A Logical DES Approach to the Design of Hybrid Control Systems

James A. Stiver, Panos J. Antsaklis, and Michael D. Lemmon
Department of Electrical Engineering
University of Notre Dame, Notre Dame, IN 46556

Abstract

Hybrid control systems, that is, systems which contain both continuous-time and discrete event dynamics are studied in this paper. First, a model is introduced that specifies the interface which connects the continuous plant dynamics with the discrete event controller. A Discrete Event System (DES) automaton description is then employed to describe the plant together with the interface and it is used to analyze the hybrid control system. Controllability is defined for hybrid control systems, enhancing existing DES control concepts. It is then used to obtain a controller design method for hybrid control systems. Finally, additional theoretic concepts such as determinism and quasideterminism that help define the interface so that the hybrid system has desired properties are introduced.

1 Introduction

The hybrid control systems considered here consist of a continuous-time system, called the plant, which is being supervised by a discrete event system, called the controller. The plant has a continuous state space with the states taking values from the real numbers. The controller, on the other hand, has a discrete, symbolic state space and is represented as a finite state automaton. A hybrid control system also contains an interface which provides a means for communication between the plant and controller via two subsystems named here the generator and the actuator. The generator converts the continuous-time output of the plant into an asynchronous sequence of symbols, understandable by the controller. The actuator translates the symbolic commands of the controller into a continuous-time input for the plant.

A simple example of a hybrid control system is the heating and cooling system of a typical home. The furnace and air conditioner, along with the heat flow characteristics of the home, form a continuous-time system which is to be controlled. The thermostat is a simple discrete event system which basically handles the symbols *too hot, too cold,* and *ok.* The temperature of the room is translated into these representations in the thermostat and the thermostat's response is translated back to electrical currents which control the furnace, air conditioner, blower, etc. Additional applications of hybrid control systems include flexible manufacturing systems and chemical process controls, among others. Note that hybrid control is an integral part of the *intelligent control* of dynamical systems [1].

Recently, efforts have been made to study hybrid systems in a unified, analytical way, [2–12]. In [2], hybrid systems are modeled using a language called SIGNAL in which discrete-time signals and events are synchronized to the fastest clock. In [3] a modeling strategy for hybrid systems is suggested and used to model a computer hard drive. [4] describes a model for a class of systems containing both continuous and discrete dynamics and shows that this class of models display causality and time monotonicity, which are essential for these models to be used in on-line monitoring. In [7] a technique is discussed which learns a set of control policies, used in the hybrid system, to control the continuous-time plant.

In [5] and [6], the development of a formal model for hybrid systems is presented. The model provides a framework to represent the interaction between the continuous and discrete portions of the system. [9] presents a hybrid system model in which the discrete dynamics are modeled by a Petri net and special attention is given to describing events and event structures. Finally, [10–12] contain background as well as some of the results reported in this work.

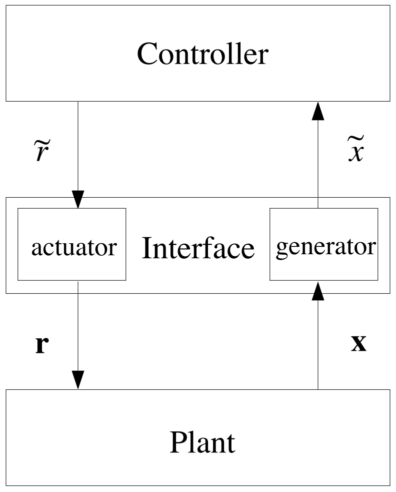


Figure 1: Hybrid Control System

In this paper, a clear methodology for analysis and controller design of hybrid control systems is presented. It is based on a mathematical model which is both general enough to represent a wide range of hybrid systems and simple enough to allow analysis and controller design. Related to this work is the work by Ramadge, *et al* [13,14], Nerode, *et al* [6,15], and DeCarlo, *et al* [9,16].

In section 2 of this paper, we present a mathematical model for hybrid control systems. We then show in section 3, how an entire hybrid control system can be represented as a pair of interacting discrete event systems, thus permitting the use of DES theoretic results in the analysis and design of hybrid control systems. Examples illustrate the concepts in section 4. In section 5, DES supervisor design techniques are extended to hybrid control systems and used to design the DES controller for a given plant, interface and control goals. Finally, section 6 describes a number of system theoretic issues, including determinism, and quasideterminism. These can be applied to the DES models of the hybrid system to provide insight into the design of the interface.

Note that early versions of the results discussed in this paper have appeared in [10–12, 17,18].

2 Hybrid Control System Modeling

A hybrid control system, can be divided into three parts, the plant, interface, and controller as shown in Figure 1. In this model, the plant represents the continuous-time components of the system, while the controller represents the discrete-event portions. The interface is the necessary mechanism by which the former two communicate. The models used for each of these three parts, as well as the way they interact are now described.

2.1 Plant

The plant is the part of the model which represents the entire continuous-time portion of the hybrid control system. The distinguishing feature of the plant is that it has a continuous state space, where the states take on values that are real numbers, and that the state evolves with time according to a set of differential equations.

Motivated by tradition, this part of the model is referred to as the plant but since it contains all the continuous dynamics, it can also contain a conventional, i.e. continuous-time, controller.

Mathematically, the plant is represented by the familiar equation

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

where $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{r} \in \mathbb{R}^m$ are the state and input vectors respectively. $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is a Lipschitz continuous function. Note that the plant input and state are continuous-time vector valued signals. Bold face letters are used to denote vectors and vector valued signals.

2.2 Controller

The controller is a discrete event system which is modeled as a deterministic automaton. This automaton is specified by a quintuple, $\{\tilde{S}, \tilde{X}, \tilde{R}, \delta, \phi\}$, where $\tilde{S}$ is the set of states, $\tilde{X}$ is the set of *plant symbols*, $\tilde{R}$ is the set of *controller symbols*, $\delta : \tilde{S} \times \tilde{X} \to \tilde{S}$ 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 $\tilde{X}$ are called plant symbols and are generated based on events in the plant. The action of the controller can be described by the equations

$$\begin{aligned} \tilde{s}[n] &= \delta(\tilde{s}[n-1], \tilde{x}[n]) \quad (2)\\ \tilde{r}[n] &= \phi(\tilde{s}[n]) \quad (3) \end{aligned}$$

where $\tilde{s}[n] \in \tilde{S}$, $\tilde{x}[n] \in \tilde{X}$, 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 the sequence. The input and output signals associated with the controller are sequences of symbols.

Tildes are used to indicate a symbol valued set or sequence. For example, $\tilde{X}$ is the set of plant symbols and $\tilde{x}[n]$ is a sequence of plant symbols. Subscripts are also used, e.g. $\tilde{x}_i$ which denotes the ith member of the symbol alphabet $\tilde{X}$.

2.3 Interface

The controller and plant cannot communicate directly in a hybrid control system because they each utilize 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 rather simple subsystems, the generator and the actuator.

The *generator* is defined by a set of continuous, differentiable functionals, $h_i : \mathbb{R}^n \to \mathbb{R}$, and a mapping, $\alpha : \mathbb{R}^n \times \mathbb{B} \to \tilde{X}$ where $\mathbb{B} = \{-1, 1\}$. The set of functionals are chosen so that each functional separates the state space of the plant into two open regions, $\{\mathbf{x} : h_i(\mathbf{x}) > 0\}$ and $\{\mathbf{x} : h_i(\mathbf{x}) < 0\}$. The null space of each functional forms the boundary (a hypersurface) between the two open regions. Whenever one of these boundaries is crossed, a *plant event* occurs. With each plant event, plant symbols are generated according to the mapping, α.

$$\alpha : \mathbf{x}(\tau_e[n]) \times b_i \mapsto \tilde{x}[n] \quad (4)$$

where $\tau_e[n]$ is the sequence of event instants or times defined as follows.

$$\begin{aligned} \tau_e[0] &= 0 \quad (5)\\ \tau_e[n] &= \lim_{\epsilon \to 0^+} \inf\{t > \tau_e[n-1] : \exists i, h_i(\mathbf{x}(t)) \cdot h_i(\mathbf{x}(\tau_e[n-1] + \epsilon)) < 0\} \quad (6) \end{aligned}$$

The binary digit, b_i, in equation 4 indicates the direction in which the boundary was crossed; $b_i = -1$ means that $h_i(\mathbf{x}) < 0$ is true for the state immediately after the event, while $b_i = 1$ means the that $h_i(\mathbf{x}) > 0$. Mathematically,

$$b_i = \begin{cases} 1 & \text{if} \quad \nabla h_i(\mathbf{x}(\tau_e[n]) \cdot \dot{\mathbf{x}}(\tau_e[n]) > 0 \\ -1 & \text{if} \quad \nabla h_i(\mathbf{x}(\tau_e[n]) \cdot \dot{\mathbf{x}}(\tau_e[n]) < 0 \end{cases} \quad (7)$$

The possibility of the trajectory crossing more than one boundary simultaneously is not considered because its probability is extremely low. Also, note that not all plant events need be represented by a plant symbol. Such events are silent, accompanied by the null symbol, λ.

The second map, called the *actuator*, $\gamma : \tilde{R} \to \mathbb{R}^m$, converts a sequence of controller symbols to a plant input as follows.

$$\mathbf{r}(t) = \sum_{n=0}^{\infty} \gamma(\tilde{r}[n]) I(t, \tau_c[n], \tau_c[n+1]) \quad (8)$$

J. A. Stiver, P. J. Antsaklis and M. D. Lemmon , "A Logical DES Approach to the Design of Hybrid Control Systems," Technical Report of the ISIS (Interdisciplinary Studies of Intelligent Systems) Group, No. ISIS-93-006, Univ of Notre Dame, October 1993.

where $I(t)$ is the following indicator function

$$I(t) = \begin{cases} 1 & \text{if} \quad \tau_c[n] \leq t < \tau_c[n+1] \\ 0 & \text{if} \quad \text{otherwise} \end{cases} \tag{9}$$

and $\tau_c[n]$ is the sequence of control instants. This sequence is based on the sequence of event instants, defined in equation 6, according to

$$\tau_c[n] = \tau_e[n] + \tau_d \tag{10}$$

where τ_d is the delay associated with the controller; following the generation of a plant symbol, it takes a time of τ_d for a control symbol to reach the plant. It will be assumed that $\tau_e[n] < \tau_c[n] < \tau_e[n+1]$.

3 DES Plant

In a hybrid control system, the plant taken together with the actuator and generator, behaves like a discrete event system. It accepts symbolic inputs via the actuator and produces symbolic outputs via the generator. This situation is analogous to the way a continuous-time plant, equipped with a zero order hold and a sampler, "looks" like a discrete-time plant. In a hybrid control system, the DES which models the plant, actuator, and generator is called the *DES plant*. From the point of view of the DES controller, it is the DES plant which is being controlled.

The DES plant is modeled by an automaton, represented mathematically by a quintuple, $\{\tilde{P}, \tilde{X}, \tilde{R}, \xi, \psi\}$. $\tilde{P}$ is the set of states, $\tilde{X}$ is the set of plant symbols, and $\tilde{R}$ is the set of control symbols. $\xi : \tilde{P} \times \tilde{R} \rightarrow \mathbb{P}(\tilde{X})$ is the *enabling function*, it specifies for a given DES plant state and a given control symbol, which plant symbols are enabled. When a plant symbol is enabled, it may be generated by the DES plant. ψ is a mapping, $\psi : \tilde{P} \times \tilde{X} \rightarrow \tilde{P}$, which is analogous to a state transition function in that it maps a state and a plant symbol to a new state. ψ is not a function because the DES plant need not be deterministic; it is entirely possible for multiple transitions to be accompanied by the same plant symbol.

The set of DES plant states, $\tilde{P}$, is based upon the set of hypersurfaces realized in the generator. It is useful at this point to introduce some notation which will facilitate the explanation of this relationship. First, associated with each function, h_i, we have a second function, $\hat{h}_i$, defined as follows

$$\hat{h}_i(x) = \begin{cases} -1 & \text{if} \quad h_i(x) < 0 \\ 0 & \text{if} \quad h_i(x) = 0 \\ 1 & \text{if} \quad h_i(x) > 0 \end{cases} \tag{11}$$

Now we can define a vector valued function, $\hat{H} : \mathbb{R}^n \rightarrow \{-1, 0, 1\}^I$, where I is the number of hypersurfaces, and the ith element of $\hat{H}(\mathbf{x})$ is equal to $\hat{h}_i(\mathbf{x})$. $\hat{H}$ will induce equivalence classes in $\mathbb{R}^n$ where the equivalence relation is defined as $\mathbf{x}_1 \equiv \mathbf{x}_2 \Leftrightarrow \hat{H}(\mathbf{x}_1) = \hat{H}(\mathbf{x}_2)$. Some of these equivalence classes lie on the separating hypersurfaces and the remainder consist of open regions bounded by the separating hypersurfaces. With each of these open regions there is an associated DES plant state. Thus it is convenient to index the set of states, $\tilde{P}$, with a binary vector, $b \in \mathbb{B}^I$, such that $\tilde{p}_b$ is associated with the preimage of b under $\hat{H}$.

Definition 1: The *DES plant state*, $\tilde{p}[n]$, is defined as follows.

$$\tilde{p}[n] = \tilde{p}_b \Leftrightarrow \lim_{\epsilon \rightarrow 0^+} \hat{H}(\mathbf{x}(\tau_e[n] + \epsilon)) = b \tag{12}$$

where $b \in \mathbb{B}^I$ and $\tilde{p}_b \in \tilde{P} = \{\tilde{p}_c : c \in \mathbb{B}^I\}$

Now we are in a position to determine the enabling function, ξ, and the mapping, ψ. The enabling function is obtained via a procedure which determines whether a particular plant symbol is enabled for a given state and control symbol. The following theorem states that for a given DES plant state, plant symbol, and control symbol, the plant symbol will be enabled if there exists a state in the continuous plant, which lies on the boundary of the relevant open set, maps to the given symbol, and crosses the boundary under the given control.

Theorem 1 *Given a hybrid control system, described by (1) - (10), with f Lipschitz continuous and h_i continuous and differentiable, $\tilde{x}_\ell \in \xi(\tilde{p}_b, \tilde{r}_k) \Leftrightarrow \exists(\mathbf{x}, i)$ such that $h_i(\mathbf{x}) = 0, (\forall\, j \neq i)\ h_j(\mathbf{x})b_j > 0$ and $\alpha : \mathbf{x} \times -b_i \mapsto \tilde{x}_\ell$ which meets the following condition:*

$$[\nabla h_i(\mathbf{x}) \cdot f(\mathbf{x}, \gamma(\tilde{r}_k))]b_i < 0$$

Proof: We will first prove the sufficiency of the theorem conditions. Assume we have a trajectory, $\mathbf{x}(t)$, which satisfies the theorem for some time, t, that is $\mathbf{x}(t) = \mathbf{x}$.

- The condition, $h_i(\mathbf{x}) = 0, (\forall j \neq i)\ h_j(\mathbf{x})b_j > 0$ assures that $\mathbf{x}(t)$ is on the boundary of region b, specifically $\mathbf{x}(t)$ is on the boundary formed by h_i of the open set $\{\mathbf{x} : \hat{H}(\mathbf{x}) = b\}$.
- The condition, $\alpha : \mathbf{x} \times -b_i \mapsto \tilde{x}_\ell$ guarantees that if the state crosses the boundary from region b at the point $\mathbf{x}(t)$, the symbol $\tilde{x}_\ell$ will be generated.
- Finally, the key condition is $[\nabla h_i(\mathbf{x}) \cdot f(\mathbf{x}, \gamma(\tilde{r}_k))]b_i < 0$. It guarantees that if a trajectory passes through $\mathbf{x}(t)$, under $f(\mathbf{x}, \gamma(\tilde{r}_k))$, it will pass from inside region b to outside region b through the boundary formed by h_i.

Together the above show that the existence of an $(\mathbf{x}, i)$ satisfying the theorem is sufficient to guarantee $\tilde{x}_\ell \in \xi(\tilde{p}_b, \tilde{r}_k)$.

Now we will prove that the conditions of the theorem are also necessary to guarantee $\tilde{x}_\ell \in \xi(\tilde{p}_b, \tilde{r}_k)$. We start by assuming $\tilde{x}_\ell \in \xi(\tilde{p}_b, \tilde{r}_k)$. Thus there exists a trajectory, $\mathbf{x}(t)$ which crosses a boundary from region b at a transition time, $\tau_e[n]$.

- Let h_i be the function forming the boundary which is crossed at the point $\mathbf{x} = \mathbf{x}(\tau_e[n])$. This establishes the first condition on $(\mathbf{x}, i)$ that $h_i(\mathbf{x}) = 0, (\forall j \neq i)\ h_j(\mathbf{x})b_j > 0$.
- Since the symbol $\tilde{x}_\ell$ is generated, we know that the second condition, $\alpha : \mathbf{x} \times -b_i \mapsto \tilde{x}_\ell$ must also be satisfied.
- Since we know that the boundary formed by h_i is crossed, from region b, at time $\tau_e[n]$, we know that for sufficiently small positive ϵ, $h_i(\mathbf{x}(\tau_e[n] - \epsilon))b_i > 0$ and $h_i(\mathbf{x}(\tau_e[n] + \epsilon))b_i < 0$. By the mean value theorem, the derivative of $h_i(\mathbf{x})b_i$, given by $[\nabla h_i(\mathbf{x}(t)) \cdot f(\mathbf{x}(t), \gamma(\tilde{r}_k))]b_i$ must be negative at some time between $\tau_e[n] - \epsilon$ and $\tau_e[n] + \epsilon$. This condition is true as $\epsilon \to 0^+$ which proves the final part of the theorem, $[\nabla h_i(\mathbf{x}) \cdot f(\mathbf{x}, \gamma(\tilde{r}_k))]b_i < 0$.

These three points prove that the conditions of the theorem are necessarily true if $\tilde{x}_\ell \in \xi(\tilde{p}_b, \tilde{r}_k)$. □

The usefulness of this theorem is that it allows the extraction of a DES automaton model of the continuous plant and interface. Note that in certain cases this is a rather straightforward task. For instance, it is known that if a particular region boundary can only be crossed in one direction under a given command and is mapped to the same plant symbol along it's extent, then the conditions of the theorem need only be tested at a single point on the boundary. This condition is true for the double integrator example which follows. In general this may not be the case, but one can restrict the area of interest to an operating region of the plant state space thus reducing the computation required.

The state transition mapping, ψ, can be found by a similar procedure described in the next corollary. It states that a given initial DES plant state and plant symbol will be mapped to a resulting DES plant state if there exists a continuous plant state and control such that the given plant symbol is enabled and upon its generation the continuous plant state enters the open region associated with the resulting DES plant state.

Corollary 1 *Given a hybrid control system, described by (1) - (10), with f Lipschitz continuous and h_i continuous and differentiable, $\psi : \tilde{p}_b \times \tilde{x}_\ell \mapsto \tilde{p}_c \Leftrightarrow \exists(\mathbf{x}, k, i)$ such that $h_i(\mathbf{x}) = 0, (\forall j \neq i)\ h_j(\mathbf{x})b_j > 0$ and $\alpha : \mathbf{x} \times -b_i \mapsto \tilde{x}_\ell$ which meets the following conditions:*

1. $[\nabla h_i(\mathbf{x}) \cdot f(\mathbf{x}, \gamma(\tilde{r}_k))]b_i < 0$
2. $\forall j, c_j = -b_j \Leftrightarrow j = i$

Proof: The first condition has already been shown to be necessary and sufficient to enable the symbol $\tilde{x}_\ell$. The second condition requires that $\tilde{p}_c$ be exactly the state which results from the transition, according to the definition of the DES plant state, definition 1. □

4 Examples

4.1 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 which point the furnace shuts off. For simplicity, we will assume that when the furnace is on it produces heat at a constant rate.

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

$$\dot{\mathbf{x}} = .0042(T_0 - \mathbf{x}) + 2step(\mathbf{r}) \tag{13}$$

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, T_0 is the outside temperature, and the constant .0042 is based on the rate of heat loss from the room. 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. The generator is a bimetal band connected to a switch. The band partitions the state space of the plant, $\mathbf{x}$, with two surfaces as follows.

$$h_1(\mathbf{x}) = \mathbf{x} - 70 \tag{14}$$
$$h_2(\mathbf{x}) = \mathbf{x} - 75 \tag{15}$$

The mapping, α, generates two plant symbols.

$$\alpha(70, -1) = \tilde{x}_1 \tag{16}$$
$$\alpha(75, 1) = \tilde{x}_2 \tag{17}$$

Notice that a plant symbol is generated whenever the temperature falls below 70 or rises above 75. The plant event which occurs when the temperature rises above 70 or falls below 75 is silent.

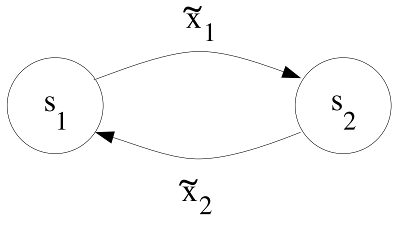
The switch attached to the bimetal band is the DES controller. It has two states because the switch has two positions, on and off. The state transition graph of this simple controller is shown in Figure 2. The output function of the controller is the following

$$\phi(\tilde{s}_1) = \tilde{r}_1 \tag{18}$$
$$\phi(\tilde{s}_2) = \tilde{r}_2 \tag{19}$$

Finally, the actuator is described by the function γ

$$\gamma(\tilde{r}_1) = 0 \tag{20}$$
$$\gamma(\tilde{r}_2) = 24 \tag{21}$$

The behavior of this hybrid system is shown in figure 3 which is a simulation of the thermostat/furnace system with the outside temperature a brisk 25 degrees Fahrenheit.



$$\begin{tikzcd} s_1 \arrow[r, bend left, "\tilde{x}_1"] & s_2 \arrow[l, bend left, "\tilde{x}_2"] \end{tikzcd}$$

Figure 2: Controller for Thermostat/Furnace System

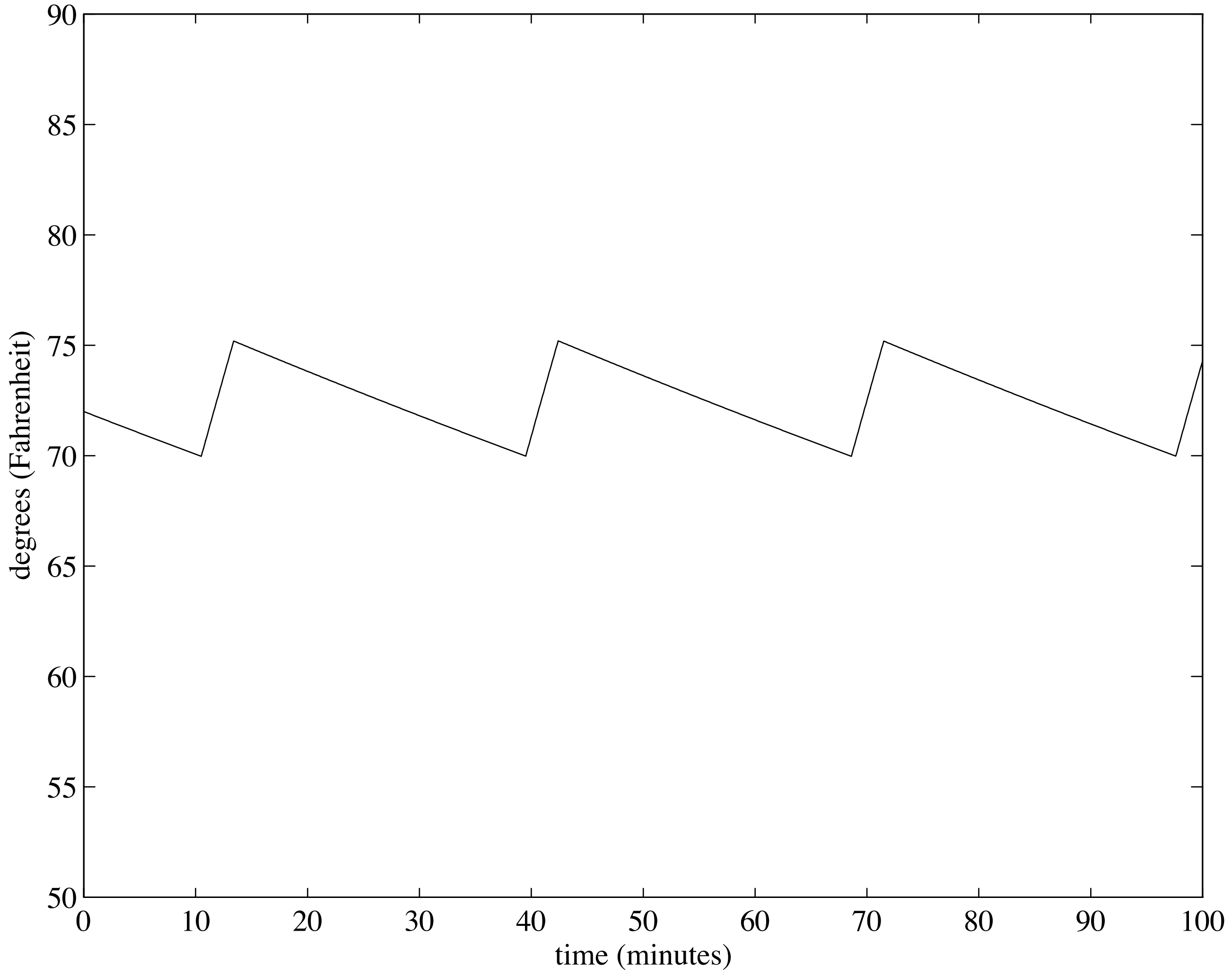


Figure 3: Simulation of Thermostat/Furnace System

J. A. Stiver, P. J. Antsaklis and M. D. Lemmon , "A Logical DES Approach to the Design of Hybrid Control Systems," Technical Report of the ISIS (Interdisciplinary Studies of Intelligent Systems) Group, No. ISIS-93-006, Univ of Notre Dame, October 1993.

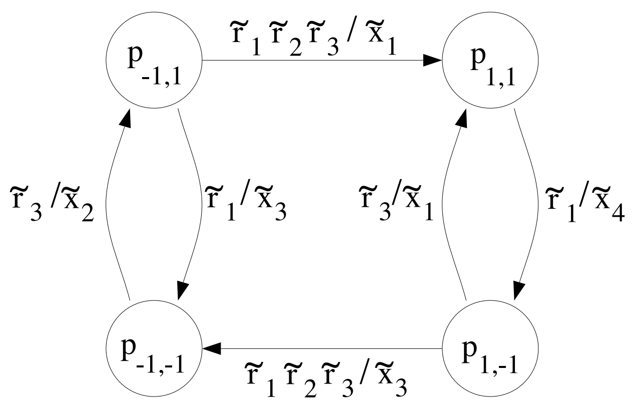

$p_{-1,1}$
$\tilde{r}_1\tilde{r}_2\tilde{r}_3/\tilde{x}_1$
$p_{1,1}$
$\tilde{r}_3/\tilde{x}_2$
$\tilde{r}_1/\tilde{x}_3$
$\tilde{r}_3/\tilde{x}_1$
$\tilde{r}_1/\tilde{x}_4$
$p_{-1,-1}$
$\tilde{r}_1\tilde{r}_2\tilde{r}_3/\tilde{x}_3$
$p_{1,-1}$

Figure 4: DES Plant for Double Integrator

4.2 Example 2 - The Double Integrator

This is another example of a hybrid system. It has appeared before, [11,12], and is used further in this work to illustrate various concepts. It consists of a double integrator which is being controlled by a discrete event system. Thus, the plant is given by the equation of a double integrator.

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \mathbf{r} \tag{22}$$

The events are formed by the following two hypersurfaces

$$h_1(\mathbf{x}) = x_1 \tag{23}$$
$$h_2(\mathbf{x}) = x_2 \tag{24}$$

and the events generate plant symbols according to α

$$\alpha(\mathbf{x}, b_i) = \begin{cases} \tilde{x}_1 & \text{if} \quad \mathbf{x}_1 = 0, x_2 > 0, b_i = 1 \text{ or } \mathbf{x}_1 > 0, x_2 = 0, b_i = 1 \\ \tilde{x}_2 & \text{if} \quad \mathbf{x}_1 = 0, x_2 > 0, b_i = -1 \text{ or } \mathbf{x}_1 < 0, x_2 = 0, b_i = 1 \\ \tilde{x}_3 & \text{if} \quad \mathbf{x}_1 = 0, x_2 < 0, b_i = -1 \text{ or } \mathbf{x}_1 < 0, x_2 = 0, b_i = -1 \\ \tilde{x}_4 & \text{if} \quad \mathbf{x}_1 = 0, x_2 < 0, b_i = 1 \text{ or } \mathbf{x}_1 > 0, x_2 = 0, b_i = -1 \end{cases} \tag{25}$$

Note that α effectively identifies which of the four regions is being entered.

The actuator provides three possible inputs to the plant.

$$\gamma(\tilde{r}) = \begin{cases} -10 & \text{if} \quad \tilde{r} = \tilde{r}_1 \\ 0 & \text{if} \quad \tilde{r} = \tilde{r}_2 \\ 10 & \text{if} \quad \tilde{r} = \tilde{r}_3 \end{cases} \tag{26}$$

These inputs were chosen so that the plant could be driven to the origin by applying them in the proper sequence. Using theorem 1, we can extract the DES plant for this system. It is shown in figure 4. To illustrate how the DES plant was extracted consider $\xi(\tilde{p}_{-1,1}, \tilde{r}_2)$. Specifically, is the plant symbol $\tilde{x}_1$ enabled? If we choose a point in $\{\mathbf{x} : h_1(\mathbf{x}) = 0\}$, say $[0\ 1]'$, we find that it satisfies theorem 1, and thus we know $\tilde{x}_1 \in \xi(\tilde{p}_{-1,1}, \tilde{r}_2)$.

5 Supervisory Control via DES Plant Models

5.1 Controllability and Supervisor Design

In this section, we use the language of the DES plant to examine the controllability of the hybrid control system. This work builds upon the work done by Ramadge and Wonham on the controllability of discrete event systems in a logical framework [19–23]. Here we generalize several of those results to apply them to the DES plant obtained from a hybrid control system.

Before existing techniques, developed in the logical DES framework can be extended, certain differences must be dealt with. The Ramadge-Wonham model (RWM) consists of two interacting DES's called here the *RWM generator* and *RWM supervisor*. The RWM generator is analogous to the DES plant and the RWM supervisor is analogous to the DES controller. The RWM generator shares its name with the generator found in the hybrid control system interface but the two should not be confused. In the RWM, the plant symbols are usually referred to as "events", but we will continue to call them plant symbols to avoid confusion. The plant symbols in the RWM are divided into two sets, those which are controllable and those which are uncontrollable: $\tilde{X} = \tilde{X}_c \cup \tilde{X}_u$. A plant symbol being controllable means that the supervisor can prevent it from being issued by the RWM generator. When the supervisor prevents a controllable plant symbol from being issued, the plant symbol is said to be *disabled*. The plant symbols in $\tilde{X}_c$ can be individually disabled, at any time and in any combination, by a command from the RWM supervisor, while the plant symbols in $\tilde{X}_u$ can never be disabled. This is in contrast to our DES plant where each command (controller symbol) from the DES controller disables a particular subset of $\tilde{X}$ determined by the complement of the set given by the enabling function, ξ. Furthermore, this set of disabled plant symbols depends not only on the controller symbol but also the present state of the DES plant. In addition, there is no guarantee that any arbitrary subset of $\tilde{X}$ can be disabled while the other plant symbols remain enabled.

The general inability to disable plant symbols individually and the enabling function's dependence upon the state of the DES plant, are what differentiate the DES models used to represent a hybrid system from those of earlier frameworks. In fact, the RWM represents a special case of the former, in which the enabling function is independent of the plant state and it is possible to disable only certain plant symbols albeit in any combination desired.

The behavior of a DES can be characterized by the set of possible sequences of symbols which it can generate. This set is referred to as the language of the DES, denoted L, and defined (using the notation of the DES plant) as

$$L = \{w : w \in \tilde{X}^*, \psi(p_0, w) \text{ is defined}\} \tag{27}$$

where $\tilde{p}_0 = \tilde{p}[0] \in \tilde{P}$ is the initial state and $\tilde{X}^*$ is the set of all possible sequences of symbols from $\tilde{X}$.

A DES is controlled by having various symbols disabled by the controller based upon the sequence of symbols which the DES has already generated. When a DES is controlled, it will generate a set of symbol sequences which lie in a subset of its language. If we denote this language of the DES under control as L_f then $L_f \subset L$.

It is possible to determine whether a given RWM generator can be controlled to a desired language [19]. That is, whether it is possible to design a controller such that the RWM generator will be restricted to some target language K. Such a controller can be designed if K is prefix closed and

$$\bar{K}\tilde{X}_u \cap L \subset \bar{K} \tag{28}$$

where $\bar{K}$ represents the set of all prefixes of K. A prefix of K is a sequence of symbols, to which another sequence can be concatenated to obtain a sequence found in K. A language is said to be prefix closed if all the prefixes of that language are also in the language.

When equation 28 is true for a given RWM generator, the desired language K is said to be controllable, and provided K is prefix closed, a controller can be designed which will restrict the generator to the language K. This condition requires that if an uncontrollable symbol occurs after the generator has produced a prefix of K, the resulting string must still be a prefix of K because the uncontrollable symbol cannot be prevented.

Since the DES plant belongs to a more general class of automata than the RWM, we present another definition for controllable language which applies to the DES plant.

Definition 2: A language, K, is controllable with respect to a given DES plant if

$$\forall w \in \bar{K} \; \exists \, \tilde{r} \in \tilde{R} \; \ni \; w\xi(\psi(\tilde{p}_0, w), \tilde{r}) \subset \bar{K} \tag{29}$$

This definition requires that for every prefix of the desired language, K, there exists a control, $\tilde{r}$, which will enable only symbols which will cause string to remain in K.

Theorem 2 *If the language K is prefix closed and controllable according to (29), then a controller can be designed which will restrict the given DES plant to the language K.*

Proof: Let the controller be given by $f : \tilde{X}^* \rightarrow \tilde{R}$ where $f(w) \in \{\tilde{r} : w\xi(\psi(\tilde{p}_0, w), \tilde{r}) \subset \bar{K}\}$. $f(w)$ is guaranteed to be non-empty by (29). We can now show by induction that $w \in L_f \Rightarrow w \in K$.

1. $\forall w \in L_f$ such that $|w| = 0$ we have $w \in K$. This is trivial because the only such w is the null string ϵ and $\epsilon \in K$ because K is prefix closed.
2. $Let L_f{}^i = \{w : w \in L_f, |w| = i\}$, that is $L_f{}^i$ is the set of all sequences of length i found in L_f. Given $L_f{}^i$, $L_f{}^{i+1} = \{v : v = w\xi(\psi(\tilde{p}_0, w), f(w)), w \in L_f{}^i\}$. Now with the definition of $f(w)$ and (29) we have $L_f{}^i \in K \Rightarrow L_f{}^{i+1} \in K$.

So $w \in L_f \Rightarrow w \in K$. □

Since the DES plant can be seen as a generalization of the RWM, the conditions in (29) should reduce to those of (28) under the appropriate restrictions. This is indeed the case.

If the desired language is not attainable for a given DES, it may be possible to find a more restricted language which is. If so, the least restricted behavior is desirable. [19] and [22] describe and provide a method for finding this behavior which is referred to as the *supremal controllable sublanguage*, $K^\uparrow$, of the desired language. The supremal controllable sublanguage is the largest subset of K which can be attained by a controller. $K^\uparrow$ can be found via the following iterative procedure.

$$\begin{aligned} K_0 &= K && (30)\\ K_{i+1} &= \{w : w \in \bar{K}, \bar{w}\tilde{X}_u \cap L \subset \overline{K_i}\} && (31)\\ K^\uparrow &= \lim_{i\rightarrow\infty} K_i && (32)\end{aligned}$$

Once again, this procedure applies to the RWM. For hybrid control systems, the supremal controllable sublanguage of the DES plant can be found by a similar but more general iterative scheme.

$$\begin{aligned} K_0 &= K && (33)\\ K_{i+1} &= \{w : w \in \bar{K}, \forall v \in \bar{w}\ \exists\ \tilde{r} \in \tilde{R} \text{ such that } v\xi(\psi(\tilde{p}_0, v), \tilde{r}) \subset \overline{K_i}\} && (34)\\ K^\uparrow &= \lim_{i\rightarrow\infty} K_i && (35)\end{aligned}$$

This result yields the following theorem.

Theorem 3 *For a DES plant and prefix closed language K, $K^\uparrow$ is controllable and contains all prefix closed, controllable sublanguages of K.*

Proof: From (34) we have

$$K^\uparrow = \{w : w \in \bar{K}, \forall v \in \bar{w}\ \exists\ \tilde{r} \in \tilde{R} \text{ such that } v\xi(\psi(\tilde{p}_0, v), \tilde{r}) \subset \overline{K^\uparrow}) \qquad (36)$$

which implies

$$w \in K^\uparrow \Rightarrow \exists\ \tilde{r} \in \tilde{R} \text{ such that } w\xi(\psi(\tilde{p}_0, w), \tilde{r}) \subset \overline{K^\uparrow} \qquad (37)$$

From (37) it is clear that $K^\uparrow$ is controllable. To show that every prefix closed, controllable subset of K is in $K^\uparrow$ we assume there exists $M \subset K$ such that M is prefix closed and controllable but $M \not\subset K^\uparrow$.

$$\exists M \text{ s.t. } M \subset K, M \not\subset K^\uparrow \tag{38}$$
$$\Rightarrow \exists w \text{ s.t. } w \in M, w \notin K^\uparrow \tag{39}$$
$$\Rightarrow \exists i \text{ s.t. } w \in K_i, w \notin K_{i+1} \tag{40}$$
$$\Rightarrow \exists v \in \bar{w} \text{ s.t. } \forall \tilde{r} \in \tilde{R}, v\xi(\psi(\tilde{p}_0, v), \tilde{r}) \not\subset K_i \tag{41}$$
$$\Rightarrow \exists w' \in v\xi(\psi(\tilde{p}_0, v), \tilde{r}) \text{ s.t. } w' \in M, w' \notin K_i \tag{42}$$
$$\Rightarrow \exists j < i \text{ s.t. } w' \in K_j, w' \notin K_{j+1} \tag{43}$$

If the sequence is repeated with $i = j$ and $w = w'$ we eventually arrive at the conclusion that $w' \in M$ but $w' \notin K_0$ which violates the assumption that $M \subset K$ and precludes the existence of such an M. □

Example 1 - Double Integrator We use the double integrator example again because the DES plant was found earlier. This DES is represented by the automaton in figure 4, and explicitly by the following sets and functions.

$$\tilde{P} = \{\tilde{p}_{1,1}, \tilde{p}_{-1,1}, \tilde{p}_{-1,-1}, \tilde{p}_{1,-1}\} \tag{44}$$
$$\tilde{X} = \{\tilde{x}_1, \tilde{x}_2, \tilde{x}_3, \tilde{x}_4\} \tag{45}$$
$$\tilde{R} = \{\tilde{r}_1, \tilde{r}_2, \tilde{r}_3\} \tag{46}$$

$$\psi(\tilde{p}_{1,1}, \tilde{x}_4) = \tilde{p}_{1,-1} \quad \psi(\tilde{p}_{-1,-1}, \tilde{x}_2) = \tilde{p}_{-1,1} \tag{47}$$
$$\psi(\tilde{p}_{-1,1}, \tilde{x}_1) = \tilde{p}_{1,1} \quad \psi(\tilde{p}_{1,-1}, \tilde{x}_1) = \tilde{p}_{1,1} \tag{48}$$
$$\psi(\tilde{p}_{-1,1}, \tilde{x}_3) = \tilde{p}_{-1,-1} \quad \psi(\tilde{p}_{1,-1}, \tilde{x}_3) = \tilde{p}_{-1,-1} \tag{49}$$

$$\xi(\tilde{p}_{1,1}, \tilde{r}_1) = \{\tilde{x}_4\} \quad \xi(\tilde{p}_{-1,-1}, \tilde{r}_3) = \{\tilde{x}_2\} \tag{50}$$
$$\xi(\tilde{p}_{-1,1}, \tilde{r}_1) = \{\tilde{x}_1, \tilde{x}_3\} \quad \xi(\tilde{p}_{1,-1}, \tilde{r}_1) = \{\tilde{x}_3\} \tag{51}$$
$$\xi(\tilde{p}_{-1,1}, \tilde{r}_2) = \{\tilde{x}_1\} \quad \xi(\tilde{p}_{1,-1}, \tilde{r}_2) = \{\tilde{x}_3\} \tag{52}$$
$$\xi(\tilde{p}_{-1,1}, \tilde{r}_3) = \{\tilde{x}_1\} \quad \xi(\tilde{p}_{1,-1}, \tilde{r}_3) = \{\tilde{x}_1, \tilde{x}_3\} \tag{53}$$

The values, for which ψ has not been stated, are undefined, and the values for which ξ has not been stated are the empty set.

Let $\{a, b, c, d\} = \{\tilde{x}_1, \tilde{x}_2, \tilde{x}_3, \tilde{x}_4\}$. Then the language generated by this automaton is $L = \overline{(da + dcb(cb)^*a)^*}$. If we want to drive the plant in clockwise circles, then the desired language is $K = \overline{(dcba)^*}$. It can be shown that this K is controllable because it satisfies Equation (29). Therefore according to theorem 2, a controller can be designed to achieve the stated control goal.

Example 2 - A More Complex DES Plant Model This example has a richer behavior and will illustrate the generation of a supremal controllable sublanguage. We start immediately with the DES plant model shown in figure 5. Only the plant symbols are shown in the figure. The enabling function, ξ, is given by the following table.

ξ	$\tilde{c}_1$	$\tilde{c}_2$	$\tilde{c}_3$	$\tilde{c}_4$
$\tilde{p}_1$	$\emptyset$	$\{\tilde{b}\}$	$\{\tilde{b}\}$	$\emptyset$
$\tilde{p}_2$	$\{\tilde{a}\}$	$\{\tilde{a}, \tilde{d}\}$	$\{\tilde{c}\}$	$\{\tilde{a}, \tilde{c}, \tilde{d}\}$
$\tilde{p}_3$	$\{\tilde{a}\}$	$\{\tilde{f}\}$	$\emptyset$	$\{\tilde{a}, \tilde{f}\}$
$\tilde{p}_4$	$\{\tilde{a}\}$	$\{\tilde{f}\}$	$\{\tilde{a}, \tilde{f}\}$	$\{\tilde{a}, \tilde{e}, \tilde{d}\}$
$\tilde{p}_5$	$\emptyset$	$\{\tilde{d}\}$	$\{\tilde{d}\}$	$\{\tilde{d}\}$
$\tilde{p}_6$	$\{\tilde{e}\}$	$\{\tilde{e}\}$	$\{\tilde{e}\}$	$\{\tilde{e}\}$

(54)

J. A. Stiver, P. J. Antsaklis and M. D. Lemmon , "A Logical DES Approach to the Design of Hybrid Control Systems," Technical Report of the ISIS (Interdisciplinary Studies of Intelligent Systems) Group, No. ISIS-93-006, Univ of Notre Dame, October 1993.

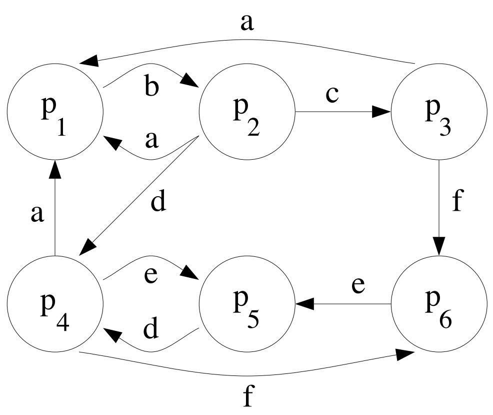


Figure 5: DES Plant Model for Example 2

The language generated by this DES is $L = \overline{L_m}$ where

$$L_m = (b(a + d\sigma^* a + c(a + fed\sigma^* a)))^* \tag{55}$$

where $\sigma = ((e + fe)d)$. Suppose we want to control the DES so that it never enters state $\tilde{p}_5$ and can always return to state $\tilde{p}_1$. The desired language is therefore

$$K = \overline{(a + b + c + d + f)^* a} \tag{56}$$

In this example, the language K is not controllable. This can be seen by considering the string $bcf \in K$, for which there exists no $\tilde{r} \in \tilde{R}$ which will prevent the DES from deviating from K by generating e and entering state $\tilde{p}_5$.

Since K is not controllable, we find the supremal controllable sublanguage of K as defined in equation (35). The supremal controllable sublanguage is

$$K^\uparrow = K_1 = \overline{(a + b + c + d + f)^* a - (bcfed\sigma^* a)^*} \tag{57}$$

Obtaining a DES controller once the supremal controllable sublanguage has been found is straight forward. The controller is a DES whose language is given by $K^\uparrow$ and the output of the controller in each state, $\phi(\tilde{s})$, is the controller symbol which enables only transitions which are found in the controller. The existence of such a controller symbol is guaranteed by the fact that $K^\uparrow$ is controllable. For Example 2, the controller is shown in Figure 6 and it's output function, ϕ, is as follows:

$$\phi(\tilde{s}_1) = \tilde{r}_2 \qquad \phi(\tilde{s}_2) = \tilde{r}_4 \tag{58}$$

$$\phi(\tilde{s}_3) = \tilde{r}_1 \qquad \phi(\tilde{s}_4) = \tilde{r}_1 \tag{59}$$

6 System Theoretic Issues

In this section we describe two properties which can be applied to discrete event systems in general and the DES plant specifically. Each of these properties is, from the DES point of view, a form of determinism. Here we call them *determinism* and *quasideterminism*. Note that a clear understanding of these properties is important because they provide guidelines on the design of the interface.

6.1 Determinism

When an automaton, such as the DES plant, is said to be *deterministic* it typically means that for a given state and event, there is only one possible subsequent state. In classical systems however, determinism refers to the property that the future state can be

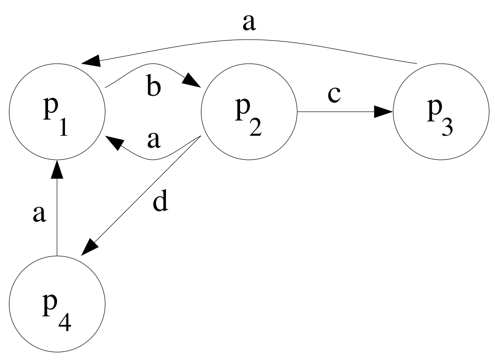


Figure 6: DES Controller for Example 2

determined based on the current state and input. Since we wish to control the DES plant via the inputs, this classical definition of determinism is useful to us and so determinism is defined in terms of the DES plant state and input symbol.

If you consider a given state of the DES plant and a given input to it, there may be multiple possible subsequent states and therefore the DES plant is not generally deterministic.

Definition 3: A DES plant is *deterministic* if

$$\forall(b,k)\ \exists c : \tilde{x} \in \xi(\tilde{p}_b, \tilde{r}_k) \Rightarrow (\psi : \tilde{p}_b \times \tilde{x} \mapsto \tilde{p}_d \Rightarrow d = c)$$

Determinism allows $\tilde{p}[n+1]$ to be predicted from $\tilde{p}[n]$ and $\tilde{r}[n]$.

6.2 Partitioning and Quasideterminism

As mentioned above, determinism is important from the standpoint of control. Unfortunately, it is difficult to attain in the DES plant model. We now introduce *quasideterminism* which means that a subsequent DES plant state can be predicted based on the current state, the current input, and the previous output. Note quasideterminism differs from determinism with the addition of the previous output.

Definition 4: A DES plant is *quasideterministic* if

$$\forall(b,k,\ell)\ \exists c : \tilde{p}[n] = \tilde{p}_b, \tilde{r}[n] = \tilde{r}_k, \tilde{x}[n] = \tilde{x}_\ell \Rightarrow \tilde{p}[n+1] = \tilde{p}_c$$

A particular problem in the design of a hybrid control system is the selection of the generator. Since the separating hypersurfaces and the mapping, α, found in the generator, are used to generate the plant symbols, they must be chosen to provide sufficient information to the controller to allow control without being so fine that they lead to an unmanageably complex system or simply degenerates the system into an essentially conventional control system.

The generator must accomplish two goals. First it must give the controller sufficient information to identify 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. The generator in an aircraft control system would thus need to be able to identify whether or not the plant was diving, for instance, by knowing the region containing the current state.

Second, the generator must provide enough additional information about the state, to enable the controller to control 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 generator must be detailed enough to answer: 1) is the current state in an acceptable region; and 2) which input (if any) can be applied to drive the state to an acceptable region.

To design a generator, we can start by selecting the hypersurfaces to meet the first goal mentioned above. These hypersurfaces will identify all the desired operating regions of the plant state space, so their design will be dictated by the control goals. The mapping, α, can then be designed to make the DES plant quasideterministic.

The constraints on α such that the DES plant will be quasideterministic are established by the following theorem. But first we slip in the definition of a function which is used in the theorem. $\hat{F}_k : \mathbb{R}^n \times \mathbb{B} \to \mathbb{R}^n$, is a function which maps a plant state at the time of a plant event to the plant state at the time of the next plant event, subject to $\dot{\mathbf{x}} = f(\mathbf{x}, \gamma(\tilde{r}_k))$. Note that $\hat{F}_k$ is defined over the same domain as α. According to the theorem, a DES plant will be quasideterministic if any two trajectories which enter the open region associated with a DES plant state must generate different plant symbols if they exit the open region through different boundaries.

Theorem 4 *The DES plant will be quasideterministic if* $\forall(b, i, j, \mathbf{x}_1, \mathbf{x}_2)$ *such that*

$$\hat{h}_i(\mathbf{x}_1) = 0, (\forall \kappa \neq i)\hat{h}_\kappa(\mathbf{x}_1) = b_\kappa$$

$$\hat{h}_j(\mathbf{x}_2) = 0, (\forall \kappa \neq j)\hat{h}_\kappa(\mathbf{x}_2) = b_\kappa$$

the following holds

$$\hat{H}(\hat{F}_k(\mathbf{x}_1, b_i)) \neq \hat{H}(\hat{F}_k(\mathbf{x}_2, b_j)) \Rightarrow \alpha(\mathbf{x}_1, b_i) \neq \alpha(\mathbf{x}_2, b_j)$$

Quasideterminism is useful as a guide in selecting α given a set of boundaries.

Proof: We can see from the theorem that if any two continuous plant state trajectories leave a given open region via different boundaries (which means they enter two different subsequent open regions) then their entries into the given open region much be accompanied by different plant symbols. Thus given a DES plant state, $\tilde{p}[n]$, and control symbol, $\tilde{r}[n]$, the next DES plant state, $\tilde{p}[n+1]$, can be predicted by noting the current plant symbol, $\tilde{x}[n]$. □

6.3 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. In this paper, we assume that the control actions available are determined by the plant (positions of the various valves, switches, etc.) and thus represent a constraint on the controller design. In general the available control actions must be selected and this is an open area of research.

6.4 More Examples

We return to the double integrator example to illustrate the properties defined in the previous section. First we show a system which is neither deterministic nor quasideterministic, next one which is quasideterministic and finally a deterministic system.

6.4.1 Nondeterministic Double Integrator System

Consider the double integrator example presented earlier and pictured in figure 4. Notice that there are two transitions from state $\tilde{p}_{-1,1}$ under input $\tilde{r}_2$ indicating the system is nondeterministic. Note also that the previous plant symbol will always be $\tilde{x}_1$. Therefore the previous plant symbol cannot be used to predict which state will result from the transition and so the system is not quasideterministic either.

6.4.2 Quasidetermistic Double Integrator System

The second example consists of the above system except with a different α. Now the mapping α associates a different symbol with each surface. When a plant event occurs, the symbol identifies which surface was crossed, but not the direction or location of crossing.

$$\alpha(\mathbf{x}, b_i) = \begin{cases} \tilde{x}_1 & \text{if} \quad h_1(\mathbf{x}) = 0 \\ \tilde{x}_2 & \text{if} \quad h_2(\mathbf{x}) = 0 \end{cases} \tag{60}$$

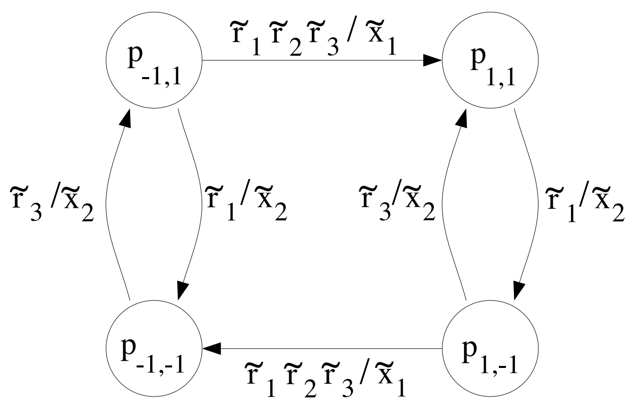

$p_{-1,1}$
$p_{1,1}$
$p_{-1,-1}$
$p_{1,-1}$
$\tilde{r}_1\tilde{r}_2\tilde{r}_3/\tilde{x}_1$
$\tilde{r}_3/\tilde{x}_2$
$\tilde{r}_1/\tilde{x}_2$
$\tilde{r}_3/\tilde{x}_2$
$\tilde{r}_1/\tilde{x}_2$
$\tilde{r}_1\tilde{r}_2\tilde{r}_3/\tilde{x}_1$

Figure 7: Quasideterministic DES Plant

The DES plant, shown in figure 7, is quasideterministic. Looking at figure 7 we see two situations where non-deterministic transitions occur, $\psi(\tilde{p}_{-1,1}, \tilde{r}_1)$ and $\psi(\tilde{p}_{1,-1}, \tilde{r}_3)$. Since the system is quasideterministic, this uncertainty can be eliminated by considering the most recent plant symbol as shown in the following where we assume $\tilde{p}[n] = \tilde{p}_{-1,1}$.

$$\psi(\tilde{p}_{-1,1}, \tilde{r}_1) = \begin{cases} \tilde{p}_{1,1} & \text{if} \quad \tilde{x}[n] = \tilde{x}_1 \\ \tilde{p}_{-1,-1} & \text{if} \quad \tilde{x}[n] = \tilde{x}_2 \end{cases} \tag{61}$$

Unfortunately, $\tilde{p}_{-1,1}$ cannot normally be reached following the plant symbol $\tilde{x}_1$ so this situation cannot arise unless the system is disturbed.

6.4.3 Deterministic Double Integrator System

Our final example features the above double integrator with the hypersurfaces defined as

$$h_1(\mathbf{x}) = x_1 + \frac{1}{2}x_2^2 \tag{62}$$

$$h_2(\mathbf{x}) = x_2 \tag{63}$$

$$h_3(\mathbf{x}) = x_1 - \frac{1}{2}x_2^2 \tag{64}$$

and α defined as

$$\alpha(\mathbf{x}, b_i) = \begin{cases} \tilde{x}_1 & \text{if} \quad h_1(\mathbf{x}) = 0 \\ \tilde{x}_2 & \text{if} \quad h_2(\mathbf{x}) = 0 \\ \tilde{x}_3 & \text{if} \quad h_3(\mathbf{x}) = 0 \end{cases} \tag{65}$$

The hypersurfaces are pictured in figure 8 and the deterministic DES plant automaton is pictured in 9. Notice that all the transitions are deterministic.

7 Conclusion

The contributions of this paper include a formal model for hybrid control systems that uses a simple interface. The model is simple enough to allow analysis and general enough to include other hybrid system models which have appeared in the literature. Local conditions are given in theorem 1 which allow the extraction of a DES model from the plant and interface. A method for DES controller design is given. Finally important system theoretic issues, such as determinism and quasidetermism were identified.

The results of this paper were used to design a DES controller for a hybrid control system, when the plant and control goals are given. These results are far from a complete treatment of this subject, however. The problem of selecting the available inputs to the plant, for example, is still largely unsolved. Also, the design of the generator has only been

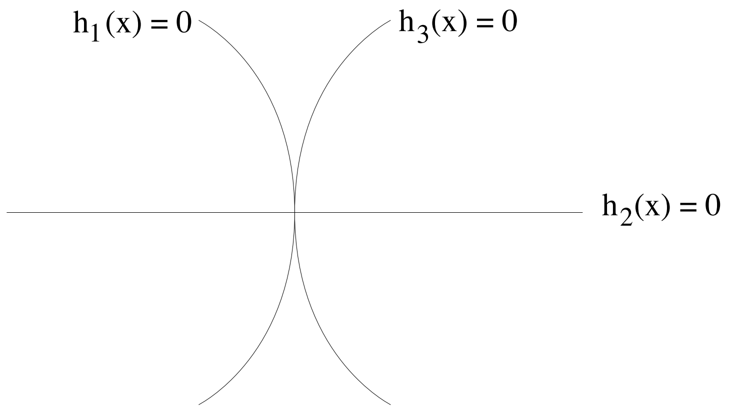


Figure 8: Hypersurfaces for Deterministic DES Plant

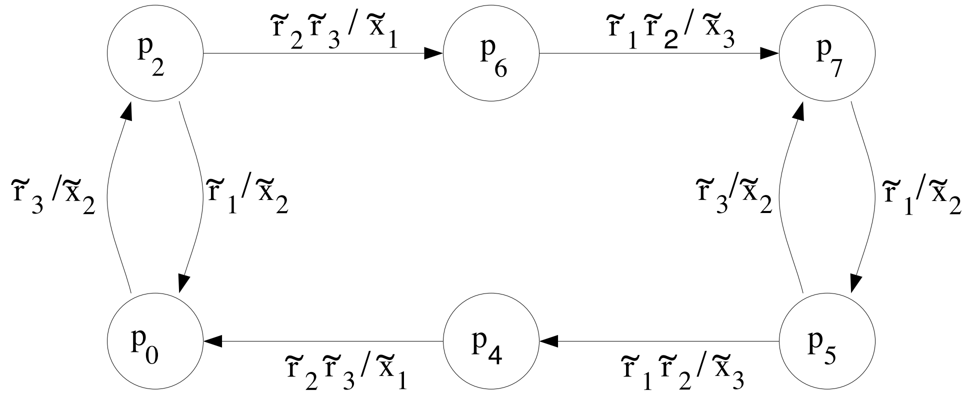

p_2
p_6
p_7
p_0
p_4
p_5
$\tilde{r}_2\tilde{r}_3/\tilde{x}_1$
$\tilde{r}_1\tilde{r}_2/\tilde{x}_3$
$\tilde{r}_3/\tilde{x}_2$
$\tilde{r}_1/\tilde{x}_2$
$\tilde{r}_3/\tilde{x}_2$
$\tilde{r}_1/\tilde{x}_2$
$\tilde{r}_2\tilde{r}_3/\tilde{x}_1$
$\tilde{r}_1\tilde{r}_2/\tilde{x}_3$

Figure 9: Deterministic DES Plant

partially solved. Using quasideterminism, it is still up to the designer to to make the initial selection of a partition, a selection which has a large impact on the resulting system. Also the computational issue, which is very important in hybrid control, was not addressed in this paper.

It is important to note that the core issue in hybrid control systems is the way in which the the interface relates the continuous plant to the DES plant. This issue is rather fundamental and quite difficult to resolve. It's solution will depend on the answer to the question of what is the minimum amount of information about the plant that will allow, with the controls available, the accomplishment of the control objectives. In this paper an approach was presented which, given the controls and control objectives, helps to design the interface and derive a controller using quasideterminism and controllability.

Acknowledgement - this work was partially supported by NSF grants MSS-9216559 and IRI91-09298.

References

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

[2] A. Benveniste and P. Le Guernic, "Hybrid dynamical systems and the signal language", *IEEE Transactions on Automatic Control*, vol. 35, no. 5, pp. 535–546, May 1990.

[3] A. Gollu and P. Varaiya, "Hybrid dynamical systems", In *Proceedings of the 28th Conference on Decision and Control*, pp. 2708–2712, Tampa, FL, Dec. 1989.

[4] L. Holloway and B. Krogh, "Properties of behavioral models for a class of hybrid dynamical systems", In *Proceedings of the 31st Conference on Decision and Control*, pp. 3752–3757, Tucson, AZ, Dec. 1992.

[5] W. Kohn and A. Nerode, "Multiple agent autonomous hybrid control systems", In *Proceedings of the 31st Conference on Decision and Control*, pp. 2956–2966, Tucson, AZ, Dec. 1992.

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

[7] M. D. Lemmon, J. A. Stiver, and P. J. Antsaklis, "Learning to coordinate control policies of hybrid systems", In *Proceedings of the American Control Conference*, pp. 31–35, San Francisco, CA, June 1993.

[8] K. M. Passino and U. Ozguner, "Modeling and analysis of hybrid systems: Examples", In *Proceedings of the 1991 IEEE International Symposium on Intelligent Control*, pp. 251–256, Arlington, VA, Aug. 1991.

[9] P. Peleties and R. DeCarlo, "A modeling strategy with event structures for hybrid systems", In *Proceedings of the 28th Conference on Decision and Control*, pp. 1308–1313, Tampa, FL, Dec. 1989.

[10] J. A. Stiver and P. J. Antsaklis, "A novel discrete event system approach to modeling and analysis of hybrid control systems", In *Proceedings of the Twenty-Ninth Annual Allerton Conference on Communication, Control, and Computing*, University of Illinois at Urbana-Champaign, Oct. 1991.

[11] J. A. Stiver and P. J. Antsaklis, "Modeling and analysis of hybrid control systems", In *Proceedings of the 31st Conference on Decision and Control*, pp. 3748–3751, Tucson, AZ, Dec. 1992.

[12] J. A. Stiver and P. J. Antsaklis, "State space partitioning for hybrid control systems", In *Proceedings of the American Control Conference*, pp. 2303–2304, San Francisco, California, June 1993.

[13] P. J. Ramadge, "On the periodicity of symbolic observations of piecewise smooth discrete-time systems", *IEEE Transactions on Automatic Control*, vol. 35, no. 7, pp. 807–812, July 1990.

[14] C. Chase and P. J. Ramadge, "Dynamics of a switched n buffer system", In *Proceedings of the Twenty-Eighth Annual Allerton Conference on Communication, Control, and Computing*, pp. 455–464, University of Illinois at Urbana-Champaign, Oct. 1991.

[15] A. Nerode and W. Kohn, "Models for Hybrid Systems: Automata, Topologies, Controllability, Observability", In *Hybrid Systems*, edited by R. L. Grossman, A. Nerode, A. P. Ravn and H. Rischel, pp. 317–356. Springer-Verlag, 1993.

[16] P. Peleties and R. DeCarlo, "Modeling of interacting continuous time and discrete event systems : An example", In *Proceedings of the 26th Annual Allerton Conference on Communication, Control, and Computing*, pp. 1150–1159, University of Illinois at Urbana-Champaign, Oct. 1988.

[17] P. J. Antsaklis, M. D. Lemmon, and J. A. Stiver, "Hybrid system modeling and event identification", Technical Report of the ISIS Group ISIS-93-002, University of Notre Dame, Notre Dame, IN, January 1993.

[18] P. J. Antsaklis, M. D. Lemmon, and J. A. Stiver, "Learning to be autonomous: Intelligent supervisory control", Technical Report of the ISIS Group ISIS-93-003, University of Notre Dame, Notre Dame, IN, April 1993, To appear as a chapter in the IEEE Press book Intelligent Control: Theory and Applications.

[19] P. J. Ramadge and W. M. Wonham, "Supervisory control of a class of discrete event processes", Systems Control Group Report 8515, University of Toronto, Toronto, Canada, Nov. 1985.

[20] P. Ramadge and 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.

[21] P. Ramadge and W. M. Wonham, "The control of discrete event systems", *Proceedings of the IEEE*, vol. 77, no. 1, pp. 81–89, Jan. 1989.

[22] W. M. Wonham and P. J. Ramadge, "On the supremal controllable sublanguage of a given language", Systems Control Group Report 8312, University of Toronto, Toronto, Canada, Nov. 1983.

[23] W. M. Wonham and 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.

[24] M. Lemmon, J. Stiver, and P. Antsaklis, "Event identification and intelligent hybrid control", In *Hybrid Systems*, edited by R. L. Grossman, A. Nerode, A. P. Ravn and H. Rischel, pp. 265–296. Springer-Verlag, 1993.