Hybrid System Modeling and Autonomous Control Systems

Panos J. Antsaklis, James A. Stiver, and Michael Lemmon *

Department of Electrical Engineering
University of Notre Dame
Notre Dame, IN 46556 USA

Abstract. Hybrid control systems contain two distinct types of systems, continuous state and discrete-state, that interact with each other. Their study is essential in designing sequential supervisory controllers for continuous- state systems, and it is central in designing control systems with high degree of autonomy.
After an introduction to intelligent autonomous control and its relation to hybrid control, models for the plant, controller, and interface are introduced. The interface contains memoryless mappings between the supervisor's symbolic domain and the plant's nonsymbolic state space. The simplicity and generality afforded by the assumed interface allows us to directly confront important system theoretic issues in the design of supervisory control systems. such as determinism, quasideterminism, and the relationship of hybrid system theory to the more mature theory of logical discrete event systems.

1 Introduction

Hybrid control systems contain two distinct types of systems, continuous and discrete-state, which interact with each other. An example of such a system is the heating and cooling system of a typical home. Here the furnace and air conditioner together with the home's heat loss dynamics can be modeled as continuous-state, (continuous-time) system which is being controlled by a discrete-state system, the thermostat. Other examples include systems controlled by bang-bang control or via methods based on variable structure control.

Hybrid control systems also appear as part of *Intelligent Autonomous Control Systems*. Being able to control a continuous-state system using a discrete-state supervisory controller is a central problem in designing control systems with high degrees of autonomy. This is further discussed below.

The analysis and design of hybrid control systems requires the development of an appropriate mathematical framework. That framework must be both powerful and simple enough so it leads to manageable descriptions and efficient algorithms for such systems. Recently, attempts have been made to study hybrid control

* The partial financial support of the National Science Foundation (IRI91-09298 and MSS92-16559) is acknowledged

systems in a unified, analytical way and a number of results have been reported in the literature [Benveniste 1990] [Gollu 1989] [Grossman 1992] [Holloway 1992] [Kohn 1992] [Lemmon 1993b] [Nerode 1992] [Passino 1991b] [Peleties 1988] [Peleties 1989] [Stiver 1991a] [Stiver 1991b] [Stiver 1991c] [Stiver 1992] [Stiver 1993].

In this chapter, a novel approach to hybrid systems modeling and control is described. Descriptions of the plant to be controlled, the controller and the interface are given in Section 3. The important role of the interface is discussed at length. In Section 4, certain system theoretic questions are addressed. In particular, the concepts of determinism and quasideterminism are introduced and results are given. It is then shown how logical Discrete Event System (DES) models can be used to formulate the hybrid control problem, thus taking full advantage of existing results on DES controller design [Cassandras 1990] [Ozveren 1991] [Passino 1989a] [Passino 1989b] [Passino 1991a] [Passino 1992a] [Ramadge 1987] [Ramadge 1989] [Wonham 1987] for hybrid control systems. When the system to be controlled is changing, these fixed controllers may not be adequate to meet the control goals. In this case it is desirable to identify the plant and derive the control law on line, and this is addressed in the companion chapter in this volume titled "Event Identification and Intelligent Hybrid Control." Inductive inference methods are used to identify plant events in a computationally efficient manner.

In Section 2, after a brief introduction to Intelligent Autonomous Control, the important role hybrid control systems play in the design of Autonomous Control Systems is discussed and explained. In this way, the hybrid control problem can be seen in the appropriate setting so that its importance in the control of very complex systems may be fully understood and appreciated. Further discussion can be found in [Antsaklis 1993b]; for more information on intelligent control see [Albus 1981] [Antsaklis 1989] [Antsaklis 1991] [Antsaklis 1993b] [Antsaklis 1993a] [Antsaklis 1993c] [IEEE Computer 1989] [Passino 1993] [Saridis 1979] [Saridis 1985] [Saridis 1987] [Saridis 1989a] [Zeigler 1984].

2 On Intelligent Autonomous Control Systems

It is appropriate to first explain what is meant by the term Intelligent Autonomous Control [Antsaklis 1989] [Antsaklis 1993b]. In the design of controllers for complex dynamical systems, there are needs today that cannot be successfully addressed with the existing conventional control theory. Heuristic methods may be needed to tune the parameters of an adaptive control law. New control laws to perform novel control functions to meet new objectives should be designed while the system is in operation. Learning from past experience and planning control actions may be necessary. Failure detection and identification is needed. Such functions have been performed in the past by human operators. To increase the speed of response, to relieve the operators from mundane tasks, to protect them from hazards, a high degree of autonomy is desired. To achieve this autonomy, high level decision making techniques for reasoning under uncertainty must be utilized. These techniques, if used by humans, may be attributed to intelligence. Hence, one way to achieve high degree of autonomy is to utilize high level decision making techniques, intelligent methods, in the autonomous controller. In our view, *higher autonomy is the objective, and intelligent controllers are one way to achieve it.* The need for quantitative methods to model and analyze the dynamical behavior of such autonomous systems presents significant challenges well beyond current capabilities. It is clear that the development of autonomous controllers requires significant interdisciplinary research effort as it integrates concepts and methods from areas such as Control, Identification, Estimation, Communication Theory, Computer Science, Artificial Intelligence, and Operations Research.

Control systems have a long history. Mathematical modeling has played a central role in its development in the last century and today conventional control theory is based on firm theoretical foundations. Designing control systems with higher degrees of autonomy has been a strong driving force in the evolution of control systems for a long time. What is new today is that with the advances of computing machines we are closer to realizing highly autonomous control systems than ever before. One of course should never ignore history but learn from it. For this reason, a brief outline of conventional control system history and methods is given below.

2.1 Conventional Control - Evolution and Quest for Autonomy

The first feedback device on record was the water clock invented by the Greek Ktesibios in Alexandria Egypt around the 3rd century B.C. This was certainly a successful device as water clocks of similar design were still being made in Baghdad when the Mongols captured the city in 1258 A.D.! The first mathematical model to describe plant behavior for control purposes is attributed to J.C. Maxwell, of the Maxwell equations' fame, who in 1868 used differential equations to explain instability problems encountered with James Watt's flyball governor; the governor was introduced in the late 18th century to regulate the speed of steam engine vehicles. Control theory made significant strides in the past 120 years, with the use of frequency domain methods and Laplace transforms in the 1930s and 1940s and the development of optimal control methods and state space analysis in the 1950s and 1960s. Optimal control in the 1950s and 1960s, followed by progress in stochastic, robust and adaptive control methods in the 1960s to today, have made it possible to control more accurately significantly more complex dynamical systems than the original flyball governor.

When J.C Maxwell used mathematical modeling and methods to explain instability problems encountered with James Watt's flyball governor, he demonstrated the importance and usefulness of mathematical models and methods in understanding complex phenomena and signaled the beginning of mathematical system and control theory. It also signaled the end of the era of intuitive invention. The performance of the flyball governor was sufficient to meet the control needs of the day. As time progressed and more demands were put on the device there came a point when better and deeper understanding of the device was necessary as it started exhibiting some undesirable and unexplained behavior, in particular oscillations. This is quite typical of the situation in man made systems even today where systems based on intuitive invention rather than quantitative theory can be rather limited. To be able to control highly complex and uncertain systems we need deeper understanding of the processes involved and systematic design methods, we need quantitative models and design techniques. Such a need is quite apparent in intelligent autonomous control systems and in particular in hybrid control systems.

Conventional control design methods: Conventional control systems are designed today using mathematical models of physical systems. A mathematical model, which captures the dynamical behavior of interest, is chosen and then control design techniques are applied, aided by Computer Aided Design (CAD) packages, to design the mathematical model of an appropriate controller. The controller is then realized via hardware or software and it is used to control the physical system. The procedure may take several iterations. The mathematical model of the system must be "simple enough" so that it can be analyzed with available mathematical techniques, and "accurate enough" to describe the important aspects of the relevant dynamical behavior. It approximates the behavior of a plant in the neighborhood of an operating point.

The control methods and the underlying mathematical theory were developed to meet the ever increasing control needs of our technology. The need to achieve the demanding control specifications for increasingly complex dynamical systems has been addressed by using more complex mathematical models and by developing more sophisticated design algorithms. The use of highly complex mathematical models however, can seriously inhibit our ability to develop control algorithms. Fortunately, simpler plant models, for example linear models, can be used in the control design; this is possible because of the feedback used in control which can tolerate significant model uncertainties. Controllers can for example be designed to meet the specifications around an operating point, where the linear model is valid and then via a scheduler a controller emerges which can accomplish the control objectives over the whole operating range. This is in fact the method typically used for aircraft flight control. When the uncertainties in the plant and environment are large, the fixed feedback controllers may not be adequate, and adaptive controllers are used. Note that adaptive control in conventional control theory has a specific and rather narrow meaning. In particular it typically refers to adapting to variations in the constant coefficients in the equations describing the linear plant: these new coefficient values are identified and then used, directly or indirectly, to reassign the values of the constant coefficients in the equations describing the linear controller. Adaptive controllers provide for wider operating ranges than fixed controllers and so conventional adaptive control systems can be considered to have higher degrees of autonomy than control systems employing fixed feedback controllers. There are many cases however where conventional adaptive controllers are not adequate to meet the needs and novel methods are necessary.

2.2 Intelligent Control for High Autonomy Systems

There are cases where we need to significantly increase the operating range of control systems. We must be able to deal effectively with significant uncertainties in models of increasingly complex dynamical systems in addition to increasing the validity range of our control methods. We need to cope with significant unmodeled and unanticipated changes in the plant, in the environment and in the control objectives. This will involve the use of intelligent decision making processes to generate control actions so that a certain performance level is maintained even though there are drastic changes in the operating conditions. It is useful to keep in mind an example, the Houston Control example . It is an example that sets goals for the future and it also teaches humility as it indicates how difficult demanding and complex autonomous systems can be. Currently, if there is a problem on the space shuttle, the problem is addressed by the large number of engineers working in Houston Control, the ground station. When the problem is solved the specific detailed instructions about how to deal with the problem are sent to the shuttle. Imagine the time when we will need the tools and expertise of all Houston Control engineers aboard the space shuttle, space vehicle, for extended space travel.

In view of the above it is quite clear that in the control of systems there are requirements today that cannot be successfully addressed with the existing conventional control theory. They mainly pertain to the area of uncertainty, present because of poor models due to lack of knowledge, or due to high level models used to avoid excessive computational complexity.

The control design approach taken here is a bottom-up approach. One turns to more sophisticated controllers only if simpler ones cannot meet the required objectives. The need to use intelligent autonomous control stems from the need for an increased level of autonomous decision making abilities in achieving complex control tasks. *Note that intelligent methods are not necessary to increase the control system's autonomy. It is possible to attain higher degrees of autonomy by using methods that are not considered intelligent. It appears however that to achieve the highest degrees of autonomy, intelligent methods are necessary indeed.*

2.3 An Intelligent High Autonomy Control System Architecture For Future Space Vehicles

To illustrate the concepts and ideas involved and to provide a more concrete framework to discuss the issues, a hierarchical functional architecture of an intelligent controller that is used to attain high degrees of autonomy in future space vehicles is briefly outlined; full details can be found in [Antsaklis 1989]. This hierarchical architecture has three levels, the Execution Level, the Coordination Level, and the Management and Organization Level. The architecture exhibits certain characteristics, which have been shown in the literature to be necessary and desirable in autonomous intelligent systems.

It is important at this point to comment on the choice for a hierarchical architecture. Hierarchies offer very convenient ways to describe the operation of complex systems and deal with computational complexity issues, and they are used extensively in the modeling of intelligent autonomous control systems. Such a hierarchical approach is taken here (and in [Passino 1993]) to study intelligent autonomous and hybrid control systems.

Architecture Overview: The overall functional architecture for an autonomous controller is given by the architectural schematic of the figure below. This is a functional architecture rather than a hardware processing one; therefore, it does not specify the arrangement and duties of the hardware used to implement the functions described. Note that the processing architecture also depends on the characteristics of the current processing technology; centralized or distributed processing may be chosen for function implementation depending on available computer technology.

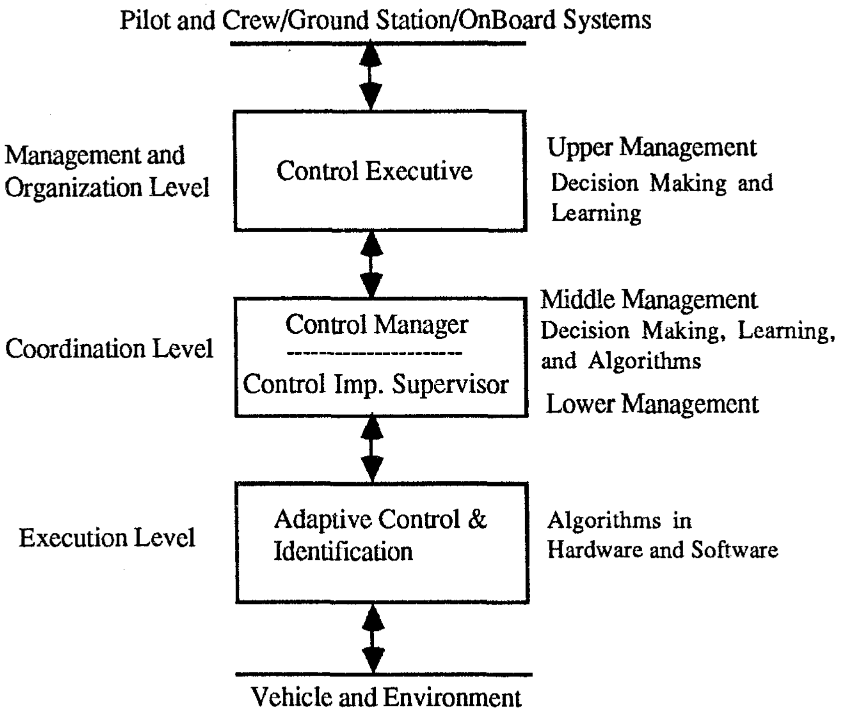

Fig. 1. Intelligent Autonomous Controller Functional Architecture

The architecture in Figure 1 has three levels; this is rather typical in the Intelligent Control literature. At the lowest level, the Execution Level, there is the interface to the vehicle and its environment via the sensors and actuators. At the highest level, the Management and Organization Level, there is the interface to the pilot and crew, ground station, or onboard systems. The middle level, called the Coordination Level, provides the link between the Execution Level and the Management Level. Note that we follow the somewhat standard viewpoint that there are three major levels in the hierarchy. It must be stressed that the system may have more or fewer than three levels. Some characteristics of the system which dictate the number of levels are the extent to which the operator can intervene in the system's operations, the degree of autonomy or level of intelligence in the various subsystems, the hierarchical characteristics of the plant. Note however that the three levels shown here in Figure 1 are applicable to most architectures of autonomous controllers, by grouping together sublevels of the architecture if necessary. As it is indicated in the figure, the lowest, Execution Level involves conventional control algorithms, while the highest, Management and Organization Level involves only higher level, intelligent, decision making methods. The Coordination Level is the level which provides the interface between the actions of the other two levels and it uses a combination of conventional and intelligent decision making methods. The sensors and actuators are implemented mainly with hardware. Software and perhaps hardware are used to implement the Execution Level. Mainly software is used for both the Coordination and Management Levels. There are multiple copies of the control functions at each level, more at the lower and fewer at the higher levels. See [Antsaklis 1989] [Antsaklis 1993b] for an extended discussion of the issues involved.

Hybrid control systems do appear in the intelligent autonomous control system framework whenever one considers the Execution level together with control functions performed in the higher Coordination and Management levels. Examples include expert systems supervising and tuning conventional controller parameters, planning systems setting the set points of local control regulators, sequential controllers deciding which from a number of conventional controllers is to be used to control a system, to mention but a few. One obtains a hybrid control system of interest whenever one considers controlling a continuous-state plant (in the Execution level) by a control algorithm that manipulates symbols, that is by a discrete-state controller (in Coordination and/or Management levels).

2.4 Quantitative Models

For highly autonomous control systems, normally the plant is so complex that it is either impossible or inappropriate to describe it with conventional mathematical system models such as differential or difference equations. Even though it might be possible to accurately describe some system with highly complex nonlinear differential equations, it may be inappropriate if this description makes subsequent analysis too difficult or too computationally complex to be useful.

The complexity of the plant model needed in design depends on both the complexity of the physical system and on how demanding the design specifications are. There is a tradeoff between model complexity and our ability to perform analysis on the system via the model. However, if the control performance specifications are not too demanding, a more abstract, higher level, model can be utilized, which will make subsequent analysis simpler. This model intentionally ignores some of the system characteristics, specifically those that need not be considered in attempting to meet the particular performance specifications. For example, a simple temperature controller could ignore almost all dynamics of the house or the office and consider only a temperature threshold model of the system to switch the furnace off or on.

The quantitative, systematic techniques for modeling, analysis, and design of control systems are of central and utmost practical importance in conventional control theory. Similar techniques for intelligent autonomous controllers do not exist. This is mainly due to the hybrid structure (nonuniform, nonhomogeneous nature) of the dynamical systems under consideration; they include both continuous-state and discrete-state systems. Modeling techniques for intelligent autonomous systems must be able to support a macroscopic view of the dynamical system, hence it is necessary to represent both numeric and symbolic information. The nonuniform components of the intelligent controller all take part in the generation of the low level control inputs to the dynamical system, therefore they all must be considered in a complete analysis. Therefore the study of modeling and control of hybrid control systems is essential in understanding highly autonomous control systems [Antsaklis 1989].

3 Hybrid Control System Modeling

The hybrid control systems considered here consist of three distinct levels; see Figure 2. The controller is a discrete-state system, a sequential machine, seen as a Discrete Event System (DES). The controller receives, manipulates and outputs events represented by symbols. The plant is a continuous-state system typically modeled by differential/difference equations and it is the system to be controlled by the discrete-state controller. The plant receives, manipulates and outputs signals represented by real variables that are typically (piecewise) continuous. The controller and the plant communicate via the interface that translates plant outputs into symbols for the controller to use, and controller output symbols into command signals for the plant input. The interface can be seen as consisting of two subsystems: the generator that senses the plant outputs and generates symbols representing plant events, and the actuator that translates the controller symbolic commands into piecewise constant plant input signals.

To develop a useful mathematical framework we keep the interface as simple as possible; this is further discussed below. The interface determines the events the controller sees and uses to decide the appropriate control action. If the plant and the interface are taken together the resulting system is a DES, called the

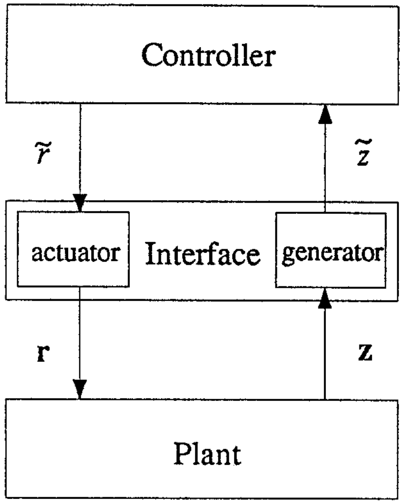

Fig. 2. Hybrid Control System

DES Plant, that the controller sees and attempts to control. Another way of expressing this is that the DES controller only sees a more abstract model of the plant; a higher level less detailed plant model than the differential / difference equation model. The complexity of this more abstract DES plant model depends on the interface. It is therefore very important to understand the issues involved in the interface design so that the appropriate DES model is simple enough so to lead to a low complexity controller. It should be noted that this lower complexity is essential for real time adaptation of hybrid control systems. All these issues pointed out here are discussed in detail later in this chapter.

It is important to identify the important concepts and develop an appropriate mathematical framework to describe hybrid control systems. Here the logical DES theory and the theory of automata are used. The aim is to take advantage as much as possible of the recent developments in the analysis and control design of DES. These include results on controllability, observability, stability of DES and algorithms for control design among others. We first present a flexible and tractable way of modeling hybrid control systems. Our goal is to develop a model which can adequately represent a wide variety of hybrid control systems, while remaining simple enough to permit analysis. We then present methods which can be used to analyze and aid in the design of hybrid control systems. These methods relate to the design of the interface which is a necessary component of a hybrid system and its particular structure reflects both the dynamics of the plant and the aims of the controller.

Below, the plant, interface and controller are described first. The assumptions made and the generality of the models are discussed. In Section 4, the DES plant model is then derived and the concepts of determinism and quasideterminism are introduced and certain results are shown. The description of the generator in the interface is discussed. Controllability of the DES plant model is studied. The selection of the interface is discussed at length and the fundamental issues are identified. Connections to Ramadge-Wonham model are shown, the difficulties involved are indicated, and some recent results are outlined. Simple examples are used throughout to illustrate and explain. Note that most of these results can be found in [Stiver 1992].

A hybrid control system, can be divided into three parts, the plant, interface, and controller as shown in Figure 2. The models we use for each of these three parts, as well as the way they interact are now described.

3.1 Plant

The system to be controlled, called the plant, is modeled as a time-invariant, continuous-time system. This part of the hybrid control system contains the entire continu-ous-time portion of the system, possibly including a continuous-time controller. Mathematically, the plant is represented by the familiar equations

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{r}) \tag{1}$$

$$\mathbf{z} = g(\mathbf{x}) \tag{2}$$

where $\mathbf{x} \in \Re^n$, $\mathbf{r} \in \Re^m$, and $\mathbf{z} \in \Re^p$ are the state, input, and output vectors respectively. $f : \Re^n \times \Re^m \to \Re^n$ and $g : \Re^n \to \Re^p$ are functions. This is the common plant model used in systems and control. In our theory, developed below, it is only necessary to have a mathematical description where the state trajectories are uniquely determined by the initial state and the input signals. For the purposes of this work we assume that $\mathbf{z} = \mathbf{x}$. Note that the plant input and output are continuous-time vector valued signals. Bold face letters are used to denote vectors and vector valued signals.

3.2 Controller

The controller is a discrete event system which is modeled as a deterministic automaton. This automaton can be specified by a quintuple, $\{\tilde{S}, \tilde{Z}, \tilde{R}, \delta, \phi\}$, where $\tilde{S}$ is the (possibly infinite) set of states, $\tilde{Z}$ is the set of plant symbols, $\tilde{R}$ is the set of controller symbols, $\delta : \tilde{S} \times \tilde{Z} \to \tilde{S}$ is the state transition function, and $\phi : \tilde{S} \to \tilde{R}$ is the output function. The symbols in set $\tilde{R}$ are called controller symbols because they are generated by the controller. Likewise, the symbols in set $\tilde{Z}$ are called plant symbols and are generated by the occurrence of events in the plant. The action of the controller can be described by the equations

$$\tilde{s}[n] = \delta(\tilde{s}[n-1], \tilde{z}[n]) \tag{3}$$

$$\tilde{r}[n] = \phi(\tilde{s}[n]) \tag{4}$$

where $\tilde{s}[n] \in S, \tilde{z}[n] \in \tilde{Z}$, and $\tilde{r}[n] \in \tilde{R}$. The index n is analogous to a time index in that it specifies the order of the symbols in a sequence. The input and output signals associated with the controller are asynchronous sequences of symbols, rather than continuous-time signals. Notice that there is no delay in the controller. The state transition, from $\tilde{s}[n-1]$ to $\tilde{s}[n]$, and the controller symbol, $\tilde{r}[n]$, occur immediately when the plant symbol $\tilde{z}[n]$ occurs.

Tildes are used to indicate that the particular set or signal is made up of symbols. For example, $\tilde{Z}$ is the set of plant symbols and $\tilde{z}[n]$ is a sequence of plant symbols. An argument in brackets, e.g. $\tilde{z}[n]$, represents the nth symbol in the sequence $\tilde{z}$. A subscript, e.g. $\tilde{z}_i$, is used to denote a particular symbol from a set.

3.3 Interface

The controller and plant cannot communicate directly in a hybrid control system because each utilizes a different type of signal. Thus an interface is required which can convert continuous-time signals to sequences of symbols and vice versa. The interface consists of two memoryless maps, γ and α. The first map, called the actuating function or actuator, $\gamma : \tilde{R} \rightarrow \Re^m$, converts a sequence of controller symbols to a piecewise constant plant input as follows

$$\mathbf{r}(t) = \gamma(\tilde{r}[n]) \tag{5}$$

The plant input, $\mathbf{r}$, can only take on certain constant values, where each value is associated with a particular controller symbol. Thus the plant input is a piecewise constant signal which may change only when a controller symbol occurs. The second map, the plant symbol generating function or generator, $\alpha : \Re^n \rightarrow \tilde{Z}$, is a function which maps the state space of the plant to the set of plant symbols as follows

$$\tilde{z}[n] = \alpha(\mathbf{x}(t)) \tag{6}$$

It would appear from Equation 6 that, as $\mathbf{x}$ changes, $\tilde{z}$ may continuously change. That is, there could be a continuous generation of plant symbols by the interface because each state is mapped to a symbol. This is not the case because α is based upon a partition of the state space where each region of the partition is associated with one plant symbol. These regions form the equivalence classes of α. A plant symbol is generated only when the plant state, $\mathbf{x}$, moves from one of these regions to another.

3.4 Comments on the Generality of the Model

The model described above may appear at first to be too limited but this is not the case. The simplicity of this model is its strength and it does not reduce its flexibility when modeling a hybrid control system. It is tempting to add complexity to the interface, however this typically leads to additional mathematical difficulties that are not necessary. Consider first the function γ which maps controller symbols to plant inputs. Our model features only constant plant inputs, no ramps, sinusoids, or feedback strategies. The reasons for this are two fold. First, in order for the interface to generate a nonconstant signal or feedback signal it must contain components which can be more appropriately included in the continuous time plant, as is done in the model above. Second, making the interface more complex will complicate the analysis of the overall system. Keeping the function γ as a simple mapping from each controller symbol to a unique numeric value is the solution.

The interface could also be made more complex by generalizing the definition of a plant symbol. A plant symbol is defined solely by the current plant state, but this could be expanded by defining a plant symbol as being generated following the occurrence of a specific series of conditions in the plant. For example, the interface could be made capable of generating a symbol which is dependent upon the current and previous values of the state. However, doing this entails including dynamics in the interface which actually belong in the controller. The controller, as a dynamic system, is capable of using its state as a memory to keep track of previous plant symbols.

The key feature of this hybrid control system model is its simple and unambiguous nature, especially with respect to the interface. To enable analysis, hybrid control systems must be described in a consistent and complete manner. Varying the nature of the interface from system to system in an ad hoc manner, or leaving its mathematical description vague causes difficulties.

3.5 Examples

Example 1 - Thermostat/Furnace System This example will show how an actual physical system can be modeled and how the parts of the physical system correspond to the parts found in the model. The particular hybrid control system in this example consists of a typical thermostat and furnace. Assuming the thermostat is set at 70 degrees Fahrenheit, the system behaves as follows. If the room temperature falls below 70 degrees the furnace starts and remains on until the room temperature exceeds 75 degrees. At 75 degrees the furnace shuts off. For simplicity, we will assume that when the furnace is on it produces a constant amount of heat per unit time.

The plant in the thermostat/furnace hybrid control system is made up of the furnace and room. It can be modeled with the following differential equation

$$\dot{\mathbf{x}} = .0042(T_0 - \mathbf{x}) + 2step(\mathbf{r}) \tag{7}$$

where the plant state, $\mathbf{x}$, is the temperature of the room in degrees Fahrenheit, the input, $\mathbf{r}$, is the voltage on the furnace control circuit, and T_0 is the outside temperature. The units for time are minutes. This model of the furnace is a simplification, but it is adequate for this example.

The remainder of the hybrid control system is found in the thermostat which is pictured in Figure 3. As the temperature of the room varies, the two strips of metal which form the bimetal band expand and contract at different rates thus causing the band to bend. As the band bends, it brings the steel closer to one side of the glass bulb. Inside the bulb, a magnet moves toward the nearest part of the steel and opens or closes the control circuit in the process. The bimetal band effectively partitions the state space of the plant, $\mathbf{x}$, as follows

$$\alpha(\mathbf{x}) = \begin{cases} \tilde{z}_1 \text{ if } \quad \mathbf{x} \leq 70 \\ \tilde{z}_2 \text{ if } 70 < \mathbf{x} \leq 75 \\ \tilde{z}_3 \text{ if } \quad \mathbf{x} > 75 \end{cases}, \tag{8}$$

where the three symbols correspond to 1) steel is moved against the left side of the bulb, 2) band is relaxed, and 3) steel is moved against the right side of the bulb.

Inside the glass bulb is a magnetic switch which is the DES controller. It has two states because the switch has two positions, on and off. The DES controller input, $\tilde{z}$, is a magnetic signal because the symbols generated by the generator are conveyed magnetically. The state transition graph of this simple controller is shown in Figure 4. The output function of the controller is essentially the following

$$\phi(\tilde{s}_1) = \tilde{r}_1 \Leftrightarrow \text{close control circuit} \tag{9}$$
$$\phi(\tilde{s}_2) = \tilde{r}_2 \Leftrightarrow \text{open control circuit} \tag{10}$$

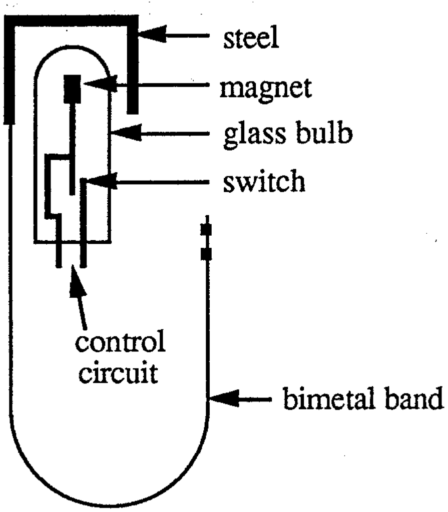

Fig. 3. Thermostat

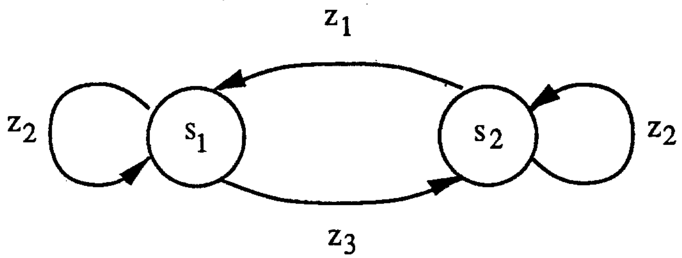

Fig. 4. Controller for Thermostat/Furnace System

The contacts on the switch which open and close the control circuit can be thought of as the actuator, although there is no logical place to separate the actuator from the DES controller. The commands from the controller to the actuator are basically a formality here because the controller and actuator are mechanically one piece. With this in mind, the actuator operates as

$$\gamma(\tilde{r}_1) = 0 \tag{11}$$

$$\gamma(\tilde{r}_2) = 24 \tag{12}$$

Example 2 - Surge Tank This is another example to illustrate how a simple hybrid control system can be modeled. The system consists of a surge tank which is draining through a fixed outlet valve, while the inlet valve is being controlled by a discrete event system. The controller allows the tank to drain to a minimum level and then opens the inlet valve to refill it. When the tank has reached a maximum level, the inlet valve is closed. The surge tank is modeled by a differential equation,

$$\dot{\mathbf{x}} = \mathbf{r} - \mathbf{x}^{1/2} \tag{13}$$

where $\mathbf{x}$ is the liquid level and $\mathbf{r}$ is the inlet flow. The interface partitions the state space into three regions as follows

$$\alpha(\mathbf{x}) = \begin{cases} \tilde{z}_1 \text{ if } & \mathbf{x} > max \\ \tilde{z}_2 \text{ if } & min < \mathbf{x} < max \\ \tilde{z}_3 \text{ if } & \mathbf{x} < min \end{cases}, \tag{14}$$

Thus when the level exceeds max, plant symbol $\tilde{z}_1$ is generated, and when the level falls below min, plant symbol $\tilde{z}_3$ is generated. The interface provides for two inputs corresponding to the two controller symbols $\tilde{r}_1$ and $\tilde{r}_2$ as follows

$$\gamma(\tilde{r}) = \begin{cases} 1 \text{ if } \tilde{r} = \tilde{r}_1 \\ 0 \text{ if } \tilde{r} = \tilde{r}_2 \end{cases}, \tag{15}$$

Since $\mathbf{r} = \gamma(\tilde{r})$, this means the inlet valve will be open following controller symbol $\tilde{r}_1$, and closed following controller symbol $\tilde{r}_2$.

The controller for the surge tank is a two state automaton which moves to state $\tilde{s}_1$ whenever $\tilde{z}_3$ is received, moves to state $\tilde{s}_2$ whenever $\tilde{z}_1$ is received and returns to the current state if $\tilde{z}_2$ is received. Furthermore $\phi(\tilde{s}_1) = \tilde{r}_1$ and $\phi(\tilde{s}_2) = \tilde{r}_2$.

4 System Theoretic Issues

4.1 The DES Plant Model

If the plant and interface of a hybrid control system are viewed as a single component, this component behaves like a discrete event system. It is advantageous to view a hybrid control system this way because it allows it to be modeled as two interacting discrete event systems which are more easily analyzed than the system in its original form. The discrete event system which models the plant and interface is called the DES Plant Model and is modeled as an automaton similar to the controller. The automaton is specified by a quintuple, $\{\tilde{P}, \tilde{Z}, \tilde{R}, \psi, \lambda\}$, where $\tilde{P}$ is the set of states, $\tilde{Z}$ and $\tilde{R}$ are the sets of plant symbols and controller symbols, $\psi : \tilde{P} \times \tilde{R} \to \tilde{P}$ is the state transition function, and $\lambda : \tilde{P} \to \tilde{Z}$ is the output function. The behavior of the DES plant is as follows

$$\tilde{p}[n+1] = \psi(\tilde{p}[n], \tilde{r}[n]) \tag{16}$$

$$\tilde{z}[n] = \lambda(\tilde{p}[n]) \tag{17}$$

where $\tilde{p}[n] \in \tilde{P}, \tilde{r}[n] \in \tilde{R}$, and $\tilde{z}[n] \in \tilde{Z}$. There are two differences between the DES plant model and the controller. First, as can be seen from Equation 16, the state transitions in the DES plant do not occur immediately when a controller symbol occurs. This is in contrast to the controller where state transitions occur immediately with the occurrence of a plant symbol. The second difference is that the automaton which models the DES plant may be non-deterministic, meaning $\tilde{p}[n+1]$ in Equation 16 is not determined exactly but rather is limited to some subset of $\tilde{P}$. The reason for these differences is that the DES plant model is a simplification of a continuous-time plant and an interface. This simplification results in a loss of information about the internal dynamics, leading to non-deterministic behavior.

The set of states, $\tilde{P}$, of the DES plant is based on the partition realized in the interface. Specifically, each state in $\tilde{P}$ corresponds to a region, in the state space of the continuous-time plant, which is equivalent under α. Thus there is a one-to-one correspondence between the set of states, $\tilde{P}$, and the set of plant symbols, $\tilde{Z}$. It is this relationship between the states of the DES plant model and the plant symbols which forms the basis for the work described in this section. It can be used to develop an expression for the state transition function, ψ. Starting with the continuous-time plant, we integrate Equation 1 to get the state after a time t, under constant input $\mathbf{r} = \gamma(\tilde{r}_k)$

$$\mathbf{x}(t) = F_k(\mathbf{x}_0, t) \tag{18}$$

Here $\mathbf{x}_0$ is the initial state, t is the elapsed time, and $\tilde{r}_k \in \tilde{R}$. $F_k(\mathbf{x}_0, t)$ is obtained by integrating $f(\mathbf{x}, \mathbf{r})$, with $\mathbf{r} = \gamma(\tilde{r}_k)$. Next we define

$$\hat{F}_k(\mathbf{x}_0) = F_k(\mathbf{x}_0, t), \tag{19}$$

where

$$\tau_0 = \inf_{\tau}\{\tau | \alpha(F(\mathbf{x}_0, \tau)) \neq \alpha(\mathbf{x}_0)\} \tag{20}$$

and

$$t = \tau_0 + \epsilon \tag{21}$$

for some infinitesimally small ϵ.

Equation 19 gives the state, $\mathbf{x}$, where it will cross into a new region. Now the dynamics of the DES plant model can be derived from Equations 5, 6, 19.

$$\tilde{z}[n+1] = \lambda(\psi(\tilde{p}[n], \tilde{r}[n])) \tag{22}$$

$$\tilde{z}[n+1] = \alpha(\hat{F}_k(\mathbf{x}_0)) \tag{23}$$

$$\psi(\tilde{p}[n], \tilde{r}[n]) = \lambda^{-1}(\alpha(\hat{F}_k(\mathbf{x}_0))) \tag{24}$$

where $\tilde{r}[n] = \tilde{r}_k$ and $\mathbf{x}_0 \in \{\mathbf{x} | \alpha(\mathbf{x}) = \lambda(\tilde{p}[n])\}$. As can be seen, the only uncertainty in Equation 24 is the value of $\mathbf{x}_0$. $\mathbf{x}_0$ is the state of the continuous-time plant at the time of the last plant symbol, $\tilde{z}[n]$, i.e. the time that the DES plant entered state $\tilde{p}[n]$. $\mathbf{x}_0$ is only known to within an equivalence class of α. The condition for a deterministic DES plant is that the state transition function, ψ, must be uneffected to this uncertainty.

Definition 1. A DES is *deterministic* iff for any state and any input, there is only one possible subsequent state.

The following theorem gives the conditions upon the hybrid control system such that the DES plant will be deterministic.

Theorem 2. *The DES plant will be deterministic iff given any $\tilde{p}[n] \in P$ and $\tilde{r}_k \in R$, there exists $\tilde{p}[n+1] \in P$ such that for every $\mathbf{x}_0 \in \{\mathbf{x} | \alpha(\mathbf{x}) = \lambda(\tilde{p}[n])\}$ we have $\alpha(\hat{F}_k(\mathbf{x}_0)) = \lambda(\tilde{p}[n+1])$.*

Proof: Notice that the set $\{\mathbf{x} | \alpha(\mathbf{x}) = \lambda(\tilde{p}[n])\}$ represents the set of all states, $\mathbf{x}$, in the continuous-time plant which could give rise to the state $\tilde{p}[n]$ in the DES plant. The theorem guarantees that the subsequent DES plant state, $\tilde{p}[n+1]$, is unique for a given input and thus the DES plant is deterministic.

To prove that the theorem is necessary, assume that it does not hold. There must then exist a $\tilde{p}[n] \in \tilde{P}$ and $\tilde{r}_k \in \tilde{R}$ such that no $\tilde{p}[n+1]$ exists to satisfy the condition: $\alpha(\hat{F}_k(\mathrm{x}_0)) = \lambda(\tilde{p}[n+1])$ for every $\mathrm{x}_0 \in \{\mathrm{x}|\alpha(\mathrm{x}) = \lambda(\tilde{p}[n])\}$. This is not a deterministic system because there is uncertainty in the state transition for at least one state and input. □

Theorem 2 states that the DES plant will be deterministic if all the state trajectories in the continuous-time plant, which start in the same region and are driven by the same input, move to the same subsequent region.

4.2 Double Integrator

To illustrate the DES plant model, an example of a hybrid control system containing a double integrator is given. Double integrators often arise in systems. For example, a satellite equipped with a thruster will behave as a double integrator when the thrust is considered the input and the velocity and position are the two states.

$$\dot{\mathrm{x}} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \mathrm{x} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \mathrm{r} \tag{25}$$

The general control goal in this system, which motivates the design of the interface, is to move the state of the double integrator between the four quadrants of the state-space. In the interface, the function a partitions the state space into four regions as follows,

$$\alpha(\mathrm{x}) = \begin{cases} \tilde{z}_1 & \text{if} \quad x_1, x_2 > 0 \\ \tilde{z}_2 & \text{if} \quad x_1 < 0, x_2 > 0 \\ \tilde{z}_3 & \text{if} \quad x_1, x_2 < 0 \\ \tilde{z}_4 & \text{if} \quad x_1 > 0, x_2 < 0 \end{cases}, \tag{26}$$

and the function γ provides three set points,

$$\gamma(\tilde{r}) = \begin{cases} -10 & \text{if } \tilde{r} = \tilde{r}_1 \\ 0 & \text{if } \tilde{r} = \tilde{r}_2 \\ 10 & \text{if } \tilde{r} = \tilde{r}_3 \end{cases}, \tag{27}$$

So whenever the state of the double integrator enters quadrant 1, for example, the plant symbol $\tilde{z}_1$ is generated. When the controller (which is unspecified) generates controller symbol $\tilde{r}_1$, the double integrator is driven with an input of -10.

Now we know that the DES plant will have four states because there are four regions in the state space of the actual plant. By examining the various state trajectories given by Equation 28, we can find the DES plant which is shown in Figure 5. Equation 28 is obtained by integrating Equation 25 and adding $\mathbf{x}(0)$.

$$\mathrm{x} = \begin{bmatrix} 1 & t \\ 0 & 1 \end{bmatrix} \mathrm{x}_0 + \begin{bmatrix} .5t^2 \\ t \end{bmatrix} \gamma(\tilde{r}) \tag{28}$$

As can be seen in Figure 5, the DES plant is not deterministic. If we consider $\tilde{p}[n] = \tilde{p}_2$ and $\tilde{r}[n] = \tilde{r}_1$, there exists no uniquely defined $\tilde{p}[n+1]$, it could be either $\tilde{p}_1$ or $\tilde{p}_3$. This could present a problem in designing a controller for this system because it is not entirely predictable. In the following section a possible remedy for lack of determinism is presented and this example is revisited.

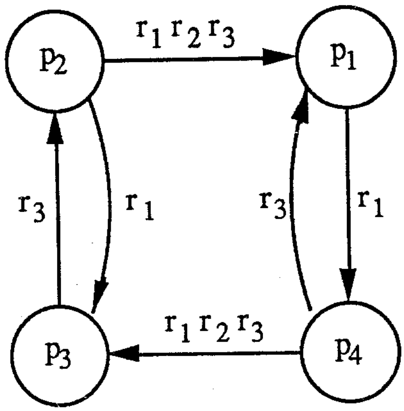

Fig. 5. DES Plant Model for Double Integrator

4.3 Partitioning and Quasideterminism

A particular problem in the design of a hybrid control system is the selection of the function α, which partitions the state-space of the plant into various regions. Since this partition is used to generate the plant symbols, it must be chosen to provide sufficient information to the controller to allow control without being so fine that it leads to an unmanageably complex system or simply degenerates the system into an essentially conventional control system.

The partition must accomplish two goals. First it must give the controller sufficient information to determine whether or not the current state is in an acceptable region. For example, in an aircraft these regions may correspond to climbing, diving, turning right, etc. Second, the partition must provide enough additional information about the state, to enable the controller to drive the plant to an acceptable region. In an aircraft, for instance, the input required to cause the plane to climb may vary depending on the current state of the plane. So to summarize, the partition must be detailed enough to answer: 1) is the current state acceptable; and 2) which input can be applied to drive the state to an acceptable region.

In a hybrid control system, the controller needs information about the plant for two reasons. First the controller must be able to assess whether the plant is operating as desired or if some new control action is needed. Second, if control action is called for, the controller needs to know which control action will achieve the desired effect. Both of these tasks require information about the plant. Consider for example a climate control system in a building. To assess the current condition, the controller needs to know whether the temperature and humidity fall within a certain range of acceptable values. If not the controller needs additional, more detailed, information about which condition is unacceptable and how much and in which direction it must be changed to reach the desired range.

To design a partition, we can start by designing a primary partition to meet the first goal mentioned above. This primary partition will identify all the desired operating regions of the plant state space, so its design will be dictated by the control goals. The final partition will represent a refinement of the primary partition which enables the controller to regulate the plant to any of the desired operating regions, thus meeting the second goal.

An obvious choice for the final partition is one which makes the DES plant deterministic and therefore guarantees that the controller will have full information about the behavior of the plant. In addition to being very hard to meet, this requirement is overly strict because the controller only needs to regulate the plant to the regions in the primary partition, not the final partition. For this reason we define quasideterminism, a weaker form of determinism. In the DES plant, the states which are in the same region of the primary partition can be grouped together, and if the DES plant is deterministic with respect to these groups, then we say it is quasideterministic. So if the DES plant is quasideterministic, then we may not be able to predict the next state exactly, but we will be able to predict its region of the primary partition and thus whether or not it is acceptable.

Definition 3. The DES plant will be quasideterministic iff given any $\tilde{p}[n] \in \tilde{P}$ and $\tilde{r}_k \in \tilde{R}$, there exists $\tilde{Q} \subset \tilde{P}$ such that for every $\mathbf{x}_0 \in \{\mathbf{x} | \alpha(\mathbf{x}) = \lambda(\tilde{p}[n])\}$ we have $\alpha_p(\hat{F}_k(\mathbf{x}_0)) = \lambda_p(\tilde{p}[n+1])$ where $\tilde{p}[n+1] \in \tilde{Q}$ and $\lambda_p(\tilde{q})$ is the same for all $\tilde{q} \in \tilde{Q}$. □

The functions α_p and λ_p are analogous to α and λ but apply to the primary partition. They are useful for comparing states but they are never implemented and their actual values are irrelevant. For example, if $\alpha_p(\mathbf{x}_1) = \alpha_p(\mathbf{x}_2)$, then $\mathbf{x}_1$ and $\mathbf{x}_2$ are in the same region of the primary partition. Or, if $\alpha_p(\mathbf{x}_1) = \lambda_p(\tilde{p}_1)$, then $\mathbf{x}_1$ is in the same region of the primary partition as $\tilde{p}_1$ in the DES plant. When used with α_p we define $\hat{F}$ as

$$\hat{F}_k(\mathbf{x}_0) = F_k(\mathbf{x}_0, t), \tag{29}$$

where

$$\tau_0 = \inf_{\tau}\{\tau | \alpha_p(F(\mathbf{x}_0, \tau)) \neq \alpha_p(\mathbf{x}_0)\} \tag{30}$$

and

$$t = \tau_0 + \epsilon \tag{31}$$

as before.

We would like to find the coarsest partition which meets the conditions of Definition 1 for a given primary partition. Such a partition is formed when the equivalence classes of α are as follows,

$$E[\alpha] = \inf\{E[\alpha_p], E[\alpha_p \circ \hat{F}_k] | \tilde{r}_k \in R\} \tag{32}$$

Where we use $E[\bullet]$ to denote the equivalence classes of $\bullet$. The infimum, in this case, means the coarsest partition which is at least as fine as any of the partitions in the set.

Theorem 4. *The regions described by Equation (32) form the coarsest partition which generates a quasideterministic DES plant.*

Proof: First we will prove that the partition does, in fact, lead to a quasideterministic system. For any two states, $\mathbf{x}_1$ and $\mathbf{x}_2$, which are in the same equivalence class of α, we apply some control $\mathbf{r} = \gamma(\tilde{r}_k)$. The two states will subsequently enter new regions of the primary partition at $\hat{F}_k(\mathbf{x}_1)$ and $\hat{F}_k(\mathbf{x}_2)$ respectively. The actual regions entered are $\alpha_p(\hat{F}_k(\mathbf{x}_1))$ and $\alpha_p(\hat{F}_k(\mathbf{x}_2))$. Now according to Equation 32, if $\mathbf{x}_1$ and $\mathbf{x}_2$ are in the same equivalence class of α, then they are also in the same equivalence class of $\alpha_p \circ \hat{F}_k$. Therefore $\alpha_p(\hat{F}_k(\mathbf{x}_1)) = \alpha_p(\hat{F}_k(\mathbf{x}_2))$ and the system is quasideterministic.

Next we will prove that the partition is as coarse as possible. Assume there is a coarser partition which also generates a quasideterministic system. That is, there exists two states, $\mathbf{x}_3$ and $\mathbf{x}_4$, in the same region of the primary partition such that $\alpha(\mathbf{x}_3) \neq \alpha(\mathbf{x}_4)$, but $\alpha_p(\hat{F}_k(\mathbf{x}_3)) = \alpha_p(\hat{F}_k(\mathbf{x}_4))$ for *any* possible k. These two states would lie in the same equivalence class of $\alpha_p \circ \hat{F}_k$ for all $\tilde{r}_k \in \tilde{R}$ and therefore in the same equivalence class of $\inf\{E[\alpha_p], E[\alpha_p \circ \hat{F}_k | \tilde{r}_k \in \tilde{R}\}$. This violates the assumption that $\mathbf{x}_3$ and $\mathbf{x}_4$ do not lie in the same equivalence class of α, so two such states could not exist and therefore a coarser partition can not exist. □

Quasideterminism accomplishes its goal by causing the trajectories of the various states within a given region of the final partition, under the same control, to be invariant with respect to the regions of the primary partition.

We can return now to the double integrator discussed previously and use it to illustrate quasideterminism. The state space of the double integrator had been partitioned into the four quadrants and this gave rise to the nondeterministic DES plant shown in Figure 5. Using those four regions as the primary partition, a final partition can be obtained according to Theorem 4. This partition is shown in Figure 6 and the resulting DES plant is shown in Figure 7. The final partition refined the regions in quadrants II and IV, and the DES plant is now quasideterministic (in fact it is deterministic but unfortunately that is not generally the result).

Note that the partition described in Equation 32 and discussed in Theorem 4 is not dependent upon any specific sequence of controller symbols. It is intended to yield a DES plant which is as "controllable" as possible, given the continuous-time plant and available inputs. If the specific control goals are known, it may be possible to derive a coarser partition which is still adequate. This can be done in an ad hoc fashion, for instance, by combining regions which are equivalent under the inputs which are anticipated when the plant is in those regions.

Selection of Control Action In hybrid control systems, the choice of the plant inputs which make up the range of the actuator, γ, play an important role in defining the system. At this time we have no way of systematically deriving a set of control actions which will achieve the desired control goals, either optimally or otherwise. We can assume that the control actions available are determined

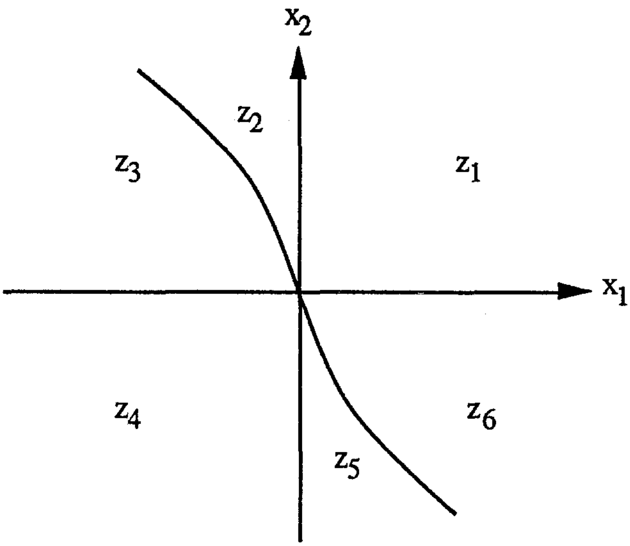

Fig. 6. State Space Partition for Double Integrator

by the plant (positions of the various valves, switches, etc.) and thus represent a constraint on the controller design.

4.4 Connections to Existing DES Control Theory

A significant amount of work has been done on the analysis and design of discrete event systems, especially the design of controllers for discrete event systems. Since the controller of a hybrid control system is a DES and we can use a DES to represent the plant in a hybrid control system, we can apply many of the theories and techniques, which were developed for DES's, to hybrid control systems. In this section, we draw on some of this work.

Stability: Several papers have been written dealing with the stability of discrete event systems, e.g. [Passino 1992a] and [Ozveren 1991]. In [Passino 1992a] the ideas of Lyapunov stability are applied to discrete event systems. These same techniques can be applied to the DES plant in a hybrid system. The states of the DES plant which are considered "desirable" are identified and a metric is defined on the remaining "undesirable" states. With a metric defined on the state space, finding a Lyapunov function will prove that the DES is stable. In the case of a hybrid control system, this interpretation of definition Lyapunov stability means the following. The state of the plant will remain within the set of states which were deemed "desirable" and if it is perturbed from this area, the state will return to it. A detailed application of these results to hybrid control systems can be found in [Stiver 1991b].

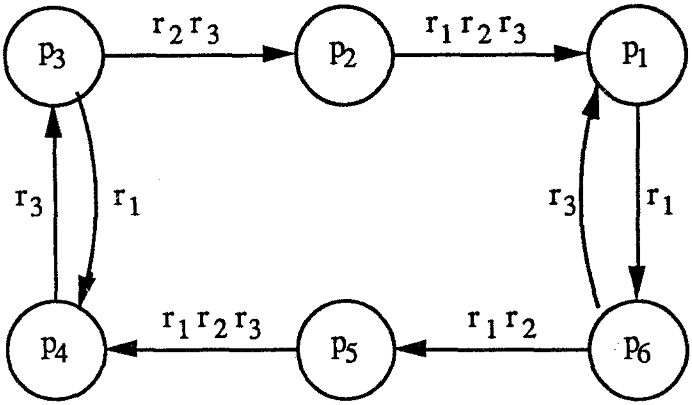

Fig. 7. DES Plant Model for Double Integrator

In [Ozveren 1991] the stability of a DES is defined as the property that the state of the DES will visit a certain subset infinitely often. This subset is analogous to the "desirable" set mention above. In a hybrid control system, this would imply that the state of the plant could leave the subset of states but would eventually return.

Controllability: Work has been done on the controllability of discrete event systems using the Ramadge-Wonham framework [Ramadge 1987] [Ramadge 1989] [Wonham 1987]. The DES models used in the Ramadge-Wonham framework differ from the models developed for hybrid control systems as described in this chapter, therefore the theorems and techniques cannot be applied directly, but must be adapted to work with a slightly different model.

The model developed by Ramadge and Wonham (henceforth RWM) features a generator and a supervisor, both DES's, which are analogous to the DES plant model and DES controller, respectively. There are, however, several differences which must be addressed first.

In the generator, the state transitions are divided into two sets, those which are controllable and those which are uncontrollable. The controllable state transitions, or symbols, can be individually enabled by a command from the supervisor, while the uncontrollable state transitions are always enabled. Also, the effect of the supervisor commands is not changed by the state of the generator. This is in contrast to our DES plant model where commands from the DES controller can enable one or more state transitions depending on the current state. The general inability to enable symbols individually and the dependence of DES controller commands upon the state of the DES plant model, are what differentiate the DES models used our work on hybrid control systems from the RWM.

The reason for the differences between the RWM and the model used for hybrid control systems is chiefly due to the fact that the RWM is suited to modeling actual discrete event systems, while the DES plant model is an abstraction of a continuous-time system. This means that a particular state of the DES plant corresponds to more than one state in the continuous-time plant.

Controllability in a DES can be characterized by the set of symbol sequences which can be made to occur in the DES plant, [Ramadge 1989]. This set is referred to as the language of the particular DES. When under control, the DES will exhibit behavior which lies in a subset of its language. A theorem has been developed to determine whether a given RWM DES can be controlled to a desired language and if not, what is the greatest portion of the desired language which can be achieved via control. With appropriate modifications this theorem can be applied to the DES plant to determine whether a given control goal is possible.

If a desired behavior (i.e. language) is not attainable for a given controlled DES, it may be possible to find a more restricted behavior which is. If so, the least restricted behavior is desirable. [Wonham 1987] provides a method for finding this behavior which is referred to as the *supremal sublanguage* of the desired language.

When a controllable language has been found for a DES plant, designing a controller is straight-forward. The controller will be another DES which produces the desired controllable language. The output from the controller enables only the symbols which are in the controller. The exact form of the above results together with their proofs are not presented here due to space limitations; they are available from the authors.

5 Concluding Remarks

This chapter has introduced a model for hybrid systems which has focused on the role of the interface between the continuous-state plant and discrete-event supervisor. An especially simple form of the interface was introduced in which symbolic events and nonsymbolic state/control vectors are related to each other via memoryless transformations. It was seen that this particular choice dichotomizes the symbolic and nonsymbolic parts of the hybrid system into two cleanly separated dynamical systems which clearly expose the relationship between plant and supervisor. With the use of the proposed interface, quasi-determinism can be used to extend controllability concepts to hybrid systems. The clear separation of symbolic and nonsymbolic domains allows the formulation of hybrid controller methodologies which are directly based on equivalent DES control methods. Finally, the acknowledgement of the different roles played by symbolic and nonsymbolic processing in hybrid systems allows the proper formulation of the hybrid system's identification problem found in the companion chapter in this volume titled "Event Identification and Intelligent Hybrid Control" [Lemmon 1993c].

The work outlined in the preceding sections is indicative of the breadth of work currently being pursued in the area of hybrid systems as a means of modeling and designing supervisory and intelligent control systems. In spite of the great strides being made in this area, there are significant issues which remain to be addressed in future work. These issues include a more rigorous examination of the traditional control concepts of controllability, observability, and stability with regard to hybrid systems. To some extent, the notions of quasi-determinism and the event identification problems are preliminary efforts to codify these extensions. Future work, however, remains before these extensions are fully understood.

References

[Acar 1990] L. Acar, U. Ozguner, "Design of Knowledge-Rich Hierarchical Controllers for Large Functional Systems", *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. 20, No. 4, pp. 791-803, July/Aug. 1990.

[Albus 1981] J. Albus, et al, "Theory and Practice of Intelligent Control", *Proc. 23rd IEEE COMPCON*, pp 19-39, 1981.

[Antsaklis 1989] P. J. Antsaklis, K. M. Passino S. J. and Wang, "Towards Intelligent Autonomous Control Systems: Architecture and Fundamental Issues", *Journal of Intelligent and Robotic Systems*, Vol.1, pp. 315-342, 1989.

[Antsaklis 1990, 1992] P. J. Antsaklis, Special Issues on 'Neural Networks in Control Systems' of the IEEE Control Systems Magazine, April 1990 and April 1992.

[Antsaklis 1991] P. J. Antsaklis, K. M. Passino and S. J. Wang, "An Introduction to Autonomous Control Systems", *IEEE Control Systems Magazine*, Vol. 11, No. 4, pp.5-13, June 1991.

[Antsaklis 1993a] P. J. Antsaklis and K. M. Passino, Eds., *An Introduction to Intelligent and Autonomous Control*, 448 p., Kluwer Academic Publishers, 1993.

[Antsaklis 1993b] P. J. Antsaklis and K. M. Passino, "Introduction to Intelligent Control Systems with High Degree of Autonomy", *An Introduction to Intelligent and Autonomous Control*, P. J. Antsaklis and K. M. Passino, Eds., Chapter 1, pp. 1-26, Kluwer Academic Publishers,1993.

[Antsaklis 1993c] P. J. Antsaklis, "Neural Networks for the Intelligent Control of High Autonomy Systems", *Mathematical Studies of Neural Networks*, J.G. Taylor, Ed., Elsevier, 1993. To appear.

[Astrom 1986] K. J. Astrom, et al, "Expert Control", *Automatica*, Vol. 22, No. 3, pp. 277-286, 1986.

[Benveniste 1990] A. Benveniste, P. Le Guernic, "Hybrid Dynamical Systems and the SIGNAL Language", *IEEE Transactions on Automatic Control*, Vol. 35, No. 5, pp. 535-546, May 1990.

[Cassandras 1990] C. Cassandras, P. Ramadge, "Toward a Control Theory for Discrete Event Systems", *IEEE Control Systems Magazine*, pp. 66-68, June 1990.

[Fishwick 1991] P. Fishwick, B. Zeigler, "Creating Qualitative and Combined Models with Discrete Events", *Proceedings of The 2nd Annual Conference on AI, Simulation and Planning in High Autonomy Systems*, pp. 306-315, Cocoa Beach, FL, April 1991.

[Gollu 1989] A. Gollu, P. Varaiya, "Hybrid Dynamical Systems", *Proceedings of the 28th Conference on Decision and Control*, pp. 2708-2712, Tampa, FL, December 1989.

[Grossman 1992] R. Grossman, R. Larson, "Viewing Hybrid Systems as Products of Control Systems and Automata", *Proceedings of the 31st Conference on Decision and Control*, pp. 2953-2955, Tucson AZ, December 1992.

[Holloway 1992] L. Holloway, B. Krogh, "Properties of Behavioral Models for a Class of Hybrid Dynamical Systems", *Proceedings of the 31st Conference on Decision and Control*, pp. 3752-3757, Tucson AZ, December 1992.

[IEEE Computer 1989] Special Issue on Autonomous Intelligent Machines, IEEE Computer, Vol. 22, No. 6, June 1989.

[Isidori 1989] A. Isidori, *Nonlinear Control Systems*, 2nd Edition, Springer-Verlag, Berlin, 1989

[Kohn 1992] W. Kohn, A. Nerode, "Multiple Agent Autonomous Hybrid Control Systems", *Proceedings of the 31st Conference on Decision and Control*, pp. 2956-2966, Tucson AZ, December 1992.

[Lemmon 1992] M. D. Lemmon, "Ellipsoidal Methods for the Estimation of Sliding Mode Domains of Variable Structure Systems", *Proc. of 26th Annual Conference on Information Sciences and Systems*, Princeton N.J., March 18-20, 1992.

[Lemmon 1993a] M. D. Lemmon, "Inductive Inference of Invariant Subspaces", *Proceedings of the American Control Conference*, San Francisco, California, June 2-4, 1993.

[Lemmon 1993b] M. D. Lemmon, J. A. Stiver, P. J. Antsaklis, "Learning to Coordinate Control Policies of Hybrid Systems", *Proceedings of the American Control Conference*, San Francisco, California, June 2-4, 1993.

[Lemmon 1993c] M. D. Lemmon, J. A. Stiver, P. J. Antsaklis, "Event Identification and Intelligent Hybrid Control", This volume.

[Mendel 1968] J. Mendel and J. Zapalac, "The Application of Techniques of Artificial Intelligence to Control System Design", in *Advances in Control Systems*, C.T. Leondes, ed., Academic Press, NY, 1968.

[Mesarovic 1970] M. Mesarovic, D. Macko and Y. Takahara, *Theory of Hierarchical, Multilevel, Systems*, Academic Press, 1970.

[Narendra 1990] K. S. Narendra and K. Parthsarathy, "Identification and Control of Dynamical Systems Using Neural Networks", *IEEE Trans. Neural Networks*, Vol 1(1):4-27.

[Nerode 1992] A. Nerode, W. Kohn, "Models for Hybrid Systems: Automata, Topologies, Stability", Private Communication, November 1992.

[Ozveren 1991] C. M. Ozveren, A. S. Willsky and P. J. Antsaklis, "Stability and Stabilizabilty of Discrete Event Dynamic Systems", *Journal of the Association of Computing Machinery*, Vol 38, No 3, pp 730-752, 1991.

[Passino 1989a] K. Passino, "Analysis and Synthesis of Discrete Event Regulator Systems", Ph. D. Dissertation, Dept. of Electrical and Computer Engineering, Univ. of Notre Dame, Notre Dame, IN, April 1989.

[Passino 1989b] K. M. Passino and P. J. Antsaklis, "On the Optimal Control of Discrete Event Systems" , *Proc. of the 28th IEEE Conf. on Decision and Control*, pp. 2713-2718, Tampa, FL, Dec. 13-15, 1989.

[Passino 1991a] K. M. Passino, A. N. Michel, P. J. Antsaklis, "Lyapunov Stability of a Class of Discrete Event Systems", *Proceedings of the American Control Conference*, Boston MA, June 1991.

[Passino 1991b] K. M. Passino, U. Ozguner, 'Modeling and Analysis of Hybrid Systems: Examples", *Proc. of the 1991 IEEE Int. Symp. on Intelligent Control*, pp. 251-256, Arlington, VA, Aug. 1991.

[Passino 1992a] K. M. Passino, A. N. Michel and P. J. Antsaklis, "Ustojchivost' po Ljapunovu klassa sistem diskretnyx sobytij", *Avtomatika i Telemekhanika*, No.8, pp. 3-18, 1992. "Lyapunov Stability of a Class of Discrete Event Systems", *Journal of Automation and Remote Control*, No.8, pp. 3-18, 1992. In Russian.

[Passino 1992b] K. M. Passino and P. J. Antsaklis, "Event Rates and Aggregation in Hierarchical Discrete Event Systems", *Journal of Discrete Event Dynamic Systems*, Vol.1, No.3, pp. 271-288, January 1992.

[Passino 1993] K. M. Passino and P. J. Antsaklis, ""Modeling and Analysis of Artificially Intelligent Planning Systems", *Introduction to Intelligent and Autonomous Control*, P.J.Antsaklis and K.M.Passino, Eds., Chapter 8, pp. 191-214, Kluwer, 1993.

[Peleties 1988] P. Peleties, R. DeCarlo, "Modeling of Interacting Continuous Time and Discrete Event Systems : An Example", *Proceedings of the 26th Annual Allerton Conference on Communication, Control, and Computing*, pp. 1150-1159, Univ. of Illinois at Urbana-Champaign, October 1988.

[Peleties 1989] P. Peleties, R. DeCarlo, "A Modeling Strategy with Event Structures for Hybrid Systems", *Proceedings of the 28th Conference on Decision and Control*, pp.1308-1313, Tampa FL, December 1989.

[Peterson 1981] J. L. Peterson, *Petri Net Theory and the Modeling of Systems*, Prentice-Hall, Englewood Cliffs, NJ, 1981.

[Ramadge 1987] P. Ramadge, W. M. Wonham, "Supervisory Control of a Class of Discrete Event Processes", *SIAM Journal of Control and Optimization*, vol. 25, no. 1, pp. 206-230, Jan 1987.

[Ramadge 1989] P. Ramadge, W. M. Wonham, "The Control of Discrete Event Systems", *Proceedings of the IEEE*, Vol. 77, No. 1, pp. 81 - 98, January 1989.

[Saridis 1979] G. N. Saridis, "Toward the Realization of Intelligent Controls", *Proc. of the IEEE*, Vol. 67, No. 8, pp. 1115-1133, August 1979.

[Saridis 1983] G. N. Saridis, "Intelligent Robot Control", *IEEE Trans. on Automatic Control*, Vol. AC-28, No. 5, pp. 547-556, May 1983.

[Saridis 1985] G. N. Saridis, "Foundations of the Theory of Intelligent Controls", *Proc. IEEE Workshop on Intelligent Control*, pp 23-28, 1985.

[Saridis 1987] G. N. Saridis, "Knowledge Implementation: Structures of Intelligent Control Systems", *Proc. IEEE International Symposium on Intelligent Control*, pp. 9-17, 1987.

[Saridis 1989a] G. N. Saridis, "Analytic Formulation of the Principle of Increasing Precision with Decreasing Intelligence for Intelligent Machines", *Automatica*, Vol.25, No.3, pp. 461-467, 1989.

[Stengel 1984] R. F. Stengel, "AI Theory and Reconfigurable Flight Control Systems", Princeton University Report 1664-MAE, June 1984.

[Stiver 1991a] J. A. Stiver, "Modeling of Hybrid Control Systems Using Discrete Event System Models", M.S. Thesis, Dept. of Electrical Engineering, Univ. of Notre Dame, Notre Dame, IN, May 1991.

[Stiver 1991b] J. A. Stiver, P. J. Antsaklis, "A Novel Discrete Event System Approach to Modeling and Analysis of Hybrid Control Systems", *Control Systems Technical Report #71*, Dept. of Electrical Engineering, University of Notre Dame, Notre Dame, IN, June 1991.

[Stiver 1991c] J. A. Stiver, P. J. Antsaklis, "A Novel Discrete Event System Approach to Modeling and Analysis of Hybrid Control Systems", *Proceedings of the Twenty-Ninth Annual Allerton Conference on Communication, Control, and Computing*, University of Illinois at Urbana-Champaign, Oct. 2-4, 1991.

[Stiver 1992] J. A. Stiver, P. J. Antsaklis, "Modeling and Analysis of Hybrid Control Systems", *Proceedins of the 31st Conference on Decision and Control*, pp. 3748-3751, Tucson AZ, December 1992.

[Stiver 1993] J. A. Stiver, P. J. Antsaklis, "State Space Partitioning for Hybrid Control Systems", *Proceedings of the American Control Conference*, San Francisco, California, June 2-4, 1993.

[Turner 1984] P. R. Turner, et al, "Autonomous Systems: Architecture and Implementation", Jet Propulsion Laboratories, Report No. JPL D- 1656, August 1984.

[Valavanis 1986] K. P. Valavanis, "A Mathematical Formulation For The Analytical Design of Intelligent Machines", PhD Dissertation, Electrical and Computer Engineering Dept., Rensselaer Polytechnic Institute, Troy NY, Nov. 1986.

[Wonham 1987] W. M. Wonham, P. J. Wonham, "On the Supremal Controllable Sublanguage of a Given Language", *SIAM Journal of Control and Optimization*, vol. 25, no. 3, pp. 637-659, May 1987.

[Zeigler 1984] B. P. Zeigler, "Multifacetted Modelling and Discrete Event Simulation", Acedemic Press, NY, 1984.

[Zeigler 1987] B. P. Zeigler, 'Knowledge Representation from Newton to Minsky and Beyond", *Journal of Applied Artificial Intelligence*, 1:87-107, 1987.

[Zeigler 1989] B. P. Zeigler, "DEVS Representation of Dynamical Systems: Event Based Intelligent Control", *Proc. of the IEEE*, Vol. 77, No. 1, pp. 72-80, 1989.