Debora Botturi

An Optimization Approach to Hybrid System Control

Ph.D. Thesis

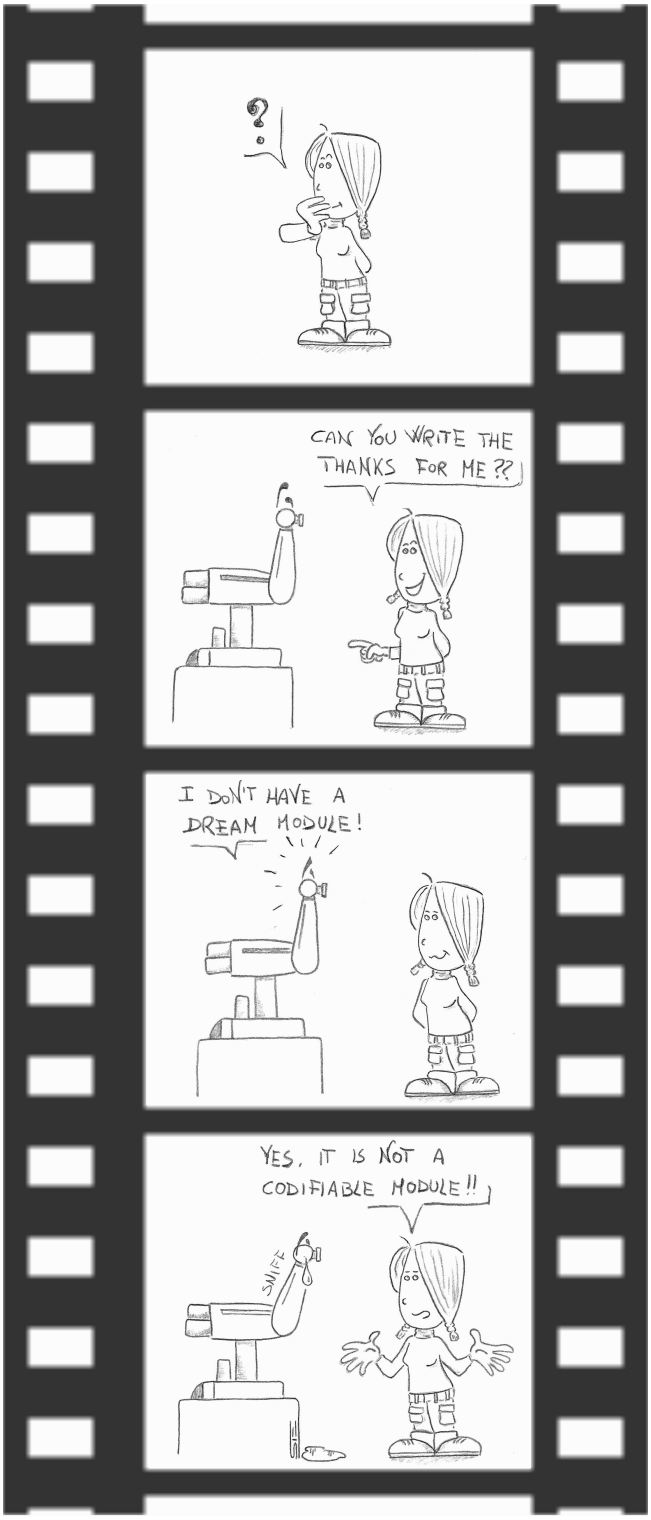
31 Marzo 2005

Università degli Studi di Verona
Dipartimento di Informatica

Advisor:
prof. Paolo Fiorini

Series N°: **TD-01-05**

Università di Verona
Dipartimento di Informatica
Strada le Grazie 15, 37134 Verona
Italy


?
CAN YOU WRITE THE THANKS FOR ME??
I DON'T HAVE A DREAM MODULE!
YES, IT IS NOT A CODIFIABLE MODULE!!
SNIFF

Summary

The continuous growth of Minimally Invasive Surgery (MIS) operations is motivated by the apparently conflicting goals of better patient care and lower health care cost that MIS has been able to achieve. However, further increase in patient care quality and cost reduction requires the development of new and more sophisticated surgical techniques, with attention to the reduction of variability in surgical outcomes and to the increase of efficiency of MIS procedures. Towards this goal, many robotic devices were developed or proposed, and some of them are used in daily hospital practice. In the robotic assisted surgery, the procedures are completely performed by the surgeon, and the robotic device is a mere, albeit sophisticated, tool. Thus, in order to keep increasing care quality and reducing its cost, it may be desirable and necessary to explore the development of algorithms that will endow a robotic assistant with the capability of collaborating with the surgeon while performing MIS procedures. In this context, collaboration means that the robot is able to carry out certain small tasks autonomously, adapting to the variability of the surgical arena, and with minimal guidance by the surgeon. This capability would reduce the strain on the surgeon, remove variability of surgeons' training levels and of procedure's outcome, and enhance the overall system efficiency by decreasing surgery time. In this Thesis we will examine the issues related to the automatic execution of robotic tasks, in uncertain environments similar to a surgical arena, and we will propose a method to compute the nominal controls for a robot that will carry out a typical surgical task.

Safety is the paramount problem in robotic surgery, and it will be even more important in the automatic execution of surgical tasks. It is also well-known that current surgical robots lack force feedback, thus depriving the human operator of a powerful way to control the robot performance. Thus, even now, a large responsibility for the correct task execution is put on the correctness of the control algorithms, because of the variability of the environment. In this Thesis, we push this concept further, by investigating safety using constraints on task execution, within an optimization approach. Variability is considered with respect to task model and constraints, and is quantified as the distance of the current behavior from the desired behavior of the robotic device during MIS. This approach allows also to continuously monitor task quality and operator performance.

Traditionally, autonomous task execution has been approached by simplifying the problem using environment engineering, i.e. by adding suitable fixtures to remove task uncertainty. The most used advanced control technique consists of pre-programmed skills, so that different tasks can be programmed and executed using the same set of basic capabilities. When an explicit model of the environment is not available, skills are parameterized to allow for a certain degree of flexibility in the task execution. If task and/or environment models are known or can be identified, it is possible to develop an explicit control law for the task execution. In this Thesis we develop an explicit control law using the theory of Hybrid Systems, and we model uncertainty with appropriate sequences of states in the Hybrid System. In particular, we use deterministic automata as task model since the surgical gestures studied are well coded in the medical literature, and can be properly represented as autonomous tasks.

We represent a surgical task as a hybrid automaton, whose elementary state represents a distinct action of the task. In this way, we can compute the nominal controls by optimizing an appropriate quality index in the subtask domain, corresponding to each action of the task. In this context, the tradeoff between explicit model knowledge and feedback compensation strongly influences system performance. We investigate the relation between a priori knowledge, as represented by hybrid task automaton, the number and form of constraints, and the type of on-line feedback to provide guidelines for the design of the complete system.

We argue that integration of techniques from hybrid system theory, constrained optimization and task constraint representation, help autonomy in uncertain environments. To demonstrate this, we simulate the autonomous execution of a suture, that is a common and very representative surgical gesture. The execution of this task requires the definition of a nominal trajectory that can drive the robot to the task successful completion. To this aim, we define an optimal strategy to compute the nominal control trajectory off-line and an on-line compensation of the deviation from the desired state trajectory. We conclude that the integration of different methods, such as hybrid systems, optimization methods, and optimal control contribute to the construction of safer autonomous executions of a realistic surgical task.

In summary, a set of tools for the synthesis of the control of a complex system has been proposed, investigated and evaluated in the context of autonomous surgical tasks. The methods developed provide a rich basis for the treatment of complex task in unknown environments and expand on the state of the art of autonomous task execution.

Acknowledgments

If I have seen further, it is by standing on the shoulders of giants.
Isaac Newton

Paolo thanks to introduce me to the robotic field and to give me the freedom to pursue my own ideas, without you I would be a different kind of person in a different kind of world, thanks for your shoulders!
Stefano thanks to be my friend, drawing for me and making the laboratory a good place to work.
Andrea the matlab guy, thanks for your help! The discussions with you are always very stimulating.
Markus you keep me down when flying is too dangerous! thanks to be my parachute when I decide to fly anyway!
All my family grazie per aver capito cosa davvero conta per me.
Dani my dear friend and example to follow, thanks for your advice.
Lars stubborn guy, sofa and newspaper are waiting for you in my home!
Vincent, Mark, Philippus lekker ding! I will never forget the philosophical matter talk!
Nicola pasta, wine and good company you can warm up your heart in a raining day!
Dado the singer, Puma has no secrets for him!
Isabella perfect doctorate-mate, I was lucky to find you on my way.
Biondo special summer schools mate, you know...I love Rome!
Robotic community thanks to give me the feeling to be part of a big group and the motivations to continue in this work, especially I want to thank Prof. Henrik Christensen to have hosted me at the very beginning, thank to the experience at CAS I decided to start the PhD, Prof. Stefano Stramigioli, thanks for your teaching and your enthusiasm, Prof. Maria Letizia Corradini, Prof. Roberto Segala, Prof. Maria Paola Bonacina and Prof. Bruno Siciliano thank you to be in my doctorate committee.

Contents

1

Introduction

The process of scientific discovery is, in effect, a continual flight from wonder.
Albert Einstein

The research described in this Thesis addresses the problem of automatic execution of a robotic task, considering uncertain environment and unmodeled dynamics. The objective is to develop a control methodology that can be used to give autonomy to a robotic device during the execution of a specific task.

The problem is motivated by many application areas, such as space exploration, custom-designed automation, service robotics, and robotic surgery. Each application area is characterized by tasks that can be carried out using structured operation sequences, which can be represented as abstract procedures. A different problem instance will require the adaptation of the abstract procedure to the specific problem conditions. In particular we focus on the application domain of robotic surgery and we want to explore the possibility of adding autonomous capability to the robotic device performing the surgery, to be of help to the surgeon. In this context, by "task" we mean a small sequence of coded surgical gestures, that is well described in the medical literature and that can, potentially, be described in algorithmic form. In this Chapter we will first present the motivations of this research in Section 1.1 and a brief overview of the scenario in Section 1.2. In Section 1.3 we describe the problem addressed in this Thesis and in Section 1.4 the outline of this Thesis is presented.

1.1 The Challenge of Surgery Automatization

Traditionally, surgery involves making large incision to access the patient organ that requires attention. This method is referred to as 'open surgery'. The incision and the significant dissection needed to allow the surgeon to visualize the surgical area are the parts of the operation that contribute to delay patient recovery and cause most of the associated pain. Minimally Invasive Surgery (MIS) is a cost-effective alternative to the open surgery, whereby essentially the same operations are performed using specialized instruments designed to enter into the

body through several tiny holes, rather than one large incision. Instead of looking directly at the part of the body being treated, the physician monitors the procedure via a special video camera inserted through one of the small holes. By eliminating the large incision and extensive dissection, much of the recovery pain and the length of hospital stay can be reduced [40, 108].

However, compared to open surgery, MIS presents additional physical, visual, motor, spatial, and force constraints. The MIS tools are constrained by the incision points. The surgeon must coordinate the hand motion that controls the tool with the remote visual display of the operation being performed by the end-tool. The limited workspace and coordination of a pair of tools further compound the challenge.

As a direct result of these constraints, there is an extended learning curve for the surgeon to gain the required skills and dexterity. Furthermore, there is a great deal of operating variability even among trained surgeons, especially with respect to the length of the operation. As demonstrated in [58], [97], [116], [152] and [164], time-motion studies of endoscopic surgeries have indicated that for activities such as suturing, knot tying, suture cutting, and tissue dissection, the operation time variation among surgeons can be as large as 50%. In suturing in particular, it was noted that the major difference lies in the proficiency at grasping the needle and moving it to desired position and orientation, without slipping or dropping it. The continuing growth of MIS operations depends in large part on the reduction of variability and increase of efficiency of MIS procedures. Towards this goal, many robotic devices have been patented or described in the technical literature [85], [152] and [168].

One class of surgical robotic devices that has been proposed to assist in endoscopic surgeries is based on the concept of teleoperation. Here, a surgeon performs the operation remotely, with a robot completely under the surgeon's command, operating on the patient. The robot motion is slaved, via mechanical linkages or computer control, to the movements and control by the surgeon. The surgeon's view of the operation may be further enhanced by remote vision to create the sense of virtual presence [108].

Various robotic positioners and stabilizer have also been proposed. In these devices, similarly to teleoperation, a robot-holding surgical tool is controlled so as to follow the surgeon's command. The role of the robot is to filter out tremor and disturbances of the surgeon's hands, so as to enhance the precision and mechanical stability of the operation. Various specialized robotic tools have also been proposed. For example, additional joints (like fingers) may be added to the end of the endoscopic tool to enhance the tool dexterity without requiring motion of the entire tool stem. This is particularly useful in cardiac operations where motion of the tool stem is limited.

It is important to note that in the various robotic surgical assistant systems described above, the surgical procedures are still completely performed by the human surgeon. The human commands are mimicked by the robotic device through computer control. The virtual presence, through visual feedback to the surgeon, creates the sensation that the surgeon is operating the tool tip instead of the tool handle, thus reducing one of the challenges of MIS. However, procedures that require a high level of skill, such as suturing, legation, and precise tissue dissection,

continue to depend on the skill of the surgeon. It is therefore desirable to have a robotic system that can collaborate while performing endoscopic procedures with the surgeon doing certain tasks autonomously to reduce the strain on the surgeon, removing variability of surgeons' training levels, and enhancing system efficiency by decreasing the operation time.

In the following Sections we will introduce the main concepts of the technologies used in this Thesis and we will formally describe the problem that we are addressing

1.2 The Big Picture

We envision autonomous capabilities enhancing a teleoperation system, in which an operator is controlling the robotic device and trades and shares control with the autonomous capabilities of the system. However, the technologies developed in this Thesis will be of interest also to fully autonomous systems, since they will help encoding the set of skills necessary to implement some form of autonomous behavior.

1.2.1 Teleoperation System

Teleoperation is defined as the control over a distance of one or more robots by a human operator. Usually it refers to a system with a master/slave configuration, where the operator works on a joystick that is kinematically compatible with the slave manipulator (see Figure 1.1). It has been shown that operator performance is improved by providing force information to the human operator [68]. Force information can be presented visually to the operator on a monitor, but the most significant performance improvement is achieved by providing force feedback to the operator, i.e. by generating forces directly with the motors of the master device. In this case the operator is said to be kinesthetically coupled to the slave and the teleoperator system is said to have bilateral control or to be force reflecting.

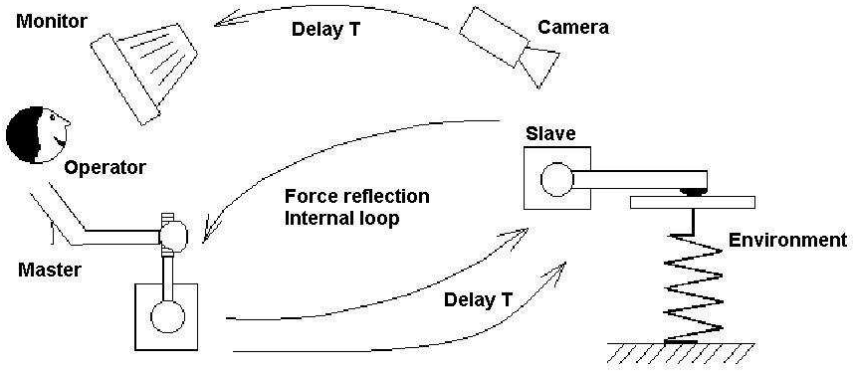


Fig. 1.1. Teleoperation schema.

Teleoperation is used in cases where the environment is not directly accessible by a human, or when it is too dangerous for on operator, or when it is necessary to scale the force exerted on the environment. Almost all of the applications of teleoperation involve contact with the remote environment, e.g. grasping, welding and puncturing. In particular, robotic applications to surgery include puncturing as one of the most usual actions.

The trade-off between stability and performance is the main consideration in the design of a teleoperation system. In fact, teleoperation control architectures analyzed in the literature can be classified in terms of their stability and performance trade-off [120]. Control algorithms for ideal kinesthetic coupling [188] are at one end of the spectrum, whereas passivity based algorithms [14] are at the other end. Conventional algorithms such as position error based force feedback and kinesthetic force feedback lie in the middle.

Both performance and stability are inherently dependent on the task for which the system is designed. Thus the need to design system controllers for the specific applications, including local and remote environments, not only in terms of parameter tuning, but also with respect to the overall control scheme. In this Thesis we focus on the control of a slave manipulator. In particular, we are interested in the interaction with soft objects. The motivation for this analysis is robotic telesurgery, i.e. surgical operations performed by robotic instruments controlled by a surgeon through teleoperation [40, 65, 108]. One of the goals of robotic telesurgery is to improve dexterity and perception during minimally invasive surgery through the use of teleoperation technology [153, 174]. Clearly, the environment in which the slave manipulator moves is not completely known and organs are characterized by partially known mechanical properties. Furthermore, by its very nature, surgery requires different modes of interaction during the course of an operation: for example free motion, contact and puncturing.

1.2.2 Surgical Systems

Robot-based surgical systems are starting to support surgeons during traditional as well as experimental procedures. These systems usually consist of a control console from which the surgeon issues manual or vocal commands to a robot which then executes them at the nearby surgical arena. Images from the surgery are returned to the console as the only sensory feedback available to the surgeon. In certain systems, a separate display shows a graphical representation of the forces applied by the robot to the patient body during the procedure. The lack of force feedback directly to the surgeon's hands is a significant drawback of today's robotic surgical systems and the cause of some criticism by practicing surgeons. A similar problem is also present in simulators of laparoscopic surgery, since a computer model replaces the patient body and therefore force feedback may not be very realistic. In the last ten years haptic interfaces have addressed this problem, producing realistic sensations in some of current simulators. Therefore, one of today's main challenges is the correct modeling of biological tissues. To determine the correct force feedback to the user, the simulator must compute the tissue deformation in real time. Of course, the deformation has to be validated by biomechanical measures.

An important issue of the research carried out in this work is the a priori knowledge, hence studies on correct force feedback and modeling of the environment were carried out, see Chapter 8, in order to take advantages of this research.

1.3 Thesis Objectives

Due to the complex and diverse nature of a generic surgical task, the use of a single control law for the whole task is not feasible. Therefore, we propose to use an approach based on *Hybrid System* theory, whereby a task is modeled as a sequence of states each endowed with its own controller. The peculiarity of hybrid systems is the interaction between continuous-time dynamics (governed by differential or difference equation), and discrete dynamics and logic rules (described by temporal logic, finite state machine, if-then-else conditions, discrete events, etc.) and discrete components (on/off switches, selectors, digital circuitry, software code, etc.) Drawbacks of this approach include the fact that the design of the controller have to take into account the continuous part and the discrete part (jumps).

This Thesis will address the problem of controlling such a hybrid system. We will consider the problem both from theoretical and practical perspectives. Our theoretical work will develop a mathematical model of control computation, and our practical work will develop techniques for the automation of complex task, such as surgical procedures. Since our analysis will be based on hybrid systems, we will investigate continuous and discrete system properties. Our model for control computation will be the hybrid automaton. A hybrid automaton is a state machine with transitions between states that are governed by a discrete logic decision. We classify hybrid automata according to what question about their behavior can be answered algorithmically. In particular, the class of hybrid automata that we will use has deterministic behavior.

The properties of a control algorithm are stated in terms of satisfying stability. However, in our case stability analysis is not enough to ensure either good performance or task termination, so our purpose is also to determine a set of parameters that can be used to determine the performance of the control system and whether or not the task is likely to terminate. In other words we have to find what does "good" and "bad" mean in the surgical environment and use "bad" as a warning for possible critical situations. The main difficulty of proposing a new control method for an application such as robotic surgery, is to ensure complete and total safety of the procedure. This imply the need to address a number of aspects, which, in a less demanding application, may be overlooked in a first development. In fact, this Thesis will deal with uncertain and variable environments, as represented by the different patient anatomies, with the true complexity of a surgical operation, without imposing unrealistic simplifications. The Thesis will also deal with the presence and the interaction of humans to provide input to the system, and with the difficulties of implementing the results.

1.4 Outline and Contributions

In summary, we will develop a framework that will expand the classical theoretical results of dynamical system control and that will be suitable for hybrid systems control. This feature should lead eventually to the extension of several control techniques that are currently available for non-hybrid systems to hybrid systems, thus rendering the analysis of hybrid system easier and more reliable.

Because of the complexity of the problem that we want to address, we need to use a number of different theories and tools. To model the system we will adopt the framework of hybrid systems and use the hybrid automata formalism, which nicely account for the mix of event and time driven dynamics characterizing task models. To develop a control for such a system, we will use the formalism of optimal control. It will allow us to use the a priori knowledge as the basis for stability analysis and for identifying the main performance properties of the complete system. In this work, however, we focus on the control aspects of the complete system, studying the optimal control for each jump and iterating the algorithm on the complete system.

The Thesis is organized as follows. After a brief explanation of the challenges of this project from the point of view of the autonomous execution, Chapter 2, we will present an overview of the state of the art of the technologies that will be the building blocks of this research. We will address the main aspects of the model and an analysis of hybrid systems in Chapter 3. In Chapter 4, together with some specific notion on suture we present the surgical aspects that are the environment and the motivation for this work. After these concepts have been sketched, we will propose a possible approach to the control design in Chapter 5. In Chapter 6 we discuss numerical solution to the optimal problems with emphasis on the solution that we adopt in this work. Finally, in Chapter 7, we give an example of application and in Chapter 8 we report some teleoperation experiments with the same setup that we are going to use for autonomous task execution experiments. In Chapter 9 the summary and an indexed list of activities for the near future concludes the Thesis.

2

Autonomous Task Execution

The question of whether computers can think is just like the question of whether submarines can swim.
Edsger W. Dijkstra

In this Chapter we will briefly summarize the state of the art of Autonomous Task Execution, i.e. the set of tools and techniques developed to guide a complex task to its completion. By complex task we mean an activity that cannot be described only in terms of reaching a set point or following a nominal trajectory, but that requires the satisfaction of logical and algebraic conditions at the beginning, during, and at its conclusion. The conditions on the task are derived from sensor measurements, logical states of the variables and a priori knowledge of the task structure. The evolution of the task must be governed by a supervisory process, which can be made explicit in the form of a monitoring agent, or implicit in the structure and the conditions of the task itself. The quality of the task, up to its successful conclusion, must be ensured and monitored by the appropriate selection of the condition set. The Chapter reviews some of the approaches developed to address the various aspects of the problem of complex task modeling and control. First, a few examples are given of complex task; then we present a description of how the research has attempted to define and model the requirements of complex tasks; finally, we review several approaches and results in the area of task modeling, execution and monitoring. The Chapter is concluded by a few considerations on the lessons learnt in this area, which motivates the approach to complex task control described in the next Chapters.

2.1 Definitions and General Concepts

To justify the difficulties in developing an infrastructure for autonomous task execution, the tasks under consideration must be extremely important and impossible to be controlled directly by a human operator. For example, automatic task execution has been implemented for the control of space probes and telerobotic devices operating in Earth orbit or on planet surface. Space probes in fact, are controlled

by uploading a series of commands, called a sequence, which has been tested on Earth for correctness on a probe simulator [75]. Furthermore, this application does not support any form of interactive control and teleoperation because of the long communication time delay between the Earth station and the remote robot. This approach has been successfully used during the Mars Pathfinder mission and proposed for next Mars exploration missions [21].

An important aspect of the tasks considered in this Thesis is that they are *complex*, i.e. cannot be controlled by Automatic Control techniques alone. Classical and Modern Control Theory, in fact, study the design of control systems whose purpose is to ensure that a vector of nominal trajectories is closely tracked by the controlled plant. The trajectories are continuous functions of time and the control actions aims at reaching an equilibrium in which the error vector goes to zero. Instead, a complex task has a structure that can be represented by the diagram in Figure 2.1. This Figure shows the mental model developed by the operators of complex systems, such as nuclear and power plants, to help manage the complexity of the operations and the flow of data [124, 150, 151]. The task, in this case, is the correct operation of the complete plant.

In these studies, the authors set the basis for the definition of the mental model of a complex task, and formulate a hypothesis on how a person interacting with the task organize the task mental representation. The hypothesis proposed is that the the task model has a layered structure, in which each level has a different degree of data abstraction. Each level processes a specific type of feedback, from raw measurements to plant states, so that the amount of data remains approximately constant at every level. With this representation, the operator can process plant information at each level, and in particular can initiate a control action at the level in the hierarchy whose data representation best describes the current situation and supports the required corrective action. In this way, for instance, an alarm signal can trigger a response action at the lowest level, without interacting with the higher levels because its importance is recognized immediately. A drift in the state of the plant instead, needs to be processed at a higher level in the hierarchy, because it is characterized by the combination of several feedback signals possibly filtered by logical operators. With such a structure, the operator adds three important features to the plant's mental model that improve her/his control performance: causal connections between layers, constant data complexity, and directed focus of attention.

Figure 2.1 as also been used as a model of the robot control paradigm *plan, react, execute* that is currently at the heart of most robotic systems. In this paradigm, the Automatic Control functions are confined to the bottom layer of the architecture, whereas techniques of Artificial Intelligence are more appropriate for the upper layers. The tasks considered in this Thesis, although much simpler than the control of a power plant, require the careful integration of execution and sensing functions, but also of reactions to unexpected events and planning complete sequences of actions. Data will be a mixture of continuous and logical signals generated by sensors and by higher level cognitive functions. Typically, these tasks will include some motion phase, some interaction with the environment, some sensor based input, and some adaptation to an uncertain environment.

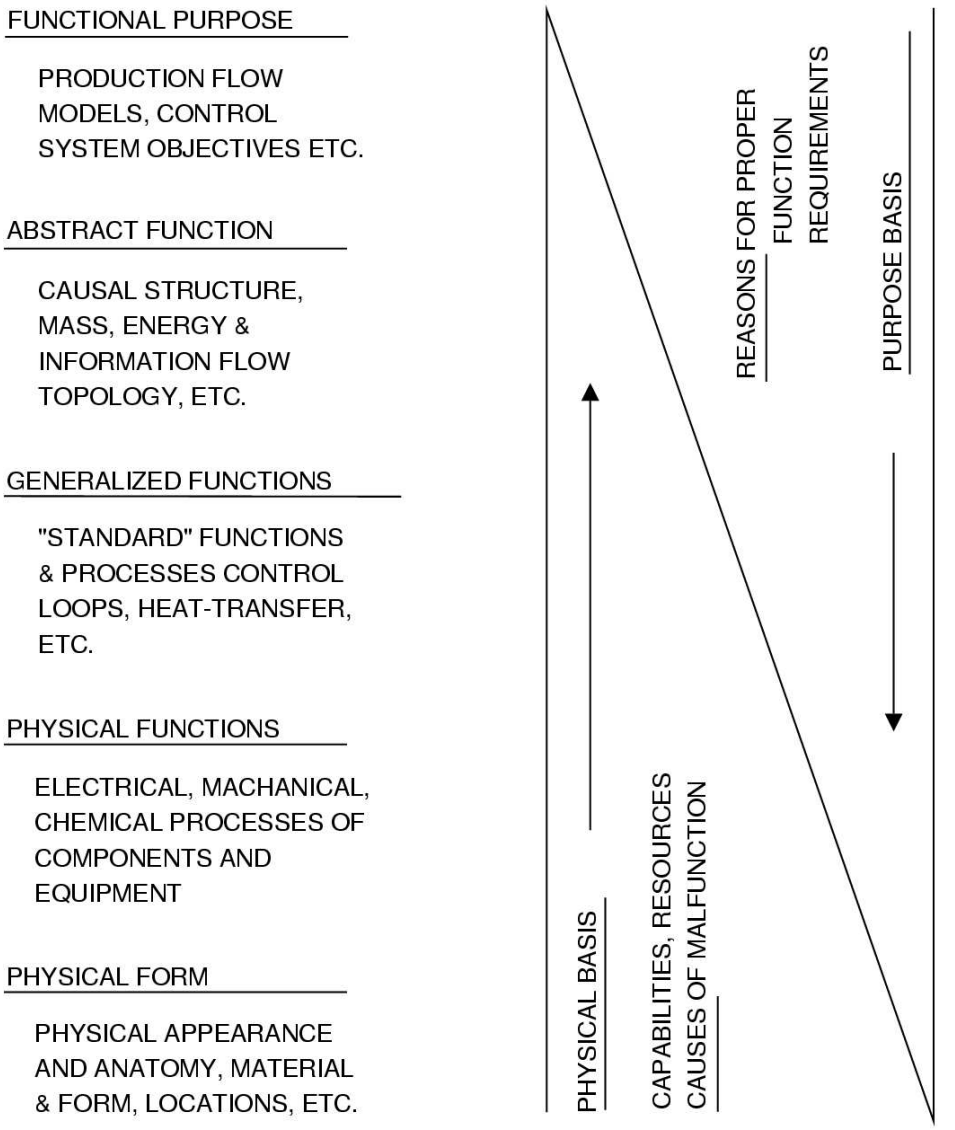


Fig. 2.1. Mental model of a plant operator.

To accomplish all these actions, the control system considered must then include at least some components of the *plan, react, execute* paradigm. In fact the architecture that is usually adopted by the robotic community to handle complex tasks is an implementation of the conceptual structure shown in Figure 2.2.

It is evident that, in this case, the term *control* acquires a much broader meaning, since the *control architecture* must include provisions for planning and reactive components. Much research has gone into identifying the best balance among the various components, in particular with respect to the planning or *deliberative* component and the reactive or *behavioral* component. The role of classical control is

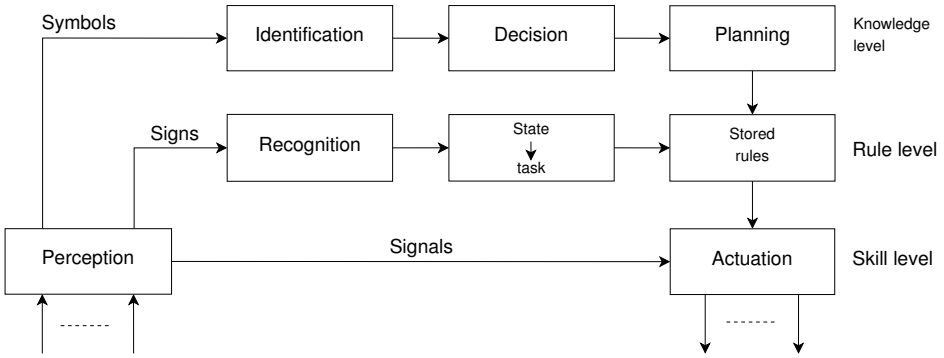


Fig. 2.2. General architecture of a robotic control system

well defined and delimited, since it is in charge of controlling the motion of the robot actuators that execute the commands generated by the upper layers of the architecture. The balance between deliberative and reactive planning elements determine the type of action that the control layer will perform. In the case of deliberative planning, the task is controlled in a *feedforward* manner, in which the nominal controls are precomputed off-line [119] by suitable planing algorithms, and executed at run time by the agent carrying out the task. When planning is predominantly reactive, a predefined set of robot skills is encoded in the control architecture. Each skill represents a behavior competing to take control of the agent. When the task is executed, the action of the robot depends on the balance of the various behaviors, whose output is weighted by coefficients depending on the desired task [50]. Clearly, these two approaches represent the two opposite ends of a spectrum of solutions, which, in general, include different percentages of deliberative and reactive planning [140].

However, as it will become clearer during the course of this Thesis, neither approach by itself, nor mixtures of deliberative and reactive components based on heuristics of various type, solve completely the problem of autonomous task execution. The reason for this inadequacy is quite simple, and it consists in the lack of a general framework that would permit analysis, simulation, testing and implementation of the complete architecture. In particular, analysis tools for the reactive part of the control architecture are missing, thus requiring the verification of the overall system performance with simulations and experiments. To overcome this limitation, *Hybrid Systems* are starting to be used in the representation, validation and execution of complex tasks. The advantage of this framework, is that it has a solid theoretical base and several analysis and execution tools have been developed for each of the layers of the architecture of Figure 2.2. In fact, Hybrid Systems include elements of control systems in their continuous states, of reactive behavior in the logic states, and of deliberative planning in their structure. Hybrid systems permit to encode the structure of a complex task with an appropriate sequence of continuous and discrete states; maintain the hierarchical structure of the standard robotic control architecture by encapsulating the continuous variables in the hybrid states; and analyses and test the overall system performance by using formal tools derived from Computer Science and Automatic Control.

To show how the research in this area has evolved in the past years, in the next Sections we will first summarize earlier work in task planning and execution, where deliberative, reactive and executive control were kept separate and gradually mixed together. Then we will describe in some detail more recent work in task planning and execution, where the awareness of the capabilities of Hybrid Systems for autonomous task execution is becoming more evident.

2.2 Early Work in Autonomous Task Planning and Execution

Earlier formalizations and architectures developed to describe and execute complex tasks included a variety of tools and procedures, but little emphasis is given to structured approaches.

Some of the earlier solutions proposed address the need of intelligent data acquisition and processing, i.e. the development of sensors capable of producing also a logical output from the data collected. This approach was called the Logical Sensor Specification [103], in which physical devices were separated from each other and from the central controller by software layers, where local processing of feedback signals and actuator commands could take place. Other proposals put more emphasis on the local reactive capabilities of the controlled system and they used mechanical devices able to produce an implicit control similar to the Human Reflex Action [30]. These approaches were directed to solve specific problems, and therefore more general structures were studied based on Expert Systems [173, 121] or on Artificial Neural Networks [54, 86]. The first family of solutions permits the system to learn some of its own control procedures and the modularity of the rule base allows for easy implementation and update. The second approach is directed towards integrating real time performance with some strategic knowledge. Finally an important area of research was Intelligent Control [51, 136], in which nested control loops organized in the hierarchical structure described in Figure 2.2 supervise all aspects of the task evolution. Intelliget control is still a general approach, proposed for a number of different applications and is not specific to robotics. However, because of this fact, it is still very application dependent and cannot easily extended to generic robotic tasks.

To address the need of higher flexibility, with specific reference to the control and execution of robotic tasks, a new approach was proposed in the context of the Telerobotic Testbed, described in [100], which was a telerobotics laboratory developed at JPL-Caltech under NASA sponsorship, completely devoted to the development of new control strategies for telerobotic tasks. During the development of the testbed, teleoperation capabilities were enhanced with the addition of dextrous manipulators [23], dual-arm operation [99], and automatic sequence generation and execution. This last development is relevant to the research described in this Thesis since it represents an approach to formulate and execute a sequence of operations in a partially structured environment. Tasks were built interactively using a series of pre-programmed *skills* such as, *guarded motion*, *move-to-push*, *slide*, *insert*, and so on [22], i.e. all actions that could be framed in the context of classical control algorithms. Skills were parameterized, so that different operations

could be programmed using the same set of basic capabilities. The execution of the sequence was carried out by maintaining two queues of commands, one requiring only feedforward commands and the other requiring some feedback from the sensors, called Reflex Actions [24]. The execution of each skill was monitored by defining termination conditions on each controlled variable. The description of the complete system, with sequence input and execution, and a discussion of robustness issues during execution is presented in [25]. This approach was tested in laboratory experiments of complex sequences: opening Orbital Replacement Unit (ORU) door, and turning bolts were successfully completed. However it is not clear how much engineering of the environment went into ensuring that the task, i.e. the complete sequence of skills, completed successfully. Furthermore, the uncertainty relative to the task is compensated by defining ranges, rather than values, of the parameters, but without including additional states into the task model.

All the approaches presented in this Section share the common aspect of being developed for a specific task, without a unifying framework for task analysis and control and thus are examples of *ad hoc* solutions rather than instances of a formal theory of complex system analysis and control. In the next Section we will briefly review some of the more recent results in task control and show how the field is evolving towards a more extensive use of the formalism of Hybrid Systems.

2.3 Recent Work in Autonomous Task Planning and Execution

As mentioned in the previous Section, the main limitation emerged from the summary of the earlier work, is the lack of a structured approach to the analysis and control of a complex task. In this Section we trace the more recent development, leading to the gradual establishment of the Hybrid System methodology as the comprehensive framework for complex task study. In particular, one of key theoretical element missing in the earlier work is the tight and effective integration of deliberative and reactive planning with execution and control. In this Section we will briefly summarize a few approaches showing how this problem is being addressed, and will describe solutions that respond to this need with various levels of integration. The approaches presented in this Sections are subdivided according to the technology that they address: single sensing and control actions, or global architectural issues.

2.3.1 Design, Sensing and Actuation Issues

Clearly, the main issues of complex task control are related to the architecture and communication characteristics of the control system. However, also the basic elements of the architecture, such as sensors and actuators, must be capable of responding to the demands of the architecture. In this Section, we briefly document some of the enhancements made to robotic devices to make them compatible with the needs of complex task control. In particular, we first describe an example of advanced processing at the low level of the architecture, then at the middle level, and finally in the top layer of the architecture.

The issues of integrating actuation and perception at the low level of the control architecture is addressed by the approaches summarized next. Visual servoing is a form of trajectory control in which feedback data are produced by image processing, thus providing a form of feedback control higher than standard signal processing. The approach described in [115], presents a few experiments in this area, with reference to the surgical robotics domain, in which the motion of a laparoscopic instrument is guided using the image of a laparoscopic camera. In these experiments, the higher level of the reference architecture of Figure 2.2 is supplied by the human operator, who teleoperates the surgical instruments with the help of visual servoing. Thus, this technique can be described as a *semi-autonomous* enhancement to teleoperation. To achieve the motion of the instruments, the authors designed a new instrument-holder equipped with laser beams projecting a known pattern on the internal surfaces of the patient. This pattern, once properly identified and measured, is the basis of visual servoing, since it identifies the current position of the instrument in the patient's body and allows to safely move the instrument even when it is not in the field of view of the endoscopic camera. With this set up, the surgeon can specify the final position of the instrument tip by indicating it on the monitor with a cursor, and bring it to a desired position without an exact knowledge of its current position. The technology described in the paper is an example of essential technologies for autonomous task execution, since it provides the low level sensing and actuation necessary to give the bottom layer of the architecture the desired processing autonomy.

The needs of the middle layer of the architecture shown in Figure 2.2, namely the capability of adapting a planned task execution to variations of the task parameters are addressed in [39]. Here, the authors describe an example of *reactive control* methods applied to a robotic surgical task. Reactive control is impemented by means of a new device, the *EndoBot*, which can provide some form of shared and traded control during a MIS procedure. The EndoBot consists of a docking station holding two four-degree-of-freedom devices, each capable of moving a MIS instrument along four axes, since two axis are constrained by the incision point. The device was designed to cooperate with the surgeon, in the sense that the surgeon could teleoperate some of the axes and the robot can move the other axes autonomously. The approach relevant to this Thesis is the role of the device during a simple suture in which it compensates autonomously unexpected variations in the environment. However no explanation is given on how the EndoBot would autonomously execute the suture and what sensory conditions and motor controls were used to progress from one phase to the next.

At the upper level of the reference architecture, there is a planning layer that is in charge computing the initial nominal task trajectory. One of the main problem arising with this configuration is the division of activities between the middle level, i.e. the reactive planner/behavior, and the top level. In particular it must be decided at what point the nominal task trajectory need to be completely replanned and the local corrections applied by the middle level are no longer sufficient to ensure task conclusion. An example of the solution to this problem is presented in [20] in the context of the autonomous landing of an airship. The task here is carried out using pure visual servoing, and is a complex control problem requiring a system with a good level of autonomy. In this case the autonomy is necessary to

compensate the severe disturbances of wind gusts, which require the replanning of the landing maneuvers when the trajectory exceeds a given tolerance.

Another type of autonomy is discussed in reference [80] where an approach is presented in which the telerobotic system improves the way the operator's commands are executed by the remote robot, by varying the ratio between the commanded and the executed velocities of the robot. In this way, for example, the robot can autonomously adjust the approach velocity of the robot to a target to improve accuracy and overall performance. This approach is demonstrated in several Fitts' law[1] [88] type experiments. When replannig is too difficult for an automatic agent, a human operator can be called upon for help, and the control functions can be shared between the robot controller and a human operator, as described in [84]. In this case, the operation of the robot is mostly autonomous, and the task is the assembly of micro MEMS components. However there are situations in which the robot cannot operate autonomously, for example when the uncertainties in the microassembly are different from those considered when developing an assembly plan. The robot then requires human assistance and, depending on the case, the control can be shared between the robot and the operator, or traded, with full control given to the human. In this research

Another aspect of complex task analysis refer to the preliminary modeling of the task structure. A correct model is essential to express the *a priori* knowledge about the task and to set up the various task evolution scenarios. An approach to represent manipulation tasks in a formal manner is described in [29]. The paper describes a set of grasping tasks in terms of the way humans carry them out. The preferences of several human subjects are coded into a knowledge base, which is then used to different grasp patterns in terms of grasping primitives. The primitives are also classified according to the shape of the object to be grasped and the type of robot hand. The task is represented as a set of discrete rules, that should be used by a controller as pre and post conditions to the various phases of a grasping task. In this example, the grasp patterns are the different instances of a grasp task, whereas the grasp primitives represent the behavior level of the task control. Another example of task modeling is described in [27] in the context of manufacturing operations. In this research, an assembly task is described by means of a graph in which nodes and edges represent the various objects involved and the actions required. However, this method does not enter into the detail of how the sequence of actions described should be executed and controlled by an automatic device. In any case, the definition of a task as an oriented graph is one of the current methods used to specify the task structure and execution flow.

Having briefly discussed the single elements and technologies concurring to the implementation of the control of a complex task, in the next Section we will examine the structure of the software system that must be set up to support advanced control functions.

[1] Fitts' law is a model of human psychomotor behavior developed in 1954. According to Fitts' Law, the time to move and point to a target of width W at a distance A is a logarithmic function of the spatial relative error (A/W).

2.3.2 Architectural Issues

Key to the effective operation of a task-level control system is the underlying software architecture and the mechanisms governing the exchange of functions and duties among the different agents in the control system. This is a very active research area which is currently being explored by a number of large projects, both in Europe and in the US, and a small sample of the results achieved so far is discussed next.

An effective balance of feedforward and feedback control is described in [157] where the authors presents the analysis and implementation of the control system for two different tasks, walking and object grasping. In this paper the authors describes the network of actions and the logical checks that form the *Task Control Architecture* used for the two tasks. It is interesting to note that the control is based upon a rather fix task model, in which a set of concurrent control actions are executed at different frequencies to ensure some concurrency. Although rather powerful, this approach seems to encode the tasks in the control and may require substantial modification to be used to control a different task. Thus the deliberative component of planning is represented by the task tree and the *a priori* knowledge by the sequence of task actions. The issue of modifying the task to adapt to a changed environment seems not to be addressed in detail.

A different approach is described in [26] in which the authors describe an architecture for the motion control of a group of autonomous vehicles based on the careful tuning of various behaviors. In this paper, the task demanded to the robots is the exploration of an area, and therefore it involves almost exclusively the motion of robots while keeping an assigned formation. In this case, the task is coded in terms of the robot formation and the maneuver goal. Because of the linearity of the task it is acceptable to have a preponderance of reactive behavior versus deliberative planning. It is questionable whether this approach is applicable to more structured tasks, where temporal and spatial sequences of actions must be combined to complete the task.

Another exmple of an architecture based on the behavior paradigm for planning and control is described in [9] in the context of exploring and mapping an indoor environment. The behaviors implemented for the task of map formation, are *go to*, *obstacle avoidance*, *wall avoidance*, *corridor following*, *door passing*, and *docking* to reach the recharging station. However, in this case the mapping task is specified in two forms. The actual measurements and memorization of the environment is carried out autonomously by the robot, whereas the higher functions of navigation and exploration patterns are not performed autonomously, but replaced by following a human guiding the robot through the environment, thus allocating resources according to the ability of an agent to perform the function.

An implementation of the deliberative and reactive framework in autonomous task execution is described in [19], where the authors describe in detail the implementation of a robot control architecture consisting of three main layers: planning, execution, and reaction. The first layer takes care of the definition of the high level goals using the environmental map. The execution layer partitions the goals into subgoals and activates the appropriate behaviors. Finally, the reactive layer interfaces the robot with sensors and actuators and modifies the plans in real time. The architecture proved very successful in an exhibition where the robots traveled more

than 3000 Km. However, tasks executed by the robots had not a fixed configuration and can be rearranged depending on the various sensor input. Furthermore, no manipulation was involved, thus constraints on task execution were somehow simpler to model and satisfy, and the switch to the deliberative planner was only required to compute a new navigation trajectory in the environment.

The integration of vision and grasp planning and execution is described in [114]. The authors describe the experiments carried out to verify the performance of a vision-based tracking system combined with a grasp planner. The tracking system is used to estimate pose and motion direction of an object, and provides these data to the grasp planner, which then plans a stable grasps and commands a robotic hand to execute. The integration of vision and grasping is a key element in the development of autonomous manipulation tasks and this paper describes the *ad hoc* architecture used to carry out the task.

Failure recovery and graceful degradation are important issues in the context of complex tasks, where safety may be crucial. Since, in most cases, complex tasks are carried by independent agents, safety depends on the correct functioning of all the agents. However, since malfunctions do occur, it is important to identify in which way a group of agents can gracefully alter its operation to maintain its basic capabilities. These same considerations can be extended also to robotic teams, as discussed in [76], where the authors examine the main causes of malfunctions: communication failure, robot partial malfunction, and robot death. The approach discussed is based on capitalistic market economy, a flavor of game theory, in which each robot is an independent agent trying to maximize its own good. However, since the payoff depends only on the team success, the satisfaction of the team goal equates to maximization of each individual profit. Using this approach, the authors show the capability of the team to reconfigure its resources as a function of detected failures and the team ability to carry out autonomously the assigned task, even in the presence of several failures. In this example, both deliberative and reactive planning are carried out in an implicit form, resulting from the optimization algorithms implemented in the control software.

The main criticism one can move to the approaches presented above, and to mainstream research in this area is the lack of a unifying method to study complex tasks, whether during modeling, analysis and, finally, execution. A possible answer to this lack of common theoretical background is given by the increasing use of *Hybrid System* methodologies to support the description and the execution of autonomous tasks.

A typical example of how this formalism can be used in the context of complex task control is given in [143]. The application described in the paper refers to the everyday action of opening a door, but it integrates different control and sensory modalities of the type needed in the real autonomous surgical procedure, which are the main focus of this Thesis. The paper first gives a model of the *open door* task, in which each action of the task is assigned to a specific control mode and qualified by appropriate sensor thresholds to ensure completion. The task is described using the *Hybrid System* formalism and the control parameters needed in each continuous state of the Hybrid Model are estimated on line during task execution. The authors indicate that experiments carried out with this approach were successful about 90% of the time, mostly due to the lack of an accurate error

recovering strategy. The challenge in this case is to extend this approach to more complex cases, such as medical procedures, where safety is paramount.

Other examples of use of the Hybrid System formalism in complex task analysis and control is represented by the following papers. In [57] the authors describe the implementation of a robot capable of autonomous installing warning sphere on high voltage cables. In this case, the task to be carried out by the robot is well defined and the authors do not provide much detail on how the various phases of the task are controlled by the underlying hybrid system monitor. In [62, 63] a hybrid automaton is used to model the cooperation of multiple mobile robots to perform a coordinated manipulation. In [62] each robot of a team is assigned a *role* and the dynamic exchange of the roles during the execution of the cooperative manipulation of an object is governed by a hybrid automaton. A suitable utility function determines when and to what role a robot should be assigned, and simulations show that this policy supports the execution of the task. The novelty of the approach presented in [63] is in the composition of the hybrid automata representing each individual robot into a single, more complex, automaton describing the complete cooperative task. The authors point out the possibility of doing formal task analysis and verification using the hybrid system methodology to identify possible faults and deadlocks in the task execution.

Finally, another area of application of Hybrid System theory, and of great potential impact, is the control of the navigation and interaction of autonomous vehicles. An example is [7], in which the authors summarize their contributions to this research area. The focus here is on intelligent multi agent systems that eventually will replace centralized control systems, as in the case of air traffic management, or will enhance human resources, as in the case of automatic vehicle control. The Hybrid System framework is ideally suited for autonomous, or semi-autonomous, agent control. In fact, at the continuous level, each agent chooses its own optimal strategy, while discrete coordination is used to solve conflicts. This approach has several advantages with respect to centralized control: has the potential to yield an optimal design, it is intrinsically reliable and scalable, and is flexible to adapt to different conditions, traffic levels, and unexpected needs. In the context of traffic management, the Hybrid System definition of *safe sets*, has a very real meaning, since it represents the state set able to guarantee, for example, collision avoidance. In particular, a hybrid automaton can be used to represent the different operating conditions of the autonomous vehicle, and thus frame and represent in a consistent way the discontinuities of navigation control.

Concluding this brief summary of approaches and techniques to the architectural design of complex task control systems, it appears that the methods satisfying most of the requirements on model definition, formal analysis, simulation and execution are based on some variation of the Hybrid System methodology. This methodology, as it is discussed in the next Chapters, is still in its infancy and therefore it has not yet reached a unified and all encompassing structure. It is still a collection of many similar definitions and methods, yet the techniques developed so far are very relevant and useful to the objective of this Thesis.

2.4 Conclusions

In this Chapter the main issues and developments relevant to autonomous task execution have been briefly described. First the distinction between simple and complex tasks has been addressed, and then some of the difficulties found in the development of autonomous systems listed in terms of modeling, analyzing, and executing the sequence of actions representing a complex task. To show how this problem has been addressed by many researchers in the past twenty years, first earlier work is summarized to show the variety of solutions proposed. Then, most recent papers are summarized emphasizing the clear trend emerging towards the development of a mathematical framework able to support all the needs in complex task planning and control. The current trend is towards the adaptation of Hybrid System methodologies to the needs of complex task execution and control. This trend is illustrated by pointing out the results described in a few key papers in the area. In the following Chapters, this trend will be further justified, by analyzing the advantages offered by the Hybrid System methodology in the development of a control strategy for a surgical suture.

3

Hybrid System

Everything should be made as simple as possible, but not simpler.
Albert Einstein

The hybrid systems of interest in this research are dynamic systems, where the behavior of interest is determined by the interaction of continuous and discrete dynamics.

There are several reasons for using hybrid models to represent the dynamic behavior of a complex task. Reducing complexity is an important reason for dealing with hybrid systems; this is accomplished by incorporating models of dynamic processes having different levels of abstraction. For example a thermostat typically sees a very simple model of the complex heat flow dynamics adequate for the task in hand. Another example is the analysis of non linear systems. In order to avoid dealing directly with the set of nonlinear equations, one may choose to work with sets of simpler equation (e.g., linear), and switch among these simpler models. The advent of digital machines has made hybrid systems very common indeed. Whenever a digital device interacts with the continuous world, the behavior involves hybrid phenomena that need to be analyzed and understood.

It is not hard to find examples of systems that motivate the need for studying hybrid systems. Many examples can be found in the literature, for instance the management of a fishery resource [141], computer disk system [92], motion control systems [49], robotics (a non exhaustive list [78,83,163,184]), power systems [105], systems in classical mechanics [48], air traffic management [172] and automated vehicles [127].

In this chapter we will give a brief overview of the main aspects of the model and analysis of Hybrid Systems, that will form the framework for this research.

3.1 Survey of systems and model

Construction of models (abstractions) of parts of reality (systems) and investigation of their properties are fundamental issues for all scientists. The models can be more or less formal but all have the property that they try to link relations

in the system to some kind of pattern. Loosely speaking, a model of a system is a tool used to represent some sort of knowledge of the system without making experiments. It is used for instance to predict the future behavior or to design a controller. The degree of agreement between the system and the model determines the usefulness of the model. One of the tasks of applied mathematics is to generate models for the description of systems in different disciplines. A mathematical model formally describes the relation between different quantities and variables in the system by a mathematical relation. Such models are commonly used in modern engineering science, for instance control engineering and computer science, and are the class of models studied in this Thesis.

Due to the close connection between a system and its model (see Figure 3.1) it is common to drop the distinction and use the meaning of the terms interchangeably. Hence, the word system is used to denote either some part of reality or its model, which in this Thesis means a mathematical relation between different variables. A system having state variables that change values as a function of time is called a dynamic system, as opposed to static systems which remain in a configuration which does not change with time.

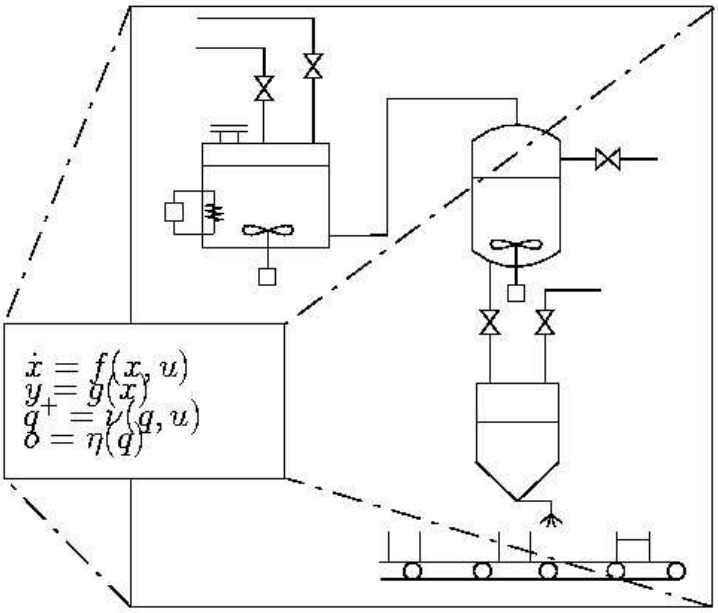


Fig. 3.1. Mathematical model describing a physical system.

3.1.1 Classification of Dynamic Systems

In [125] a useful characterization was proposed of dynamical system. Roughly speaking, a dynamical system describes the evolution of a state over time. As a well-known illustration we could consider the following differential equation:

$$\dot{x}(t) = f(x(t), u(t)) \tag{3.1}$$

The state variables at time t in this case are given by an array $x(t)$ (e.g. position and velocity of a rigid body) and typically take values in $X \subseteq \Re^n$ and evolve over time $t \in T \subseteq \Re$ according to (3.1). The variable $u(t) \in U \subseteq \Re^m$ at time t denotes either control inputs, that may be chosen as we like (e.g. commands of the user), or disturbance (e.g. sensor error). Loosely speaking, one could say that the state $x(\tau)$ at time τ summarizes all the information from the past (for times $t \leq \tau$) of the system that is needed in order to understand the future behavior of the state $x(t)$ for $t > \tau$ except for the purely external effects due to the inputs and disturbances [159]. In principle one could say that a dynamical system is defined by a relation between the current state and an applied input or disturbance and a state at a later time instant. For the differential equation given in (3.1) this means that we have a map ϕ from the (initial) state $x \in X$, the initial time $\tau \in T$, the (final) time $\sigma \in T$ with $\sigma \geq \tau$ and a function $u : [\tau, \sigma] \to U$ to the values of the state array at time $x(\sigma)$. Hence,

$$x(\sigma) = \phi(\sigma, \tau, x, u) \tag{3.2}$$

This is an interesting way to look at systems which yields a nicely unifying way to include also computer science models. Sontag [159] formalized the concepts as follows.

Definition 3.1. *A time set T is a subgroup of $(\Re, +)$.*

For any such set, T_+ is the set of nonnegative elements $\{t \in T | t \geq 0\}$. By notational convention, when the time set T is understood from the context, all intervals are assumed to be restricted to T.

For each set U and interval I, the set of all maps from I into U is denoted by

$$U^I = \{\omega \mid \omega : I \to U\} \tag{3.3}$$

If T and k is a nonnegative integer, the set $U^{[0,k)}$ can be identified naturally with the set of all sequences

$$\omega(0), \ldots, \omega(k-1)$$

of length k consisting of elements of U, i.e. the Cartesian product U^k. In the particular case in which I is an empty interval , the set in (3.3) consists of just one element, which we denote as $\diamond$; this can be thought of as the "empty sequence" of zero length.

The next definition provides the abstraction of the concept of system.

Definition 3.2. *A system (or machine) $\Sigma = (T, X, U, \phi)$ consist of:*

- *a time set T*

- *a nonempty set X called the state space of Σ*
- *a nonempty set U called the control-value or input-value space of Σ*
- *a map $\phi : D_\phi \to X$ called the transition map of Σ, which is defined on the subset D_ϕ of*

$$\{(\tau, \sigma, x, \omega) \mid \tau, \sigma \in T, \sigma \leq \tau, x \in X, \omega : [\sigma, \tau) \to U\}$$

such that the following properties hold:

Non-triviality: *for each state $x \in X$, there is at least one pair $\sigma < \tau$ in T and some $\omega \in U^{\sigma,\tau}$ such that ω is admissible for x, that is, so that $(\tau, \sigma, x, \omega) \in D_\phi$;*

Restriction: *if $\omega \in U^{[\sigma,\mu)}$ is admissible for x, then for each $\tau \in [\sigma, \mu)$ the restriction $\omega_1 = \omega|_{[\sigma,\tau)}$ of ω to the subinterval $[\sigma, \tau)$ is also admissible for x and the restriction $\omega_2 = \omega|_{[\sigma,\mu)}$ is admissible for $\phi(\tau, \sigma, x, \omega_1)$;*

Semigroup: *if ω, τ, μ are any three elements of T so that $\sigma < \tau < \mu$, if $\omega_1 \in U^{[\sigma,\tau)}$ and $\omega_2 \in U^{[\tau,\mu)}$, and if x is a state so that*

$$\phi(\tau, \sigma, x, \omega_1) = x_1 \text{ and } \phi(\tau, \sigma, x, \omega_2) = x_2$$

then $\omega = \omega_1\omega_2$ is also admissible for x and

$$\phi(\tau, \sigma, x, \omega) = x_2;$$

Identity: *for each $\sigma \in T$ and each $x \in X$, the empty sequence $\diamond \in U^{[\sigma,\sigma)}$ is admissible for x and*

$$\phi(\tau, \sigma, x, \diamond) = x.$$

Based on the fact that limited measurements are available, the following concept is then natural.

Definition 3.3. *A system or machine with outputs is given by a system Σ together with*

- *A set Y called the measurement-value or output-value space;*
- *A map $h : T \times X \to Y$ called the readout or measurement map.*

Elements of X are called states, elements of U are control values or input values, and those of Y are output values or measurement values. The function $\omega \in U^{[\sigma,\tau)}$ are called controls or inputs.

The definition of system is intended to capture the intuitive notion of a machine that evolves in time according to the transition rules specified by ϕ. At each instant, the state x summarizes all of the information needed in order to know the future evolution of the system.

Based on state type, the following classification systems can be made:

Continuous state: the state takes values in a continuous set, e.g. $\Re^n$ for some $n \geq 1$.

Discrete state: the state takes values in a countable or finite discrete set $\{q_1, q_2, \cdot\}$.

Based on the set of times T over which the states evolves, dynamical systems can be categorized in :

Continuous time: T is a continuous set, e.g. a subset of $\Re$.
Discrete time: T is a discrete set, e.g. a subset of the integers $\mathbb{Z}$.

Third, we distinguish systems, based on the mechanism that drives their evolution, which can be:

Time-driven: the state of the system changes as time progresses, i.e. continuously (for continuous time systems), or at every tick of the clock (for discrete time systems). It is common to have a model described by a continuous time continuous state system expressed by a differential equation (3.1). These systems are usually referred to as continuous systems.

Event-driven: the state of the system changes due to the occurrence of an event. An event corresponds to the start or the end of an activity. In general, event-driven systems are asynchronous and the event occurrence times are not equidistant. Typical examples of event-driven systems are manufacturing systems, telecommunication networks, parallel processing systems, and logistic systems. For a manufacturing system possible events are: the completion of a part on a machine, a machine breakdown, or a buffer becoming empty.

We can also have combinations of continuous and discrete states, of continuous and discrete time, or of time-driven and event-driven dynamics. The resulting systems are called hybrid. In this Thesis a hybrid system essentially is a system the evolution of which is over continuous time, but there are also discrete time instants when "something happens" (e.g., the occurrence of an event that results in a mode change of the system. Usually the mode is then characterized by a discrete state variable).

3.1.2 Discrete Event Systems

Discrete Event Systems (DES) are models that arise naturally for a large class of systems, mostly man-made and highly complex. The interest in discrete event systems in control engineering has been intensified in the last decade, and a reference describing their properties is for example [5].

The DES mathematical models is described by asynchronous discrete-time discrete state systems, where the times $\{t_k \in \Re \mid k \in \mathcal{N}\}$ are not known a priori.

Automata or finite state machines are the most common models for discrete time and discrete state (event-driven) systems, see Figure 3.2. We need some notation before we can give a formal definition of automaton. For a set V we denote the collection of all subsets of V (the power set) by $P(V)$. Moreover, if we have two sets V and W a partial function from V to W is a mapping that is not necessarily defined for all values of V , but only for a subset D of V (its domain). Hence, if f is a partial function from V to W, then there is a subset $D(f) \subseteq V$ such that f is a function from $D(f)$ to W.

Definition 3.4. *An Automaton is defined by the triple $\Sigma = (Q, U, \phi)$ with*

- *Q a finite or countable set of discrete states;*
- *U a finite or countable set of discrete inputs or the input alphabet;*

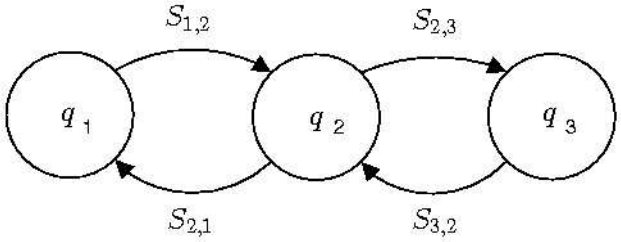


Fig. 3.2. State transition diagram.

- $\phi : Q \times U \to P(Q)$ *is a partial transition function.*

In case Q and U are finite, we speak of a finite automaton.

The evolution of an automaton is rather simple; given a discrete state $q \in Q$ and a discrete input symbol $u \in U$, the transition function defines the collection of next possible states $\phi(q, u) \subseteq Q$. Note that since ϕ needs not be defined for all combinations of q and u, this means that not from all discrete states all input symbols can be applied. On the other hand, sometimes the set of next possible states may have more than one element. This is the so-called *nondeterminism*. In case this nondeterminism is absent, i.e. $\phi(q, u)$ has zero or one element, we speak of a *deterministic* automaton, which fits more or less directly in the definition of a system as given in Definition 3.2.

3.2 Hybrid Systems

Complex systems typically posses a hierarchical structure, characterized by continuous variable dynamics at the lowest level and logical decision-making at the highest. Virtually all control systems today perform computer-coded checks and issue logical as well as continuous-variable control commands. Such systems are "hybrid" systems. Traditionally the hybrid nature of these systems is suppressed by converting them into either purely discrete or continuous entities. Motivated by real-world problems, we introduce "hybrid systems" as interacting collections of dynamical systems, evolving on continuous-variable state space, and subject to continuous control and discrete phenomena.

A look at the literature shows that there are many approaches to the modeling, analysis and synthesis of hybrid systems. They can be characterized and described along several dimensions. In broad terms, approaches differ with respect to the emphasis on or the complexity of the continuous and discrete dynamics, and on whether they emphasize analysis and synthesis results or analysis only or simulation only. On one end of the spectrum there are approaches to hybrid systems that represent extensions of system theoretic ideas for systems (with continuous-valued variables and continuous time) that are described by ordinary differential equations to include discrete time and variables that exhibit jumps, or extend results to switching systems. Typically these approaches are able to deal with complex

continuous dynamics and emphasize stability results. On the other end of the spectrum there are approaches to hybrid systems that are embedded in computer science models and methods, that represent extensions of verification methodologies from discrete systems to hybrid systems. Typically these approaches are able to deal with complex discrete dynamics described by finite automata and emphasize analysis results (verification) and simulation methodologies. There are additional methodologies spanning the rest of the spectrum that combine concepts from continuous control systems described by linear and nonlinear differential/difference equations, and from supervisory control of discrete event system that are described by finite automata and Petri nets to derive, with varying success, analysis and synthesis results.

There are analogies between certain current approaches to hybrid control and digital control systems methodologies. Specifically, in digital control one could carry out the control design in the continuous time domain, then approximate or emulate the controller by a discrete controller and implement it using an interface consisting of sampler and a hold device. Alternatively, one could obtain first a discrete model of the plant taken together with the interface and then carry out the controller design in the discrete domain. In hybrid systems, in a manner analogous to the latter case, one may obtain a discrete event model of the plant together with the interface using automata or Petri nets; the controller is then designed using discrete event system (DES) supervisor methodologies. The model consists of three basic parts: continuous-time plant, finite control automaton, and interface. The interface in turn consists of two parts, viz. an analog-to-digital (AD) converter and digital-to-analog (DA) converter. The supervisor model is illustrated in Figure 3.3.

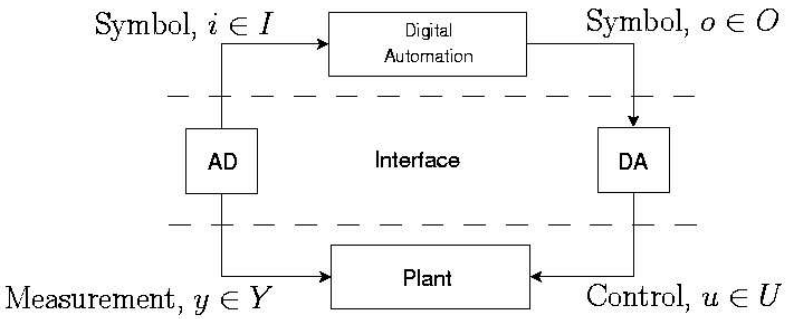


Fig. 3.3. Deterministic optimum control problem.

Associated to the plant are an input space U, a state space X, and an output space Y , while the controller automaton has a (finite) input space I, a state space Q and an output space O.

Further information on hybrid systems may be found in references [13, 16, 17, 18, 94, 137, 145].

3.2.1 Modeling Approaches

Models are the ultimate tools for obtaining and dealing with knowledge. Recently, the modeling of hybrid system, with special emphasis on process control application, has given rise to an abundance of research activity. There has been considerable effort to develop theoretical frameworks and models for such systems, and different directions have been pursued, depending on chosen goals. In this research, we approach the study of modeling hybrid dynamical systems with the aim of designing control laws. With this purpose in mind, hybrid automata are discussed next.

Different mathematical paradigms have been used for modeling hybrid systems reveal the diversity of the researches. Tavernini [166] used differential automata; Nerode and Kohn [94] took an automata theoretic approach to systems composed of interacting ODEs and finite automata; Antsaklis in the same Lecture Notes took a discrete event dynamical systems approach; Brockett [49] combined ODEs and discrete phenomena to describe motion systems; Back [94] provided a framework suitable for numerical simulation. Alur [11] used hybrid automata, and an extension of timed automata is used in [12,128]; Chaochen [64] used Duration Calculus for hybrid real-time systems, and Benveniste [38] proposed a behavioral framework of hybrid systems modeling with emphasis on compositionality and use of multiform time. Clearly, these models were developed for different purposes with assumption arising accordingly.

The choice of a suitable framework is a trade-off between two conflicting criteria: the modeling power and the decisive power. The modeling power indicates the size of the class of systems allowing a reformulation in terms of the chosen model description. The decisive power is the ability to prove quantitative and qualitative properties of individual systems in the framework. A model structure that is too broad cannot reveal specific properties of a particular element in a model class. The size of a model class is often taken too large for analysis purpose. Even for the easiest hybrid systems analysis and control problems are often undecidable[1], NP-complete or NP-hard[2], or require a high computational load.

Although hybrid automata can be considered as one of the most descriptive and general models for hybrid systems, analysis and control design based on these models is often results in computationally hard problems ([5,61]). However, some

[1] A problem is undecidable if there cannot exist generally applicable algorithms that solve the problem.

[2] A *decision* problem is a problem that has only two possible solutions: either the answer "yes" or the answer "no". A *search* problem is a problem for which we either have to give a solution or have to establish that the problem has no solution. A search problem (such as an optimization problem or a design problem) is called NP-hard if the corresponding decision problem (e.g., deciding whether or not there exists an optimal solution or an optimal design) is NP-complete. An NP-complete problem can only be solved in polynomial time if the class P would coincide with the class NP [2]. With the present state of knowledge it is still an open question whether the class P coincides with the class NP. However, since no NP-complete problem is known to be solvable in polynomial time despite the efforts of many excellent researchers, it is widely conjectured that no NP-complete problem can be solved by a polynomial time algorithm.

special classes of hybrid systems for which tractable analysis and control design techniques are available. An overview of these kind of modeling techniques includes:

- mixed logical dynamical (MLD) systems [36, 37],
- piecewise-affine (PWA) systems [160],
- linear complementarity (LC) systems [102, 176],
- extended linear complementarity (ELC) systems,
- max-min-plus scaling (MMPS) systems [70],
- timed automata,
- timed Petri nets.

We would like to emphasize that this list is by no means exhaustive. We are not going to treat all these classes here, but focus on the most well-known classes. The common feature of all the modeling paradigms, and in fact of all hybrid systems, is in the interaction of different dynamics. This indicates that also the model structure should mix two modeling formalisms. Typically, one might think of the interaction of time-driven models (governed by differential or difference equations) on one hand, and event-driven systems (described by, e.g., temporal logic, automata, finite state machines, etc.) or logic rules on the other. In some way these features should be combined in a unifying model structure. One of the nicest ways to look at hybrid systems is via hybrid automata, which can be as a cross production of finite state machines and differential or difference equations (depending on whether we use discrete time or continuous time formalism).

Hybrid Automata

The Hybrid Automaton (see [47, 45, 138]) model is described briefly as follows. The discrete part of the dynamics is modeled by means of a graph whose vertices are called *locations, discrete states or modes*, and whose edges are *transitions*. The continuous state takes values in a vector space χ. For each mode there is a set of trajectories, which represents the continuous dynamics of the system. Interaction between the discrete dynamics and continuous dynamics takes place through *invariants* and *transition relations*. Each mode has an invariant associated to it, which describes the conditions that the continuous state has to satisfy at this mode. Each transition has an associated transition relation, which describes the conditions on the continuous state under which that particular transition may take place and effect that the transition will have on the continuous state.

Various ramifications of the hybrid automaton model have been proposed in the literature. Sometimes the notion of a transition relation is split up into two components, namely a *guard* which specifies the subset of the state space where a certain transition is enabled, and a *reset map* which is a (set-valued) function that specifies how new continuous states are related to previous continuous states for a particular transition. We opt here for the following description of *hybrid automata*.

Definition 3.5. *A hybrid automaton H is a collection $H = (Q, X, f, Init, Inv, E, G, R)$ with*

- $Q = \{q_1, \cdots, q_N\}$ *is a finite set of discrete states;*

- $X = \Re^n$ *is a set of continuous states;*
- $f : Q \times X \to X$ *is a vector field;*
- $Init \subseteq Q \times X$ *is a set of initial states;*
- $Inv : Q \to P(X)$ *describe the invariants;*
- $E \subseteq Q \times Q$ *is a set of edges;*
- $G : E \to P(X)$ *is a guard condition;*
- $R : E \to P(X \times X)$ *is a reset map.*

Note that $P(X)$ is a power set of X, i.e., the collection of all subset of X. Hence, note that the guard condition G gives for each mode a subset of X. The *(hybrid) state* variable of the system H is given by $(q, x) \in Q \times X$. The evolution of this dynamical system behaves as shown in Figure 3.4. The initial hybrid state (q_0, x_0) of trajectories of a hybrid automaton lies in the initial set $Init$. From this hybrid state the continuous state x evolves according to the differential equation

$$\dot{x} = f(q_0, x) \text{ with } x(0) = x_0$$

and the discrete state q remains constant $q(t) = q_0$. The continuous evolution can go on as long as x stays in $Inv(q_0)$. If at some point the continuous state x reaches the guard $G(q_0, q_1)$, we say that the transition is enabled. The discrete state may then change to q_1, and the continuous state jumps from the current value x^- to the new value x^+ with $(x^-, x^+) \in R(q_0, q_1)$. After this transition, the continuous evolution resumes and the whole process is repeated.

This framework indicates the behavior of a hybrid system: continuous phases separated by events at which (maybe multiple) discrete actions (re-initialization of the continuous state x and discrete state q) take place. It is obvious that these systems switch between many operating modes where each mode is governed by its own characteristic dynamical laws. Mode transitions are triggered by variables crossing specific thresholds (state events) and by the elapse of certain time periods (time events) due to the invariants and guards. With a change of mode, discontinuities in the time-continuous variable may occur as given by the reset map. Extension of this formalism are possible, so as to include also external inputs (inputs events) as triggering a mode change, as defined in [44].

A description format (e.g. differential equations or finite state machines) for a class of dynamical systems only specifies a collection of trajectories if one provides a notion of solution. Formally speaking, description formats are a matter of syntax: they specify what is a well-formed expression. The notion of solution provides semantics: to each well-formed expression it associates a collection of functions of time (called the behavior of the system in [182]).

Hybrid Time Trajectories

Definition 3.6. *A hybrid time trajectory τ is a finite or infinite sequence of intervals $\tau = \{I_i\}_{i=0}^N$ such that*

- $I_i = [\tau_i, \tau_i']$ *for* $i < N$, *and, if* $N < \infty$, $I_N = [\tau_N, \tau_N')$
- $\tau_i \leq \tau_i' = \tau_{i+1}$ *for* $i \geq 0$

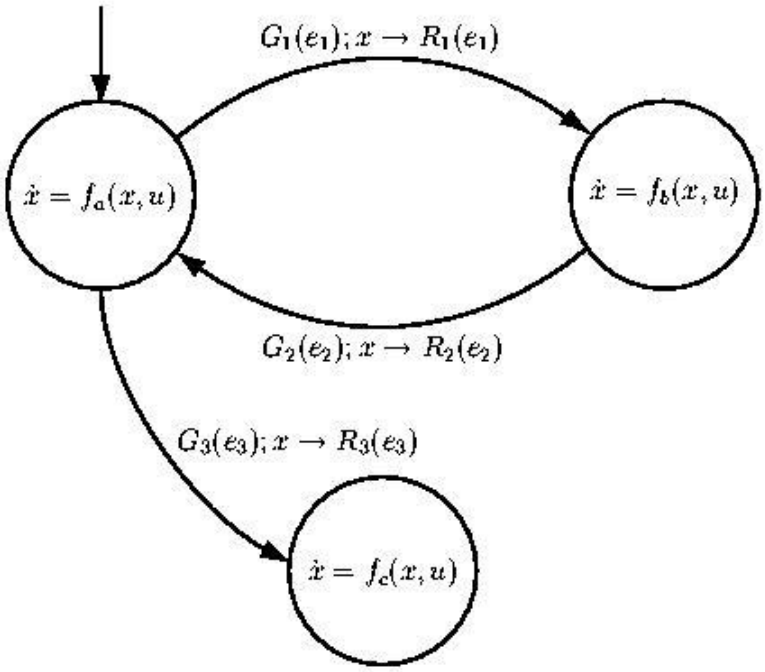


Fig. 3.4. Example of a general hybrid system: a finite number of discrete states are interconnected by edges representing events/guards. Each discrete state contains a continuous dynamical system with corresponding differential equation.

A hybrid time trajectory is a sequence of intervals of the real line, whose end points overlap. The interpretation is that the end points of the intervals are the times at which discrete transitions take place. Note that $\tau_i = \tau_i'$ is allowed, therefore multiple discrete transitions may take place at the same time.

Hybrid time trajectories can extend to infinity if τ is an infinite sequence or if it is a finite sequence ending with an interval of the form $[\tau_N, \infty)$. We denote by T the set of all hybrid time trajectories and use $t \in \tau$ as shorthand notation for that there exists i such that $t \in I_i$ with $I_i \in \tau$.

We use q and x to also denote the time evolution of the discrete and continuous state, respectively. For each $i \in \{1, \ldots, N\}$, they will be defined as functions from the interval I_i to Q and $\Re^n$, respectively. We use $q : \tau \to Q$ and $x : \tau \to \Re^n$ as short hand notations for the maps assigning values from Q and $\Re^n$ to each $t \in \tau$. Note q and x are not functions on the real line, as they assign multiple values to the same $t \in \Re$ at $t = \tau_i' = \tau_{i+1}$ for all $i \geq 0$.

Each $\tau \in T$ is fully ordered by the relation $\prec$ defined by $t_1 \prec t_2$ for $t_1 \in [\tau_i, \tau_i']$ and $t \in [\tau_j, \tau_j']$ if $i < j$, or if $i = j$ and $t_1 < t_2$.

Executions

Next we introduce a concept similar to a solution of a continuous dynamical systems for hybrid automata. This concept is, however, richer the regular solutions, so to distinguish them we introduce the notion of executions of hybrid automata.

Definition 3.7. *An execution χ of a hybrid automaton H is a collection $\chi = (\tau, q, x)$ with $\tau \in T$, $q : \tau \to Q$, and $x : \tau \to \Re_n$, satisfying*

Initial condition: $(q(\tau_0), x(\tau_0)) \in Init$;
Continuous evolution: *for all i with $\tau_i < \tau_i'$, $q(\cdot)$ is constant and $x(\cdot)$ is a solution to the differential equation $\dot{x} = f(q(t), x(t))$ over $[\tau_i, \tau_i']$ and for all $t \in [\tau_i, \tau_i')$, $(q(t), x(t)) \in Inv$;*
Discrete evolution: *for all i, $(q(\tau_{i+1}), x(\tau_{i+1})) \in R(q(\tau_i'), x(\tau_i'))$.*

Figure 3.5 illustrates an execution. We say a hybrid automaton accepts an exe-

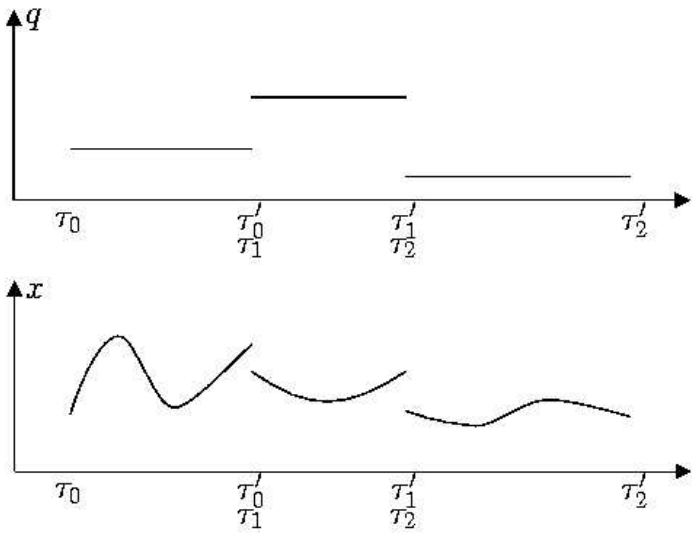


Fig. 3.5. Example of an execution.

cution χ or not. The execution time $\tau_\infty(\chi)$ is defined as $\tau_\infty(\chi) = \sum_{i=0}^{N}(\tau_i' - \tau_i)$, where $N+1$ is the number of intervals in the hybrid time trajectory. An execution is finite if τ is a finite sequence ending with a compact interval, it is called infinite if τ is either an infinite sequence or if $\tau_\infty(\chi) = \infty$, and it is called Zeno if it is infinite but $\tau_\infty(\chi) < \infty$. The execution time of a Zeno execution is also called the Zeno time.

We use $\varepsilon_H(q_0, x_0)$ to denote the set of all executions of H with initial condition $(q_0, x_0) \in Init$, $\varepsilon_H^M(q_0, x_0)$ to denote the set of all maximal executions, and $\varepsilon_H^\infty(q_0, x_0)$ to denote the set of all infinite executions.

Given the previous definitions, we can formalize the following property of hybrid automata.

Definition 3.8. *A hybrid automaton H is deterministic if $\varepsilon_H^M(q_0, x_0)$ contains at most one element for all $(q_0, x_0) \in Init$*

With this in mind we model for example a suture task with a deterministic finite hybrid automaton, using well know subdivision coming from a surgeon "a priori" knowledge of the task, as shown in Figure 4.2. This model will be discussed in Section 4.4.

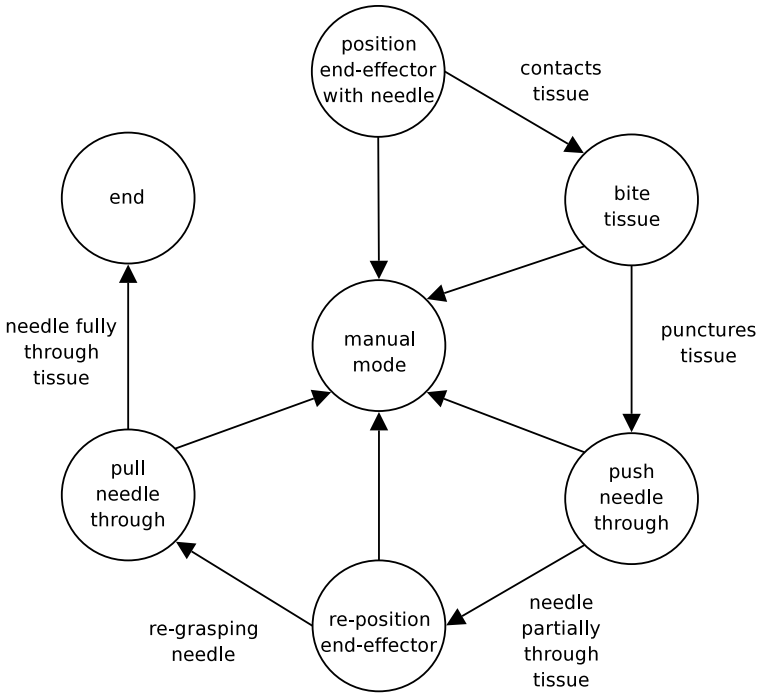


Fig. 3.6. Example of a hybrid automaton for the suture task.

3.3 Lyapunov Stability

Stability is one of the central properties of system theory and engineering. From a controlling point of view, one of the most important properties that a closed-loop system must satisfy is that it has to be stable, since the system is otherwise useless and potentially dangerous. Loosely speaking, a stable operating (equilibrium) point means that a trajectory starting somewhere near this point will stay near it for all future time. A formal definition of stable equilibrium points in hybrid systems together with stability conditions verifying this property are given in this section.

Consider a dynamical system which satisfies

$$\dot{x} = f(x,t) \qquad x(t_0) = x_0 \qquad x \in \Re^n \tag{3.4}$$

We assume that $f(x,t)$ satisfies the standard conditions for the existence and uniqueness of solutions. Such conditions are, for instance, that $f(x,t)$ is Lipschitz continuous with respect to x, uniformly in t, and piecewise continuous in t. A point $x \in \Re^n$ is an equilibrium point of (3.4) if $f(x^*,t) \equiv 0$. By shifting the origin of the system, we may assume that the equilibrium point of interest occurs at $x^* = 0$. Intuitively, we say an equilibrium point is *locally stable* if all solutions which start near x^* (meaning that the initial conditions are in a neighborhood of x^*) remain near x^* for all time. The equilibrium point x^* is said to be *locally asymptotically stable* if x^* is locally stable and, furthermore, all solutions starting near x^* tend towards x^* as $t \to \infty$.

The most general and useful approach for studying stability of nonlinear systems is the theory introduced in the late 19th century by the Russian mathematician A. M. Lyapunov [89].

The *stability in the sense of Lyapunov* is defined as following:

Definition 3.9. *The equilibrium point $x^* = 0$ of (3.4) is stable (in the sense of Lyapunov) at $t = t_0$ if for any $\epsilon > 0$ there exists a $\delta(t_0, \epsilon) > 0$ such that*

$$\mid x(t_0) \mid < \delta \Longrightarrow \mid x(t) \mid < \epsilon, \quad \forall t \geq t_0$$

The theory proposed by Lyapunov showing stability for nonlinear systems now carries his name. Many refinements of Lyapunov's methods have been developed since then.

The *indirect method of Lyapunov* uses the linearization of a system to determine the local stability of the original system. If it is assumed that the nonlinear vector field is continuously differentiable, then the nonlinear system possesses the same stability properties as the system that is linearized around the origin. Unfortunately, the analysis only implies local properties.

On the other hand, more global stability properties can be concluded by applying *Lyapunov's direct method* (also called the second method of Lyapunov). Applying this method for nonlinear systems, stability can be shown by verifying the existence of a scalar auxiliary function satisfying certain conditions [4, 6] without explicitly integrating the differential equation. Such a function is called a Lyapunov function and plays the role of a measure of the system's (abstract) energy. The method is a generalization of the idea that if there is some measure of energy in a system, then we can study the rate of change of the energy of the system to ascertain stability.

The geometrical implications of stability, instability, asymptotic stability and exponential stability in the sense of Lyapunov for a continuous equilibrium point of an autonomous hybrid system are shown in Figure 3.7. The concept of exponential stability is used here with the purpose of knowing an explicit bound on the trajectory state at any time.

The stability analysis of hybrid systems using Lyapunov functions is complicated by the fact that there are switches of the discrete states and hence the vector fields describing the continuous evolution in a hybrid system.

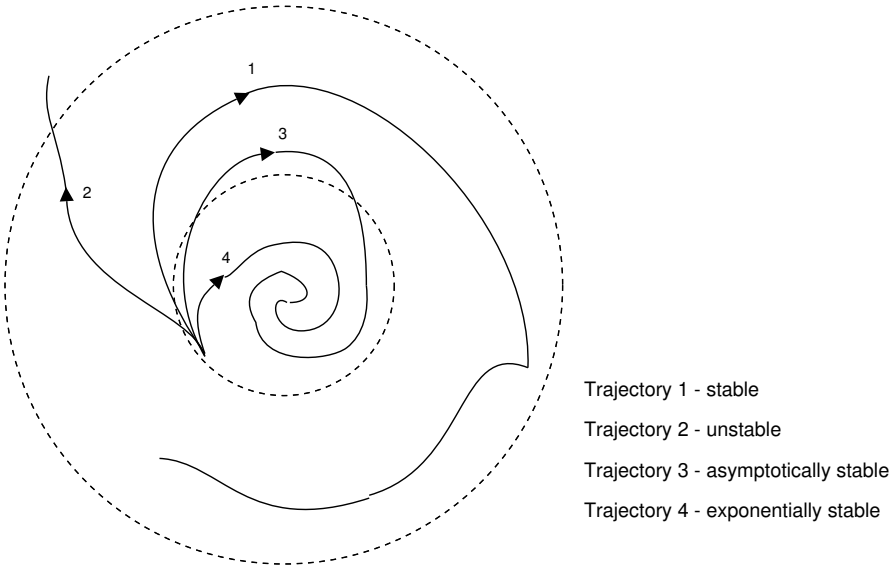


Fig. 3.7. Concepts of stability.

In this Thesis, Lyapunov's theory is applied to autonomous hybrid systems. Lyapunov functions are considered as dynamic constraints in the optimization problem. Due to the results explained in the following sections a stability condition for an hybrid system is computed and added to the formulation of the problem treated in this work. We obtain the optimum control law under the stability assuntions, whose are necessary but not sufficient for a safe system behavior.

3.3.1 Basic Definitions

This section introduces stability for hybrid automata. By generalizing some classical concepts from continuous dynamical systems, we are able to derive a Lyapunov stability theorem for hybrid automata.

Definition 3.10. *The continuous state* $x^* = 0 \in \Re^n$ *is an* **equilibrium point of an hybrid automaton** H *if there exists a non-empty set* $\bar{Q} \subset Q$ *such that for all* $q \in \bar{Q}$

- $(q', z'^3) \in Jump^4(q, 0)$ *implies that* $z' = 0$ *and* $q' \in \bar{Q}$
- $f(q, 0) = 0$

If $Jump(q, 0) \neq \emptyset$, then we require that the vector field should vanish in the origin for all reachable discrete states.

An equilibrium point $x^* = 0$ together with $\bar{Q}$ define an invariant set in the following sense.

[3] We refer to a point $(q, z) \in Q \times \Re^n$ where z is the evaluation of x as the state of H

[4] A set-valued function $Jump : Q \times \Re^n \rightarrow P(Q \times \Re^n)$ is called the jump condition. It specified if a jump from one discrete mode to another is possible and what new value should be assigned to the continuous variable after the jump.

Definition 3.11. *A set $M \subset Init$ is called* **invariant** *if for all $(q_0, x_0) \in M$, $(\tau, q, x) \in \varepsilon_H(q_0, x_0)$, and $t \in \tau$, it holds that $(q(t), x(t)) \in M$.*

We can now define the notion of stability and enunciate the Lyapunov's Stability Theorem for Hybrid Automata.

Definition 3.12. *Assume $x^* = 0$ is an equilibrium point of a hybrid automaton H. It is called* **stable***, if for all $\epsilon > 0$ there exists $\delta > 0$ such that for all executions $\chi = (\tau, x, q) \in \varepsilon_H(q_0, x_0)$ with $\|x_0\| < \delta$ it holds that $\|x(t)\| < \epsilon$ for all $t \in \tau$.*

The equilibrium point $x^* = 0$ is **asymptotically stable** *if it is stable and $\delta < 0$ can be chosen such that for all executions $\chi = (\tau, x, q) \in \varepsilon_H^\infty(q_0, x_0)$ with $\|x_0\| < \delta$ it holds that $lim_{t \to \tau_\infty} \|x(t)\| = 0$.*

Note that in the definition of stability , the discrete state of hybrid automata are not taken into account. This means that an equilibrium point of a hybrid automaton is considered asymptotically stable, although the discrete evolution not converge.

Theorem 3.13. *Consider a hybrid automaton H, such that $(q, z) \in Q \times \Re^n$ and $(q', z') \in Jump(q, z)$ imply that $z' = z$. Assume there exists an open set $\Omega \subset Q \times \Re^n$, such that $(q, 0) \in \Omega$ for some $q \in Q$ and that $x^* = 0$ is an equilibrium point of H. Let $V : \Omega \to \Re$ be a continuously differentiable function in its second argument such that for all $q \in Q$*

- $V(q, 0) = 0$
- $V(q, x) > 0, \forall x, (q, x) \in \Omega \backslash 0$
- $\frac{\partial V}{\partial x} f(q, x) \leq 0, \forall x, (q, x) \in \Omega$

If for all $\chi = (\tau, q, x) \in \varepsilon_H(q_0, x_0)$ with $(q_0, x_0) \in Init \cap \Omega$ and for all $\hat{q} \in Q$, the sequence $\{V(q(\tau_i), x(\tau_i)) : q(\tau_i) = \hat{q}\}$ is non-increasing (or empty), then $x^* = 0$ is stable.

Theorem 3.13 gives sufficient conditions for the stability of the origin of a system. It does not, however, give a prescription for determining the Lyapunov function $V(q, x)$. Since the theorem only gives sufficient conditions, the search for a Lyapunov function establishing stability of an equilibrium point could be arduous. However, it is a remarkable fact that the converse of Theorem 3.13 also exists: if an equilibrium point is stable, then there exists a function $V(q, x)$ satisfying the conditions of the theorem. However, the utility of this and other converse theorems is limited by the lack of a computable technique for generating Lyapunov functions. Theorem 3.13 also stops short of giving explicit rates of convergence of solutions to the equilibrium. It may be modified to do so in the case of exponentially stable equilibria.

3.3.2 Existing stability results

There exist several results in the literature using Lyapunov theory to show stability for hybrid systems and to illustrate that stability of hybrid systems in general depends on the switchings of discrete states and corresponding vector fields. The purpose of this section is to present some of these.

Peletis-DeCarlo An early theorem guaranteeing (asymptotic) stability of linear hybrid systems is proposed in [142] by Peletis and DeCarlo. It is assumed that the time between vector field switchings is bounded above and below by some constant, implying that an infinite number of switching cannot occur in finite time but there will be an infinite number of switching when time goes to infinity. By introducing multiple Lyapunov functions, one Lyapunov function V_i for each linear vector field f_i, which are positive definite and continuously differentiable (and hence continuous), (asymptotic) stability is guaranteed by requiring the energy at the consecutive times (just before switching) to be a decreasing sequence, as shown in Figure 3.8. When there are switchings to another linear vector field at the times t_1, t_2 and so on, the values of the corresponding Lyapunov function just before the switchings are marked by filled circles. The sequence of consecutive values marked by the circles must be (strictly) decreasing according to the (asymptotic) stability result.

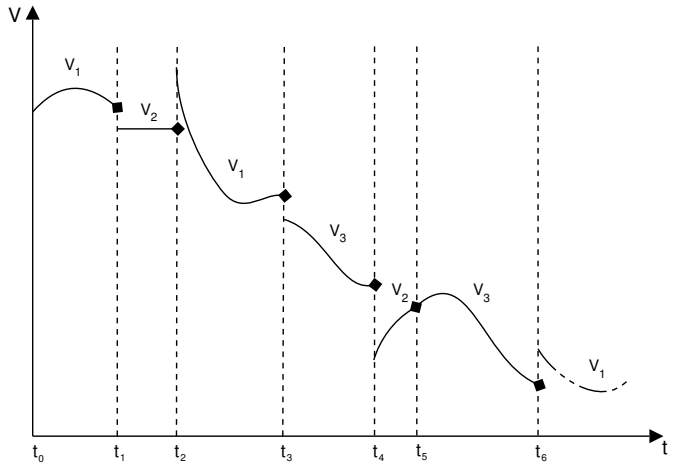


Fig. 3.8. Illustration of the decrease of energy in the (asymptotic) stability result of Peleties-DeCarlo.

Dogruel-Özgüner Another proposed (asymptotic) stability result for hybrid systems is the theorem presented in [78] by Dogruel and Özgüner. It is applied to hybrid systems with nonlinear vector fields, where possible sliding modes have been replaced by an equivalent continuous dynamics in such a way that the system spends equal time in each of the discrete states involved in the switchings. It is assumed that the origin is an equilibrium point for all vector fields. By introducing a common Lyapunov function which is positive definite and continuously differentiable (and hence continuous), (asymptotic) stability is guaranteed by requiring the energy to be decreasing all the time for all vector fields that are possible in a certain region, as shown in Figure 3.9.

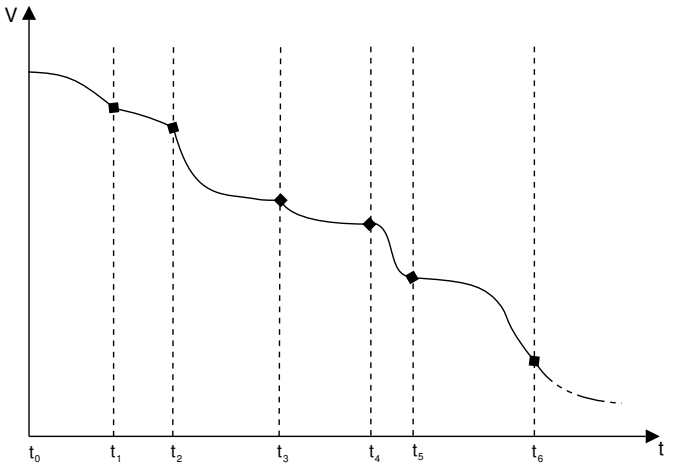


Fig. 3.9. Illustration of the decrease of energy in the stability result of Dogruel and Özgüner

Branicky The stability result proposed by Branicky in [46] is applicable to hybrid systems with nonlinear vector fields, each assumed to be globally Lipschitz[5] and having the origin as an equilibrium point. It is assumed that there are a finite number of vector field switching in finite time. By introducing multiple Lyapunov functions, one Lyapunov function V_i for each vector field f_i which are positive definite and continuously differentiable (and hence continuous), stability is guaranteed by requiring the energy not to increase when no vector field switchings occur and to be a non-increasing sequence at the consecutive times obtained when switching to the different vector fields, as shown in Figure 3.10.

Ye-Michel-Hou The proposed stability result in [186, 187] by Ye, Michel and Hou is very general in the sense that it can be applied to different types of systems, for instance continuous differential equations together with difference equations, or differential equations together with discrete event systems. A continuous Lyapunov function is introduced which guarantees stability if the sequence of values of the Lyapunov function at consecutive switching times is non-increasing and the energy between these times is bounded by a continuous function which is zero in the origin. Asymptotic stability is guaranteed by requiring the sequence of values of the Lyapunov function at consecutive switching times to be decreasing (it is assumed that there are an infinite number of switches), as shown in Figure3.11. Multiple Lyapunov functions are introduced, one Lyapunov function for each vector field, which are positive definite. The requirement that the energy is not allowed to increase when no vector field switchings occur in Branicky's result is weakened by the condition that the energy only has to be bounded by a continuous function which is zero at the origin.

[5] A function F is globally Lipschitz continuous if there exist $L > 0$ (indipendent of r) such that $\|F(z) - F(y)\| \leq L\|z - y\|$ for all $\|z\|, \|y\| < r$ for any $r > 0$

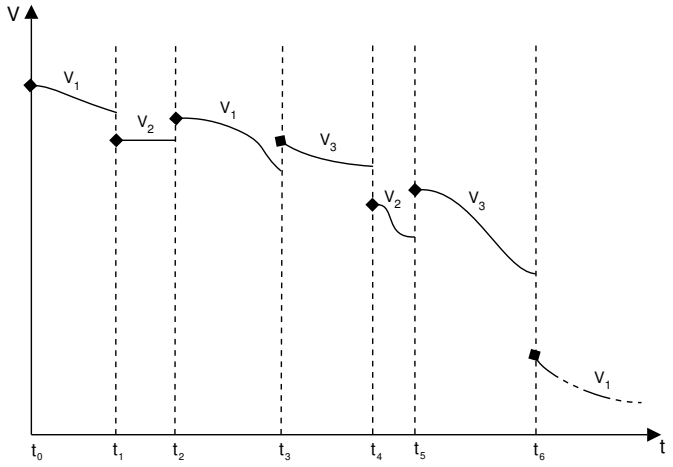


Fig. 3.10. Illustration of the decrease of energy in the stability result of Branicky.

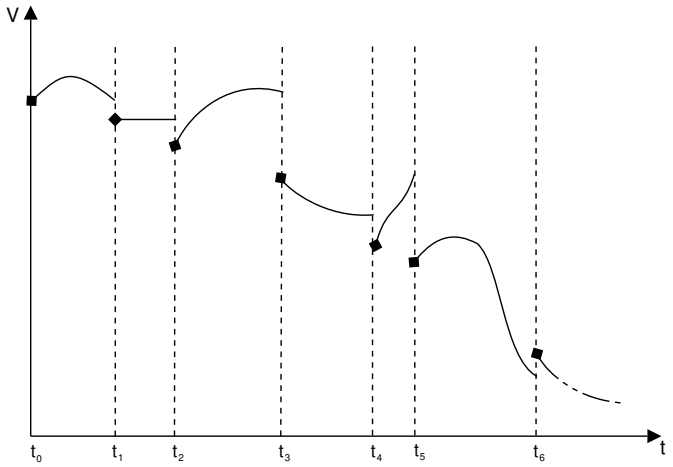


Fig. 3.11. Illustration of the decrease of energy in the stability result of Ye-Michel-Hou.

We can do some consideration on the results presented from the literature. The above stability results are restricted to hybrid systems of a certain structure and model. All results assume that the origin is an equilibrium point for all vector fields. However, the vector fields in hybrid systems do not satisfy this property in general. Furthermore, all results, except the one proposed by Dogruel and Özgüner, assume that there are a finite number of vector field switchings in finite time, restricting the direct application of the stability results to hybrid systems without sliding motions. In the same work, the stability conditions proposed by the authors are only sufficient conditions for stability. It is not hard to construct hybrid examples which are stable but do not have a common Lyapunov function.

In this work we assume as a model a deterministic automaton, where the number of switching is finite. Hence, we assure the stability in the synthesis of

the control law, introducing constraints, i.e. convergence to an equilibrium point in the last subsystems of the trajectory execution, as explained in Chapter 7, ensuring the stability result.

3.3.3 A Literature Review

Research into hybrid systems may be broken down into four broad categories:

- **Modeling:** formulating precise models that capture the rich behavior of hybrid systems.
- **Analysis:** developing tools for the simulation, analysis, and verification of hybrid systems.
- **Synthesis:** synthesizing hybrid controllers, which issue continuous control and make discrete decisions, that achieve certain prescribed safety and performance goal for hybrid systems.
- **Simulation:** conceiving new tools that lead to easier modeling, verification, and control of hybrid systems.

This section summarizes a few key papers, ranging from system theoretic results to verification and simulation.

Modeling and Analysis

A starting point for understanding the properties of Hybrid Systems is the definition of an appropriate model, or range of models, expressive enough for the properties under investigation. The work in [186], [46], [107], [144], [176], [112], [95], [131], [162], and [170] is mostly concerned with modeling and analysis related to system stability.

A model suitable for qualitative analysis of hybrid dynamical systems is presented in Ye [186]. An invariant set (e.g. an equilibrium set) is defined using (Lyapunov-like) stability concepts. Sufficient conditions for uniform stability, uniform boundedness of motion, uniform asymptotic stability, exponential stability and instability are established. Converse necessary conditions for some stability type are also defined. Examples are presented from control theory, including sampled-data feedback control systems, systems with impulse effects, and switched systems.

Multiple Lyapunov functions are used for stability analyses of switched systems and iterated function systems are used for Lagrange stability in Branicky [46], see in the following for more details.

Stability analysis of nonlinear and hybrid systems is addressed in Johansson and Rantzer [107] by searching for piecewise quadratic Lyapunov functions as a convex optimization problem using linear matrix inequalities.

Stability and robustness issues are also addressed using Lyapunov theory in Pettersson and Lennartson [144]. The applicability of the results is discussed, and strong conditions for stability are formulated to compute Lyapunov functions as solutions of an Linear Matrix Inequalities (LMI) problem.

The paper by Van de Schaft and Schumacher [176] is concerned with the study of the well-posedness (existence and uniqueness of solutions) of complementarity

systems, special hybrid systems that are related to the linear complementary problem of mathematical programming. Complementarity modeling is presented first, and well-posedness is defined. The paper, then establishes sufficient conditions for the uniqueness of smooth continuations of complementary systems of arbitrary number of discrete states.

Feedback stabilization of a class on nonlinear systems using hybrid feedback controllers is addressed by Kolmanovsky and McClamroch [112]. The authors study the general structure of systems represented as a cascade of a linear time invariant and a nonlinear system. The controller is constructed to induce a slow and a fast dynamics, and feedback control linearizes the original nonlinear system at the two different time scales.

The domain of attraction of a nonlinear control systems is expanded by Guckenheimer [95] and McClarmroch [131] using switching. The resulting nonlinear control system admits a family of equilibria corresponding to constant control inputs. Switching occurs at discrete times from one control input to another so that the system gradually progresses from one equilibrium point to another towards a final equilibrium.

A Discrete Event System (DES) automaton is used in Stiver [162] to describe a continuous plant, its discrete event controller, and the interface and the complete system is analyzed as a hybrid control system. The system is proved to be a deterministic and controllable hybrid system for which a controller design method is proposed. To extend this approach, invariant based methods for control design are also presented. This method is based on the natural invariants of the continuous part and has appeared in [161].

The control design of a class of hybrid systems with continuous dynamic described by pure integrators is studied by Tittus and Egardt in [170]. This class includes a small set of hybrid systems, but it models the important class of control batch processes. The notion of controllability for this class is proposed and a controllability analysis formulated as a backward reachability problem is derived. The analysis is based on a hybrid automaton model, which includes a hybrid plant and a hybrid controller that interact in a feedback loop.

Synthesis

The design of controllers suitable for hybrid systems, or based on hybrid system theory, is address in papers [149], [67], [56], [139], [126], [123].

The paper by Raisch and O'Young [149] addresses the problem of a continuous plant controlled via symbolic feedback. The hybrid problem is first translated into a purely discrete problem by approximating the continuous plant model by a non deterministic finite state machine. Past measurements and control symbols are used to improve approximation accuracy and adjusted it to required specifications. An optimal controller enforcing the specifications is then computed by applying supervisory control theory.

Supervisory control for a class of continuous-time hybrid systems is studied by Cury [67]. The supervisor is allowed to switch the discrete-valued input signal when threshold events are observed. The objective of the control is to synthesize a nonblocking supervisor that keeps the set of possible control sequences and threshold events for the closed-loop system between given upper and lower bounds in the

sense of set containment. The paper shows that this problem can be converted into a supervisor synthesis problem for a DES. A finite representation may not exist for the exact DES model of the hybrid system, however. The algorithm bypasses this difficulty by constructing finite-state Muller automata that accept outer approximations to the exact controlled threshold-event language is presented. The paper shows that supervisors synthesized for the approximating automata are able to achieve the original specifications when applied to the true hybrid system.

Hierarchical hybrid control systems are defined by Caines and Wei [56] using the notion of dynamic consistency. This notion is extended to hybrid systems by defining a set of dynamically consistent hybrid partition machines, with between-block and in-block controllability properties. The hybrid partition machines are organized in a lattice structure, whose properties are investigated.

Nerode and Kohn [139] propose a Multiple Agent Hybrid Control Architecture to implement a real-time distributed software controller. The architecture is based on principles of declarative control, concurrent programming and dynamical hybrid systems. Each agent computes its control action by solving a relaxed convex optimization problem. An evolution of this approach is proposed in (Kohn et al., 1998) by finding an unbiased estimate of the plant state given dynamic and noisy measurements represented in a multiplicity of forms, without converting all data to a common representation. The paper defines the Multiple Agent Hybrid Estimation Architecture to allow hetereogenous data to flow between individual agents in the network and to improve their individual capability of estimating the current plant state.

Hybrid control of large scale, multiagent systems based on optimal control and game theory is addressed by Lygeros, Godbole and Sastry [126]. The hybrid design is seen as a game between two players: the control, which is chosen by the designer, and noise including the actions of other agents, and unmodeled environmental disturbances. The two players compete over cost functions that encode properties that the closed loop hybrid system needs to satisfy, e.g. safety. The control "wins" the game if it can keep the system safe for any allowable disturbance. The game theoretical methodology is used to compute the continuous controllers as well as the sets of safe states where the control "wins" the game. The sets of safe states are used to construct an interface to the discrete domain that guarantees the safe operation of the combined hybrid system. This approach has been used in air traffic management [172] and in control of automated highway systems [127].

The integration of timed automata and robust control methods for the control of complex dynamical systems is analyzed by Lemmon and Antsaklis [123]. Recent results from computer science and robust control are presented and integration guidelines are given to study stability and boundedness conditions of switched systems. Robust control methods have also been used for hybrid control [16, 17].

Simulation

Papers on simulation and performance verification are the final group of this brief summary of theoretical aspects of Hybrid Systems, and include [12], [16], [104], [73]. Here a most relevant distinction is that between using timed automata and hybrid automata.

Timed automata are proposed by Alur and Dill [12] to model the behavior or real-time systems over time. Their performance is studied from the perspective of formal language theory (closure properties, decision problems, and subclasses). The theory is applied to automatic verification of real-time requirements of finite state machines.

Hybrid automata are introduced as a model and specification language for hybrid systems by Alur et al. [10]. Hybrid automata can be viewed as a generalization of timed automata, in which the behavior of variables is governed in each state by a set of differential equations. This paper shows that the reachability problem is undecidable even for very restricted classes of hybrid automata. Two semidecision procedures are presented for verifying safety properties of piecewise-linear hybrid automata, in which all variables change at constant rates.

Puri and Varaiya [16] present two methods for verification of hybrid systems. The authors model these systems with hybrid automata. Verification is based on abstracting the continuous dynamics in the hybrid system. The methodology is presented using a train-gate-controller example. In [13], Puri, Borkar and Varaiya present a method to compute an arbitrarily close approximation of the reach set of a Lipschitz differential inclusion. Deshpande and Varaiya [74] use nondeterministic finite automata to model the discrete behavior and differential inclusions to model the continuous behavior of hybrid systems. Safety and fairness properties over the system's state trajectories are expressed using the concept of viability, i.e. the system ability to perform an infinite number of discrete transitions. To ensure viability, the system's evolution must be restricted so that the discrete transitions occur within specific subsets of their enabling conditions which are called viability kernel. Results pertaining to continuity properties of the viability kernel are given and conditions under which it can be computed in a finite number of steps are established. Finally, a hybrid controller that yields all viable trajectories is synthesized.

The methodology used by Henzinger et al. [104] to algorithmically analyze nonlinear hybrid systems consists of first translating them to linear hybrid automata, and then using automated model-checking tools. Two translation methods are presented. The first is called clock translation, and it replaces, when possible, constraints on nonlinear variables with constraints on variables with constant unitary derivative. This method is efficient but has limited applicability. The second method, linear phase-portrait approximation, conservatively over-approximates the phase-portrait of a nonlinear hybrid system using piecewise-constant polyhedral differential inclusions.

Numerous simulation tools have been proposed for the simulation, verification and implementation of hybrid systems. `SHIFT` is proposed by Deshpande [73] and consists of a programming language for describing dynamic networks of hybrid automata. The `SHIFT` models offers the proper level of abstraction for describing complex applications such as automated highway systems whose operation cannot be captured easily by conventional systems. Henzinger and Ho [16] proposed `HYTECH` as an automatic tool for analyzing hybrid systems. Daws et al. [13] developed `KRONOS` as a verification platform for complex real-time systems. Taylor and Kebede [167] developed Matlab tools for modeling and simulation of hybrid systems. Bemporad et al. [147] proposed `HYSDEL` which models the system as a

discrete hybrid automaton (DHA) using the high level modeling language `HYSDEL` (HYbrid System DEscription Language). Such a model can be later translated into Mixed Integer Dynamical (MLD) modeling framework. Other computer simulation and verification tools that are (also) used for hybrid systems are BaSiP, Modelica, Chi, 20-sim, UPPAAL, and many others. Simulation models can represent the plant with a high degree of detail, providing a close correspondence between simulated behavior and real plant behavior. This approach is, for any large system, computationally very demanding, and moreover it is difficult to understand from a simulation how the behavior depends on model parameters. This difficulty is even more pronounced in the case of large hybrid systems which consist of many interacting modules. Fast simulation techniques based on variance reduction, and perturbation analysis techniques [5] have been developed in order to partially overcome these limitations.

3.4 Conclusions

In this chapter an overview on hybrid system framework is presented. Because the literature on hybrid systems is broad and covers many areas in engineering and computer science, here we present only some aspects strictly related to this research. We define a hybrid automaton, that is the model that we choose to describe our task. We summarized the state of the art on the analysis of hybrid systems with emphasis on the stability results and some consideration on synthesis and simulation. We use the concepts and definitions summarized here as the base to set up the approach presented in the following chapters.

4

Robotic Surgery

The more a machine is developed the more it disappears.
José Maria Galvan

Surgical theaters are changing rapidly. Also surgical techniques are changing. Minimally Invasive Surgery (MIS) has already been introduced some time ago. After the introduction of MIS, computers and computer controlled surgical tools were more and more integrated into the surgical theater. The next step was to introduce the robot for surgical purposes. In the future surgeons will not even have to touch the patient while operating.

To understand the problems that arise when the new robotic techniques is introduced, an introduction to medical robotics is given in this Section. The terms 'minimally invasive surgery' and 'microsurgery' are introduced. Their advantages, disadvantages and shortcomings are discussed. The most important advantages are fast recovery and less pain for the patient. The most important disadvantages are the less ergonomic situation for the surgeon and the bad visibility of the operation area. A difference is made between 'robotic surgery' and 'computer aided surgery' (CAS). CAS system is powered by the surgeon, while a medical robot is not. Medical robots can be divided in four groups, (i) passive robots, (ii) active robots, (iii) synergistic systems and (iv) master-slave systems. Especially master-slave systems seem to be very promising to use on large scale, because they are able to handle soft tissues and can be used for a large range of operations.

As stated, robots seem to be very promising for use during surgery. However, a lot of aspects have to be improved to increase their performance, not only with respect to mechanical design, but also with respect to control and sensing. Besides these technical aspects also clinical and social aspects play a role. Questions such as: 'Is the result of robotic surgery really better than the result of conventional surgery? Is such a big investment worthwhile? Are there any surgeons who are able and willing to handle such a robot?' arise.

Many studies were carried out in order to introduce new robotic techniques or simply to improve the existing ones. In the literature studies on decomposition of complex task, especially surgical task, have the objective to develop simulators for evaluating surgical skill. Few works address the problem of automatic execution

of surgical task, which requires the identification of an underlying mental model to derive a possible task control sequence. The model aims at analyzing and segmenting the task in simpler sub-tasks. As an example of surgical task at the end of this Chapter we consider suture, which is well defined and parametric in nature. This approach generates a model, hybrid automaton, where the states are basic skills and the transitions between states are governed by knowledge of the task and issues of good performance with respect to a given task.

In this Chapter we will summarize the main aspects of the research area to which this Thesis contributes. In Section 4.1 some definitions of medical aspects are presented, in Section 4.2 attention is given to clinical and social aspects of robot surgery. In Section 4.3, we introduce segmentation as a technique widely used to subdivide and analyze task. In Section 4.4 we describe the surgical task that we have been using in this work, as representative task of the spectrum of surgical procedures.

4.1 Medical Aspects

Minimally Invasive Surgery

As long as operations exist, procedures are developed to decrease the size of the surgical incision. Smaller incisions lead to less trauma and less pain for the patient. This results in a faster recovery of the patient. Patients can return earlier to their homes and freeing place for other patients. Beside the advantages for the patients care, a fast recovery of the patient has also several economical advantages, e.g. lower medical costs and the patient is able to return earlier to work. The search for smaller incisions has led to minimally invasive surgery. The small incisions of MIS are used to introduce special instruments with a long rod transmission mechanism through a cannula into the body of the patient. By using a small camera, the surgeon is able to follow the progress of the operation on a screen. The term minimally invasive surgery covers all surgery with small incisions and endoscopes like thorascopy (chest cavity), arthroscopy (joints), laparoscopy (abdominal cavity), pelviscopy (pelvis) and angioscopy (bloodvessels).

Although MIS fastens the rehabilitation of the patients, it also has some disadvantages. In open surgery the surgeon has more space and freedom to move in a large operation area. With MIS, the operation area is smaller and the freedom to move the operation tool decreases with the use of an endoscope. Laparoscopic tools are long and have only four degrees of freedom. These four degrees of freedom include rotation around and translation along the axis perpendicular to the incision surface, and rotation around the two axes on the incision surface, as shown in Figure 4.1. Because of the long instruments the operative situation is less ergonomic for the surgeon. It takes more energy to use the laparoscopic tools because of the rod transmission between the surgeons hands and the tip of the instruments. Furthermore, the surgeon has to concentrate on the reverse motion direction of his hands and the tip of the instrument. For example, a motion to the right of the surgeon is followed by a motion to the left of the instrument tip. The surgeon has to develop more skills to perform the same operation than without MIS techniques. Another disadvantage of MIS is the interaction between the surgeon and

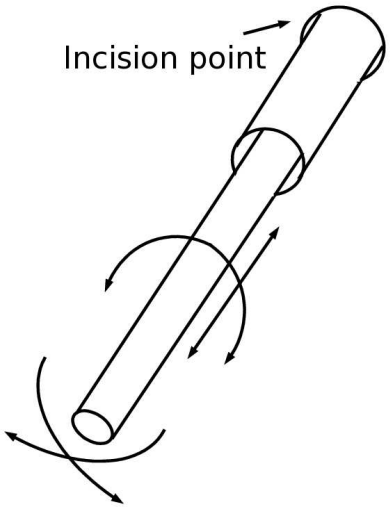


Fig. 4.1. The four degree of freedom of a laparoscopic instrument

the operative area. There is a loss of visibility for surgeons, because they have to follow his own movements on a 2-dimensional screen with a limited angle of view. There is also a loss of tactile sensation and the contact sensation with the long endoscopic tools is totally different than in open surgery. This makes it hard to recognize or to distinguish different tissues or structures, see [71] for more details.

Microsurgery

With microsurgery the technology scales down the surgeons' motions and forces and scales up the field of view, allowing surgeons to skillfully operate on microscopic anatomy with relative ease. The technology enables surgeons to have a more accurate view and a better control of motion and forces, than it is possible with their own eyes and hands. Human perception is not lost, which enables opportunities for new micro-surgical procedures and an improved performance. The disorder, e.g. a tumor, can be approached and removed more accurately without damaging surrounding tissues. The small forces and motions used in microsurgery cannot be achieved with conventional hand-held surgical tools.

Computer aided surgery and medical robotics

Although Computer Aided Surgery (CAS) and Medical or Surgical Robotics are often confused, they differ significantly. The main difference between the terms 'robotic' and 'computer aided' surgery is how they are powered. Robots are powered by an actuated system, while a CAS system is generally powered by the surgeon [69, 169, 174]. A robot can assist the surgeon in two ways. The robot can position the surgical tools or perform operative tasks very accurately. The position can be a predefined location or a predefined complex path. The target area has

to be defined very accurately, because the robot has to be programmed for its surgical task. The target operation area is defined during the pre-operative phase. Both operations with medical robotics or with CAS systems use a pre-operative phase. With CAS systems the pre-operative phase is only used for simulating and planning the operative procedure, programming of instruments is not necessary. Unexpected situations can be avoided and the operation time can be decreased. However, robots are able to provide a greater accuracy than CAS systems. Another reason why robots are superior to CAS systems is the ability to constrain the motions of the system. A robotic system has a predefined safety concept, which does not change during operation. The movements of a CAS system are established by the decisions of surgeons, who can change the operation plan every moment they want. They can neglect warnings or cut into unsafe regions.

Robotic systems

Robots are used to improve the outcome of operations. Different kinds of surgical robots exists. It is possible to divide them in groups related to their operative function or in groups related to the way they function. There exist (i) passive robots, (ii) active robots, (iii) synergistic systems and (iv) master-slave systems [69].

Passive robots are used as tool-holders and do not have an operative task. Their advantage is, that they do not get tired and keep tools accurately in position for a long time. An active robot must be able to carry out more complex motions then passive robots. They have an operative task, which they perform autonomously. That is why most active robots are developed for one specific task within the total operation procedure. The safety demands are high for active robots. Some examples of active robots are laparoscopic cameras or robots used for arthroscopy. Because arthroscopy (joints) deals with bones, the modeling and calibration phase are easier than to deal with soft tissue.

The third group of medical robots contains the synergistic systems. Synergistic systems are controlled by both the surgeon and a computer. The surgeon is able to use the machine within a predefined motion and force region. Oppositely to active robots, synergistic systems do not work autonomously. The operational task is performed by the surgeon, but the synergistic robot system constrains the surgeon. This reduces the risk of failures without loosing the surgeons skills and judgment.

The fourth group consists of the master-slave systems, also called teleoperation systems. Master-slave systems are also non-autonomous. At the master side there are the surgeon, the master robot and visual and haptic displays. The master robot is controlled by the surgeon. At the slave side there are the patient, the slave robot, haptic sensors and cameras. The slave robot is in contact with the patient and follows the instructions of the master robot. In this way the slave robot performs the actual operation. The surgeon controls the master robot on the basis of visual feedback and, if present, haptic feedback from the operation area. Visual feedback can either be two or three dimensional and is established with cameras at the operation area. The cameras are a part of the slave robot and, accordingly can be controlled with the master robot. The slave robot can use instruments for conventional surgery, instruments for MIS, instruments for microsurgery or a combination of them. The big advantages of master- slave systems above active robots

is that they can handle with soft tissue and be used for several operative tasks, because the surgeon is in control of the master robot. The ergonomic situation of the surgeon improves when master-slave systems are used. They can perform the operation sitting at the console and does not have to hold the instruments for a long time at the same position. The master and the slave robot can be decoupled. While the master robot is moved, the slave robot keeps holding its position. Although the distance between master and slave robot normally is a few meters, master-slave systems have the potential to be used in long distance tele-surgery. In that case the data transmission has to be fast because of time delays between the master and the slave side. Master-slave systems have the potential to use motion and force scaling. Motion scaling has the advantage that the surgical tools can be positioned very accurately. Also tremor of the surgeons hands can be filtered out. Force scaling can only be used if haptic feedback is present. Forces can be scaled upward from slave to master to let the surgeon have better notice of properties of the operation area. However, commercial systems at this moment lack the presence of total haptic feedback.

A special group of master-slave systems are the surgical simulators. Surgical simulators are not used for surgical intervention, but as training facility for trainees or as operation planning facility. A surgical simulator has the same structure as the master-slave system. Instead of a real patient and a real slave robot, a virtual patient and a virtual slave robot are introduced by a computer. The computer produces images of the operative area and computes force feedback for the master robot of the contact forces between organs and surgical instruments. The motion images and contact forces are reproduced and displayed as they are in real situations.

The main benefits of using robots for surgical applications are improved precision, stability and dexterity. To take the patients and surgeons advantage of these benefits, improvements have to be made in mechanical design, sensing and control [65, 106].

4.2 Clinical and Social Aspects

The reason why robots integrate slower into the surgical theater than in industrial environments is not only due to technical aspects. The most important social aspect is safety. The safety requirements for medical robots are a lot more stringent than industrial robots. Safety of a system can be achieved by active and passive safety mechanisms in the mechanical design of the system. Passive mechanisms include the materials used and passive joints. An example of an active mechanism is a switch that turns off the instrument when force or motion limits are exceeded. Safety concepts can also be programmed into the software of the robot. Another possibility to ensure safety is to keep the robot under supervision of the surgeon. Synergistic robots are good examples of this possibility. Another way is supervision of the progress of the operation by surgeons on a screen, while they are able to stop the robot in an easy way. The startup times of most medical robots, especially master-slave systems, is quite long, sometimes up to fifteen minutes. Every time the system is stopped for safety reasons, it takes another quarter of an

hour to continue the operation. Robots will only be used for operations when their advantages in outcome above conventional techniques are proven. This can take some time. For example the advantage of hip replacement by use of a robot can only be proved after several years [148]. Another example is the use of master-slave systems. Master-slave systems have the potential to perform the same endoscopic operations than conventional endoscopic operations and even more sophisticated ones. However, with the current master-slave systems the operating and anastomosis time are not decreased. Improvements of the instrument and manipulator design have the ability to perform operations that are not possible with conventional endoscopic techniques and to decrease the operating time [59,117,165]. The cost of a medical robot is high with respect to conventional instruments. When the benefits of a robot are not really clear, this might set up a threshold to invest in medical robots. Furthermore, acceptance by patients, who may not be used to robots, as well as acceptance by surgeons, who have to get familiar with working with robots, are important aspects for the introduction of robots into the surgical theater.

4.3 Analysis and Segmentation of Surgical Task

The analysis of complex task could help improving the performance during task execution, knowledge of the task state could add important features in the robotics systems, i.e. safety and autonomy. Especially teleoperation data are studies with the purpose of evaluating, testing and training surgical skills.

During task execution it is advisable that a supervisory algorithm analyses the teleoperation data as an additional safety measure. This algorithm should have the ability to monitor the system by using feedback signals. Because of the variability of complex teleoperation tasks (a sequence of simpler but different sub-tasks) the knowledge of the task state could help improving performance. For example, a single control algorithm may not be the most appropriate choice for every sub-task. A better choice could be to use a different control strategy for every sub-task, in this way each controller can be made more precise. To identify the various sub-tasks of a teleoperation, it is necessary to segment the teleoperation data in order to recognize the changes in the task state and to mark them as jumps from a sub-task to another. The segmentation gives informations on the system behavior identifying the changes of the model states.

In the literature the problem of data segmentation has been addressed in different ways. In [97] a Hidden Markov Model (HMM) is used to carry out the task segmentation with a number of states equal to the teleoperation sub-tasks. The state transition is computed using the Viterbi algorithm, which returns the more probable state sequence of the HMM, and the parameters of the HMM are computed using the Baum-Welch algorithm or, equivalently, the Expectation Maximization method (EM). However this approach returns the segmentation only in off-line analysis. In [87] a partially Recurrent Neural Network (RNN) with fixed feedback is trained in order to segment the task on-line. This approach produced good results but the use of a neural network hides the use of prior information about the task. In [81,82] auto-regressive models are presented where the segmentation or the jump between a state and the next is obtained using the Sequential

Likelihood Ratio Test (SLRT) technique [28]. This technique is based on work on failure detection [183] and speech segmentation [15]. The teleoperation task and the signals produced during a teleoperation task are very unpredictable. They strongly depend on the operator and they are also quite variable when the same operator executes the same task. For these reason in [60] HMM model, where the emission probability distribution are computed using a Support Vector Machine (SVM) Classifier, is used and a surgical task model is developed as input for possible teleoperation control algorithm, and as basis for automatic task execution.

Simulators for evaluating surgical skills and testing of dexterity in MIS can be roughly divided into three categories: (i) training box including physical objects or latex organ packs (ii) virtual reality simulator including graphical representation of virtual objects or virtual anatomy, and (iii) virtual reality simulator with a force feedback device (haptic display) for simulating forces and torques generated as a result of interaction between the virtual objects or organs and the surgical tools.

One of the most used surgical simulator is the laparoscopic trainer box covered by an opaque membrane through which different trocars are placed at different working angles. The trainee is required to complete several structured laparoscopic tasks which are scored for both precision and speed of performance. Several studies [72, 134] have found the laparoscopic training box a valuable teaching tool for training and evaluation of basic laparoscopic skills.

A virtual reality simulator for laparoscopic surgery models the movements needed to perform MIS and can generate a score for various aspects of psychomotor skill. The use of virtual reality models for teaching complex surgical skills while simulating realistic human/tool and tool/tissue interaction has been a long-term goal of numerous investigators [41, 155] et al..

Although the haptic devices providing force feedback to the surgical tool while interacting with the virtual tissue/organ are commercially available, simulating a realistic force feedback based on biomechanical models of soft tissue is still under active research. The complexity of these biomechanical models is due to the viscoelasticity and non-linear characteristics of soft tissues. Moreover, the F/T data measured in-vivo [108, 152] is crucial for designing and evaluating haptic devices force-feedback telerobotic systems and virtual reality simulators.

The methodology developed in the current studies was based on the Hidden Markov Modeling (HMM) and the goal was to define the learning curve of MIS based on new quantitative knowledge of the F/T applied by surgeons on their instruments, and the types of tool/tissue interactions used during the course of MIS surgery. This goal was pursued through several steps: (i) developing instrumented endoscopic tools which contain embedded sensors capable of measuring and recording F/T information (ii) creating a database of F/T signals acquired during actual operating conditions on experimental animals, (iii) performing a task decomposition in terms of tool/tissue interactions existed in MIS based on video analysis (iv) developing statistical models (HMM) for evaluating an objective laparoscopic skill level. These studies were not intended to test knowledge of the procedure, but whether the surgeon-subjects could technically perform the procedure.

4.4 Suture

4.4.1 Suture characteristics

With respect to other surgical procedures, sutures have the advantage that can be described by a set of well defined rules and characteristics. For example, they should be manufactured to assure:

- Sterility
- Uniform diameter and size
- Pliability for ease of handling and knot security
- Uniform tensile strength by suture type and size
- Freedom from irritants or impurities that would elicit tissue reaction

Also the suture material is described by well defined characteristics, some of which are essential in the development of an automatic procedure, such as:

- Breaking strength, i.e. limit of tensile strength at which suture failure occurs
- Elasticity, i.e. measure of the ability of the material to regain its original form and length after deformation
- Knot-pull tensile strength
- Breaking strength of knotted suture material (10-40% weaker after deformation by knot placement)
- Knot strength, i.e. amount of force necessary to cause a knot to slip (related to the coefficient of static friction and plasticity of a given material)
- Memory, i.e. inherent capability of suture to return to or maintain its original gross shape (related to elasticity, plasticity, and diameter)
- Straight-pull tensile strength, i.e. linear breaking strength of suture material
- Suture pullout value, i.e. the application of force to a loop of suture located where tissue failure occurs, which measures the strength of a particular tissue; variable depending on anatomic site and histologic composition (fat, 0.2 kg; muscle, 1.27 kg; skin, 1.82 kg; fascia, 3.77 kg)
- Tensile strength, i.e. measure of a material or tissue's ability to resist deformation and breakage
- Wound breaking strength, i.e. limit of tensile strength of a healing wound at which separation of the wound edges occurs

The United States Pharmacopeia classification system was established in 1937 for standardization and comparison of suture materials, corresponding to metric measures. The three classes of sutures are collagen, synthetic absorbable, and non-absorbable. Size refers to the diameter of the suture strand and is denoted as zeroes. The more zeroes characterizing a suture size, the smaller the resultant strand diameter (e.g., 4-0 or 0000 is larger than 5-0 or 00000). The smaller the suture, the less tensile strength of the strand.

Sutures were originally manufactured ranging from #1 to #6. (To give an idea about these numbers, a #4 suture would be more or less the diameter of a tennis racket string.) The manufacturing techniques, derived at the beginning from the production of musical strings, did not allow thinner diameters. As the procedures improved, #0 was added to the suture diameters, and later, thinner and thinner threads were manufactured, which were identified as #2/0 to #6/0. The body of

the needle is available also in different makes, like circular, with edge on the outer side, with edge on the inner side, and others [175].

We can estimate the parameter of a mathematical model with a good level of accuracy, using the approach that we formalize in the following section.

4.4.2 Suture Model

We classify the suture pattern into two categories: the first is a discontinuous pattern, in which each loop is fixed with a knot (possibly of different types); and a continuous pattern where there are many loops whose number is related to the length of the cut. Each phase of a suture is described in few steps:

1. reach the tissue
2. penetrate the tissue
3. recover the needle on the other side

It is possible to execute a suture step from different positions and with different orientations and a classification of the different combinations is available in the surgical literature. We consider here the simplest case described in the literature called "classical suture step". Thanks to the repeatability of this surgical gesture, it is possible to represent the single step with an algorithm, and iterate it to form the complete suture pattern.

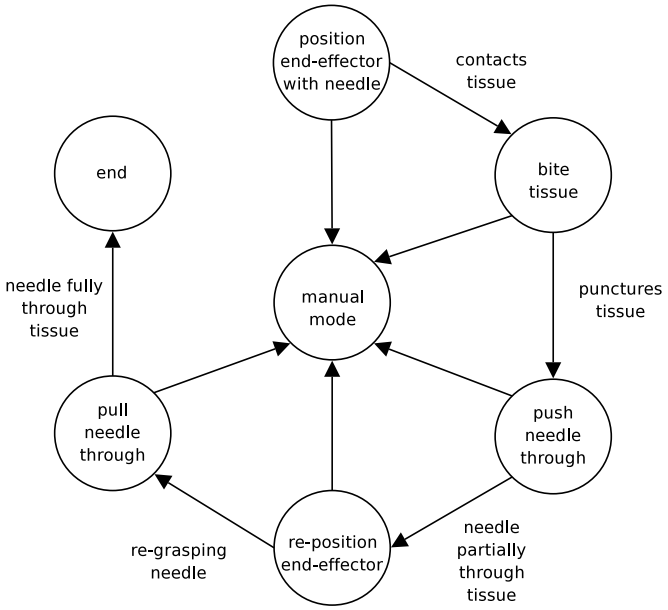


Fig. 4.2. The automaton representing a suture.

With this in mind, we model a suture with a deterministic finite hybrid automaton, using the phase subdivision given by the a priori knowledge of the surgical

task, as shown in Figure 4.2. This figure describes the elementary step of the "simple continuous" suture pattern as shown in simplified form in Figure 4.3.

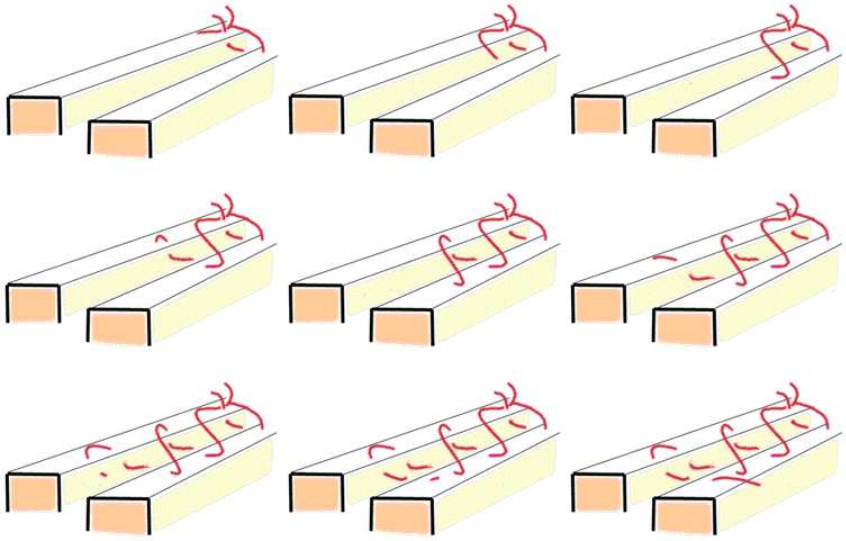


Fig. 4.3. Steps of the simple continuous suture.

The off-line computation of the control function will be determined by estimating the parameters of the cut and of the surgical tool, at each phase of the suture. In particular, we will use a trajectory tracking to set the robotic end effector at the beginning of the suture. In this case, the guard, i.e. the logical flag determining the state change will be the needle position. During the penetration and the extraction of the needle form the tissue, we will use a force measure as the state guard. At each jump between states we will reset the parameters, to start the new execution of the control law in the new state.

In a dynamic system, the problem complexity is due to the uncertain environment and the unmodeled dynamics of the elements involved in the task execution, i.e. the environment and the surgical manipulator. A surgical task is an example of a complex dynamic system where the environment can not be precisely classified. The uncertainty is compensated by sensors, such as a force sensor or a camera, that in real time capture the interaction of the robot with the environment. This behavior can be coded to obtain the correct action of the robotic device, for example jump to the next state of the task automaton. Using a task model derived from a priori knowledge and sensor feedback, a robotic device can perform surgical tasks by teleoperation or, in the case of simple actions, autonomously. Force feedback and autonomous execution can introduce stability and safety problems into the compete surgical system. We encode system safety properties into the requirement that state trajectories remain within certain safe subsets of the state space and we design all the states of our system following the safe property, i.e. constraints in the control computation. We also account for the possibility to reach

the manual state from each state. We guarantee the continuous evolution (invariant set property) and the stability of the complete system by enforcing constraints on the control laws. As an example, we consider the positioning state, the first state of the suture automaton, and the jump to the following state, i.e. contact with the tissue. In the first state we use a control law based on the knowledge of the exact position and orientation that the needle should have to bite the tissue. The constraints that we take into account are: needle shape, grasp position of the manipulator end effector with respect to the needle, tissue position, and tissue properties. The jump guard is the force, i.e. when the needle contacts the tissue, a force value is acquired from the force sensor. The state transition constraints that we use to preserve stability and safety are: bounded control, this is to ensure bounded output, and requires the knowledge of the tissue viscosity; small device movements, so that the kinematic constraints are satisfied during the movement to the contact. In the second state, i.e. tissue perforation, a control law based on the property of the tissue is used. The manual mode is always reachable when an event is triggered by the supervisory software or by the surgeon monitoring the operation. We consider an index weighted on the safety of the task, i.e. how far is the current execution from the nominal task performance. The higher this index the lower the probability that the task will be safely completed.

4.5 Conclusion

In this chapter we have described the state of the art in robotic surgery, with emphasis on teleoperation and segmentation. The classification is used mainly for safety or training or evaluation issues as described in this Chapter. In the last Section we describe a suture in terms of a deterministic automaton. We base our algorithm on the parameters determined from suture analysis, and we consider the safety of the execution as a model property.

5

Optimal Control Design

Insofar as the laws of mathematics are certain, they do not refer to reality; and insofar as they refer to reality, they are not certain.
Albert Einstein

In this Chapter we will present the approach we propose to control the autonomous execution of a complex task. The task that we consider is a surgical suture, using the model developed in Chapter 4. This task, although simple, has the characteristics of a complex task, since it require different control modes and involves various logical conditions. To guarantee the correct task execution we first compute off-line the nominal trajectory of the state variables, using optimization techniques. At execution time, uncertainties are compensated by a regulator. Roughly speaking, the computation of the nominal trajectory is an instance of optimal control problem, i.e. to drive the system to a desirable state while minimizing a cost function that depends on the path followed. It typically involves a *terminal cost* (depending on the terminal state), an *integral cost* accumulated along continuous evolution, and a series of *jump cost* associated with discrete transitions. This is a classical problem for continuous systems, extended more recently to discrete systems [156], and a class of hybrid systems with simple continuous dynamics. The approach has been extended to general hybrid systems using a dynamic programming formulation [44], and a variational formulation, extending Pontryagin Maximum Principle [93]. In this Chapter we introduce the optimum system control in Section 5.1 and its mathematical background in Section 5.1.2. We explore the optimum control from hybrid system point of view in Section 5.2. In Section 5.3 we present the formulation of the problem addressed in this Thesis and in Section 5.4 we give an explanation of the choices carried out.

5.1 Optimum System Control

The fundamental problem of optimization theory may be subdivided into four interrelated parts:

- Definition of a goal.

- Knowledge of our current position with respect to the goal.
- Knowledge of all environmental factors influencing the past, present, and future of the process.
- Determination of the best policy to define a goal, to know current state and environment.

To solve an optimization problem, we must first define a goal (the cost function) for the process we are attempting to optimize. This requires an adequate definition of the problem in physical terms and a translation of this physical description into mathematical terms.To effectively control a process, we must know the current state of the process (state estimation). Also, we must be able to characterize the process by an effective model which will depend upon various environmental factors (system identification). With a knowledge of the cost function, and the system states and parameters, we then determine the best control which minimize (or maximize) the cost function. Thus we may define five problems, which are again interrelated, and which we must solve in order to determine the best, or optimum, system solution:

- The control problem (Figure 5.1).

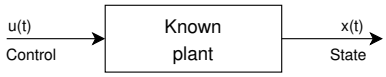


Fig. 5.1. Deterministic optimum control problem.

- The state estimation problem (Figure 5.2).

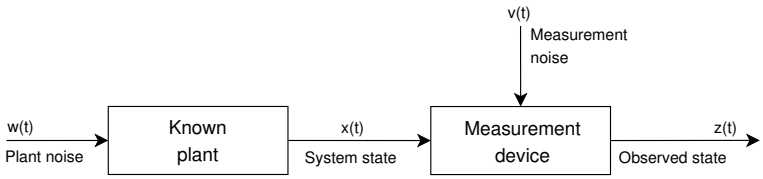


Fig. 5.2. State estimation problem.

- The stochastic control problem (Figure 5.3). We desire to determine a control $u(t)$ such that the output state $x(t)$ is changed in accordance with some desired objective.
- The parameter estimation problem (Figure 5.4). We desire to determine the best estimate of certain plant parameters based upon a knowledge of a deterministic input $u(t)$, the measured output $z(t)$, and possibly some a priori knowledge of the system plant structure.
- The adaptive control problem. We desire to determine a control $\mathbf{u}(t)$ to best accomplish some desired objective in terms of the measurement noise and plant noise as well as the uncertainty in system dynamics.

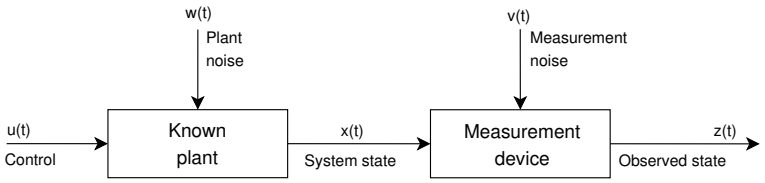


Fig. 5.3. Stochastic control problem.

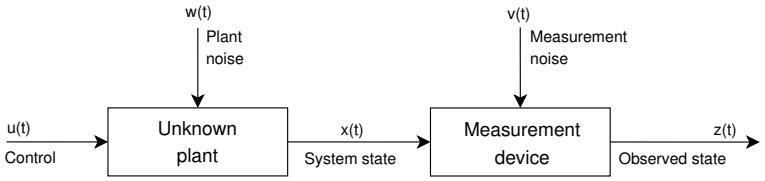


Fig. 5.4. Parameter estimation problem.

In this work we focalize our attention on the control problem, without considering state estimation and stochastic control. A parameter estimation comes from the modeling framework presented in Chapter 4 and from the literature [97, 152, 148, 58, 158, 129, 111], where surgical environment is explored and measured. We use optimization techniques for determining the nominal control $u^*(t)$. Measurement noise and plant noise as well as the uncertainty in the systems dynamics are determined as a function of the measured output, and therefore we have a closed-loop adaptive system. In the following Sections notions of calculus of extrema, variational calculus and maximum principle are briefly reviewed.

5.1.1 Decision Processes

Many problems in modern system theory may be simply stated as extreme value problems. These can be resolved via the calculus of extrema which is the natural solution method whenever one desires to find parameter values which minimize or maximize a quantity dependent upon them. In this Section, the method of Lagrange multipliers is introduced to solve constrained extrema problems for single stage decision processes. Multistage decision processes, which can also be treated by the calculus of extrema, are reserved for a variational treatment which results in a discrete maximum principle. We introduce the subject of variational calculus through a derivation of the Euler-Lagrange equations and associated transversality conditions. The existence and boundness of the integrals defining the cost function is assumed.

We now consider the problem of selecting a continuously differentiable function $x : [t_0, t_f] \rightarrow \Re$ to minimize the cost function

$$J(x) = \int_{t_0}^{t_f} L[x(t), \dot{x}(t), t]dt \tag{5.1}$$

with respect to the set of real-valued, continuously differentiable functions u on the interval $[t_0, t_f]$. Such functions will be referred to as *admissible trajectories*.

Throughout this Section, we assume that L is continuous in $x, \dot{x}$, and t and has continuous partial derivatives with respect to x and $\dot{x}$. It is of interest to note that the cost function presented in (5.1) called the Lagrange form, is equivalent under certain smoothness assumptions to several other useful cost function descriptions, two of which are the Bolza form

$$J(x) = \theta[x(t_f), t_f] + \int_{t_0}^{t_f} \phi[x(t), \dot{x}(t), t]dt$$

and the Mayer form

$$J(x) = \sigma[x(t_f), t_f]$$

Both the Mayer and Lagrange forms are special cases of the Bolza form.

5.1.2 Nominal Trajectory

The computation of the nominal trajectory can be formulated mathematically as an optimization problem in the following way. Find the control $u(t) \in \mathbf{U}$ in $t_0 \leq t \leq t_f$ that minimizes the performance index J:

$$J = \theta[\mathbf{x}(t_f), t_f] + \int_{t_0}^{t_f} \phi[\mathbf{x}(t), \mathbf{u}(t), t]dt \tag{5.2}$$

subject to:

- **kinematic constraints**:
  - initial manifold:
$$\Gamma[\mathbf{x}(t_0), t_0] = \mathbf{0} \tag{5.3}$$
  - terminal manifold:
$$\Omega[\mathbf{x}(t_f), t_f] = \mathbf{0} \tag{5.4}$$
- **dynamic constraints**:
  - systems dynamics:
$$\dot{\mathbf{x}} = \mathbf{f}[\mathbf{x}(t), \mathbf{u}(t), t] \tag{5.5}$$
  - admissible controls:
$$\mathbf{u}(t) \in \mho, \quad t \in [t_0, t_f], \quad \mho \subset \Re^m$$
$$\mathbf{g}[\mathbf{u}, t] \geq \mathbf{0} \tag{5.6}$$
  - state constraints:
$$\mathbf{x}(t) \in \mathbf{X}, \quad t \in [t_0, t_f], \quad \mathbf{X} \subset \Re^n$$
$$\mathbf{h}[\mathbf{x}(t), t] \geq \mathbf{0} \tag{5.7}$$

We consider the problem of determining an admissible control function $\mathbf{u}$ in order to minimize the criterion function of (5.2) subject to these constraints where each component of $\mathbf{h}$ is assumed to be continuously differentiable in state space. We assume also that the terminal manifold equation is a function of terminal time, and the terminal time is unspecified. The initial time and the initial state vector are specified. Therefore the problem becomes one of minimizing the cost function (5.2) for the system described by (5.5) with $\mathbf{x}(t_0) = \mathbf{x}_0$ where, at the unspecified terminal time $t = t_f$, the terminal manifold equation (5.4) and the control and state constraints are satisfied.

We may convert this inequality constraint (5.6), where $\mathbf{g} : \Re^{m+1} \rightarrow \Re^r$, into an equality constraint by writing for each component of $\mathbf{g}$

$$y_i{}^2 = g_i[\mathbf{u}(t), t] \qquad i = 1, 2, ..., r \tag{5.8}$$

We adjoin, via Lagrange multipliers, constraints (5.4), (5.5), which is embedded in the Hamiltonian, and (5.8) to the cost function (5.2) to obtain

$$J = \theta[\mathbf{x}(t_f), t_f] + \nu^T \Omega[\mathbf{x}(t_f), t_f] + \int_{t_0}^{t_f} \{\mathcal{H}[\mathbf{x}(t), \mathbf{u}(t), \lambda(t), t)] - \lambda^T(t)\dot{\mathbf{x}} - \gamma^T(t)\{\mathbf{g}[\mathbf{u}(t), t] - y^2\}\}dt \tag{5.9}$$

where $\mathcal{H}$ is the Hamilton's equations defined as

$$\mathcal{H}[\mathbf{x}(t), \mathbf{u}(t), \lambda(t), t)] = \phi[\mathbf{x}(t), \mathbf{u}(t), t] + \lambda^T \mathbf{f}[\mathbf{x}(t), \mathbf{u}(t), t] \tag{5.10}$$

There are several methods whereby we may convert the inequality constraint represented by the $s-$vector equation (5.7) to an equality constraint. We choose to define a new variable x_{n+1} by

$$\dot{x}_{n+1} = f_{n+1} = [h_1(\mathbf{x}, t)]^2 H(h_1) + [h_2(\mathbf{x}, t)]^2 H(h_2) + \cdot + [h_s(\mathbf{x}, t)]^2 H(h_s) \tag{5.11}$$

where $H[h_s(\mathbf{x}, t)]$ is a modified Heaviside step defined such that

$$H[h_s(\mathbf{x}, t)] = \begin{cases} 0 \text{ if } h_s(\mathbf{x}, t) \geq 0 \\ K_s \text{ if } h_s(\mathbf{x}, t) < 0 \end{cases} \qquad K_s > 0 \quad s = 1, 2, ..., s \tag{5.12}$$

and where the initial condition is

$$x_{n+1} = 0$$

Thus we see that $x_{n+1}(t_f)$ is a direct measure of penetration of the state variable inequality constraint

$$x_{n+1}(t_f) = \int_{t_0}^{t_f} \dot{x}_{n+1}(t)dt = \int_{t_0}^{t_f} \{[h_1(\mathbf{x}, t)]^2 H(h_1) + [h_2(\mathbf{x}, t)]^2 H(h_2) + \cdot + [h_s(\mathbf{x}, t)]^2 H(h_s)\}dt \tag{5.13}$$

We require that the final value of $x_{n+1}(t_f)$ is zero

$$x_{n+1}(t_f) = 0$$

which imposes the restriction that the solution does not violate the inequality constraint. This approach is a modification by McGill [132] of a similar procedure by Kelley [110] which converts the s-inequality constraints to s equality constraints of the form

$$\begin{array}{ccc} \dot{x}_{n+1} = [h_1(\mathbf{x},t)]^2 H(h_1) & x_{n+1}(t_0) = 0 \\ \dot{x}_{n+2} = [h_2(\mathbf{x},t)]^2 H(h_2) & x_{n+2}(t_0) = 0 \\ \vdots \qquad\qquad \vdots & \vdots \\ \dot{x}_{n+s} = [h_s(\mathbf{x},t)]^2 H(h_s) & x_{n+s}(t_0) = 0 \end{array} \tag{5.14}$$

which are then added to the cost function to obtain

$$J_{\text{modified}} = J_{\text{original}} + \sum_{j=1}^{s} x_{n+j}(t_f) \tag{5.15}$$

The multipliers K_s are thus the penalty functions, and J_{modified} is minimized such that the constraint region is entered only slightly, if at all. If we require $x_{n+j}(t_f) = 0$ for $j = 1, 2, ..., s$ the constraint is, of course, not exceeded at all.

The dynamic optimization presented in this work is based on the necessary conditions of optimality derived from Pontryagin's Maximum Principle [122]:

Theorem 5.1. *Given a dynamical system governed by the state equations:*

$$\frac{dx_j}{dt} = f_j(x_1, x_2, .., x_n, u_1, u_2, .., u_m) j = 1, 2, .., n$$

where $\mathbf{u} \in \mathbf{U}$ *and* $\mathbf{U} \subset E^m$*, independent of* $\mathbf{x}$ *and* t*, with the performance index defined as:*

$$J = \int_{t_0}^{t_f} L(\mathbf{x}(t), \mathbf{u}(t))dt$$

from a given initial manifold Γ *at time* t_0 *to a given terminal manifold* Ω *at time* t_f*, with* t_f *not prescribed, and defining*

$$H(\mathbf{\Lambda}, \mathbf{x}, \mathbf{u}) = L(\mathbf{x}(t), \mathbf{u}(t)) + \mathbf{\Lambda} \cdot f(\mathbf{x}, \mathbf{u})$$

where $\mathbf{\Lambda}$ *is a vector of Lagrange multiplier functions, then:*
if $\mathbf{u}^*(t)$, $t_0 \leq t \leq t_f$*, is an optimal control, then there exists a nonzero continuous vector function* $\mathbf{\Lambda}(t) \in \Re^{n+1}$*, satisfying the co-state equations:*

$$\frac{d\Lambda_j}{dt} = -\sum_{i=0}^{n} \frac{\partial f_i(\mathbf{x}, \mathbf{u}^*(t))}{\partial x_j}\bigg|_{\mathbf{x}=\mathbf{x}^*(t)} \Lambda_i \quad j = 0, 1, \ldots, n$$

with x_0 *defined by* $\frac{dx_0}{dt} = L(\mathbf{x}, \mathbf{u})$*, such that:*

$$\left.\begin{array}{ll} min_{\mathbf{u}\in\mathbf{U}} H(\mathbf{\Lambda}(t), \mathbf{x}^*(t), \mathbf{u}) & = H(\mathbf{\Lambda}(t), \mathbf{x}^*(t), \mathbf{u}^*(t)) \\ H(\mathbf{\Lambda}(t), \mathbf{x}^*(t), \mathbf{u}^*(t)) & = 0 \\ \Lambda_0(t) & = constant \geq 0 \end{array}\right\} \quad t \in [t_0, t_f]$$

$\Lambda_1(t), \Lambda_2(t), \ldots, \Lambda_n(t)$ *is normal to the end manifolds* Γ *and* Ω *at* $t = t_0$ *and* $t = t_f$*, respectively.*

We now apply the Euler-Lagrange equations to the cost function (5.15) in order to obtain the necessary conditions for a minimum, due to McShane [133] and Pontryagin [146], and others. It is thus convenient to define a scalar function $\mathbf{\Phi}$, the Lagrangian for no inequality state constraint, as

$$\begin{aligned}&\mathbf{\Phi}[\mathbf{x}(t), \dot{\mathbf{x}}(t), \mathbf{u}(t), \lambda(t), \gamma(t), \mathbf{y}(t), t] = \\ &\mathcal{H}[\mathbf{x}(t), \mathbf{u}(t), \lambda(t), t)] - \lambda^T(t)\dot{\mathbf{x}} - \gamma^T(t)\{\mathbf{g}[\mathbf{u}(t), t] - y^2\}\end{aligned} \tag{5.16}$$

From (5.16) it follows that the Lagrangian for the problem at hand is

$$\tilde{\mathbf{\Phi}} = \mathbf{\Phi} + \lambda_{\mathbf{n+1}}[\mathbf{f_{n+1}} - \dot{\mathbf{x}}_{\mathbf{n+1}}] \tag{5.17}$$

We write the Euler-Lagrange equations as

$$\begin{aligned}\frac{d}{dt}\frac{\partial \mathbf{\Phi}}{\partial \dot{\mathbf{x}}} - \frac{\partial \mathbf{\Phi}}{\partial \mathbf{x}} - \frac{\partial f_{n+1}}{\partial \mathbf{x}}\lambda_{n+1} &= 0 \\ \frac{d}{dt}\frac{\partial \mathbf{\Phi}}{\partial \mathbf{u}} &= 0 \\ \frac{d}{dt}\frac{\partial \mathbf{\Phi}}{\partial \mathbf{y}} &= 0\end{aligned} \tag{5.18}$$

We call each piecewise continuously differentiable solution of the Euler-Lagrange equations (5.18) extremal trajectory of the associated variational problem. The transversality conditions for this problem are

$$\begin{aligned}\Omega[\mathbf{x}(t_f), t_f] &= \mathbf{0} \\ \mathbf{x}(t_0) &= \mathbf{x_0} \\ \frac{\partial \theta}{\partial t_f} + \frac{\partial \mathbf{\Omega}^T}{\partial t_f}\nu + \mathcal{H} &= 0 \quad \text{for} t = t_f \\ \frac{\partial \theta}{\partial \mathbf{x}} + \frac{\partial \mathbf{\Omega}^T}{\partial \mathbf{x}}\nu - \lambda &= 0 \quad \text{for} t = t_f \\ x_{n+1}(t_0) = x_{n+1}(t_f) &= 0\end{aligned} \tag{5.19}$$

The optimization problem given by equations (5.18) and (5.19) is essentially a Multi Point Boundary Value Problem, since its solution must satisfy the terminal manifold Ω, the state and co-state constraints. The optimal solution $u_i^*(t)$ for the i-subtask is reached when the above necessary conditions of optimality, derived from Pontryagin's Maximum Principle, are satisfied. We compute the global optimal solution, i.e. nominal trajectory of the complete task

$$u^*(t) = \sum_{i=1}^{n}\sum_{t=0}^{N} u_i^*(t)$$

The optimality of the u^* is assured by null jump cost θ_i for all i. These conditions (5.19) can be used to check the i optimal solution because we know for each subtask i the terminal configuration manifold Ω.

In particular, they must be satisfied by the trajectory computed by the numerical method. It will be shown in Chapter 6, that a steepest descent algorithm computes a solution satisfying the necessary conditions and, therefore, converges to a target behavior. In Chapter 6 an overview on the computational approach for a Two Point Boundary Value Problem (TPBVP) is presented. The development carried out for the TPVVP can be easily extended to the Multi Point Boundary Value Problem (MPBVP).

5.2 Hybrid Optimal Control

In the context of Hybrid Systems, the computation of the optimal solution involves the selection of a path among the possible states of the hybrid system and the computation of the optimal times to jumps from one state to the next.

Controlling the switching times, when possible, and choosing among several possible states, whenever such choices are available, gives rise to a rich class of optimal control problems. This has motivated efforts to extend classical optimal control principles [44] and to apply dynamic programming techniques [185, 101]. While in principle this is possible, the computational complexity of this computation becomes prohibitive: not only does one have to deal with the well-known curse of dimensionality in such problems, but there are at least two additional sources of complexity to deal with, i.e., the presence of switching events causing transitions from one state to another (which introduces a combinatorial element into the control), and the presence of event-driven dynamics for the switching times (which introduce non differentiability). Therefore, key to the successful development of optimal control methods for hybrid systems is the identification of structural properties that allow the decomposition of such systems into simpler components, and making use of efficient numerical techniques. Along these lines, progress has been reported for classes of hybrid systems whose structure may be exploited. For example, in [35] a Mixed Logical Dynamical (MDL) system framework is proposed, which allows the use of efficient methods developed for piecewise affine systems, and in [96] optimal controllers are presented for the class of autonomous switched linear systems.

The problem statement defined in Section 5.1.2 must be modified for the case of Hybrid Systems. The Hybrid Optimal Control Problem (HOCP) is to find optimal hybrid (i.e., continuous $\mathbf{u}$ and discrete $\mathbf{v}$) control trajectories such that an integral cost index, typically an integral of a function of the hybrid system state and control input, is minimized subject to the system dynamics, initial, terminal, and further equality or inequality constraints.

The Hybrid Optimal Control Problem is defined as the minimization of the real valued, hybrid cost index J:

$$\min_{\mathbf{u},\mathbf{v}} J(\mathbf{u},\mathbf{v}) = \theta[\mathbf{x}(t_0^+),\ldots,\mathbf{x}(t_N^-);\mathbf{q}(t_0^+),\ldots,\mathbf{q}(t_N^-);t_0,\ldots,t_N] + \int_{t_0}^{t_f} \phi[\mathbf{x},\mathbf{u},\mathbf{q},\mathbf{v},t]dt \tag{5.20}$$

subject to:

- **kinematic constraints**:

  - initial manifold:
$$\Gamma[\mathbf{x}(t_0), \mathbf{q}(t_0)] = \mathbf{0} \tag{5.21}$$
  - terminal manifold:
$$\Omega[\mathbf{x}(t_f), \mathbf{q}(t_f)] = \mathbf{0} \tag{5.22}$$

- **dynamic constraints**:
  - systems dynamics:
$$\dot{\mathbf{x}} = \mathbf{f}[\mathbf{x}, \mathbf{u}, \mathbf{q}, \mathbf{v}, t] \quad \text{if} \quad R_j(\mathbf{x}, \mathbf{u}, \mathbf{q}, \mathbf{v}, t) \neq 0 \quad j = 1, \ldots, n_s \tag{5.23}$$

$$\begin{bmatrix} \mathbf{x}(t_i^+) \\ \mathbf{q}(t_i^+) \end{bmatrix} = G_j(\mathbf{x}, \mathbf{u}, \mathbf{q}, \mathbf{v}, t_i^-) \quad \text{if} \quad R_j(\mathbf{x}, \mathbf{u}, \mathbf{q}, \mathbf{v}, t_i^-) = 0 \quad j = 1, \ldots, n_s \tag{5.24}$$
  - admissible controls:
$$\mathbf{u}(t) \in \mathcal{U}, \quad \mathcal{U} \subset \Re^{n_u}, \quad \mathbf{v}(t) \in \mathbf{V}, t \in [t_0, t_f], \quad \mathbf{V} \subset Z^{n_v}$$

$$\mathbf{g}[\mathbf{u}, \mathbf{v}, t] \geq \mathbf{0} \quad t \in [t_0, t_f] \tag{5.25}$$
  - state constraints:
$$\mathbf{x}(t) \in \mathbf{X}, \quad \mathbf{X} \subset \Re^{n_x}, \quad \mathbf{q}(t) \in \mathbf{Q}, t \in [t_0, t_f], \quad \mathbf{Q} \subset Z^{n_q}$$

$$\mathbf{h}[\mathbf{x}, \mathbf{q}, t] \geq \mathbf{0} \quad t \in [t_0, t_f] \tag{5.26}$$

where the initial and final times, t_0, t_f, are free or fixed, R_j are the n_s switching functions and G_j denotes the explicit phase transition conditions occurring at the zeros of the one of the switching functions. θ is a general function of the phase transition time $t_i, i = 1, \ldots, N$ and of the continuous $\mathbf{x}(t_i^-)$, $\mathbf{x}(t_i^+)$ and discrete states $\mathbf{q}(t_i^-)$, $\mathbf{q}(t_i^+)$ just before and just after the $N-1$ interior transition events and at the beginning and final times respectively. The number of phases N may be given or free. The integrand ψ is a real-valued function of the continuous/discrete state and control variables and of time.

The solutions of the HOCPs described in the previous equations are deterministic open-loop trajectories. As in conventional optimal control this class of problems can be generalized to a stochastic setting or to treat issues like optimal closed-loop feedback control. The numerical solution of closed-loop hybrid feedback control problems, however, is at even a much earlier stage and the primarily finite-element based solution strategies that have been presented for their solution [101, 171] cannot readily handle nonlinear systems of more than three dimensions due to the well-known curse of dimensionality. A framework for modeling and (optimally) controlling mixed logical dynamical systems described by linear dynamic equations subject to linear inequalities involving real and integer variables has been proposed by [37]. The on-line optimization problems resulting from a predictive control scheme are solved numerically by application of a mixed-integer quadratic programming branch-and-bound method. However, the approach is not applicable to our class of HOCPs with nonlinear dynamics equations subject to nonlinear constraints. For these reasons we adopt Pontryagin's Maximum Principle to the specific requirements of Hybrid Systems

5.3 Problem Formulation

The control problem is cast in a form compatible with Pontryagin's Principle with state-dependent control constraints. We consider the case where the terminal manifold equation is a function of the terminal time, and the terminal time is unspecified, as shown in Figure 5.5

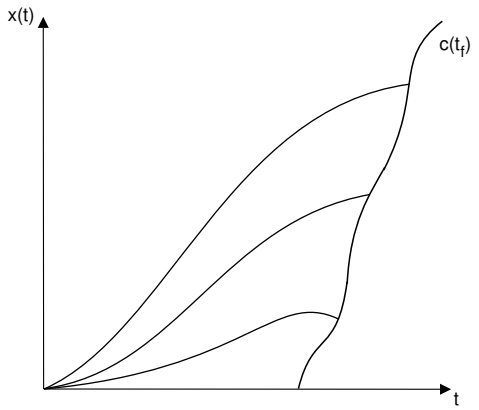


Fig. 5.5. Illustration of variable terminal time problem where $x(t_f) = c(t_f)$

The optimal control is computed minimizing the cost of a performance index function of the continuous state and of the jumps necessary to reach the subsequent state in the optimal configuration respect to the subtask. We iterate this concept for each subtask. By collecting the nominal trajectories computed for each state, we obtain the optimal control law for the complete task.

Remark 5.2. The continuous evolution is assured by the kinematic constraints, i.e. the configuration at time t_f belongs to the intersection of the terminal configuration manifold $\Omega[\mathbf{x}(t_f)]$ of state $i-1$ and the initial configuration manifold $\Gamma[\mathbf{x}(t_0)]$ of state i.

We use the variational approach where the terminal time is not fixed and where the control and state vectors are smooth functions.

Remark 5.3. Among state constraints we include Lyapunov functions which ensure that state trajectories are smooth.

From a practical point of view, we are given a system with known relation between states and control input, and we desire to find the control which changes the state $\mathbf{x}$ so as to accomplish some desirable objective.

Since our problem is to find an optimal controller for hybrid system, modeled with hybrid automaton, we use the cost function (5.9), to calculate the optimal control for a continuous state and the following jump computation to minimize the cost of the jump.

Definition 5.4. *We define the cost of the jump as the distance between the initial position on manifold $\Gamma[\mathbf{x}(t_0)]$ of the state i and the final position of terminal manifold $\Omega[\mathbf{x}(t_f)]$ of the state $i-1$. We instantiate the function $\theta[\mathbf{x}(t_f), t_f]$ as such distance function.*

Remark 5.5. θ is an indicator of successful completion of the task: the bigger θ the lower the probability to complete the task.

We define:

Definition 5.6. *Safety in a task execution is the property of completing the task without uncontrolled behaviors.*

Thus we can state the following:

Remark 5.7. The distance θ is inversely proportional to the safety of the complete system execution.

For our problem the function $\theta[\mathbf{x}(t_f), t_f]$ is the distance between two points on the same manifold because we assume that:

Remark 5.8. The final configuration manifold $\Omega[\mathbf{x}(t_f)]$ of state $i-1$ and the initial configuration manifold $\Gamma[\mathbf{x}(t_0)]$ of state i intersect at $t_f = t_0$. The final point of the trajectory of the state $i-1$ and the initial point of the trajectory of the state i are on this manifold.

To define the concept of the jump cost we introduce the following definitions.

Definition 5.9. *A metric is a nonnegative function $g(x, y)$ describing the "distance" between neighboring points for a given set. A metric satisfies the triangle inequality*

$$g(x, y) + g(y, z) \geq g(x, z)$$

and is symmetric, so

$$g(x, x) = 0$$

and

$$g(x, y) = 0 \quad \textit{implies} \quad x = y$$

Definition 5.10. *The distance between two points is the length of the path connecting them. In the plane, the distance between points (x_1, y_1) and (x_2, y_2) is given by the Pythagorean theorem. In general, the distance between points x and y in a Euclidean space $\Re^n$ is given by*

$$d = |\mathbf{x} - \mathbf{y}| = \sqrt{\sum_{i=1}^{n} |x_i - y_i|^2}$$

Definition 5.11. *Suppose for every point x in a manifold M, an inner product $\langle \cdot, \cdot \rangle$ is defined on a tangent space $T_x M$ of M at x. Then the collection of all these inner products is called the Riemannian metric.*

For curved or more complicated surfaces, the *metric* used to compute the distance between two points is an integration. With distance we mean the shortest distance between two points computed on the surface, i.e. the *geodesic*. The geodesics in a space depend on the Riemannian metric, which affects the notions of distance and acceleration, see [77].

Definition 5.12. *Let $\delta(s)$ be a smooth curve on a manifold $\mathcal{M}$ from a to b with $\delta(0) = a$ and $\delta(1) = b$. Then $\delta\prime(s) \in \mathcal{T}_{\delta(s)}$ where $\mathcal{T}_a$ is the tangent space of $\mathcal{M}$ at a. The curve length of δ with respect to the Riemannian structure is given by*

$$\int_0^1 \mid \delta\prime(s) \mid_{\delta(s)} ds \tag{5.27}$$

and the distance $d(a, b)$ between a and b is the shortest distance between a and b given by

$$d(a,b) = \inf_{\delta: a \to b} \int_0^1 \mid \delta\prime(s) \mid_{\delta(s)} ds \tag{5.28}$$

Thus, we can describe the θ function as a geodesic, where $\mathcal{M}[\mathbf{x}(t_f)] = \mathbf{0}$[1] is the manifold representing a subset of the configuration space, considering (5.8). Following (5.28), we can write

$$\theta[\mathbf{x}(t_f)] = \inf_{\delta: a \to b} \int_0^1 \mid \delta\prime(s) \mid_{\delta(s)} ds \tag{5.29}$$

where

$$\delta = \mid \Gamma_i[\mathbf{x}(t_0)] - \Omega_{i-1}[\mathbf{x}(t_f)] \mid \quad \text{with} \;\; t_f = t_0 \tag{5.30}$$

For example for a surface given parametrically by $x(u, v)$, $y(u, v)$, and $z(u, v)$, the geodesic can be found by minimizing the arc length

$$L = \int ds = \int \sqrt{dx^2 + dy^2 + dz^2}$$

where

$$dx = \frac{\partial x}{\partial u} du + \frac{\partial x}{\partial v} dv$$
$$dx^2 = (\frac{\partial x}{\partial u})^2 (du)^2 + 2 \frac{\partial x}{\partial u} \frac{\partial x}{\partial v} du dv + (\frac{\partial x}{\partial v})^2 (dv)^2$$

5.4 Discussion and Conclusions

In summary, the problem addressed in this work can be subdivided in the following steps:

- Task modeling, described in Chapter 4.
- Formalization as an optimal control problem, presented in Section 5.1.2.

[1] $\mathcal{M}$ is not explicitly dependent on time as Γ and Ω in this case

- Off-line computation of the nominal trajectory satisfying task constraints, described in Chapter 6.
- On-line simulation of the real system and calculation of the distance function θ, carried out in this Section.

In this Chapter we focus on the various approaches to optimal trajectory computation for different kind of dynamical systems. We define the theoretical framework used in this research and motivate the various choices made.

In particular we use a deterministic behavior to describe task evolution (see Chapter 4), i.e. the optimal trajectory accounts for a single switch at each jump condition. Hence, we compute the nominal trajectory as explained in Section 5.1.2, by transforming the discrete part of the hybrid system in a set of constraints for the continuous optimal trajectory computation. This choice of task representation is motivated by the nature of the task we are studying.

With respect to the computation of the optimal solution we do not follow the general HOCP approach making the optimal choice among many switching possibilities. Rather, we use a priori knowledge in the switching selection to choose among jump alternatives to emphasize good performance in each subtask and safety of the complete execution. This is accomplished by optimizing a performance index based on the the knowledge of the task. We compute the nominal trajectory implementing the desired task behavior. In the on-line implementation a weight, a.k.a the distance θ is used as index of good performance. We use Pontryagin Maximum Principle because the constraints of the problem at hand can be mapped directly into this formulation.

With the approach proposed in this Thesis we want to assure safety, good performance and efficiency. In our approach a priory knowledge plays a big role in the definition of the task model and in the computation of the optimal solution, however this is not a limitation to the solution of more general hybrid control problems because of the large number of situations where the optimal execution is well defined and structured. In fact, it is more appropriate to indicate the approach proposed as the solution to an optimal control problem rather for an hybrid optimal control problem because we have used the hybrid systems model mostly as modeling and solution framework. The results of this approach are demonstrated in Chapter 7 with appropriate examples.

6

Computational Issues

You might wish to follow the advice of your authors as notorious computer gunslingers: We always shoot first, and only then relax.
[3]

Theoretical work on controllability properties of nonlinear hybrid dynamical systems is still in its early stages and to date only a few problems with low state and control dimensionability can be throughly understood [177]. Nevertheless, there has been a strong interest in numerical methods for determining controllers for these systems, inspired from the success of such approaches in conventional nonlinear optimal control problems. Nonlinear optimal control plays a key role in modern mechatronics and robotics, in particular in the area of path, trajectory, and action planning. To mention some of the many applications: walking pattern and trajectory planning [98], mobile robot path planning [113], optimal payload (weight) lifting, and acrobatics [8, 130], etc.. Numerical algorithms designed for hybrid optimal control problems (HOCPs) with variable structure and nonlinear differential equations have recently been published [55,101,171]. The key to numerically solving HOCPs seems to be the combination of efficient numerical solvers, such as direct collocation, for optimal control problems together with (heuristic) approaches to reduce the combinatorial complexity of the discrete event aspect in HOCPs [55, 180].

The organization of this Chapter is as follows: in Section 6.1 a broad class of HOCPs is defined and numerical strategies to obtain suboptimal solutions on interior point constraints on grids are summarized, Section 6.1.1, and a branch-and-bound strategy, Section 6.1.2, is proposed. The approach of this work is presented in Section 6.2 and in particular in Section 6.3 the algorithm implemented is discussed, that relies on an efficient numerical tool developed in Matlab [53].

6.1 HOCP Solutions

A set of several different numerical strategies is presented here to compute the solution to the HOCP. Two alternatives HOCP solution strategies will be described: (i)

suboptimal solution with interior event time and state constraints fixed on a grid combined with graph search, and (ii) transformation to a mixed-binary-optimal control problem and its subsequent solution using a branch-and-bound algorithm.

6.1.1 Suboptimal Solution Technique

Suboptimal solutions may be obtained by assigning interior point time instants and states to predefined values on a (fine) grid. Between grid points, a standard optimal control problem with fixed boundary conditions is solved. Finally, the suboptimal solution to the HOCP is obtained by a graph search with each grid point forming the nodes of the graph and the optimal cost weighing the vertices of the graph. This solution strategy is applied to solve cooperative multi-arm transport problem [55]. Disadvantages of this approach are the possibly high number of multi-point boundary value problems to be solved and the inherent suboptimality of the solution obtained. On the other hand, an appealing advantage is that by problem understanding one often has good insight as to how the grid needs to be specified, and a useful solution usually can be obtained easily.

Sparse Direct Collocation

The basis for the suboptimal solution strategies is the highly efficient direct collocation method implemented in the software package Dircol [179] to approximately solve optimal control problems using solutions to (sparse) nonlinear programming.

The numerical method of sparse direct collocation implemented in Dircol can efficiently solve multi-phase optimal control problems with a fixed discrete state trajectory. The state $\mathbf{x}$ is approximated by cubic Hermite polynomials $\tilde{x} = \sum_j \alpha_j \hat{x}$ and the control vector $\mathbf{u}$ by piecewise linear functions $\tilde{u} = \sum_k \alpha_k \hat{x}$ on a discretized grid in each phase. The state differential equations are pointwise fulfilled at the grid points and grid midpoints, resulting in a set of nonlinear program (NLP) equality constraints. The control or state inequality constraints are to be satisfied at the grid points resulting in a set of nonlinear NLP inequality constraints. The transcription of the optimal control problem to an NLP is made by Dircol, the NLP is solved efficiently with the advanced SQP-based (Sequential Quadratic Programming) sparse nonlinear program solver SNOPT [91], and subsequently Dircol processes the solution to provide state and control trajectories, error estimates and output that may be used to verify the optimality of the solution. Important features of the method are:

- As the grid becomes finer, the discretized solution converges to a solution of the Euler-Lagrange differential equations (EL-DEQs) according to the Maximum Principle.
- Reliable estimates of the adjoint variable trajectories λ along the discretization grid may be derived from the Lagrange multipliers of the NLP. They enable a verification of the optimality conditions of the discretized solution without solving explicitly the EL-DEQs.
- Local optimality error estimates can be derived which enable efficient strategies for successively refining a first solution on a coarse grid.

- Computation is fast because ODE simulation and control optimization are performed simultaneously (unlike shooting methods).
- The method is also applicable to systems described by differential-algebraic equations of differential index 1. In this case, the algebraic state variables are discretized analogously to the control variables by piecewise linear functions.

6.1.2 Branch-and-Bound

The solution method for mixed-binary optimal control problems (MBOCP) using a combination of sparse direct collocation and branch-and-bound was first presented in [178] and further investigated in [180,55]. Given certain assumptions (finite and known number of phases, constant state and control variables), the HOCP may be transformed into a MBOCP with a simple transformation of its discrete variables.

The solution of the MBOCP yields the optimal (open loop) trajectories, the optimal phase transition times, the possibly free final time and the optimal binary control vector.

Remark 6.1. The nature of the binary control vector appearing in the MBOCP is twofold. On the one hand it represents the discrete control variable v that controls the order and types of phase transitions, on the other hand it also represents the discrete state q in each phase.

A branch-and-bound strategy in combination with a binary search tree is employed. The subproblems solved provide approximate upper and lower bounds to the MBOCP performance index. If the lower bound at a node is greater than the global upper bound, that branch is discarded. The comparison of subproblem solutions is additionally aided by the use of the optimality error estimate (confidence interval). The MBOCP is thus reduced to a continuous multi-phase optimal control problem, that is a relaxation method (see definition in the following).

6.2 TPBVP Solutions

Unless system equations, performance index, and constraints are quite simple, we must employ numerical methods to solve optimal programming and control problems. Optimal programming and control problems are at least two point boundary value problems (TPBVP) and, in some cases, multi point boundary value problems (MPBVP), e.g. when there are interior point constraints or state variable inequality constraints.

Finding solutions to these nonlinear two-point boundary-value problem is, in many cases, not a trivial extension of finding solutions to initial-value (one-point boundary-value) problems. The crucial distinction between initial value problems and TPBVP is that in the former case we are able to start an acceptable solution at its beginning (initial values) and just march along it by numerical integration to its end (final values). In the present case, instead the boundary conditions at the starting point do not determine a unique solution to start with, and a random choice among the solutions which satisfy these (incomplete) starting boundary conditions is almost certain not to satisfy the boundary conditions at the other

specified points. Iteration is generally required to mold these spatially scattered boundary conditions into a single global solution of the differential equations. For this reason, TPBVP require considerably more effort to solve than do initial value problems.

The nonlinear two-point boundary value problem can be defined as

Definition 6.2. *Find the n state variables, $x(t)$, the n influence functions, $\lambda(t)$ and the m control variables, $u(t)$ to satisfy, simultaneously, the n system differential equations (involving x, u), the n influence (adjoint, Euler-Lagrange) differential equations (involving λ, x, u), the m optimality conditions (involving λ, x, u) and the initial and final boundary conditions (involving λ, x)*

There are two distinct classes of numerical methods for solving TPBVPs. In the *shooting (or direct) method* we choose values for all of the dependent variables at one boundary. These values must be consistent with any boundary conditions for that boundary, but otherwise are arranged to depend on arbitrary free parameters whose values we initially 'randomly' guess. We then integrate the ODEs by initial value methods, arriving to the other boundary (and/or interior points with boundary conditions specified). The shooting method provides a systematic approach to taking a set of 'ranging' shots that allow us to improve our aim systematically. In all shooting methods, trial solutions satisfy the differential equations 'exactly', but the trial solutions come to satisfy the required boundary conditions only after the iterations are finished.

In the *relaxation (or indirect) methods* differential equations are replaced by finite difference equations on a mesh of points that cover the range of the integration. The iteration, called relaxation, consists of adjusting all the values on the mesh so as to bring them into successively closer agreement with the finite difference equations and, simultaneously, with the boundary conditions. Good initial guesses are the secret of efficient relaxation methods.

Relaxation works best when the solution is smooth and not highly oscillatory, such oscillations would require many grid points for accurate representation. The number and position of required points may not be known a priori. Shooting methods are usually preferred in such cases, because their variable step-size integrations adjust naturally to a solution's peculiarities.

6.2.1 The Calculus of Variation and Optimal Control

The extrema-finding techniques, although quite sufficient for many different situations, will not, in general, yield the solution to many problems associated with control systems. Whereas the previously discussed techniques deal with methods for extremizing functions of one or several independent variables, in control system design we are typically concerned with extremizing certain types of functions whose independent variables are actually other functions. This type of function is called a functional. Although, as we might expect, many of the basic approaches for extremizing functionals are similar to those for extremizing functions, the results are sometimes quite different. The body of mathematics developed for extremizing functionals is variational calculus. Variational principles have been applied to physical problems, such as wave propagation. The Hamiltonian formulation of the

variational problem has existed since the early nineteenth century in the works of Hamilton, Jacobi, and others. The most significant contribution in recent times was made by L.S. Pontryagin. The work of Pontryagin [146] extended the variational method to include problems wherein the available control and state vector is bounded, as we have seen in Chapter 5.

We formalize the optimum control problem as MPBVP using variational calculus. This set of differential equation may not be solved in a straightforward manner because of the occurrence of state conditions together with terminal conditions. We use direct methods to change the variational problem into a problem of ordinary maxima and minima. The techniques commonly used for this purpose are the Rayleigh-Ritz and the finite elements methods (FEM) [66, 109, 90]; discrete dynamic programming [79, 32, 33] and gradient method or method of steepest descend [110, 53].

The Ritz and FEM methods [66, 109, 90] are based on finding a sequence of functions which give successively smaller values to the functional to be minimized. In the Ritz method, for example, the trajectory and the control are expanded in terms of a weighted sum of a suitable set of functions, and a minimizing set of coefficients for these functions is found. These methods apparently have not been too popular due to the difficulties in finding a suitable set of basis functions and in determining the number of terms to use in the expansion.

Another direct method is that of *discrete dynamic programming* [79, 31, 32, 33]. Discrete dynamic programming is, in effect, the repeated, sequential, stage-by-stage application of the Hamilton-Jacobi equations or the optimality principle of Bellman. Discrete dynamic programming has an advantage of actually being simplified by the addition of control and state-space constraints. Its biggest disadvantage is the requirement for a truly fantastic amount of computer memory for any but very low order problems. The polynomial approximation method of Bellman [33] and the state increment method of Larson [118] are two methods for the reduction of the memory requirements in the computational solution of optimal control problems by discrete dynamic programming.

A third and significantly different type of direct method is the *gradient method* or *method of steepest descent*. The gradient method consists of searching for an optimum by making each new iteration move in the direction of the gradient which points locally in the direction of the maximum decrease (or increase) of the cost function. The method was developed and applied to optimal control problems by Kelly [110], and Bryson and Denham [52]. Various modifications to include penalty functions and other methods of treating equality and inequality constraints have been presented [53]. We use the approach of Bryson [53]. These gradient methods have the ability to generate successively improved trajectory with very poor starting values. However they tend to converge slowly and often require the selection and adjustment of several convergence parameter. The gradient method is a first-order method since it is based on finding first-order effects of control upon terminal constraints and the cost function. To improve the convergence of the gradient method near the optimal trajectory, second-order term can be added. Due to the similarity of these terms to the terms present in the second variation, these methods are know as *methods based on second variations* [135, 53]. They

do accelerate convergence of the gradient method but they require good initial estimates of the initial control and trajectory for convergence.

The method of quasilinearization for the resolution of boundary-value problems arising in the solution of nonlinear differential equation has been widely developed and applied by Bellman and Kalaba in various works [34]. This method generates a monotone iteration scheme whose iterates converge quadratically to a unique solution of the problem at hand. This overview does not aim at reviewing the entire literature of the two-point boundary value problem. A more complete summary you can find in [154].

To compute the nominal control function, we use a modified gradient method because we do not fix a target function, a.k.a. the control law, and we do not compute with the gradient method only a set of parameters but also a control function $u(t)$ that respects the behavioral constraints imposed by the subtask analysis. In Chapter 7 we present an application of this methods to the simulation of a robot task touching and breaking a compliant surface. In Section 6.3 we describe the algorithm that we use in this work for the generation of the subtask nominal trajectory.

6.3 The Algorithm

We describe in this Section the procedure used to resolve the multi-point boundary-value problem. We transform the variational problem into a minimization problem with constraints. We minimize a functional and the output of the minimization is the control law.

The mathematical background for this kind of problem is well defined but very few implementation results are present in the literature for such kind of minimization, where we do not have an objective function to minimize, neither we are checking for optimal parameters of this function but we compute a functional. We adapt Brayson's algorithm [53] to our problem formulation. The algorithm gives the possibility to choose between a fixed time with interval $0 < t < T$ and open final time, where t is variable. In the second case the optimization algorithm returns an optimal control law, with $0 < t < T$ where T is determined from the problem constrains. We resolve a dynamic optimization problem with open final time, with state and control constraints. The inequality in the state and control constrains are added to the problem as shown in Equations (6.1), (6.2) and (6.3). Consider an optimization problem with only one control variable $u(t)$, where u is bounded between u_{min} and u_{max}. We first introduce a change of control variables from u to u_1 where

$$u = \frac{(u_{max} + u_{min})}{2} + u_1 \frac{(u_{max} - u_{min})}{2} \tag{6.1}$$

Using the new control variable u_1, we have

$$-1 \leq u_1 \leq 1 \tag{6.2}$$

To handle this constraint, we next introduce a *slack control* variable u_2 where

$$u_1 = \cos(u_2) \tag{6.3}$$

Numerical solutions using u_2 as control are then relatively straightforward using the gradient projection algorithm `seqopt`. However, if the exact solution contains abrupt switches from $u_1 = 1$ to $u_1 = -1$ or vice-versa, the solution using u_2 will have rapid but not abrupt changes, so it is an approximate solution. By increasing the number of integration steps, one can come as close as desired to the exact solution. If the inequality constraint involves only the state variables, we have to introduce one or more *slack state variables* as well as a slack control variable.

Necessary Condition for a Stationary Solution

We state the problem as to find the control vector sequence $u(i)$, $i = 0, \ldots, N-1$ and the time step Δ to minimize

$$J = \phi[x(N), \Delta] \tag{6.4}$$

where the number of steps N is specified and

$$N\Delta = t_f \tag{6.5}$$

subject to the constraints

$$x(i+1) = f[x(i), u(i), \Delta], \quad x(0) = x_0 \tag{6.6}$$

$$\psi[x(N), \Delta] = 0 \tag{6.7}$$

The new element depends on the performance index, terminal constraints, and dynamic equations on the time step Δ. Differential changes in the performance index and the terminal constraints are given by

$$d\phi = \phi_\Delta d\Delta + \sum_{i=0}^{N-1} [{H_u}^{\phi}(i) du(i) + {H_\Delta}^{\phi}(i) d\Delta] \tag{6.8}$$

$$d\psi = \psi_\Delta d\Delta + \sum_{i=0}^{N-1} [{H_u}^{\psi}(i) du(i) + {H_\Delta}^{\psi}(i) d\Delta] \tag{6.9}$$

where H_u and H_Δ are computed by backward sequencing the adjoint equation for the performance index

$$({H_u}^{\phi}(i))^T = {f_u}^T \lambda^{\phi}(i+1), \quad \lambda^{\phi}(N) = {\phi_x}^T \tag{6.10}$$

$${H_\Delta}^{\phi}(i) = {f_\Delta}^T \lambda^{\phi}(i+1) \tag{6.11}$$

$$\lambda^{\phi}(i) = {f_x}^T \lambda^{\phi}(i+1) \tag{6.12}$$

and the adjoint equations for the terminal constraints

$$({H_u}^{\psi}(i))^T = {f_u}^T \lambda^{\psi}(i+1), \quad \lambda^{\psi}(N) = {\psi_x}^T \tag{6.13}$$

$${H_\Delta}^{\psi}(i) = {f_\Delta}^T \lambda^{\psi}(i+1) \tag{6.14}$$

$$\lambda^{\psi}(i) = {f_x}^T \lambda^{\psi}(i+1) \tag{6.15}$$

Note that ψ and hence ${H_u}^{\psi}$ and ${H_\Delta}^{\psi}$ are q-vectors, and λ^{ψ} is an $n \times q$ matrix. To simplify the notation, we define

$$\bar{\phi}_\Delta \stackrel{\triangle}{=} \phi_\Delta + \sum_{i=0}^{N-1} {H_\Delta}^{\phi}(i) \tag{6.16}$$

$$\bar{\psi}_\Delta \stackrel{\triangle}{=} \psi_\Delta + \sum_{i=0}^{N-1} {H_\Delta}^{\psi}(i) \tag{6.17}$$

We adjoin the terminal constraint changes to the performance index changes with constant Lagrange multipliers ν

$$\begin{aligned} d\Phi &= d\phi + \nu^T d\psi \\ &= (\bar{\phi}_\Delta + \nu^T \bar{\psi}_\Delta) d\Delta + \sum_{i=0}^{N-1} [{H_u}^{\phi}(i) + \nu^T {H_u}^{\psi}(i)] du(i) \end{aligned} \tag{6.18}$$

For $d\Phi = 0$ for arbitrary $d\Delta$ and $du(i)$, with $d\psi = 0$, it is necessary that

$$\Phi_\Delta \stackrel{\triangle}{=} \bar{\phi}_\Delta + \nu^T \bar{\psi}_\Delta = 0 \tag{6.19}$$

and that

$$H_u(i) \stackrel{\triangle}{=} {H_u}^{\phi}(i) + \nu^T {H_u}^{\psi}(i) = 0 \tag{6.20}$$

for $i = 0, \ldots, N-1$, where ν is chosen to satisfy $\psi = 0$.

Remark 6.3. 6.20 is the same condition also with fixed final time.

Remark 6.4. 6.19 is the transversality condition. It is the additional condition that determines the optimal time step Δ.

Numerical Solution with Gradient Methods

Using a guess for the optimal $u(i)$ and optimal Δ, we may calculate the gradients in (6.19) and (6.20). In general, they will not be zero, so we construct improved estimates by making differential changes in the negative direction of these gradients:

$$d\Delta = -k_\Delta \Phi_\Delta \tag{6.21}$$

$$du(i) = -k_u H_u(i), \quad i = 0, \ldots, N-1 \tag{6.22}$$

Substituting (6.21)-(6.22) into (6.18) gives

$$d\Phi = -k_\Delta (\Phi_\Delta)^2 - k_u \sum_{i=0}^{N-1} [H_u(i) {H_u}^T \tag{6.23}$$

which is non-positive if k_Δ and k_u are chosen positive.

Substituting (6.21)-(6.22) into (6.9) gives q linear equations for the q-vector ν:

$$\nu = -(k_\Delta \bar{\psi}_\Delta \bar{\psi}_\Delta^T + k_u Q)^{-1}(d\psi + k_\Delta \bar{\phi}_\Delta \bar{\psi}_\Delta + k_u g) \tag{6.24}$$

and

$$Q = \sum_{i=0}^{N-1} [{H_u}^{\psi}(i)[{H_u}^{\psi}(i)]^T \tag{6.25}$$

$$g = \sum_{i=0}^{N-1} [{H_u}^{\phi}(i)[{H_u}^{\psi}(i)]^T \tag{6.26}$$

Remark 6.5. $(k_\Delta \bar{\psi}_\Delta \bar{\psi}_\Delta^T + k_u Q)$ is non-singular.

A Matlab code for Problems with Open final Time

A Matlab code for solving open final time problems, called `seqopt`, is an iterative algorithm and it follows these summarized steps:

- Compute an initial Δ, t_f is divided by N, the number of elements of the control vector, that is the function to be minimized.
- Initialize states of the systems.
- Compute the entire state sequence.
- Compute the partial derivative for the dynamics, the objective function and the constraints respect to the states the control and the time.
- Compute the new control in the gradient direction given in the previous step. Also a new t_f is given.
- Iteratively perform the algorithm from the initialization step till a maximum number of steps (fixed) or the difference between the solution of one steps and the following is less than a threshold.

6.4 Conclusions

In this chapter an overview on numerical algorithms to resolve the optimal control problem is given. Some specific methods are proposed in the literature to resolve the Hybrid Optimal Control Problem. The focus of this algorithms is on the discrete evolution, with a search along the graph of the discrete states they compute an optimal time trajectory. Instead, we focus on the continuous part and we minimize dynamic parameters strictly correlated to the optimal task execution criteria. We constrain the continuous trajectory in each subtask in order to minimize the cost and have a null jump cost. The discrete part is considered only at the final step where the partial results coming from the computation of the nominal trajectory are putted together to obtain the optimal trajectory for the hybrid system.

7

Computation Results and Simulations

'Testing can show the presence of bugs, but not their absence.
Edsger W. Dijkstra

The simulation of technical processes by scientific computing has become an important tool for the development of new technologies. In many applications, the required theoretical validation and experimental research can be replaced in part by numerical computations. In this Chapter, we focus on the simulation of the hybrid system optimization described in the previous Chapter and on the presentation and discussion of the computation results.

7.1 Introduction

Several recently developed efficient numerical methods have been presented in Chapter 6. Their impact in engineering is demonstrated by their application to three different classes of problems, namely trajectory optimization, parameter estimation and design optimization.

The application of numerical optimization methods to trajectory optimization for industrial robots has shown that large improvements are possible compared with traditional path planning methods (e.g. [181]). But the use of sophisticated numerical optimization methods requires several necessities as, e.g., the proper description of the dynamic behavior of the robot as an n-body system (in minimal coordinates, e.g., the relative angles between successive joints $q = (q_1, \ldots, q_n)^T$ of the robot)

$$M(q(t)\ddot{q}(t) = u(t) + \chi(\dot{q}(t), q(t)), \quad t \in [0, t_f] \tag{7.1}$$

where $u = (u_1(t), \cdots, u_n(t))^T$ is the torque control, M is the positive definite and symmetric $(n \times n)$-matrix of moments of inertia and χ are the moments resulting from centrifugal, coriolis, gravitational and frictional forces. Thus the first basic problem is the modeling of the dynamic system (7.1) of the robot. The identification of unknown dynamic parameters of the specific robot (as, e.g., moments of inertia or friction parameters) is a second must, see Chapter 8.

Once the dynamic equations have been modeled and determined in a proper way, the optimization of trajectories can be investigated. Different robot tasks require different objectives for optimal trajectories evolution, e.g., in welding optimal tracking of the prescribed path might be required. For example if only the initial and the final position of the robot are prescribed a fast point-to-point movement might be requested.

Remark 7.1. The fastest point-to-point motion is calculated with this objective function: $t_f \rightarrow \min$

Remark 7.2. Time minimization in a robotics task exhibits quite often a surprisingly result, impacts enormous stress on manipulator links and can even often not be realized in practice.

Remark 7.3. Fast minimum energy motions has $\int_0^{t_f} \sum_{i=1}^{n} {u_i}^2(t)dt \rightarrow \min$ as objective function.

Remark 7.4. Energy minimization in a robotics task offers a compromise between time and stress, [181].

The robot trajectories execution also have to satisfy several constraints as, e.,g., constraints on the controls, the angles and the angular velocities (dynamical constraints). Further constraints result from the geometrical design of the robot's workspace and require an efficient modeling of collision avoidance (kinematic constraints).

In this Chapter the concepts presented in this Thesis are explained. In Section 7.2 computation results of the algorithm explained in the Section 6.3 for a 1-DOF manipulator puncturing a membrane are proposed. In Section 7.3 we use these results, i.e. the nominal control law, as a reference in the closed-loop system simulation, where a feedback control law is introduced to compensate the disturbances of the model use for the nominal trajectory computation.

7.2 Nominal Trajectory Computation

In this Section we study a simple robotic switching task, whose hybrid model is shown in Figure 7.1. This diagram represent a simplified surgical action, i.e. the initial steps of a suture as describes in Section 4.4. The first state represents the subtask of positioning the end effector of the manipulator in contact with a compliant surface; in the second state an elastic force is opposing the tool tip until the force applied by robot breaks the surface. At the breaking point there is a transition to the third state in which the end effector of the manipulator moves through the surface and stop as soon as possible. For the sake of simplicity, we consider a 1-DOF manipulator in order to have simple dynamics and simplified constraints. Given the cost function (5.2) and the dynamic and kinematic constraint, we instantiate them for each state in the automaton (Figure 7.1). In the first positioning state we want to minimize the dissipated energy, i.e. $\phi[u(t)] = u^2$ while satisfying the dynamic of our system, i.e. for the 1DOF manipulator:

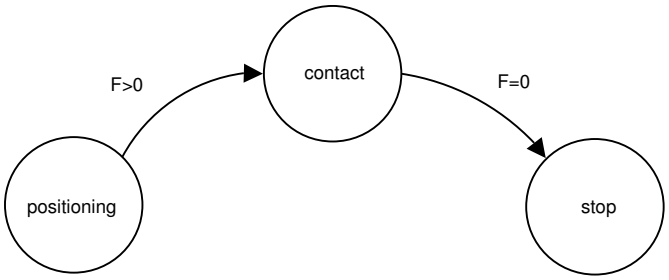


Fig. 7.1. State transition diagram.

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= \frac{\tau_1}{m_{l_1} {l_1}^2 + I_{l_1}} \end{aligned} \tag{7.2}$$

The control bounds are due to the standard range of joint torque, i.e. $-10Nm < u < 10Nm$. The state constraints take into account the stability of our system, see Section 3.3. In the more general case when an infinite number of switching occurs we need constraints on each Lyapunov function of the system (non-increasing). In our system instead we force the jumps, therefore we have a finite number of switch and we assure the convergence with a restriction on the states of the 'stop' subtask. Thus, only in the last subtask computation we take into account state constraints for stability purpose.

The kinematic constraints are related to the terminal manifold, since the initial manifold is given. In the first state we have

$$\begin{aligned} x_2(t_f) &= 0 \\ x_1(t_f) &= c \end{aligned} \tag{7.3}$$

where c is the desired initial position for the contact state, that should be the final position for the first state. Figure 7.2 shows the results of the computation algorithm. In the position plot is shown that a predefined position is reached smoothly, i.e. without spending too much effort. The velocity profile shows the that the manipulator start and arrive at the target with zero velocity. The control, third plot, change direction in correspondence of the point where the velocity starts to decrease.

In the second state we consider $\phi[u(t)] = u^2$ and the constraints due to the interaction with a compliant surface. The computation has to take into account the external force applied by the membrane (F). The dynamics of the system is

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= \frac{\tau_1 - F}{m_{l_1} {l_1}^2 + I_{l_1}} \end{aligned} \tag{7.4}$$

We bound the velocity, (7.5), in order to have a smoother behavior during the contact and a better performance in the following stop state.

$$\begin{aligned} x_2(t) &< v_{max} \\ x_2(t) &> v_{min} \end{aligned} \tag{7.5}$$

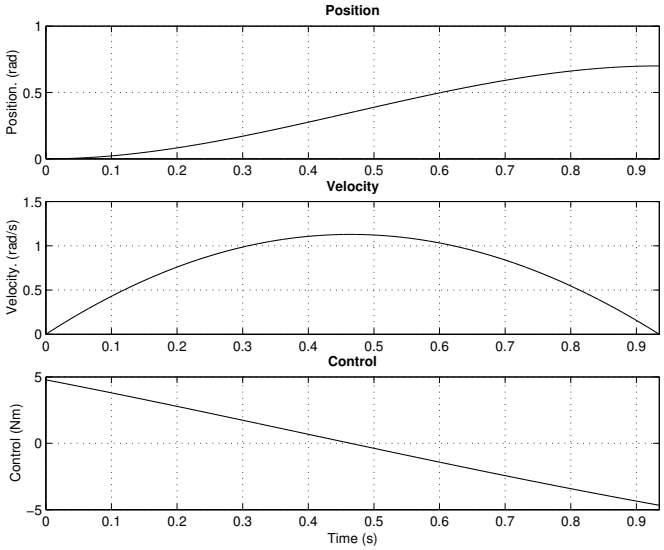


Fig. 7.2. Positioning subtask: position, velocity and control function

The results of the simulation are shown in Figure 7.3. It is possible to recognize the

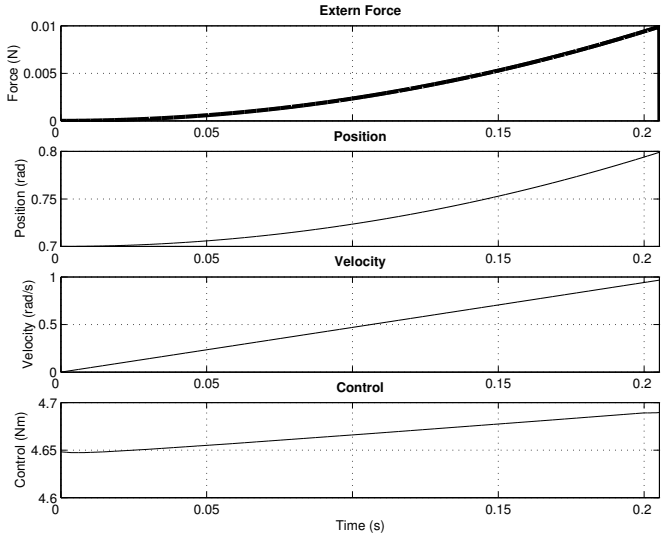


Fig. 7.3. Contact subtask: position, velocity and control function

breaking point (final point in the plot) where the force F is zero. In this subtask position, control and velocity profile increase according to the subtask behavior, i.e. the manipulator does not stop at the end of this subtask.

For the third state we consider a different cost function because we want to minimize the time to stop after the membrane breaking, i.e. we use $\phi = 1$. We have the kinematic constraint $x_2(t_f) = 0$ because the desired behavior for this subtask is to arrive at t_f with zero velocity and we have dynamic constraints due to system dynamics (7.2) and the stability constraint (7.7). We impose the convergence of the Lyapunov function (7.6) to the equilibrium point, i.e. zero.

$$V(\mathbf{x}) = \frac{1}{2}{x_2}^2 + (1 + m\cos x_1) \tag{7.6}$$

$$u \leq m \sin x_1 \tag{7.7}$$

where m is a multiplicative factor. The results are shown in Figure 7.4. The position

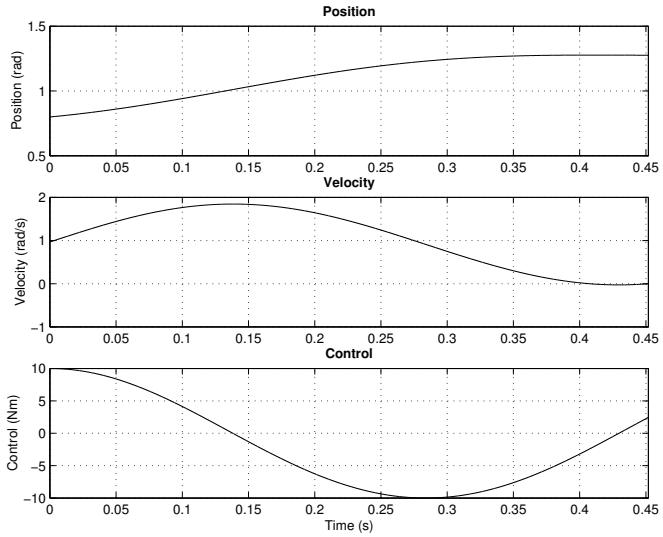


Fig. 7.4. Stop subtask

increases, the velocity instead is still influenced by conditions of the end of the previous state, and it shows an initial increase, after which it goes to zero. The control decreases from the initial time of the subtask, as we expect. In the final part of the state the velocity is controlled so it starts to decrease still maintaining a negative value.

The control function of the complete task is shown in Figure 7.5. In the position plot we notice the contact with the membrane because the position increases slowly. The velocity profile is not linear because of the constraints that are implemented in this state of the system. In fact the velocity describes better than other state variable the behavior of the subtask, because of the clear separation between subtasks. Also the control profile is not linear due to the different subtask. We can recognize the first subtask where the profile decreases, the second where the increase is less evident in this plot scale, and the third subtask where we have an oscillatory behavior.

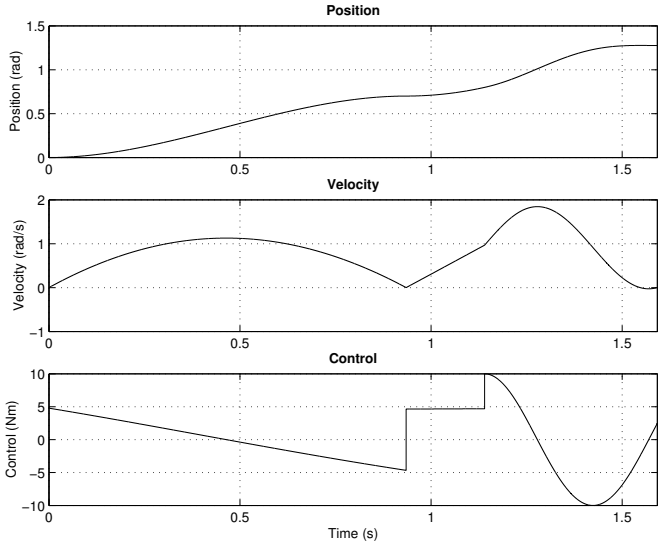


Fig. 7.5. Control function

Remark 7.5. The complete plot shows that the switching among different performance indices is effective in computing the optimal controls for a complex task.

7.3 Simulation Results

After the calculation of the off-line nominal trajectory is performed, on-line feedback must be used to correct the error due to environment and model uncertainties. Figure 7.6 shows how is modified the automaton of the puncturing task to take

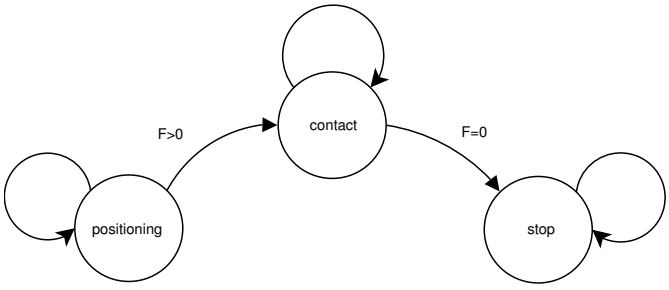


Fig. 7.6. Model of the task with feedback.

into account the on-line execution. The feedback is represented with a loop on each state, because it regulates the state behavior until the distance θ among the manifolds Ω' and the target Ω is zero.

Simulations of the task evolution with noise measurements and a constant noise in the dynamics are carried out. The off-line optimal trajectory is used as reference

signal in a feedback control loop. A regulator corrects the error see Figure 7.7, of the system performing the puncturing task. In Figure 7.7 on the right is shown

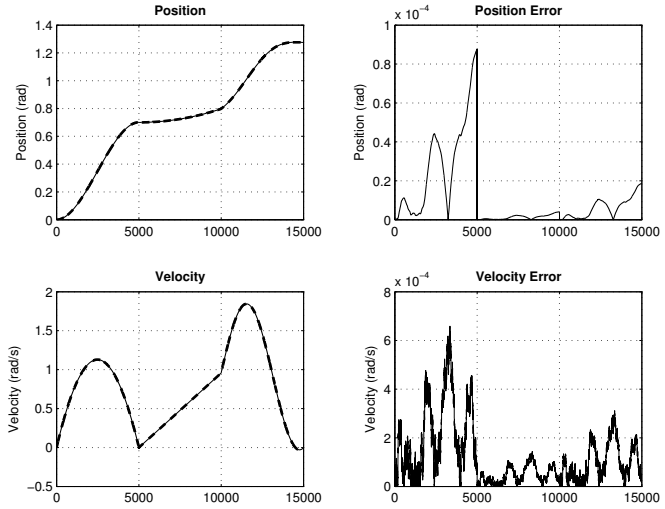


Fig. 7.7. Off-line and on-line position and velocity trajectory of the overall system with the error between the two trajectories (on the left)

the error of the simulated trajectory and the nominal trajectory. We can notice that the error is compensated by the regulator as shown in Figure 7.7 on the left.

7.4 Conclusions

An hybrid system model of a compliant task is analyzed in order to design the control law. The execution of this task requires the definition of a nominal trajectory that can drive the robot to the task successful completion. To this aim, we define an optimal strategy to compute the nominal trajectory off-line in the Section 6.3 and in this Chapter an example of the computational methods to implement this approach in presented together with some results. To obtain an online solution we use a feedback on each subtask to ensure that uncertainties and disturbances during task execution will be properly compensated. We use the distance function θ as cost of the jump, because it gives the correctness of the execution. This simulation results indicate the feasibility of the approach proposed. In the next Chapter are shown some experimental results.

8

Experimental Verification

Hope is not the conviction that something will turn out well, but the certainty that something makes sense, no matter how it turns out.
Vaclav Havel

In this Chapter we present experimental results of an autonomous task execution by a robot in a teleoperation system whose slave end effector comes in contact with a compliant surface. These experiments are carried out in a teleoperation system to identify the salient features of the task and as a test based for the experiments on autonomous execution of the same task, carried out using the slave manipulator alone.

The Chapter is divided into two parts. In the first part a task is carried out in teleoperation to extract and verify the model of the manipulator and tune its control algorithms. In the second part of the Chapter, we describe the experiments of autonomous task execution implementing the simulations of the previous Chapter (to be done).

A salient feature of the teleoperation experiments is the comparison of different control schemes: a single regulator PID, impedance control and hybrid system approach. The performance of the various schemes is compared on a contact and puncturing task of an elastic membrane, shown in Figure 8.1, to determine the best controllers for the autonomous.

The performance measures that we use to evaluate the controllers include approach to the membrane, puncturing, and position overshoot.

The autonomous task executions are carried out with the same setup, but in this case the commands are not generated by an operator but by the optimization procedure described in the previous Chapters. A comparison and discussion of the task execution in teleoperation and autonomously validates the quality of the nominal trajectory computed by the optimization procedure.

8.1 The Experimental Setup

Since we are mostly concerned with the part of the teleoperation system interacting with the environment, we will only describe the robot and the environment. A

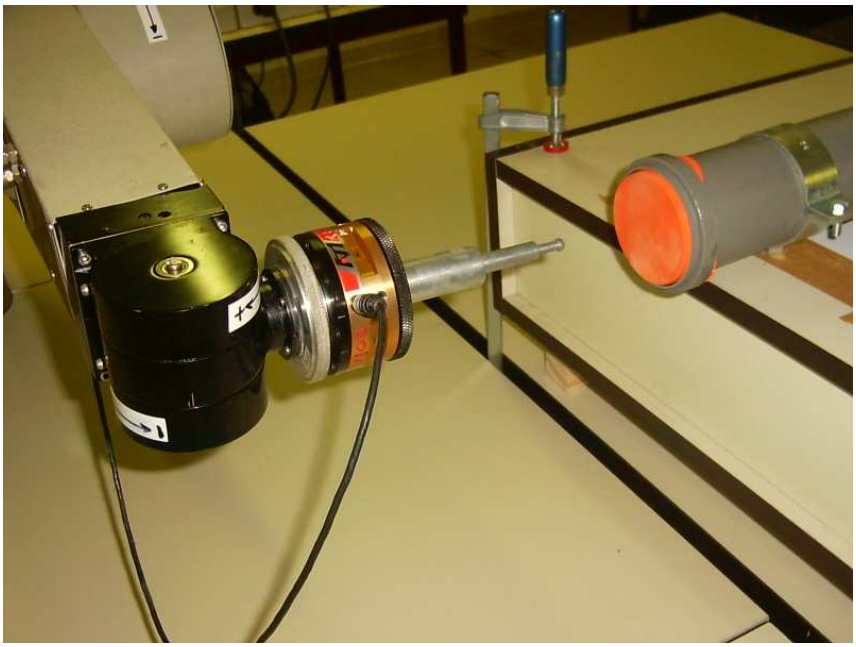


Fig. 8.1. The experimental setup. PUMA robot with a force/torque sensor is pushing against an elastic membrane.

description of the complete teleoperation system can be found in [14, 40, 43, 65, 108, 120, 188, 153, 174]

The Slave Robot

To carry out the experiments we use as slave robot a PUMA 560 manipulator shown in Figure 8.1. As a dynamical model for the optimization we use a model obtained with identification techniques, instead of considering the complete manipulator dynamics. The identification is carried out using the MatLab System Identification Toolbox [1] with the robot in a fixed position and moving only the first joint in a configuration similar to the simulations of the Chapter 7. The control input is the torque τ applied by the motor, and the measured output is the angular position on the manipulator, Φ.

Efficient methods for the solution of parameter identification problems are based on multiple shooting in combination with generalized Gauss-Newton- or adapted SQP-methods [42]. These identification algorithms only require some functions of the states and no derivatives to be measured in order to identify the unknown parameters.

The identification process computes matrices A, B, C, D, K of the classical state-space model in the form:

$$\begin{aligned} \dot{\mathbf{x}} &= A\mathbf{x} + B\mathbf{u} + K\mathbf{e}, \\ \mathbf{y} &= C\mathbf{x} + D\mathbf{u} + \mathbf{e}; \end{aligned} \tag{8.1}$$

where u is the system input vector, y the system output vector and e is white noise. Matrices A, B, C, D, K are computed so that the training input samples u match the output training samples y.

The moving joint is modeled as two masses connected by a spring-damper. The states of the system are position and velocity of the motor, and position and velocity of the robot arm.

With these state variables the manipulator model is:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{k}{M_1} & -\frac{D_1+d}{M_1} & \frac{k}{M_1} & \frac{d}{M_1} \\ 0 & 0 & 0 & 1 \\ \frac{k}{M_2} & \frac{d}{M_2} & -\frac{k}{M_2} & -\frac{D_2+d}{M_2} \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ \frac{1}{M_1} \\ 0 \\ 0 \end{bmatrix} \quad C^T = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{8.2}$$

where k, d are spring and damper coefficients, respectively; b_i is friction of mass i and M_i is the inertia of mass i, and e is a noise term, considered equal to zero.

The identified model produced the results plotted in figure 8.2. To validate this result we tested the model using several input data sets that produced an RMS error of about 1.29 degrees with respect to the true system output.

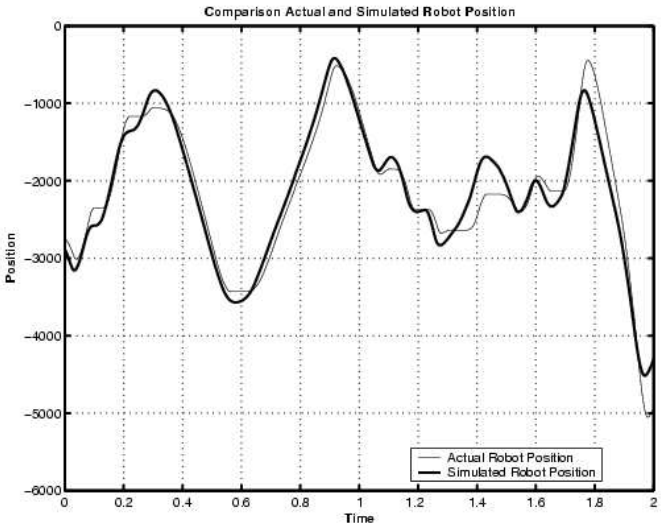


Fig. 8.2. Identification results.

Environment

In general, every environment posses a certain amount of flexibility. Whether an environment is flexible or rigid is completely dependent on its mechanical impedance. However, if an environment posses comparable or lower mechanical impedance than that of the robot, it is considered as a flexible environment. Soft springs and soft tissues are flexible environments.

The type of environment considered in the various tests described in this Chapter consists of a planar compliant surface of medium stiffness, with an estimated contact stiffness in the order of 32 kg/m for translational displacements. This choice is motivated by the desire of simulating the mechanical properties of a soft biological tissue to simulate a surgical suture. The interaction with the environment encompasses an unplanned transition from non-contact to contact at non negligible end-effector speed and must take into account that the sensor in highly stiffness.

8.2 Teleoperated Task Experiments

The experimental setup available in the laboratory includes the industrial robot Unimation Puma 560 with a control unit equipped with a force/torque sensor ATI FT 30/10 and a six degree of freedom master device. An open control architecture has been implemented which allows testing of advanced model-based control algorithms on a conventional industrial robot. For the experiments we use the elastic membrane attached to a rigid structure shown in Figure 8.1.

Figure 8.3 shows experiment results of teleoperating a contact with the membrane. The position difference between master and slave during the contact with the membrane is proportional to the force feedback to the operator, who adjusts the control input.

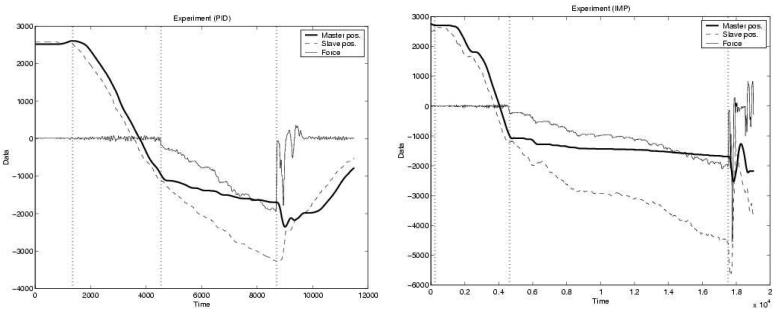
The performance used are:

- **Contact time**: time needed to reach the membrane.
- **Puncturing time**: time needed to puncture the membrane.
- **Position overshoot**: overshoot after puncturing the membrane.

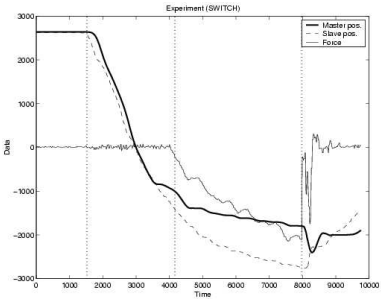
In Figure 8.3 we see the interaction of the robot with the environment, the difference in position between master and slave and the force applied by the slave to the membrane. We note that in certain points the operator adjusts his/her command input to account for his/her view of the task, thus changing the nominal control of the task.

8.3 Autonomous Task Experiments

We use the experimental setup described in the previous Section to carry out experiments using as input the nominal trajectory computed off-line with the steepest descent algorithm described in Section 6.3. The input is corrected by a feedback control loop that take into account the error due to measurements noise, environment and model uncertainties introduced with the real execution. We constrain



(a) (b)

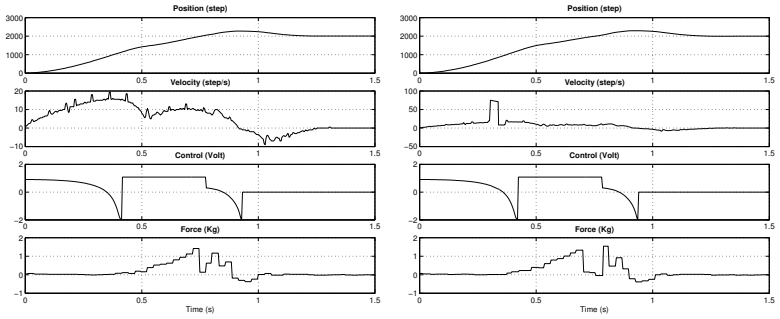


(c)

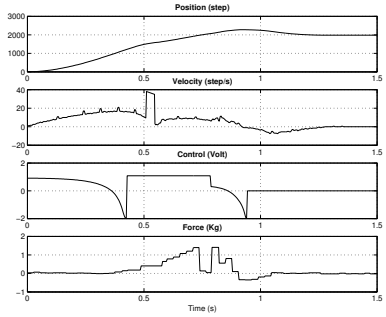
Fig. 8.3. Experiment data for teleoperated membrane puncturing for three types of robot controller. Each figure shows Master, Slave position and Force measured. Vertical lines divide the task in: idle region (when operator does not move the joystick), move region, contact region (during membrane contact), puncturing region.

the system following the performance results of the teleoperated execution, taking into account position, velocity and force constraints in the nominal trajectory computation. Thus the results (shown in Figure 8.4) verify the feasibility of this approach to the autonomous execution of a puncturing task.

A qualitative analysis on this results compared to the teleoperated results (Figure 8.5) can encourage the prosecution of the work. We obtain better performance respect to the teleoperated task in contact time and puncturing time because the manipulator goes without uncertanty through the membrane, and especially in position overshoot we have an improvement due the fast reaction of the algorithm respect to the human operator.

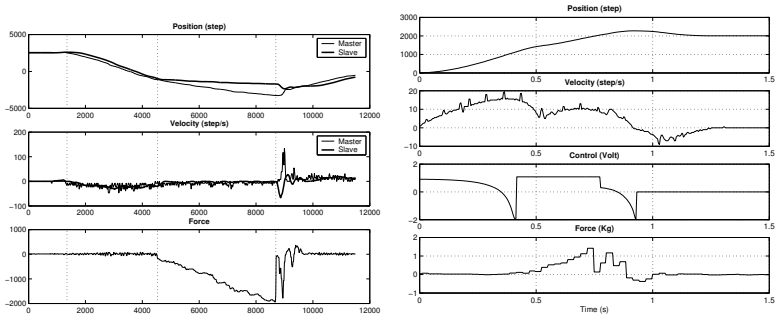


(a) (b)



(c)

Fig. 8.4. Experiment data for autonomous membrane puncturing.



(a) (b)

Fig. 8.5. (a) Experiment data for teleoperated membrane puncturing (b)Autonomous task execution.

8.4 Conclusions

In this Chapter we have presented experiments task of puncturing a membrane as a representative surgical task. The experiments were first carried out by humans to tune the system and to generate a performance baseline against which evaluate the autonomous execution. To this goal we defined a set of performance measures and identified a range of 'good' values of human performance.

Autonomous execution was carried out using the nominal controls computed off-line following the algorithm presented in Chapter 6. According to the performance measure selected, the autonomous controller and the teleoperated task are similar in the executions and the autonomous execution has better performance.

9

Summary and Recomandation for Future Research

The most exciting phrase to hear in science, the one that heralds new discoveries, is not 'Eureka!' (I found it!) but 'That's funny....
Isaac Asimov

In this Thesis we have developed a novel approach for the modellization, analysis and execution of complex tasks, based on the formalism of *Hybrid System Theory*. The approach has been used to model, simulate and experimentally verify a simple surgical task, a suture, which includes, in a small context, most of the difficulties of autonomous task execution. To our knowledge, the enhancement of teleoperation using hybrid automaton has not been done so far, and the results of the experiments show the significance of the enhancements supported by this approach, since it provides a unified framework for modeling, formal verification, and testing of the execution of sequences of elementary tasks.

A new quality measure of task execution is introduced to monitor the progress of the task. Performance is evaluated only at discrete jumps between task phases, which summarize the history of the continuous portion of the hybrid system.

Dynamic Optimization has been adapted to the computation of the nominal controls for the task execution, and a direct numerical method has been used to compute the nominal trajectory that satisfies the dynamics of the robot and satisfies the constraints of task environment.

The combination of Hybrid System formalism and of the Dynamic Optimization has lead to an algorithm for the computation of the safe execution of a complex task. First, the model of the task is structured, based on a priori knowledge of the task, and task requirements are represented as constraints on the hybrid model, either on the continuous or on the discrete states. Then an optimal trajectory, satisfying task constraints is computed using the Dynamic Optimization. Task quality is measured at each discrete state, providing for warning in case of performance degradation either in the case of autonomous or in the case of human assisted execution.

The approach is first validated in computer simulations, which indicate the feasibility of this approach.

The experimental verification of the proposed approach is based on the teleoperation equipment available in the *Altair* robotics laboratory and it consists in a 6-DOF manipulator and a 1-DOF force feedback joystick and an open architecture Corba based.

9.1 Contributions

The major contributions of this Thesis can be summarized as follows.

- The problem of the autonomous execution of a complex task has been formulated as a Hybrid Automaton optimization, with applications to the robot assisted execution of surgical procedures.
- An in depth review of the state of the art of autonomous task execution has been produced.
- A method to formalize a complex task using hybrid system tools has been outlined, yielding a representation of task phases, environment constraints, and quality measures.
- The concepts of Safety and Stability for complex task has been introduced, which is considered in the optimal control design.
- Pontryagin's Maximum Principle has been adapted to the computation of optimal hybrid trajectories, and the relative necessary conditions have been derived.
- Autonomous task execution has been formulated as the computation of the optimal trajectory in space state of a hybrid automaton, and subject to state dependent constraints.
- The numerical gradient procedure for computing the time optimal solution has been adapted to the hybrid system formalism, and implemented.
- This approach has combined hybrid system analysis with dynamic optimization.

9.2 Recommendations for Future Research

The following subjects are suggested for further research:

1. **Hybrid System**:
   - The development of automatic procedures for the automatic representation of a task model, given a task execution, for example, by a skilled operator.
   - Development of more complex and structured task quality measures, that can provide real time warnings on the insurgence of critical situations.
   - The development and analysis or different real time execution environments and control suitable for HS.
   - Lesser reliance on a priori knowledge.
2. **Dynamic Optimization**:
   - The numerical computation of optimal trajectories is an important issue. It should be made fast enough so that it could be integrated into a real time execution system. Furthermore, different numerical algorithms should

be investigated to determine the most appropriate method for this specific optimization problem.

3. **Complete Algorithm**:
   - Fault Recognition and Fault Recovery should be investigated and implemented in the algorithm.
   - More complex automa should be considered.
4. **Validation and Verification**:
   - Application of Formal Methods to verify and validate the model, i.e. model checking techniques.

9.3 Conclusion

This Thesis has developed three main aspects of autonomous execution of complex tasks: the formalization of a task as a hybrid automaton, the adaptation of necessary optimality conditions to the case of hybrid automaton, and the development of a numerical procedure for the computation of the nominal task trajectory.

The Hybrid System Formalism is very suitable and powerful to represent the structure and the constraints of a complex task. Certain features of Hybrid Systems, e.g. safe sets, can be interpreted in the context of autonomous tasks, and allow high level analysis of task properties and quality. The use of a priori knowledge of the task allows to avoid the curse of dimensionality of hybrid system analysis and to compute the optimal solution in short time. When initial analysis and simulation verification is needed, the off-line computation of the optimal trajectory yields a nominal trajectory that can be used to verify task sequence and organization. When the task is to be executed on a hardware system, the addition of a simple feedback controller, allows the easy transition from simulation to experimental verification.

Dynamic optimization is well suited to solve a Hybrid System planning and execution problem, and to explore the effects of the various constraints on the final solution. The selection of the parameters in the numerical optimization is a tool that can be used to examine the relative weight of the task constraints. The careful adjustment of the parameters allows the computation of different solutions that give a better understanding of the underlying solution space.

Finally, by combining Hybrid System formalism with Dynamic Optimization methods, the result is the computation of a feasible optimal trajectory that can be directly validated in simulation and tested with experiments. The lack of accurate models in certain phases of the task is compensated by the dynamic optimization. Similarly, the use of feedback control during task experiments, compensates for some of the uncertainties not accounted for in the optimization algorithm. Although still in its infancy, the combination of fast heuristics hybrid system modeling and dynamic optimization has demonstrated its power in computing the controls and the sensory conditions for the autonomous execution of a complex surgical task.

References

1. Matlab identification toolbox. The Mathworks Inc. http: www.mathworks.com.
2. *Computers and Intractability: A Guide to the Theory of NPCompleteness*. San Francisco, 1979.
3. *Numerical Recipes in C*. Press Syndicate, 1988.
4. *Applied Nonlinear Control*. Prentice-Hall, 1991.
5. *Discrete Event Systems, Modeling and Performance Analysis*. Richard D. Irwin, Inc. The Aksen Associates Series in Electrical and Computer Engineering, Illinois, burr ridge edition, 1993.
6. *Nonlinear Systems*. Prentice-Hall, 1996.
7. *Air traffic automation: a case study in distributed decentralized control*. Springer, 1998.
8. J. Albro and J. Bobrow. Optimal motion primitives for a 5 dof experimental hopper. Seoul, Korea, 2001. IEEE International Conference on Robotics and Automation.
9. P. Althaus and H. Christensen. Automatic map acquisition for navigating in domestic environments. In *IEEE International Conference on Robotics and Automation*, pages 1551–1556, Taipei, Taiwan, September 14-19 2003.
10. R. Alur, C. Courcoubetis, N. Halbwachs, T.A. Henzinger, P-H. Ho, X. Nicollin, A. Oliviero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical and Computer Science*, 138:3–34, 1995.
11. R. Alur, C. Courcoubetis, T.A. Henzinger, and P-H Ho. Hybrid automata: an algorithmic approach to the specification and verification of hybrid systems. In *Lecture Notes in Computer Science* [94].
12. R. Alur and D. Dill. The theory of timed automata. In *Theoretical and Computer Scisnce*, volume 126, 1994.
13. R. Alur, T.A. Henzinger, and Sontag. Hybrid systems iii, verification and control. In *Lecture Notes in Computer Science*, volume 1066. Springer, 1996.
14. R.J. Anderson and W Spong. Asymptotic stability for force reflecting teleoperators with time delay. *Journal of Robotics Research*, 11(2):135–148, April 1992.
15. R. Andre-Obrecht. A new statistical approach for the automatic segmentation of continuous speeech signals. *IEEE Transactions on Acustics Speech and Signal Processing*, 36(1), 1988.
16. P. Antsaklis, W. Kohn, A. Nerode, and S. Sastry. Hybrid systems ii. In *Lecture Notes in Computer Science*. Springer, 1995.
17. P. Antsaklis, W. Kohn, A. Nerode, and S. Sastry. Hybrid systems iv. In *Lecture Notes in Computer Science*. Springer, 1997.

18. P. Antsaklis, W. Kohn, A. Nerode, and S. Sastry. Hybrid systems v. In *Lecture Notes in Computer Science*. Springer, 1998.
19. K. Arras, R. Philippsen, N. Tomatis, M. de Battista, M. Schilt, and R. Siegwart. A navigation framework for multiple mobile robots and its applicaiton at the Expo02 Exhibition. In *IEEE International Conference on Robotics and Automation*, pages 1992–1997, Taipei, Taiwan, September 14-19 2003.
20. J. Azinheira, P. Rives, J. Carvalho, G. Silveira, E. de Paiva, and S. Bueno. Visual control for the hovering of an outdoor robotic airship. In *IEEE International Conference on Robotics and Automation*, pages 2787–2792, Washington, DC, May 2002.
21. G. Backes, P. Rabideau, K. Tao, and S. Chien. Automated planning and scheduling for planetary rover distributed operations. In *IEEE Int. Conf. on Robotics and Automation*, pages 1096–1101, Detroit,MI, May 1999.
22. P. Backes, J. Beahan, and B. Bon. Interactive command building and sequencing for supervisory autonomy. In *IEEE Int. Conf. on Robotics and Automation*, pages 795–801, 1993.
23. P. Backes and M. Long. Merging concurrent behaviors on a redundant manipulator. In *IEEE Int. Conf. on Robotics and Automation*, pages 638–645, 1993.
24. P. Backes, M. Long, and R. Steele. The modular telerobot task eecution system for space telerobotics. In *IEEE Int. Conf. on Robotics and Automation*, pages 524–329, 1993.
25. P. Backes and K. Tao. Umi: An interactive supervisory and shared control system for telerobotics. In *IEEE Int. Conf. on Robotics and Automation*, pages 1096–1101, 1990.
26. T. Balch and R. Arkin. Behavior-based formation control for multirobot teams. *IEEE Transactions of Robotics and Automation*, 14(6):926–939, December 1998.
27. A. Banerjee, P. Banerjee, T. DeFanti, A. Hudson, B. Dodds, and J. Curtis. A behavioral layer architecture for telecollaborative virtual manufacturing operations. *IEEE Transactions of Robotics and Automation*, 16(3):218–227, June 2000.
28. M. Basseville. Detecting chages in signals and systems - a survey. *Automatica*, 24(3), 1988.
29. G. Bekey, H. Liu, R. Tomovich, and W. Karplus. Knowledge based control of grasping in robot hands using heuristics from human motor skills. *IEEE Transactions of Robotics and Automation*, 9(6):709–722, December 1993.
30. G. Bekey and R. Tomovich. Robot control by reflex action. In *IEEE International Conference on Robotics and Automation*, pages 240–247, San Francisco, CA, April 7-10 1986.
31. R. Bellman. *Dynamic Programming*. Princeton University Press, New Jersy, 1957.
32. R. Bellman. *Adaptive Control Processes, A Giuded Tour*. Princeton University Press, New Jersy, 1961.
33. R. Bellman and G. Leitman. *On the Determination of Optimal Trajectories Via Dynamic Programming*. Academic Press, New York, optimization techniques edition, 1962.
34. R. E. Bellman, H. H. Kagiwada, R. E. Kalaba, and R. Vasudevan. Numerical analysis: Quasilinearization and the estimation of differential operators from eigenvalues. *Commun. ACM*, 11(4):255–256, 1968.
35. A. Bemporad, F. Borelli, and M. Morari. Optimal controllers for hybrid systems: Stability and piecewise linear explicit form. In *39th IEEE Conf. On Decision and Control*, pages 1810–1815, Dec. 2000.
36. A. Bemporad, G. Ferrari-Trecate, and M. Morari. Observability and controllability of piecewise af ne and hybrid systems. *IEEE Transactions on Automatic Control*, 45(10):1864–1876, 2000.

37. A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3):407–427, 1999.
38. A Benveniste. Compositional and uniform modeling of hybrid systems. *IEEE Transactions on automatic control, Special Issue on Hybrid Systems*, 1998.
39. C. Bernard, H. Kang, S. Singh, and J. Wen. Robotic system for collaborative control of minimally invasive surgery. *Industrial Robot: an International Journal*, 26(6):476–484, 1999.
40. C. Bernard, H. Kang, S.K. Singh, and Wen J.T. Robotic system for collaborative control in minimally invasivesurgery. *Industrial Robot:An international Journal*, 26(6):476–484, 1999.
41. A. et al. Bicchi. A sensorized minimally invasive surgery tool for detecting tissue elastic properties. In *1996 IEEE International Conference on Robotics and Automation:*, 1996.
42. H.G. Bock, E. Eich, and J.P. Schlöder. Numerical solution of constrained least squares boundary value problems in differential-algebraic equations. In K. Strehmel, editor, *Numerical Treatment of Differential Equations*.
43. D. Botturi, A. Castellani, D. Moschini, and P. Fiorini. Performance evaluation of task control in teleoperation. In *International Conference on robotics and Automation*, New Orleans, 2004.
44. M. S. Branicky, V. S. Borkar, and S. K. Mitter. A unified framework for hybrid control: Model and optimal optimal control theory. *IEEE Transaction on Automatic Control*, 43(1):31–45, April 1998.
45. M.S. Branicky. Topology of hybrid systems. In *32nd IEEE CDC*, pages 2309–2314, San Antonio,TX, december 1993.
46. M.S. Branicky. Multiple lyapunov functions and other analysis tools for switched and hybrid systems. *IEEE TRansaction on Automatic Control*, 43(4), 1998.
47. M.S. Braniky. Universal computation and other capabilities of hybrid and continuous dynamical systems. *Theoretical Computer Science*, 138(1):67–100, 1995.
48. R. Brockett. Hybrid systems in classical mechanics. In *13th IFAC*, pages 473–476, 1996.
49. R.W. Brockett. Hybrid models for motion control systems. In H.L. Tentleman and J.C. Willems, editors, *Essays on Control: Perspectives in Theory and its Applications*, Boston, 1993. Birkhäuser.
50. R. Brooks. A robust layered control system for a mobile robot. *IEEE Transactions of Robotics and Automation*, 2(1), 1986.
51. R. Brooks. A hardware retargetable distribuited layered architecture for mobile robot control. In *IEEE International Conference on Robotics and Automation*, pages 240–247, Raleigh, NC, March 31 - April 3 1987.
52. A.E. Bryson and W.F. Denham. A steepest ascent method for solving optimum programming problems. *J.Applied Machanics*, (29):247–57, 1962.
53. A.E. Bryson and Yu-Chi Ho. *Applied Optimal Control*. Hemisphere Publishing Corporation, 1975.
54. D. Bullock and S. Grossberg. A neural network architecture for automatic trajectory formation and coordination of multiple effectors during variable-speed arm movements. In *IEEE International Conference on Neural Network*, pages 559–566, San Diego, CA, June 1987.
55. M. Buss, O. Von Stryk, R. Bulirsch, and G. Schmidt. Towards hybrid optimal control. Technical report, University of Berlin, 2000.
56. P.E. Caines and Y-J. Wei. Hierarchical hybrid control systems: A lattice formulation. *Transactions on Automatic Control, Special Issue on Hybrid Systems*, 1998.
57. M. Campos, G. Pereira, S. Vale, A. Bracarense, G. Pinheiro, and M. Oliveira. A mobile manipulator fo rinstallation and removal of aircraft warning spheres on aerial

power transmission lines. In *IEEE International Conference on Robotics and Automation*, pages 3559–3564, Washington, DC, May 2002.
58. C. Cao, C. MacKenzie, and S. Payandeh. Task and motion analyses in endoscopic surgery. In *ASME IMECE Conference: 5th Annual Symposium on Haptic Interface for Virtual Environment and Teleoperator Systems*, pages 583–590, Atlanta,GA, 1996.
59. C. Casadei, S. Martelli, and P. Fiorini. A workcell for the development of robot-assisted surgical procedures. *Journal of Intelligent and Robotic Systems*, (28):301–324, 2000.
60. A. Castellani, D. Botturi, M. Bicego, and P. Fiorini. Hybrid hmm/svm model for the analysis and segmentation of teleoperation tasks. In *IEEE International Conference on Robotics and Automation*, New Orleans,LA, 2004.
61. M. C. Cavusoglu, J. Yan, and S. S. Sastry. A hybrid system approch to contact stability and force control in robotic manipulator. In *12th IEEE International Symposium on Intelligent Control*, pages 143–148, July 1997.
62. L. Chaimowicz, M. Campos, and V. Kumar. Dynamic role assigment for cooperative robots. In *IEEE International Conference on Robotics and Automation*, pages 293–298, Washington, DC, May 2002.
63. L. Chaimowicz, M. Campos, and V. Kumar. Hybrid systems modeling of cooperative robots. In *IEEE International Conference on Robotics and Automation*, pages 4086–4091, Taipei, Taiwan, September 2003.
64. Z. Chaochen, A.P. Ravn, and M.R. Hansen. An extended duration calculus for hybrid real-time systems. In *Lecture Notes in Computer Science* [94].
65. S. Charles, H. Das, T. Ohm, C. Boswell, G. Rodriguez, R. Steele, and D. Istrate. Dexterity-enhanced telerobotics microsurgery. In *8th International Conference on Advanced Robotics (ICAR97)*, Monterey, CA, July 1997.
66. R. Courant and D. Hilbert. *Methods of Methematical Physics*, volume 1. Interscience, New York, 1953.
67. J.E.R. Cury, B.H. Krogh, and T. Niinomi. Synthesis of supervisory controllers for hybrid systems based on approximating automata. *Transactions on Automatic Control, Special Issue on Hybrid Systems*, 1998.
68. H. Das, H. Zak, W. S. Kim, A. K. Bejczy, and P. S. Schenker. Operator performance with alternative manual control modes in teleoperation. *Presence*, 1(2):201–218, Spring 1992.
69. B. Davies. A review of robotics in surgery. volume Part H, pages 129–140. Institution of Mechanical Engineers, 2000.
70. B. De Schutter. Optimal control of a class of linear hybrid systems with saturation. *SIAM Journal on Control and Optimization,*, 2000.
71. K.T Den Boer. *Surgical Task Performance, Assessment using Time-Action Analysis*. PhD thesis, Delft University of Technology, 2001.
72. A.M. Derossis, G.M. Fried, M. Abrahamowicz, H.H. Sigman, J.S. Barkun, and J.L. Meakins. Development of a model for training and evaluation of laparoscopic skills. *Am.J.Surg*, 1998.
73. A. Deshpande, A. Gollu, and P. Varaiya. The shift programming language and run-time system for dynamic networks of hybrid automata. *IEEE Transactions on automatic control, Special Issue on Hybrid Systems*, 1998.
74. A. Deshpande and P. Varaiya. Viable control of hybrid systems. In *35th IEEE Conference on Decision and Control*, Kobe, Japan.
75. The Galileo development team. Galileo program description document: command and data subsystem. Technical Report 625-355-060000, D-535 Rev. G, NASA Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA (USA), May 1989.

76. M.B. Dias, M. Zinck, R. Zlot, and A. Stentz. Robust multirobot coordination in dynamic environments. In *IEEE International Conference on Robotics and Automation*, pages 3435–3441, New Orleans, LA, April 2004.
77. P. M. Do Carmo. *Differential Geometry of Curves and Surfaces*. Prentice-Hall, Upper Side river, New Jersey, 1976.
78. M. Dogruel, S. Drakunov, and Oẗguñer. Sliding mode control in discrete state systems. In *32nd Conference on Decision and Control*, pages 1194–1199, 1993.
79. S.E. Dreyfus. *Dynamic Programming and the Calculus of Variations*. Academic Press, New York, 1965.
80. R. Dubey, S. Everett, N. Pernalete, and K. Manocha. Teleoperation assistance through variable velocity mapping. *IEEE Transactions of Robotics and Automation*, 17(5):761–766, October 2001.
81. B. Eberman. A model-based approach to cartesian manipulation contact sensing. *International Jurnal of Robotics Research*, 16(4), 1997.
82. B. Eberman and J.K. Salisbury. Application of change detection to dynamic contact sensing. *International Jurnal of Robotics Research*, 13(5), 1994.
83. M. Egerstedt. Behavior based robotics using hybrid automata. In Springer-Verlag, editor, *Lecture Notes in Computer Science: Hybrid Systems III*, pages 103–116. 2000.
84. A. Ferreira. Strategies for human robot interaction for automatic microassembly. In *IEEE International Conference on Robotics and Automation*, pages 3076–3081, Taipei, Taiwan, September 2003.
85. P.A. Finlay and M.H. Ornstein. Controlling the movement of a surgical laparoscope. *IEEE Engineeing in Medicine and Biology*, pages 289–291, May/June 1995.
86. P. Fiorini. Autonomous organization of grasping tasks. In *SPIE Symposia on Aerospace Sensing, Artificial Intelligence VII*, Orlando, FL, March 27-31 1989.
87. P. Fiorini, A. Giancaspro, S. Losito, and G. Pasquariello. Neural networks for the segmentation of teleoperation tasks. In *Presence*, pages Vol.2, Number I, Winter 1993.
88. P.M. Fitts. The information capacity of the human motor system in controlling the amplitude of movements. *Journal of Experimental Psychology*, 47(6):381–391, June 1954.
89. A.F. Fuller. Lyapunov centenary issue. *International Journal of Control*, 55(3), 1992.
90. I.M. Gelfand and S.V. Fomin. *Calculus of Variations*, volume 1. Prentice-Hall, New Jersy, 1963.
91. P. Gill, W. Murray, and M. Saunders. *Users guide for SNOPT 5.3: a fortran package for large-scale nonlinear programming.* Department of Mathematics, Univ. of California, San Diego, 1997.
92. A. Göllü and P. Varaiya. Hybrid dynamical systems. In *28th IEEE Conference on Decision and Control*, pages 2708–2712, Tampa, Florida, December 1989.
93. G. Grammel. Maximum principle for hybrid system via singular pertrubations. *SIAM Control and Optimization*, 37(4):1162–1175, 1999.
94. A. Grossman, A. Nerode, A.P. Ravan, and H. Rischel. Hybrid systems. In *Lecture Notes in Computer Science*, volume 736. Springer-Verlag, 1993.
95. J. Guckenheimer. A robust hybrid stabilization strategy for equilibria. *IEEE Transactions on Automatic control*, 40(2):321–326, 1995.
96. A. Guia, C. Seatzu, and C. V. D. Mee. Optimal control of switched autonomous linear systems. In *40th IEEE Conf. On Decision and Control*, pages 2472–2477, Dec. 2000.
97. B. Hannaford and P. Lee. Hidden markov model analysis of force/torque information in telemanipulatio. *The International Journal of Robotics Research*, 10(5):528–539, October 1991.

98. M. Hardt, J. Helton, and K. Kreutz-Delgado. Numerical solution of nonlinear h2 and h1 control problems with application to jet engine compressors. *IEEE Transactions on Control Systems Technology*, 2000.
99. S. Hayati, T. Lee, K. Tso, and P. Backes. A testbed for a unified teleoperated-autonomous dual-arm robotic system. In *IEEE Int. Conf. on Robotics and Automation*, pages 1090–1095, 1990.
100. S. Hayati, T. Lee, K. Tso, P. Backes, and E. Kan. The jpl telerobot manipulator control and mechanization subsystem (mcm). pages 173–181.
101. S. Hedlund and A Rantzer. Optimal control of hybrid systems. In *38th IEEE Conf. On Decision and Control*, pages 3972–3977, Dec 1999.
102. W.P.M.H. Heemels, J.M. Schumacher, and S. Weiland. Linear complementarity systems. *SIAM Journal on Applied Mathematics*, 60(4):1234–1269, 2000.
103. T. Henderson et al. The specification of distributed sensing and control. *Journal of Robotic Systems*, 2(4), April 1985.
104. T.A. Henzinger, P-H. Ho, and H. Wong-Toi. Algorithmic analysis of nonlinear hybrid systems. *IEEE Transactions on automatic control, Special Issue on Hybrid Systems*, 1998.
105. I.A. Hiskens. Analysis tools for power systems - contending with nonlinearities. 1995.
106. R.D. Howe and Y. Matsuoka. Robotics for surgery. In *Annual Review of Biomedical Engineering*, volume 1. 1999.
107. M. Johansson and A. Rantzer. Computation of piecewise lyapunov function for hybrid systems. *IEEE Transaction on Automatic Control*, 43(4):555–559, April 1998.
108. H. Kang and J.T. Wen. Robotic assistants aid surgeons during minimally invasive procedures. *IEEE Engineering in Medicine and Biology*, pages 94–104, January/February 2001.
109. L.V. Kantorovich and V.I. Krylov. *Approximate Methods of Higher Analysis*, volume 1. Interscience and P.Noordhoff Ltd., New York and Groningen, The Netherlands, 1958.
110. H.J. Kelley and G. Leitman. *Method of Gradients*. Academic Press, New York, optimization techniques edition, 1962. Chapter 6.
111. M. Kitagawa, M. Okamura, B.T. Bethea, V.L. Gott, and A. Baumgartner. Analysis of suture manipulation forces for teleoperation with force feedback. In *Fifth International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI)*, 2002.
112. I. Kolmanovsky and N.H. McClamroch. Hybrid feedback laws for a class of nonlinear cascade systems. *IEEE Transaction on Automatic Control*, 41:1271–1281, 1996.
113. K. Kondak and G. Hommel. Computation of time optimal movements for autonomous parking of non-holonomic mobile platforms. Seoul, Korea, 2001. IEEE International Conference on Robotics and Automation.
114. D. Kragic, A. Miller, and P. Allen. Realtime tracking meets on line grasping planning. In *IEEE International Conference on Robotics and Automation*, pages 2460–2465, Seoul, Korea, May, 21-26 2001.
115. A. Krupa, C. Doignon, J. Gangloff, M. de Mathelin, G. Morel, L. Soler, J. leroy, and M. Ghodoussi. Towards semi-autonomy in laparoscopic surgery: first live experiments. Sant'Angelo d'Ischia, Italy, July 2002.
116. A Krupa, C. Doignon, J. Gengloff, M. de Mathelin, L. Soler, G. Morel, J. Leroy, and M. Ghodoussi. Towards semi-autonomy in laparoscopic surgery: first live experiment. In *8th International Symposium on Experimental Robotics*, Sant'Angelo d'Ischia, Italy, July 2002.

117. Y.S. Kwoh, E. Jonckheere, and S. Hayati. A robot with improved absolute positioning accuracy for ct guided stereotactic surgery. *IEEE Transaction on Biomedical Engineering*, pages 153–161, February 1988.
118. R.E. Larson. Dynamic programming with reduced computational requirements. *IEEE Transaction on Automatic Control*, (10):135–143, 1965.
119. J.C. Latombe, editor. *Robot Motion Planning*. Kluwer Academic Publisher, Boston, MA, 1991.
120. D.A. Lawrence. Stability and transparency in bilateral teleoperation. *IEEE Transaction on Robotics and Automation*, 9(5):624–637, October 1993.
121. S. Lee and M.H. Kim. Cognitive contorl of dynamic systems. In *IEEE International Symposium on Intelligent Control*, pages 455–460, Philadelphia, PA, January 19-20 1987.
122. G. Leitman. *An Introduction to Optimal Control*. McGraw-Hill Book Co., New York, NY, 1966.
123. M.D. Lemmon and P.J. Antsaklis. Timed automata and robust control: Can we now control complex dynamical systems? In *36th IEEE Conference on Decision and Control*.
124. M. Lind. The use of flow models for automated plant diagnosis. In *NATO Sy,posium on human detection and diagnosis of system failures*, 1981.
125. J. Lygeros, D.N. Godbole, and S. Sastry, editors. *A Game Theoretic Approach to Hybrid System Design*. University of California, Berkeley, 1995.
126. J. Lygeros, D.N. Godbole, and S. Sastry. Multiagent hybrid system design using game theory and optimal control. In *35th IEEE Conference on Decision and Control*, pages 1190–1195, Kobe, Japan, 1996.
127. J. Lygeros, D.N. Godbole, and S. Sastry. Verified hybrid controllers for automated vehicles. *Transactions on Automatic Control, Special Issue on Hybrid Systems*, 1998.
128. N. Lynch, R. Segala, and F. Vaandrager. Hybrid i/o automata. *Inf. Comput.*, 185(1):105–157, 2003.
129. H. Maa and U. Kühnapfel. Noninvasive measurement of elastic properties of living tissue. Technical report, Institut fr Angewandte Informatik, Forschungszentrum Karlsruhe,, 1997.
130. B. Martin and J. Bobrow. Minimum e ort motions for open chain manipulators with task-dependent end-e ector constraints. Albuquerque, New Mexiko, 1997. In Proceedings of the IEEE International Conference on Robotics and Automation.
131. N.H. McClamroch, C. Rui, I. Kolmanvsky, and M. Reyhanoglu. Hybrid closed loop systems: A nonlinear control perspective. In *36th IEEE Conference on Decision and Control*, 1997.
132. R. McGill. Optimal control, inequality state constraint, and the generalized newton-raphson algorithm. *SIAM J.Control*, 1965.
133. E.J. McShane. *On Multipliers for Lagrange Problems*, volume 61. American J. Math., New York, 1939.
134. W.S. Melvin and E.C Johnson, J.A.and Ellison. Laparoscopic skill enhancement. *Am.J.Surg.*, 1996.
135. C.W. Merriam. *Optimization Theory and Design of Feedback Control Systems*. McGraw-Hill Book Co., New York, 1964.
136. A. Meystel. Intelligent control in robotics. *Journal of Robotic Systems*, 5(4):269–308, August 1988.
137. A.S. Morse. Control using logic-based switching. In *Lecture Notes in Control and Information Science*, volume 222. Springer, 1997.
138. M.S.Branicky. *Studies in Hybrid Systems:Modeling, Analysis, and Control*. PhD thesis, Massachusetts Institute of Technology, 1995. In English.

139. A. Nerode and W. Kohn. Multiple agent hybrid control architecture. In *Lecture Notes in Computer Science* [94].
140. K. Passino and P. Antsaklis. A system and control theoretic perspective on artificial intelligence planning systems. *Applied Artificial Intelligence*, 3:1–32, 1989.
141. P. Peleties and R.A. DeCarlo. Modeling of interacting continuous time and discrete event systems: An example. In *26th Annual Allerton Conference on comunication, control and Computing*, pages 1150–1159, 1988.
142. P. Peleties and R.A. DeCarlo. Asymptotic stability of *m*-switched systems using lyapunov-like functions. In *American Control Conference*, pages 1679–1684, 1991.
143. L. Petersson, D. Austin, and D. Kragic. High-level control of a manipulator for door opening. volume 3, pages 2333–2338, Takamatsu, Kagawa, Japan, 2000.
144. S. Pettersson and B. Lennartson. Stability and robusteness of hybrid systems. In *35th Conference on Decision and Control*, Kobe, Japan, 1996.
145. A. Pnueli and J. Sifakis. Special issue on hybrid systems. In *Theoretical Computer Scinence*, volume 138, 1995.
146. L.S. Pontryagin. *The Mathematical Theory of Optimal Processes*. Wiley, New York, 1962.
147. A. Potocnik, B.and Bemporad, F.D. Torrisi, G. Music, and B. Zupancic. Hysdel modeling and simulation of hybrid dynamical systems. In *MATHMOD Conference*, Vienna, February 2003.
148. K. Radermacher, H.W. Staudte, and G. Rau. Computer assisted orthopaedic surgery by means of individual templates. aspects and analysis of potential applications. In *1st International Symposium on Medical Robotics and Computer Assisted Surgery (MRCAS'94)*, pages 42–48, Pittsburgh, PA, September 1994.
149. J. Raisch and S.D. O'Young. Discrete approximation and supervisory control of continuous systems. *Transactions on Automatic Control, Special Issue on Hybrid Systems*, 1998.
150. J. Rasmussen and M. Lind. Coping with complexity. In *European Annual Conference of Human Decision and Manual Control*, Delft, The Netherlands, 1981.
151. J. Rasmussen and M. Lind. Model of human decision making in complex systems and its use for design of control strategies. In *American Control Conference*, Arlington, VA, June, 14-16 1982.
152. J. Rosen, J.D. Brown, L. Chang, M. Barreca, M. Sinanan, and B. Hannaford. The bluedragon - a system for measuring the kinematics and the dynamics of minimally invasive surgical tools in-vivo. In *2002 IEEE International Conference on Robotics and Automation*, pages 1876–1881, Washington, DC, May 2002.
153. Jee-Hwan Ryn, Dong-Soo Kwon, and B. Hannaford. Stable teleoperation with time domain passivity control. In *2002 IEEE International Conference on Robotics & Automation*, pages 3260–3264, Waschintong, US, May 2002.
154. A.P. Sage and C.C.III White. *Optimum System Control*. Prentice-Hall, New Jersy, 1977.
155. R. Satava. Virtual reality surgical simulator. *Surg Endosc*, 1993.
156. R. Sengupta and S. Lafortune. An optimal control theory for discrete event systems. *SIAM Control and Optimization*, 36(2):488–541, 1998.
157. R. Simmons. Structured control for autonomous robots. *IEEE Transactions of Robotics and Automation*, 10(1):34–43, February 1984.
158. C. Simone. *Modeling of needle insertion forces for percutaneous therapies*. PhD thesis, The Johns Hopkins University, Baltimore, Mariland, 2002.
159. E. D. Sontag, editor. *Mathematical Control Theory: Deterministic Finite Dimensional Systems*. Springer, New York, 1998.
160. E.D. Sontag. Nonlinear regulation: the piecewise linear approach. *IEEE Transactions on Automatic Control*, 26(2):346–357, 1981.

161. J.A. Stiver, P.J. Antsaklis, and M.D Lemmon. An invariant based approach to the design of hybrid contol systems. volume J, San Francisco, 1996.
162. J.A. Stiver, P.J. Antsaklis, and M.D. Lemmon. A logical des approach to the design of hybrid control systems. *Mathl. Comput. Modelling*, 23(11/12):55–76, 1996.
163. Y. Sun, N. Xi, and Y. Wang. Modeling and analysis of perceptive robot controller based on hybrid automata. In *IEEE International conference on robotics and Automation*, New Orleans, LA, 2004.
164. G.T. Sung and I.S. Gill. Robotic laparoscopic surgery: a comparison of the *da vinci* and *Zeus* systems. *Elsevier Science*, Urology(58):893–898, 2001.
165. L.W. Tang, G. D'Ancona, J. Bergsland, and H.L. Kawaguchi, A.and Karamanoukian. Robotically assisted video-enhanced-endoscopic coronary artery bypass graft surgery. *Angiology*, 52:99–102, 2001.
166. L. Tavernini. Differential automata and their discrete simulator. In *Nonlinear Analysis, Theory, Methods and Applications*, volume 11, pages 665–683, 1987.
167. J.H. Taylor and D. Kebede. Modeling and simulation of hybrid systems in matlab. volume J, San Francisco, 1996.
168. R.H. Taylor, J. Funda, B. Eldridge, S. Gomory, K. Gruben, D. LaRose, M. Talamini, L. Kavoussi, and J. Anderson. A telerobotic assistant for laparoscopic surgery. *IEEE Engineeing in Medicine and Biology*, pages 279–288, May/June 1995.
169. R.H. Taylor, S. Lavallé, G.C. Burdea, and R. Mosges, editors. *Computer - Integrated Surgery*. MIT Press, Cambridge, MA, 1996.
170. M. Tittus and B. Egart. Control design for integrator hybrid systems. *IEEE Transaction on Automatic Control*, 43(4):491–500, April 1998.
171. C. Tomlin. Towards effcient computation of solutions to hybrid systems. Phoenix, AZ, 1999. 38th IEEE Conference on Decision and Control.
172. C. Tomlin, G.J. Pappas, and S. Sastry. Conflict resolution for air traffic management: A study in multi-agent hybrid systems. *Transactions on Automatic Control, Special Issue on Hybrid Systems*, 1998.
173. R. Tomovich et al. A strategy for grasp synthesis with multifingered robot hands. In *IEEE International Conference on Robotics and Automation*, pages 83–89, Raleigh, NC, March 31 - April 3 1987.
174. J. Troccaz, M. Peshkin, and B. Davies. The use of localizers, robots and synergistic devices in cas. In *4th International Symposium on Medical Robotics and Computer Assisted Surgery (CVRMed-MRCAS'97)*, pages 727–735, Grenoble, France, March 1997.
175. University of Pennsylvania School of Veterinary Medicine. *Principles of Surgery*.
176. A.J. Van der Schaft and M. Schumacher. Complementarity modeling of hybrid systems. *IEEE Transaction on Automatic Control*, 43(4):483–490, April 1998.
177. A.J. Van der Schaft and M. Schumacher. An introduction to hybrid dynamical systems. In Springer Verlag, editor, *In Lecture Notes in Control and Information Sciences*, volume 251. 2000.
178. O. Von Stryk. *Numerical hybrid optimal control and related topics*. PhD thesis, Technical University of Munich, 2000. In English.
179. O. Von Stryk. *User s guide for DIRCOL version 2.1: A direct collocation method for the numerical solution of optimal control problems*. Tecnical University of Darmstadt, 2001. WWW:www.sim.informatik.tu-darmstadt.de/sw/.
180. O. Von Stryk and M. Glocker. Decomposition of mixed-integer optimal control problems using branch and bound and sparse direct collocation. In ADPM 4th Int l Conf. on Automation of Mixed Processes: Hybrid Dynamic Systems, 2000.
181. O. Von Stryk and M. Schlemmer. Optimal control of the industrial robot manutec r3. In R. Bulirsch and D. Kraft, editors, *Computational Optimal Control*, volume 115 of *International Series of Numerical Mathematics*, pages 367–382. 1994.

182. J.C. Willems. Paradigms and puzzles in the theory of dynamical systems. *IEEE Transactions on Automatic Control*, 36:259–294, 1991.
183. A.S. Willsky. A survey of design methods for failure detection in dynamic systems. *Automatica*, 12(6), 1974.
184. N. Xi, Tarn T.J., and A.K. Bejczy. Intelligent planning and control for multirobot coordination: An event-based approach. *IEEE Transaction on Robotics and Automation*, 12(2), June 1996.
185. X. Xu and J. Antsaklis. A dynamic programmin approach for optimal control of switched systems. In *39th IEEE Conf. On Decision and Control*, pages 1822–1827, Dec. 2000.
186. H. Ye, A.N. Michel, and L. Hou. Stability theory for hybrid dynamical systems. pages 2779–2684, New Orelans, 1995. IEEE Conference on Decision Control.
187. H. Ye, N. Michel, and L. Hou. Stability theory for hybrid dynamical systems. *IEEE Transaction on Automatic Control*, 43(4):461–474, April 1998.
188. Y. Yokokohji and T. Yoshikawa. Bilateral control of master-slave manipulators for ideal kinestetic coupling–formulation and experiment. *IEEE Transactions on Robotics and Automation*, 10(5):605–620, October 1994.

Sommario

La continua crescita della Chirurgia Minimamente Invasiva é motivata dalla volontá di migliorare le cure dei pazienti e contemporaneamente abbassare i costi della sanitá. Questi obiettivi richiedono lo sviluppo di tecniche chirurgiche, con particolare attenzione alla riduzione della variabilitá e la crescita dell'efficienza delle stesse.

Molti dispositivi robotici sono stati sviluppati o proposti per cercare di raggiungere tali obiettivi, ed alcuni di essi sono usati in ambito ospedaliero. Nella robotica assistita le procedure sono completamente eseguite dal chirurgo e il robot svolge solamente la funzione di strumento, a volte anche troppo sofisticato, per svolgere tale operazione.

Quindi per aumentare la qualitá delle cure e ridurre il loro costo, sarebbe desiderabile e necessario esplorare lo sviluppo di algoritmi che diano la capacitá al robot di collaborare con il chirurgo durante una procedura minimamente invasiva. In questo contesto, con collaborazione si intende la capacitá di un robot di portare avanti autonomamente sottoprocedure, adattandosi alla forte variabilitá tipica dell' ambiente chirurgico. Questa collaborazione potrebbe ridurre le fatiche del chirurgo, eliminare la variabilitá presente tra i vari specializzandi e conseguentemente nella esecuzione delle procedure chirurgiche ed infine aumentare l'efficienza dell'intero sistema riducendo il tempo di impiego del chirurgo.

In questa tesi viene esaminata l'esecuzione automatica di compiti robotici in ambiente non completamente conosciuto come quello chirurgico, e viene proposto un metodo per il calcolo del controllo nominale per un robot che esegue una tipica procedura chirurgica.

La sicurezza é un problema molto importante nella chirurgia robotica, ed é ancora piú importante nell'esecuzione automatica ti procedure chirurgiche. I robot chirurghi attualmente in commercio non utilizzano la retroazione di forza, privando l'operatore umano di una modalitá di controllo delle prestazione del robot molto potente. Quindi, ancora oggi, una grande responsabilitá per la corretta esecuzione della procedura é messa nella correttezza dell'algoritmo di controllo, vista la variabilitá dell'ambiente. In questa Tesi viene investigata la sicurezza dell'algoritmo utilizzando l'ottimizzazione con vincoli sull'esecuzione. La variabilitá é considerata rispetto al modello del compito ed ai vincoli ed é quantificata come la distanza del comportamento corrente del robot da quello desiderato durante un'operazione minimamente invasiva. Questo approccio consente anche di monitorare continuamente la qualitá della procedurae le prestazioni dell'operatore.

Tradizionalmente, l'esecuzione automatica di procedure é stata affrontata semplificando il problema ingegnerizzando l'ambiente, ad esempio aggiungendo opportuni punti fissi per cercare di rimuovere l'incertezza del compito. Le piú avanzate tecniche di controllo consistono di comportamenti pre-programmati, cosí che differenti compiti possano essere programmati ed eseguiti usando lo stesso insieme di comportamenti di base. Quando non é disponibile un modello esplicito dell'ambiente, i comportamenti vengono parametrizzati in modo da consentire alcuni gradi di flessibilitá nell'esecuzione del compito. Se il compito e/o il modello

dell'ambiente sono conosciuti o possono essere identificati, é possibile sviluppare una legge di controllo esplicita per il controllo dell'esecuzione della procedura. In questa Tesi abbiamo sviluppatouna legge di controllo esplicita usando le teoria dei Sistemi Ibridi, e abbiamo modellato l'incertezza con una appropiata sequanza di stati nel sistema ibrido. In particolare, abbiamo usato un automaton deterministico come modello della procedura considerato che le procedure chirurgiche studiate sono ben codificate nella letteratura medica, e possono essere propiamente rappresentate come procedure autonome.

Rappresentiamo una procedura chirurgicacome un automaton ibrido i quali stati elementari rappresentanouna distinta azione della procedura. In questo modo, possiamo calcolare il controllo nominale ottimizzando un appropiato indice di qualitá nel dominio del sottocompito, che corrisponde ad ogni azione del compito. In questo contesto, il compromesso tra la conoscenza esplicita del modello e la compensazione con retroazione influenza pesantemente le prestazioni del sistema. Abbiamo indagato la relazione tra conoscenza a priori, rappresentata dall'automaton ibrido, il numero e la forma dei vincoli ed il tipo di retroazione on-line per garantire delle linee guida per il progetto dell'intero sistema.

Auspichiamo che l'integrazione di tecniche dalla teoria dei sistemi ibridi, dell' ottimizzazione vincolata e della rappresentazione dei vincoli, faciliti l'autonomia in ambiente incerto. Per dimostrare questo , abbiamo simulato l'esecuzione automatica di una sutura, che é una comoda e rappresentativa procedura chirurgica. L'esecuzione della sutura richiede la definizione di una traiettoria nominale che guidi il robot alla completa e corretta esecuzione del compito. A questo riguardo, definiamo una strategia ottima per calcolare la traiettoria ottima off-line ed una compensazione on-line dalla deviazione dalla traiettoria ottima. Possiamo concludere che l'integrazione di metodi diversi come sistemi ibridi, metodi di ottimizzazione e controllo ottimo hanno facilitato la costruzione di una piú sicura esecuzione automatica di una realistica procedura chirurgica.

Riassumendo, un insieme di strumenti per la sintesi del controllo un un compito complesso é stato proposto, investigato e valutato nel contesto delle procedure chirurgiche autonome. Questo metodo sviluppato costituisce una ricca base per trattare compiti complessi in ambiente non conosciuto ed espande la parte sullo stato dell'arte dell'esecuzione automatica di compiti.