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

Control Systems Technical Report #71

A NOVEL DISCRETE EVENT SYSTEM APPROACH TO MODELING AND ANALYSIS OF HYBRID CONTROL SYSTEMS

June 1991

James A. Stiver and Panos J. Antsaklis
Department of Electrical Engineering
University of Notre Dame
Notre Dame, Indiana 46556

The authors gratefully acknowledge the partial support of the Jet Propulsion Laboratory under contract No. 957856.

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

A NOVEL DISCRETE EVENT SYSTEM APPROACH TO MODELING AND ANALYSIS OF HYBRID CONTROL SYSTEMS

James A. Stiver and Panos J. Antsaklis
Dept. of Electrical Engineering
University of Notre Dame
Notre Dame, IN 46556
Phone: (219) 239-6916, (219) 239-5792
E-mail: jstiver@bach.helios.nd.edu, flxfsn@irishmvs.cc.nd.edu

Abstract

In this report, a hybrid control system consists of a continuous-state system, which may be continuous-time or discrete-time, that is being supervised and controlled by a higher level decision making system, typically a discrete-state symbolic system. In particular, a hybrid control system consists of two rather distinct parts: a system adequately described by a set of difference or differential equations, this may include the plant to be controlled and perhaps a conventional controller, and a higher level decision making controller, modeled as a discrete event system.

Since hybrid control systems show great promise, techniques for hybrid systems, analogous to those existing for more conventional control systems, would be very useful. However, until recently little has been attempted in the area of hybrid systems. In this report, a general and useful modeling technique for hybrid control systems is presented and this model is used to perform some analysis on the hybrid control system. A number of examples are given to illustrate the concepts; they include a double integrator controlled by a discrete event symbolic system and a surge tank controller.

More specifically, the model developed in this report includes an explicitly defined interface for carrying information between the continuous-state and discrete-state parts of the system. This interface allows for a complete mathematical model of the system to be defined, so that it can be simulated or analyzed in a general fashion. A method is presented for converting the continuous-state portion of the system, along with the interface, to a discrete-event system representation. This discrete-event system represents the model, of the lower level system, which the higher level system sees. Conditions are given to determine whether the discrete-event representation will be deterministic. A beginning treatment of the stability of hybrid control systems, using the discrete-event system representation, is also given.

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

1. INTRODUCTION

A Hybrid Control System (HCS) consists of a continuous-state system that may be continuous-time or discrete-time, which is being supervised and controlled by a higher level decision making system, typically a discrete-state symbolic system. In particular, a hybrid control system consists of two rather distinct parts: a system adequately described by a set of difference or differential equations, this may include the plant to be controlled and perhaps a conventional controller, and a higher level decision making controller, modeled as a discrete-event system. Note that the use of the term "hybrid" is quite distinct from its common use in the control field referring to systems with both digital and analog components. Hybrid control systems are quite common in practice. A familiar hybrid control system is the temperature control of a typical home, if the furnace and air conditioner are modeled as continuous-time systems as they are being controlled by a thermostat which can be regarded as a decision making system. Recently, attempts have been made to study hybrid control systems in a unified, analytical framework [4, 12, 17, 18].

Models and techniques for the analysis of systems adequately described by a set of difference or differential equations exist, in fact, this has been the main thrust of control theory in the past century. Significant work has also been done for the modeling and analysis of discrete-event systems, e.g. [14, 16, 20, 26]. Since hybrid control systems show great promise, techniques for hybrid systems, analogous to those existing for the above more conventional control systems, would be very useful. However, little has been attempted in the area of hybrid systems. In [12], a model is developed for a specific hybrid system, a computer disk and controller. In [17] and [18] a model for a more general class of hybrid systems is proposed. Finally, in [4] a method of representing a wide class of hybrid systems via a programming language is presented and used to draw some conclusions about the hybrid system.

In this report, in keeping with conventional control systems, the continuous-state system shall be called the plant, and the decision making system shall be called the controller. This is in spite of the fact that there may be some control action taking place in what is called the plant. The difficulty in modeling hybrid systems arises from the heterogeneous nature of this system. The plant is a continuous-state system which evolves in real time. Signals involving the plant may be continuous-time or discrete-time but they are always continuously valued, being assigned a real numeric value. The controller, on the other hand, is an event driven system; it evolves in logical time. Signals involving the controller are discrete and asynchronous, and their values are generally qualitative, being

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

symbolic rather than numeric. In order for the plant and controller to interact, the hybrid system must also contain an additional part which serves as an interface and allows an exchange of information between the plant and controller. The three parts of a hybrid control system, controller, interface, and plant, can be envisioned as a hierarchical system with the controller as the highest layer, the interface in the middle, and the plant as the lowest layer.

2. AN APPROACH TO MODELING HYBRID CONTROL SYSTEMS

A hybrid control system can be seen as having three layers, one layer for each of the three essential parts of a hybrid control system. The top layer is made up of the controller which acts as a supervisor and decision maker. The bottom layer consists of the plant, which represents the system being controlled. And the middle layer is an interface which allows two-way communication between the top and bottom layers.

The model described here, and used throughout this work, represents a refinement of the above representation. It is made up of four interacting blocks, each with a specific form and function. The first of these four blocks represents the top layer or controller, and it is therefore called the controller. The second block represents the bottom layer and is the plant. Each of the remaining two blocks represent a portion of the interface layer and they are called the actuator and aggregator blocks. The actuator relays commands from the controller to the plant, and the aggregator carries information in the opposite direction, from the plant to the controller. Figure 2.1 shows the arrangement of these four blocks as well as the names and locations of the signals, represented by arrows, which provide communication between the blocks. Each of the blocks and its associated signals will be described now.

2.1 Plant

The plant is modeled as a time-invariant, discrete-time system (DTS). This can be used to represent an actual discrete-time plant, a sampled continuous-time plant, or any type of system which evolves in real time and can be suitably approximated by a time-invariant, discrete-time system. Formally the plant is specified by the following equations:

$$x(k+1) = f(x(k), r(k)) \tag{1}$$

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

where

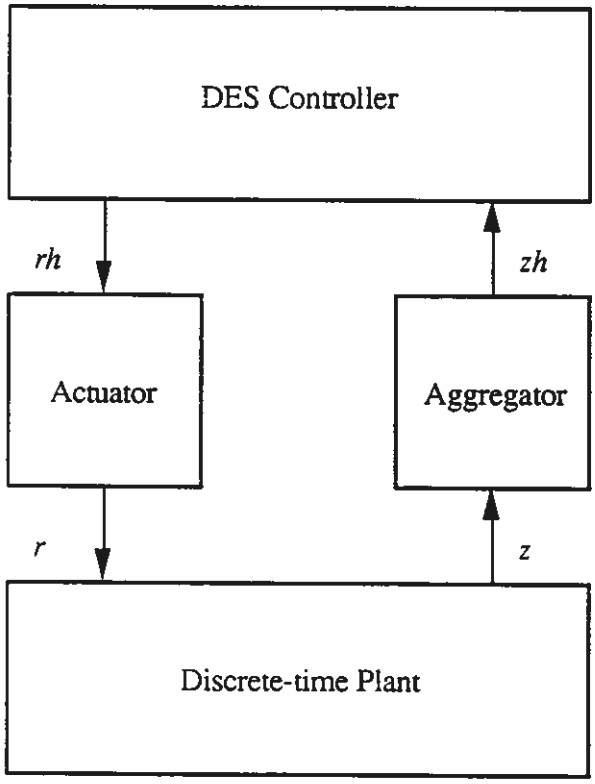

Figure 2.1

$$f : \mathbf{R}^n \times \mathbf{R}^m \rightarrow \mathbf{R}^n \text{ and } g : \mathbf{R}^n \rightarrow \mathbf{R}^p$$

are the system functions, and

$x(k) \in \mathbf{R}^n$	plant state
$r(k) \in \mathbf{R}^m$	plant input
$z(k) \in \mathbf{R}^p$	measured plant output

are discrete-time signals. For the purposes of this work we assume that $z(k) = x(k)$ and therefore $p = n$. It should be pointed out that the plant could easily be changed to represent a continuous-time system by changing Equation (1) to $\dot{x} = f(x, r)$.

2.2 Aggregator

The aggregator contains the portion of the interface layer responsible for relaying information from the plant to the controller. The input to the aggregator is the measured

output, z, which in our case is equal to the state of the plant, x, and the output is zh. The task of the aggregator, therefore, is to convert the discrete-time signal, z, to a discrete-event signal, zh. To accomplish this the aggregator partitions the state space of the plant into a number of regions, where each region is associated with an event, $e_i \in E$, where E is a set of possible events. It is possible that more than one closed region can be associated with the same event. Whenever the state of the plant moves from one region to another, the aggregator outputs the event associated with the new region. The aggregator will then remain silent until the plant state moves into another new region.

Mathematically the aggregator can be specified by a function, α, which maps each point in $\mathbf{R}^n$ to an event in E. α is not an injective function and it induces equivalence classes in $\mathbf{R}^n$. Each of these equivalence classes forms one or more contiguous regions in $\mathbf{R}^n$ which partition the state space, and are called *aggregator regions*.

Definition 3.1: The *aggregator regions* generated by the function α in the space $\mathbf{R}^n$ form a partition of $\mathbf{R}^n$ such that two points, $r_1, r_2 \in R^n$, are in the same aggregator region iff there exists a curve in $\mathbf{R}^n$ from r_1 to r_2 such that $\alpha(r) = \alpha(r_1) = \alpha(r_2)$ for any point, r, on the curve. The set of aggregator regions shall be denoted by **A**. □

Thus

$\alpha : \mathbf{R}^n \rightarrow E$	alpha
$z(k) \in \mathbf{R}^n$	aggregator input (plant state)
$zh_i \in E$	aggregator output (DES controller input)

The action of the aggregator can be written as

$$zh = sp[\alpha(x)] \tag{3}$$

where we abuse notation slightly by allowing α to operate on a signal. *sp* is a function which suppresses repeated events so that the aggregator generates an event only when the plant state first enters a region. The function α is intended to induce equivalence classes which generate reasonably large regions in the state space, large enough such that the plant state can evolve for a number of time steps and remain in the same region. As a result, zh will be a discrete and asynchronous signal which has a value only at the times that z moves from one equivalence class of α to another. To help clarify, Figure 2.2 shows a sample time line comparing z(k), α(z(k)), and zh for some system. The model for the aggregator

may seem a bit restrictive, but its ability to handle a variety of systems will be discussed later.

k	1	2	3	4	5	6
z(k)	0	1	2	4	3	1
$\alpha(z(k))$	e_1	e_1	e_3	e_3	e_3	e_1
zh_i	e_1		e_3			e_1
i	1		2			3

Figure 2.2: Sample Aggregator Time Line

2.3 DES Controller

In a hybrid control system, the controller acts as a supervision and decision making system. It is an event driven system, meaning it reacts to discrete asynchronous input and generates discrete asynchronous output. In this case, the input and output consist of an asynchronous sequence of symbols which are called events. In the case of input, each of these events represents a condition which can be reported to the controller, and in the case of output, the events represent commands which can be issued by the controller. This type of system is referred to as a discrete-event system (DES), and the input and output are called discrete-event signals. Several modeling strategies have been developed for discrete-event systems. Here the controller is modeled by a deterministic automaton which may have an infinite number of states. The automaton is specified by a quintuple, $\{S, E, C, \delta, \phi\}$, as follows

$S = \{s_1, s_2, s_3, \ldots\}$	set of controller states
$E = \{e_1, e_2, e_3, \ldots\}$	set of events (controller inputs)
$C = \{c_1, c_2, c_3, \ldots\}$	set of commands (controller outputs)
$\delta : S \times E \rightarrow S$	state transition function
$\phi : S \rightarrow C$	output function

As mentioned previously, the automaton may have an infinite number of states, in fact it may have an infinite number of events and commands as well. This model can be used to represent a wide variety of discrete-event systems, it follows the model found in [14] and is similar to those found in [20, 26].

The input signal to the controller is *zh* and the output signal is *rh*. The names reflect the fact that they are the high level signals corresponding to the discrete-time signals *z* and *r*. Whenever an event is received by the controller on *zh*, the controller immediately undergoes a state transition determined by the state transition function:

$$t_i = \delta(t_{i-1}, zh_i) \tag{4}$$

where $t_i \in S$, $zh_i \in E$, and i is a time index. With each state transition, the controller generates an output by the output function.

$$rh_i = \phi(t_i) \tag{5}$$

where $zh_i \in C$. Note that since the state transitions and outputs of the DES controller occur imediately upon receiving an input event, there is no need to include time in the model. The delay between successive state transitions and outputs is determined completely by the input signal, *zh*. Also note that the subscript on a signal, e.g. zh_i, is used to indicate the logical order of the event, state, or command carried on that signal. The subscript on a set member, e.g. e_i, however, refers to a particular member of the set.

2.4 Actuator

In this model the actuator receives a discrete-event signal, *rh*, carrying commands from the controller. The unit in turn produces a discrete-time signal, *r*, which serves as the input to the plant. To produce *r*, the actuator must be programmed with a method to translate each command in C into a numeric value for *r*. For command sets with a finite number of elements this method could consist of a table lookup function, but for infinite command sets another method of translation must be used. The signal, *r*, is piecewise constant for it depends only on the most recent controller command and will not change until the next command is received. In this way the actuator operation is similar to a hold circuit. The actuator must also have access to the clock signal associated with the plant because *r* is synchronized to the plant. The action of the actuator unit is determined by the function γ, where

$\gamma : C \rightarrow R^m$	the actuator function
$r(k) \in R^m$	the actuator output
$rh_i \in C$	the actuator input

and described by

$$r = \text{hold}[\gamma(rh)] \tag{6}$$

where we allow γ to operate on a signal and *hold* is a function which retains the current value until a new argument appears. Figure 2.3 shows a sample time line comparing rh, $\gamma(rh)$, and r.

i	1		2	3		
rh_i	c_5		c_1	c_4		
$\gamma(rh_i)$	3.2		4.6	4.8		
r(k)	3.2	3.2	4.6	4.8	4.8	4.8
k	1	2	3	4	5	6

Figure 2.3: Sample Actuator Time Line

2.5 Example: Double Integrator

In this example the HCS model will be used to model a system consisting of a sampled continuous-time plant, which is a double integrator, and a simple DES controller which is attempting to stabilize it. The algorithm used by the controller is to apply a negative input to the plant whenever the plant output is positive and increasing, a positive input whenever the output is negative and decreasing, and a zero input at all other times. This algorithm was chosen because it works and can be implemented by a DES.

The first step in modeling this system is to convert the sampled continuous-time plant to a discrete-time plant. A continuous-time double integrator is as follows

$$\frac{dx}{dt} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} r(t) \tag{7a}$$

$$y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} x(t) \tag{7b}$$

The zero order hold equivalent of this system is:

$$x(k+1) = \begin{bmatrix} 1 & h \\ 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} h^2/2 \\ h \end{bmatrix} r(k) \tag{8a}$$

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

$$y(k) = \begin{bmatrix} 1 & 0 \end{bmatrix} x(k) \tag{8b}$$

where h is the sampling period, and as always

$$z(k) = x(k). \tag{8c}$$

The next step is to model the interface which conveys information to the controller as an aggregator. Notice that the condition "positive and increasing output" is true exactly when the discrete-time plant state is in the first quadrant, likewise the condition "negative and decreasing output" is true exactly when the state is in the third quadrant. Therefore an aggregator which detects whenever the plant state enters or leaves one of these two regions would provide the necessary information to the DES controller. Such an aggregator would partition the state space into the four quadrants, but it would only produce three different events because the same event would be associated with both quadrant two and quadrant four. The regions and associated events for this aggregator are shown in Figure 2.4.

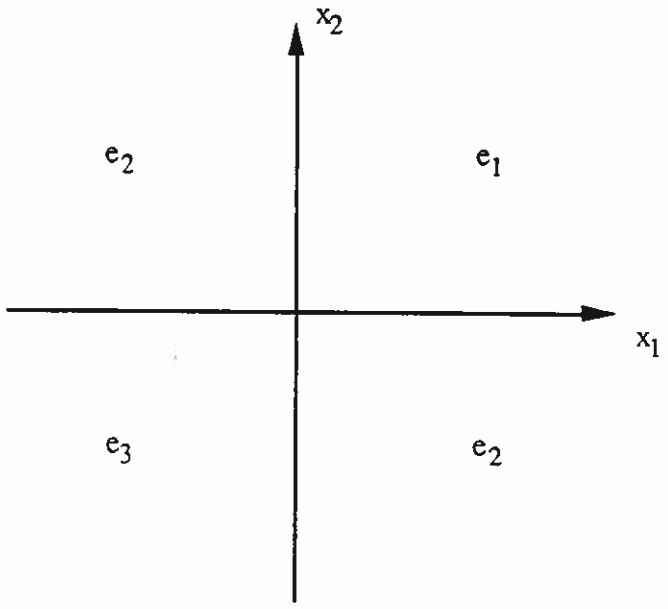

Figure 2.4

The aggregator regions are chosen depending on the requirements of the DES controller as was done in this example. Conversely, the type of control law used is influenced by the range of possible aggregator regions.

With the plant and aggregator modeled, it is quite simple to model the controller as an automaton with the desired behavior. Three states are required in the controller, one for each of the three desired commands. The state transition graph of the controller is shown in Figure 2.5 and $\phi(s_i) = c_i$ for $i \in \{1, 2, 3\}$. Lastly, the portion of the interface which translates controller commands into plant input must be modeled as an actuator. This is done by assigning the function γ as follows:

$$\gamma(c_1) = 10,\ \gamma(c_2) = -10,\ \gamma(c_3) = 0$$

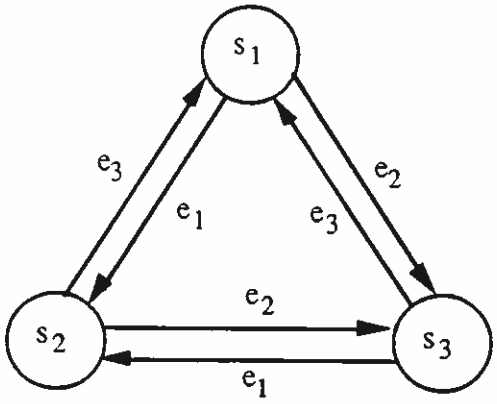

Figure 2.5

Figure 2.6 shows the results of simulations of this system with various initial states for the plant, verifying that it is stable, (though not strictly since the state will hover in an arbitrarily small area around the origin.)

3. MODELING FLEXIBILITY

At first glance, the proposed hybrid control system model may seem rather limited in its capability to represent systems because of the specific nature of the interface. The actuator generates only step inputs for the plant, it is incapable of providing a sinusoidal input, for example. The aggregator is limited as well, for it cannot provide any information other than that found in the current output, which in our case is the current state, of the plant. To appreciate the flexibility of the HCS model it is important to recall that the plant represents the entire continuous-state, dynamic portion of the system and may be made up of several parts including a continuous-time or discrete-time controller. Also, the interface of the HCS is simply a pair of static mappings between the discrete-time and discrete-event

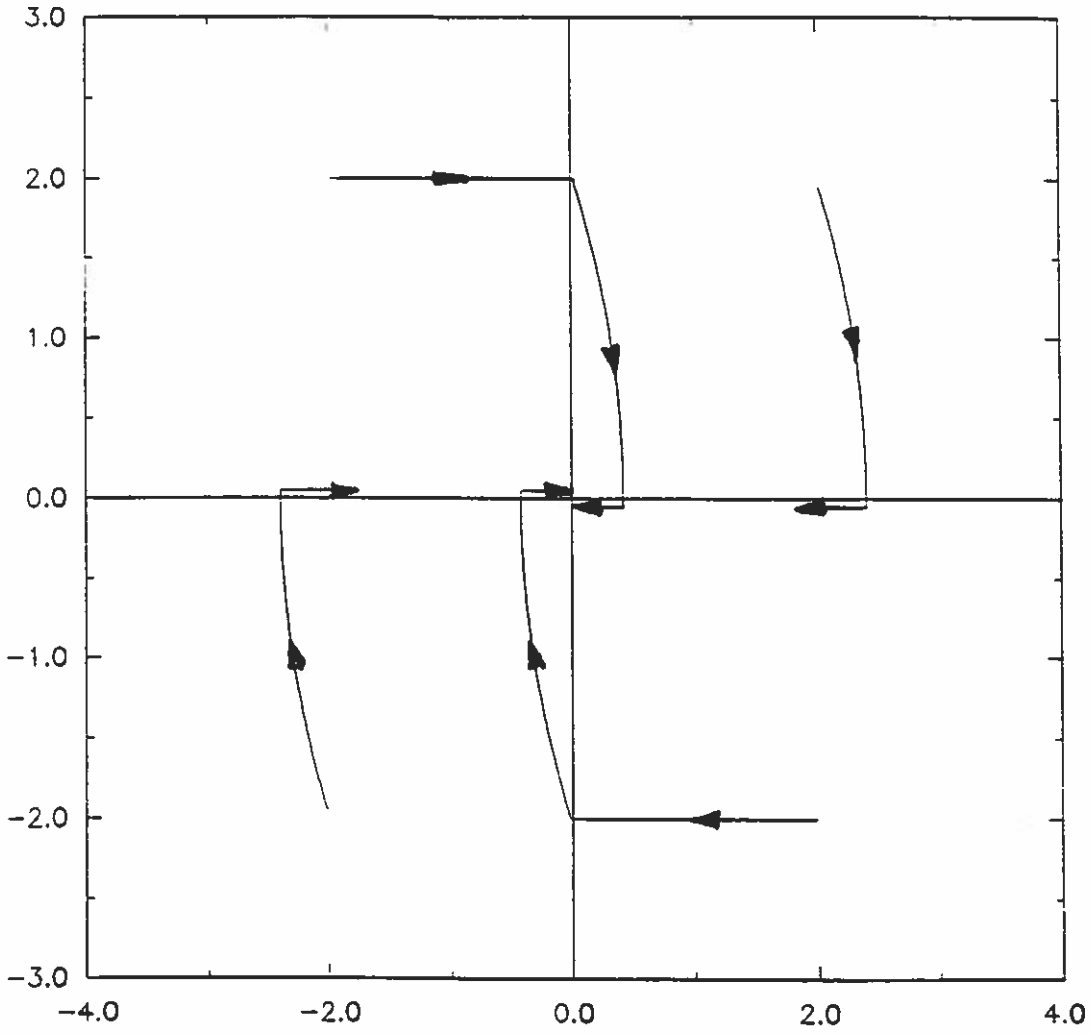

Figure 2.6

worlds, it is not intended to include any states of its own. In this section it will be shown that this model can represent a wider variety of hybrid control systems than is apparent at first glance.

3.1 Actuator and Plant Inputs

The reason the actuator in the HCS model is only capable of producing step functions is that the DES controller is idle between subsequent events. Therefore when the DES controller issues a command it can provide no new information to the actuator until the next command, and consequently the actuator is 'stuck' at that setting until the next command. To give the actuator additional signal generating capabilities, would be to model it with components which are dynamic and therefore belong in the plant.

Many times it is necessary to drive a plant with some other type of input such as a ramp or sinusoid. To accomplish this using the proposed HCS model, a model of a system which generates the ramp or sinusoid must be included in the plant model. Note that a theoretical actuator which is capable of generating various types of input signals can be realized by using two blocks in series: an actuator of the type found in the HCS model followed by a dynamnic system which responds to that actuator with the desired signals. This setup can be made to fit the current HCS model by including the dynamic portion of the theoretical actuator in the plant. The addition of the dynamic portion of the theoretical actuator to the plant would form a new plant with an augmented state space.

Example

Consider the following discrete-time plant:

$$A = \begin{bmatrix} 0 & 1 \\ .4 & -.3 \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{9}$$

We would like to drive the above plant with a sinusoidal input. To accommodate this system in the HCS model first divide the sinusoid generator into two parts: the first part outputs a step function and forms the actuator block of the HCS model, the second part is a linear system which converts the step function into a sinusoid and is incorporated into the plant of the HCS system. The following system could be used for the second part of the sinusoid generator

$$A_2 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \qquad B_2 = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \tag{10a}$$

$$C_2 = [\ 1 \quad 0\ ] \tag{10b}$$

By combining the original plant with this system, the following augmented plant is obtained

$$A_{aug} = \begin{bmatrix} A & BC_2 \\ 0 & A_2 \end{bmatrix} \qquad B_{aug} = \begin{bmatrix} 0 \\ B_2 \end{bmatrix} \tag{11}$$

The state vector of the original plant is found in the upper n elements of the state vector of the augmented plant.

3.2 Reconfigurable Plant

It has been suggested that the DES controller in a hybrid control system could be used to select different feedback laws for the plant [12], or even different plant dynamics [18]. This is quite different from merely controlling the input to the plant, but it can still be accomplished using our model for a hybrid control system. The plant input, *r*, must be used to accomplish whatever reconfiguration is required in the plant. This will tend to produce a plant model which is highly nonlinear, but a system which allows reconfiguration is generally nonlinear and this nonlinearity will have to be dealt with somewhere.

Example

Suppose we would like to model a hybrid control system which includes a first order plant which can be reconfigured by changing the response time. Consider a plant which can behave as

$$x(k+1) = [.8]\ x(k) + [.2]\ r(k) \tag{12a}$$

or

$$x(k+1) = [.9]\ x(k) + [.1]\ r(k) \tag{12b}$$

depending how it is configured. This plant would be modeled as follows in the hybrid control system model

$$x(k+1) = [1 - r_1(k)]\, x(k) + [r_1(k)]\, r_2(k) \tag{13}$$

where r_1 and r_2 are the elements of the plant input vector r which control the response time and provide the conventional input respectively. To accurately model the system the actuator will only be capable of generating two values for r_1, i.e. 0.2 and 0.1.

3.3 Aggregator and Controller Inputs

It is not uncommon in hybrid control systems for the DES controller to require information about the system it is controlling which is not included in the current state of that system. An example of this would be an event which occurs whenever a particular state of the plant has been increasing for more than 10 seconds, regardless of its actual value. It may be tempting to model the aggregator so that it can generate more complex events such as this, but that would require making the aggregator a dynamic system. The solution is to model any dynamic sensing systems as part of the plant, just as dynamic signal generating systems are modeled in the plant.

3.4 Example: Surge Tank

To further show the generality of the hybrid control system model, consider a surge tank with a DES controller. This example will demonstrate a hybrid control system in which the plant is a continuous-time, non-linear system, and the DES controller has an infinite number of states.

As stated, the plant in this example consists of a surge tank, which is a water tank with an inlet valve and an outlet valve. The outlet valve is opened and closed unpredictably by some user, while the inlet valve is connected to an actuator which is positioned by the controller. Control of the surge tank involves maintaining the water level of the tank within a preset range. The behavior of the surge tank can be expressed by the following first-order, nonlinear, differential equation [15]:

$$dh(t)/dt = \{F(t) - 1000c(t)[h(t)]^{1/2}\}/\rho A \tag{15}$$

where $\rho = 99.8$, the fluid density of water, and $A = 1$, the cross-sectional area of the tank. $h(t)$ is the height of the water, $c(t)$ is the area of the outlet valve opening, and $F(t)$ is the flow through the inlet valve. For this tank, $0 \le h(t) \le 1$, $0 \le c(t) \le 0.1$, and $0 \le F(t) \le 173$.

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

To model the surge tank as the plant of a hybrid system, the plant state, x(t), will be defined as the height of the water, the plant input will be the inlet flow rate, and the zero order hold equivalent of the plant will be used to model the sampled plant. Thus

$$\begin{aligned} x(k+1) &= f(x(k), r(k)) \\ &= x(k) + T\{r(k) - 1000c(t)[x(k)]^{1/2}\}/99.8 \end{aligned} \tag{16}$$

where

$x(k) = h(kT)$, $r(k) = F(kT)$, $z(k) = x(k)$, and T is the sampling interval.

The control scheme used to regulate the level of the water in the tank is designed to maintain the height between 0.7 and 0.9 To accomplish this the controller will open the inlet valve a preset amount if the water level drops below 0.8, and double the opening each time the water level falls half the remaining distance to 0.7. If the water level reaches 0.9 the controller will shut off the inlet valve. To supply the controller with the necessary information to implement this control scheme, the aggregator divides the state space into an infinite number of regions as follows:

$$\alpha(x(k)) = e_i \text{ when } 0.7 + 0.2(0.5)^{i-1} \le x(k) < 0.7 + 0.2(0.5)^{i-2} \tag{17}$$

The controller is described by the infinite state automata shown in Figure 3.1, where $\phi(s_i) = v_i$, and the actuator behaves as follows:

$$\gamma(c_1) = 0 \text{ and } \gamma(c_i) = (2)^{i-2},\ i > 1 \tag{18}$$

This particular control scheme was selected to reduce fluctuations in water level while also keeping the rate of commands to the actuator low. The scheme was not selected in any systematic way, however.

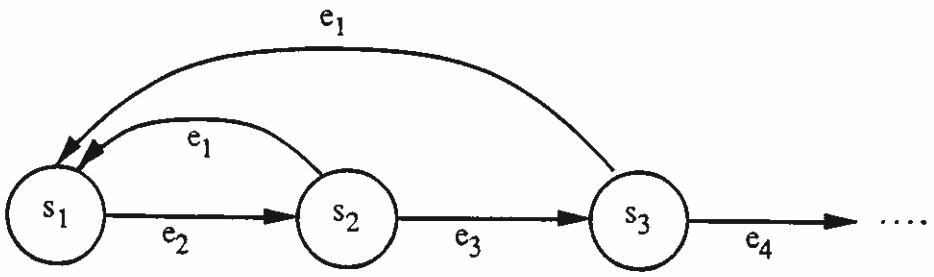

Figure 3.1

It would appear that this system has an unlimited amount of control action, because the actuator is modeled as if the inlet valve can be opened infinitely far. Because this would not be the case with an actual physical system, this model is only valid over a certain range of states. It is very common in conventional control to use models that have a region of validity so it should not be unexpected to find this situation in an HCS as well.

The behavior of this system is illustrated in Figure 3.2, which shows the water level, the controlled inlet flow, and the outlet flow. The system behaves appropriately.

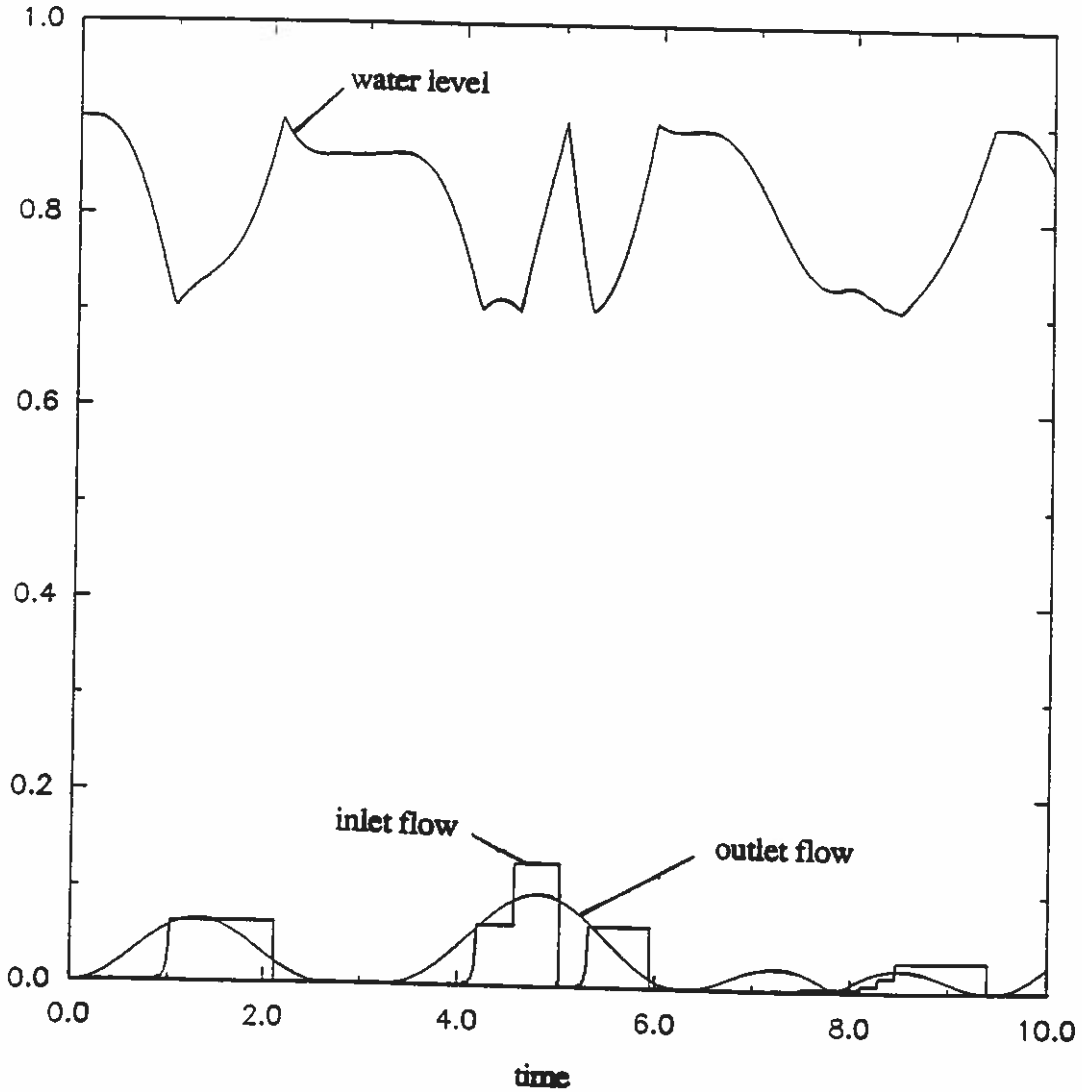

Figure 3.2

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

4. A DISCRETE EVENT SYSTEM MODEL FOR HYBRID CONTROL SYSTEMS

The difficulty in analysis presented by Hybrid Control Systems is due to the incompatibility between the models employed in the various blocks which make up the HCS, namely the DES controller and the DTS plant. A scheme in which a single modeling strategy is used would allow analysis which is impossible to perform with the current HCS model. In this section, a method for representing the entire HCS as two interacting DES's will be described. The benefits and limitations of the new model will be described as well as the relationship between the results obtained from the new model and the nature of the original model.

4.1 DES Equivalent Plant

Notice that from the point of view of the DES controller the remainder of the HCS appears as a DES (see Figure 4.1). This is because the input and output of the actuator, plant, and aggregator, taken together, are discrete-event signals. This allows the HCS model described in Section 2 to be converted into a discrete-event system model by combining the actuator, plant, and aggregator into a single block. The discrete-event controller remains the same in the new model and the newly formed block constitutes another DES which shall be referred to as the *DES equivalent plant.* This new model is shown in Figure 4.2. Like the controller, the DES equivalent plant is modeled by an automaton, but unlike the controller this automaton might not be deterministic and the state transitions do not occur instantly when an input is received. However, one can determine exactly when transitions occur becuase of the time dependence of the underlying system. It is specified by the quintuple $\{P, C, E, \psi, \phi_p\}$ as follows

$P = \{p_1, p_2, p_3, \ldots\}$	the set of states
$C = \{c_1, c_2, c_3, \ldots\}$	the set of inputs (controller commands)
$E = \{e_1, e_2, e_3, \ldots\}$	the set of outputs (events)
$\psi : P \times C \rightarrow P(P)$	the state transition function
$\phi_p : P \rightarrow E$	the output function

The set of DES equivalent plant states, P, is based on the aggregator regions generated by the equivalence classes of α in the state space of the original plant. Each state in P is uniquely associated with exactly one of these regions, i.e. there is a one-to-one

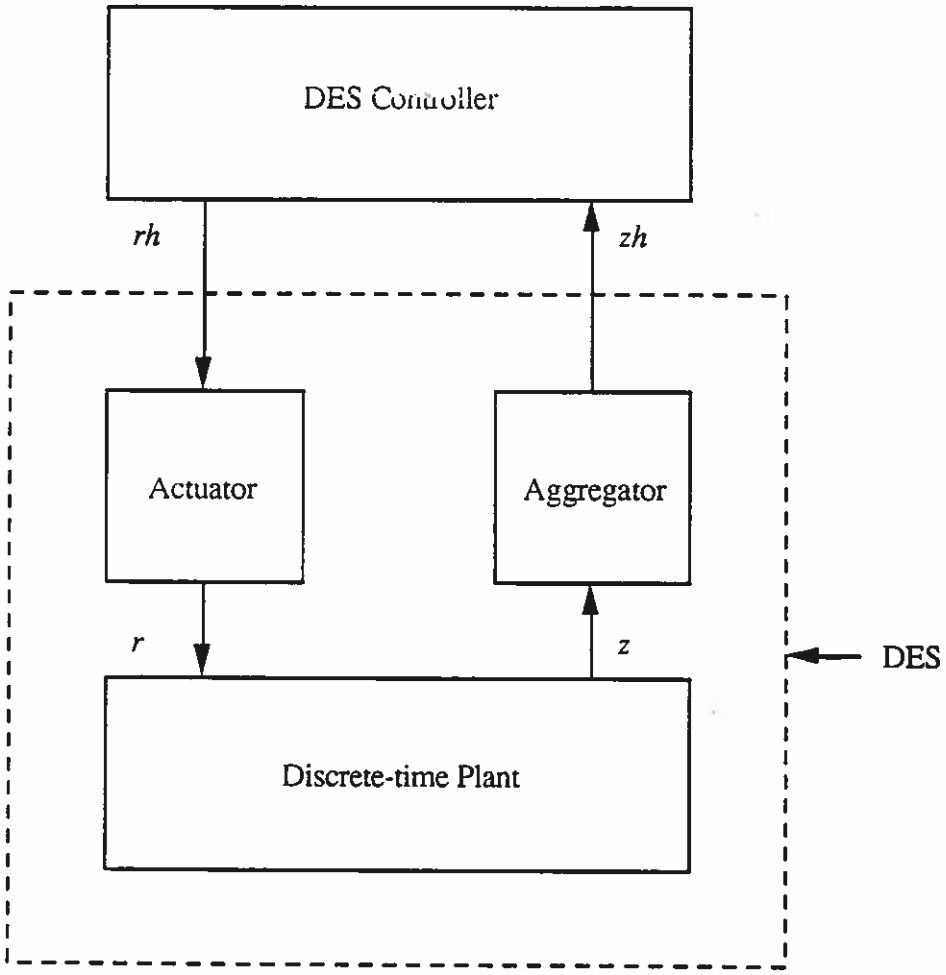

Figure 4.1

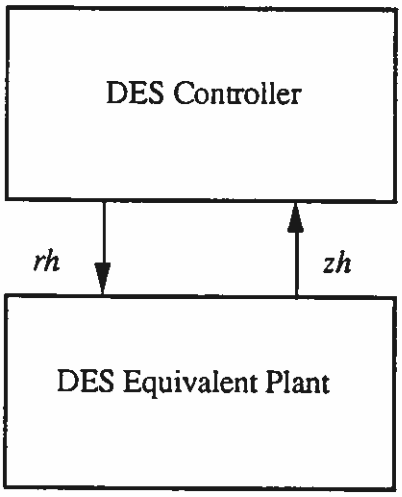

Figure 4.2

correspondence between the members of the set P and the set A. The sets E and C are the same sets as those involved with the original HCS model and defined in Section 2. The function ϕ_p plays the same role as α because it maps the plant state to an event in E. This leaves ψ to be determined. Notice that the state transition function, ψ, maps to an element of the power set of P, indicating that the automaton is nondeterministic in general. The function ψ is also a partial function in general.

We will now derive the state transition function, ψ, given information about the Hybrid Control System. We use $t_i \in P$ to denote the state of the DES equivalent plant at $time_h(i)$.

Given:

$$zh_i = sp[\alpha(x(k))] \tag{26a}$$

$$r(k) = hold[\gamma(rh_i)] \tag{26b}$$

then

$$\psi(t_i, rh_i) = \phi_p^{inv}(zh_{i+1}) \tag{27a}$$

$$= \phi_p^{inv}(sp[\alpha(f(x(k), r(k))]) \tag{27b}$$

$$\psi(t_i, rh_i) = \phi_p^{inv}(sp[\alpha(f(\alpha^{inv}(\phi_p(t_i)), \gamma(rh_i)))]) \tag{27c}$$

where α^{inv} is inverse image under α and $time_h(i) = time(k)$. The DES equivalent plant is non-deterministic in general because α^{inv} maps a single event to a subset of $\mathbf{R}^n$

$$\alpha^{inv} : E \rightarrow \mathbf{P}(\mathbf{R}^n)$$

which will typically result in a set, containing more than one state, as the value of $\psi(t_i, rh_i)$. In addition, ϕ_p^{inv} may map a single event to several DES equivalent plant states

$$\phi_p^{inv} : E \rightarrow \mathbf{P}(P)$$

resulting in an even larger set for $\psi(t_i, rh_i)$. For values of t_i and rh_i for which ψ (or any other function) is undefined the following notation may be used

$$\psi(t_i, rh_i) \in \{void\} \tag{28}$$

The DES equivalent plant represents the extent to which the DES controller in a particular hybrid control system can regulate the plant. The control of the plant is limited by two factors. First, the plant input is limited to those set points which can be generated by the actuator. This limitation reduces the number of possible state trajectories which can

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

be realized in the plant, and this is reflected in the DES equivalent plant by a reduced number of possible state transitions.

The second limitation on the control of the plant is caused by the aggregator. The aggregator does not provide the DES controller with the exact state of the plant, but rather with a sort of approximation. Thus, based solely on information from the aggregator, the response of the plant to a given input is not known precisely. This effect appears in the DES equivalent plant as nondeterministic state transitions. That is, for a given input and state, the DES equivalent plant may have several possible transitions. The example in Section 4.3 clearly shows this effect. First however, we derive the conditions for determinism.

4.2 Determinism in the DES Equivalent Plant

We would like to know what restrictions can be placed on the hybrid system to ensure that the DES equivalent plant can be represented by a deterministic automaton. This is because the exact behavior of the system can only be determined if the system is deterministic.

Definition 4.1 : A DES equivalent plant is said to be deterministic iff $\forall\ p_i \in P, c_j \in C$, we have $\psi(p_i, c_j) \in \{p_l, \text{void}\}$ where $p_l \in P$. □

If ψ is to represent a deterministic automaton, then the expression on the right side of equation (25b) must take on values which are sets with only one member. Theorem 4.1 below gives the condition for a deterministic DES equivalent plant under the restriction that ϕ_p is a bijective function.

Theorem 4.1 : Given ϕ_p bijective, the DES equivalent plant will be deterministic iff

$$\forall\ zh_i \in E, rh_i \in C\ \exists\ e \in E \text{ s.t. } \alpha(x(k+1)) \in \{zh_i, e\}$$

where

$$x(k) \text{ is any} \in \mathbf{R}^n \text{ s.t. } \alpha(x(k)) = zh_i$$

and

$$x(k+1) = f(x(k), \gamma(rh_i)).$$

Proof

To prove sufficiency, assume the condition holds then by equation (27b)

$$\psi(t_i, rh_i) = \phi_p^{inv}(sp[\alpha(f(x(k), r(k))])$$

and since $r(k) = \gamma(rh_i)$ and $x(k+1) = f(x(k), \gamma(rh_i))$

$$\psi(t_i, rh_i) = \phi_p^{inv}(sp[\alpha(x(k+1))]).$$

Now, by the theorem we have

$$\psi(t_i, rh_i) \in \{\phi_p^{inv}(sp[\alpha(x(k))]),\ \phi_p^{inv}(sp[e])\},$$

and since *sp* suppresses repeated events

$$\psi(t_i, rh_i) \in \{void, \phi_p^{inv}(sp[e])\}$$
$$\psi(t_i, rh_i) \in \{\phi_p^{inv}(sp[e])\}.$$

Recall that ϕ_p is bijective and therefore $\psi(t_i, rh_i)$ can take at most one value and ψ is thus deterministic.

To prove that the condition of Theorem 4.1 is necessary, we will start by assuming a system is deterministic and then showing that the theorem holds. Assuming the system is deterministic implies there is at most one possible value for ψ, i.e.

$$\psi(t_i, rh_i) \in \{void, p\},$$

and by equation (27b)

$$\phi_p^{inv}(sp[\alpha(f(x(k), r(k))]) \in \{void, p\}.$$

If we define $e \equiv \phi_p(p)$, then

$$sp[\alpha(f(x(k), r(k))] \in \{void, e\},$$

and since *sp* suppresses repeated events,

$$\alpha(f(x(k), r(k)) \in \{zh_i, e\}$$

follows because zh_i is the only event which can be repeated and be suppressed. Finally, by equation (1)

$$\alpha(x(k+1)) \in \{zh_i, e\}$$

and the theorem is proven. □

If the restriction that ϕ_p be a bijective function is removed then equivalence classes of α will not necessarily coincide with the aggregator regions and a more general theorem is required to describe a deterministic system.

Theorem 4.2 : The DES equivalent plant will be deterministic iff

$$\forall\ a_1 \in \mathbf{A}, rh_i \in C\ \exists\ a_2 \in \mathbf{A}\ \text{ s.t. } x(k+1) \in a_1 \cup a_2$$

where

$$x(k) \in a_1$$

and

$$x(k+1) = f(x(k), \gamma(rh_i)).$$

Proof

Recall that **A** is the set of aggregator regions and that there is a one-to-one correspondence between the aggregator regions and the states of the DES equivalent plant. Thus by making the correspondence

$$p_i \Leftrightarrow a_1 \text{ and } p_j \Leftrightarrow a_2,$$

the theorem can be rewritten as

$$\forall\ p_i \in P, rh_i \in C\ \exists\ p_j \in P\ \text{ s.t. } p \in \{p_i, p_j\}$$

where p_i and p are the DES equivalent plant states at time(k) and time(k+1) respectively. Thus Theorem 4.2 is a basically a restatement of the definition a deterministic DES equivalent plant, saying that if the state changes, there is a unique state to which it will change. □

The restriction of Theorems 4.1 and 4.2 guarantee that if the current value of *zh* and *rh* are known, then the next value of *zh* can be determined uniquely and thus ψ is a function in the conventional sense because it returns only one value. Another way of stating the theorems is that, under any given value of *rh*, all real-time plant trajectories in any given aggregator region lead to the same subsequent aggregator region. This restriction is necessary if the plant is to be controlled in an arbitrary way because a controller cannot be designed to arbitrarily control a plant which reacts unpredictably.

Often the restrictions of Theorems 4.1 and 4.2 are unnecessary. If a particular control objective is to be attained, the non-deterministic transitions may not be relevant to the design of the controller. In any case, the state transition function of a DES equivalent

plant can be separated into a deterministic part and a non-deterministic part. The deterministic part will be a partial function in general because it may not be defined for all states and inputs. Let the deterministic part of the state transition function be ψ_d such that

$$\psi_d : P \times C \rightarrow P$$

where

$$\psi_d(t_i, rh_i) \in \psi(t_i, rh_i)$$

and ψ_d is defined only for those pairs (p_i, c_j) where $\|\psi(p_i, c_j)\| = 1$. The notation $\|X\|$ refers to the cardinality of set X. Furthermore if the restriction of Theorem 4.2 is met, then

$$\psi(t_i, rh_i) = \{\psi_d(t_i, rh_i)\}.$$

The deterministic portion of the DES equivalent plant, described by the function ψ_d, represents in a sense the controllable portion of the DES equivalent plant. The state transitions included in ψ_d can be enabled by the controller and there is no uncertainty in which transition will occur. If these deterministic transitions are sufficient to achieve the control objectives for the system, then the non-deterministic part of the DES equivalent plant is of no consequence.

4.3 Example: Double Integrator

The double integrator system from Section 2.5 is a case of a Hybrid Control System with a non-deterministic DES equivalent plant. Figure 4.3 shows the state trajectories of the real-time plant under the three available control inputs, along with the aggregator regions. The trajectories which are roughly parallel are under the same input, *r*. As can be seen the aggregator regions this example do not lead to a deterministic DES plant. The reason for this situation is that the aggregator does not preserve enough information about the state of the actual plant. Therefore, by changing the aggregator to add more regions as shown in Figure 4.4, the system can be made to have a deterministic DES equivalent plant. The new aggregator is a refinement of the original which preserves more information about the actual plant state and leads to the deterministic DES equivalent plant shown in Figure 4.5. The original aggregator was sufficient for the stabilizing DES controller of the original example, but the new aggregator will allow any controller to be designed without the worry of unpredictable plant behavior.

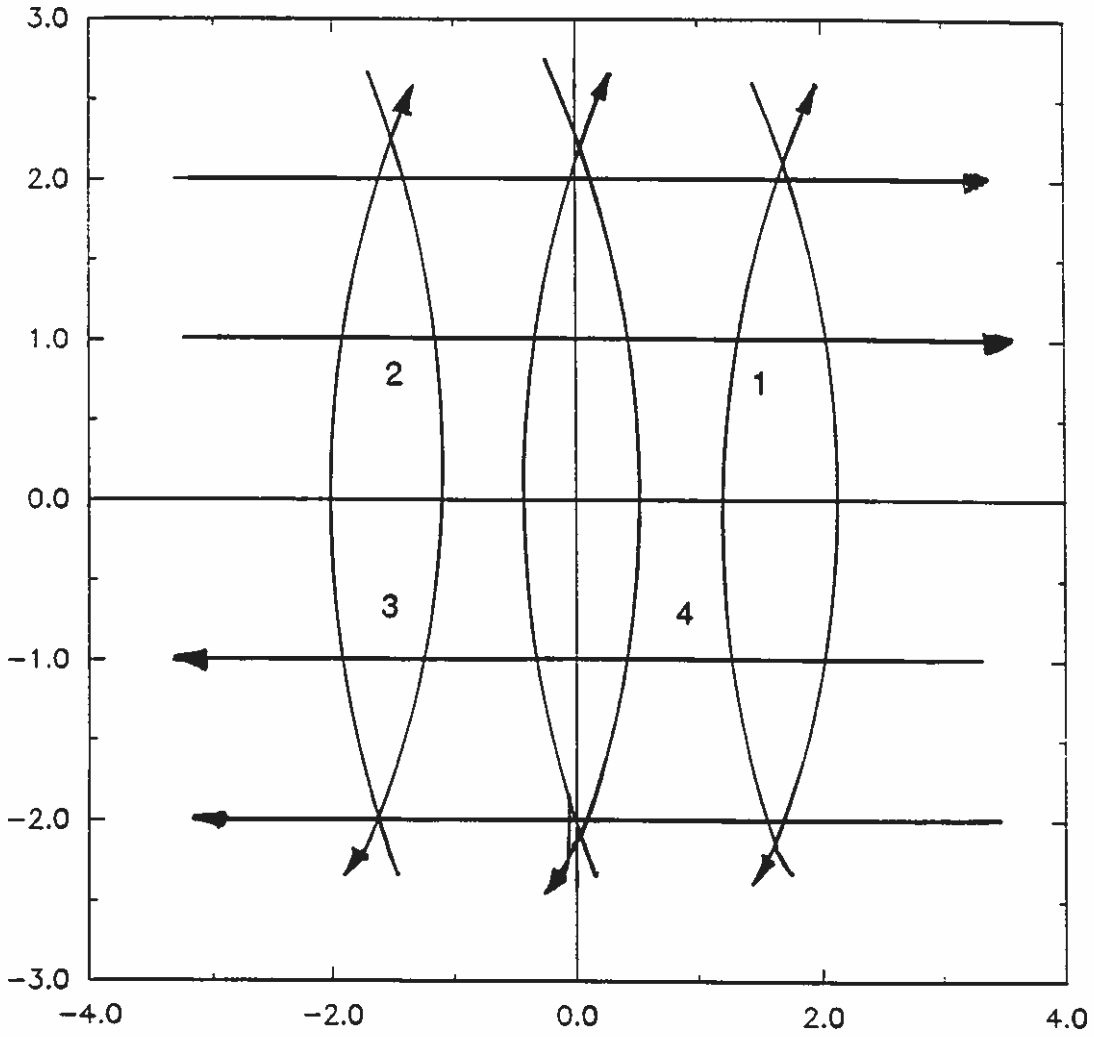

Figure 4.3

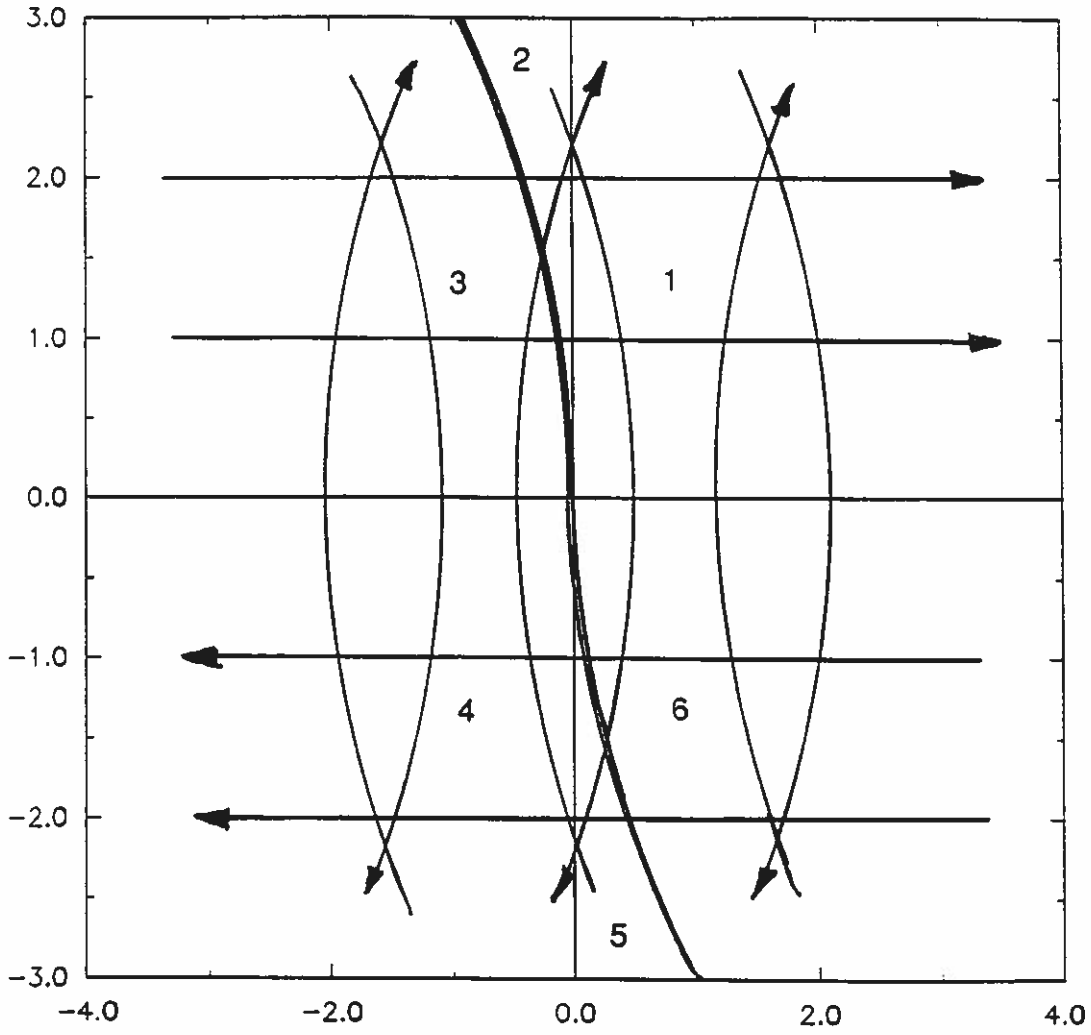

Figure 4.4

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

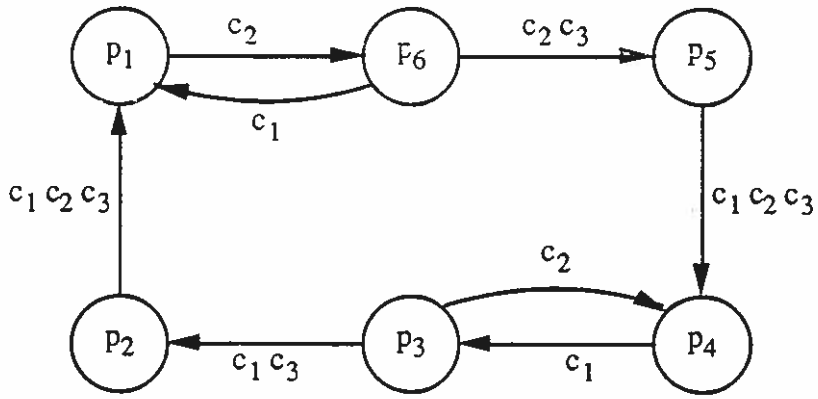

Figure 4.5

4.4 Example: Surge Tank

As a another example of the alternate representation consider again the HCS containing the surge tank found in Section 3.4. This system has unique features because the real-time plant has an unmodeled component, namely the user which opens and closes the outlet valve. This unmodeled component prevents the state trajectories from being determined exactly as a function of the input *r*. The only thing that is certain is that the state will be increasing for inputs greater than r(k) = 100, because that exceeds the maximum drain due to the user. With this information the DES equivalent plant can be found for this system and it even has a deterministic part. Figure 4.6 shows the DES equivalent plant. The fact that the only deterministic transitions correspond to inputs of c_i where i = 1 or i > 9, means that a DES controller for this system must use these inputs or risk losing control of the plant.

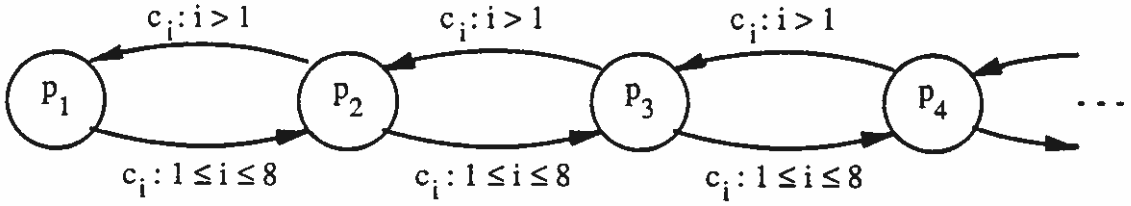

Figure 4.6

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

5. ON THE STABILITY OF HYBRID CONTROL SYSTEMS

In this section, a definition of stability which can be applied to Hybrid Control Systems is presented, and the implications of this notion of stability, for the individual blocks of the HCS, are described. The definition of stability used here is taken from the Lyapunov stability for Discrete-Event Systems which was developed in [16], and this criteria is applied to the DES equivalent of the HCS. The work described in this chapter is by no means complete, but represents some first steps toward developing a theory of stability for hybrid control systems.

5.1 Lyapunov Stability for Discrete Event Systems

A Discrete-Event System, such as the DES equivalent plant of a HCS as defined in Section 4, can be described by a quintuple $\{P, C, E, \psi, \phi_p\}$, where

$P = \{p_1, p_2, p_3, \ldots\}$	the set of plant states
$C = \{c_1, c_2, c_3, \ldots\}$	the set of plant inputs
$E = \{e_1, e_2, e_3, \ldots\}$	the set of plant output
$\psi : P \times C \rightarrow P(P)$	the state transition function
$\phi_p : P \rightarrow E$	the plant output function.

The definitions and theorems for the Lyapunov stability of discrete-event systems found in the remainder of this section were developed in [16] by Passino, et. al. and are presented here with a change in notation so that they can be applied to the discrete-event systems used in this paper.

First let us have the following notation

$P_m \subset P$

$\mathbf{E}$ is the set of all sequences of events from E

$E_k \in \mathbf{E}$ is any sequence of k events

$\mathbf{E}_v \subset \mathbf{E}$ is the set of valid event sequences i.e. physically possible

$\mathbf{E}_v(p) \subset \mathbf{E}_v$ is the set of valid event sequences that begin at state $p \in P$

$X : P \times \mathbf{E} \rightarrow P$ function which maps an initial state and a sequence of events to a final state

$\rho : P \times P \rightarrow \mathbf{R}$ is a metric on P

$\rho(p, P_m) = \inf\{\rho(p, p') : p' \in P_m\}$

A sequence of events is a sequence of symbols from the set E. Valid sequences refer to sequences which could actually be generated by the system in question. The function X is similar to an extended state transition function except that it gives the final state after a certain series of events which are outputs rather than inputs.

Definition 5.1: The *r-neighborhood* of an arbitrary set $P_m \subset P$ is denoted by the set $S(P_m; r) = \{p \in P: 0 < \rho(x,P_m) < r \}$ where $r > 0$.

Definition 5.2: The set $P_m \subset P$ is called *invariant* if from $p \in P_m$ it follows that $X(p,E_k) \in P_m$ for all E_k such that $E_k \in E_v(p)$ and $k \in N$.

Definition 5.3: A closed invariant set $P_m \subset P$ is called *stable in the sense of Lyapunov w.r.t.* E_v if for any $\varepsilon > 0$ it is possible to find a quantity $\delta > 0$ such that when $\rho(p, P_m) < \delta$ we have $\rho(X(p, E_k), P_m) < \varepsilon$ for all $E_k \in E_v(p)$ and $k \in N$. If furthermore $\rho(X(p, E_k), P_m) \rightarrow 0$ for all $E_k \in E_v(p)$ as $k \rightarrow \infty$, then the closed invariant set P_m is called *asymptotically stable w.r.t.* E_v.

Definition 5.4: A closed invariant set P_m is called *unstable in the sense of Lyapunov w.r.t.* E_v if it is not stable in the sense of Lyapunov w.r.t E_v.

Definition 5.5: If the closed invariant set P_m is asymptotically stable in the sense of Lyapunov w.r.t. E_v, then the set P_a of all states $p \in P$ and $p \notin P_m$ having the property $\rho(X(p, E_k), P_m) \rightarrow 0$ for all $E_k \in E_v(p)$ as $k \rightarrow \infty$ is called the *region of asymptotic stability of* P_m *w.r.t.* E_v.

Definition 5.6: The closed invariant set P_m with region of asymptotic stability P_a w.r.t. E_v is called *asymptotically stable in the large w.r.t.* E_v if $P_a = P$.

The following Theorem provides necessary and sufficient conditions for stability of the DES defined above.

Theorem 5.1: In order for a closed invariant set P_m to be stable in the sense of Lyapunov w.r.t E_v it is necessary and sufficient that in a sufficiently small neighborhood $S(P_m; r)$ of the set P_m there exists a specified functional V with the following properties:

(i) For sufficiently small $r_1 > 0$, it is possible to find a $r_2 > 0$ such that $V(p) > r_2$ for $p \in S(P_m; r)$ and $\rho(p,P_m) > r_1$.

(ii) For any $r_4 > 0$ as small as desired, it is possible to find a $r_3 > 0$ so small that when $r(p,P_m) < r_3$ for $p \in S(P_m; r)$ we have $V(p) \leq r_4$.

(iii) $V(X(p,E_k))$ is a non-increasing function for $k \in N$, for $p \in S(P_m; r)$, for all $k \in N$, as long as $X(p,E_k) \in S(P_m; r)$ for all $E_k \in E_v(p)$.

A proof of this theorem can be found in [16].

5.2 Lyapunov Stability for Hybrid Control Systems

Since the lower portion of a Hybrid Control System, the actuator, plant, and aggregator, can be modeled as a Discrete-Event System, namely the DES equivalent plant, the requirements for Lyapunov stability in a DES can be applied to the DES equivalent plant of a HCS. There are two situations which can be considered. First we can test the stability of the DES equivalent plant separately, with no input and therefore subject only to initial conditions. Second, we can consider the entire system, both DES equivalent plant and DES controller.

The first step in testing for Lyapunov stability is to choose a suitable invariant set, P_m, and metric, ρ. P_m is intended to represent those states which are desirable, in other words, the states in which the plant is supposed to operate. Therefore choose P_m to be the subset of P which defines the acceptable operating range of the DES equivalent plant. A useful metric is the following

$$\rho(p_i, p_j) = \inf \| \{E_k : X(p_i, E_k) = p_j\} \| \tag{31}$$

which defines the distance between two states as the minimum number of state transitions, not necessarily valid, required to move from the first state to the second. It it easy to see that this function meets the requirements for a metric, namely

$$\rho(p_i, p_j) = 0 \Leftrightarrow i = j$$
$$\rho(p_i, p_j) = \rho(p_j, p_i)$$

and

$$\rho(p_i, p_k) \leq \rho(p_i, p_j) + \rho(p_j, p_k).$$

5.3 Example: Double Integrator

5.3 Example: Double Integrator

The double integrator system first presented in Section 2.5 makes a good choice for an example here because the purpose of the system is to achieve stability. First the DES equivalent plant will be analyzed alone to see if it is stable according to our definition. Figure 5.1 shows the DES equivalent plant for the double integrator system. A problem is evident immediately when the invariant set, P_m, is determined, because all four states are in the desired operating range. Therefore it is not possible for the DES equivalent plant to go unstable. In fact, any time a system has a finite number of possible states it will be stable according to our definition because the state is inherently bounded. However, depending on the invariant set, the system will not necessarily be asymptotically stable.

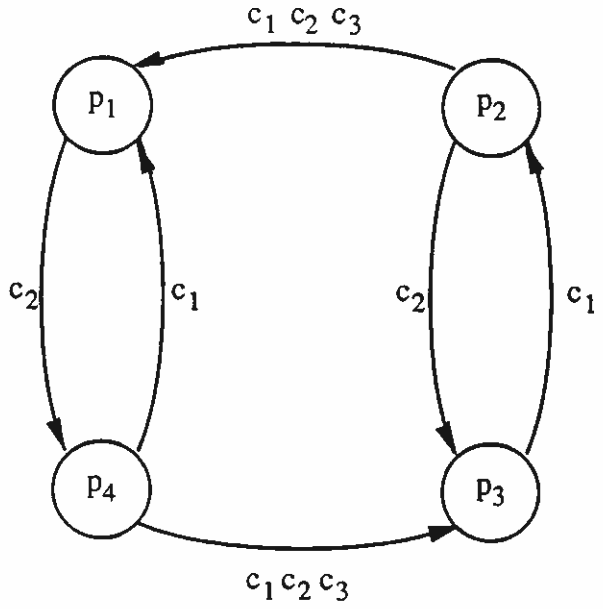

Figure 5.1: DES Equivalent Plant for Double Integrator

Let us define a new aggregator for the double integrator system which generates an unbounded state space in the DES equivalent plant and thus allows a meaningful stability analysis. The new aggregator generates the regions shown in Figure 5.2. In fact Figure 5.2 only shows a part of the state space but it establishes the pattern of the aggregator regions. The curved boundaries are coincident with certain state trajectories so that the DES equivalent plant has a manageable number of nondeterministic transitions. A portion of the DES equivalent plant is shown in Figure 5.3, and the invariant set has been indicated. The invariant set is

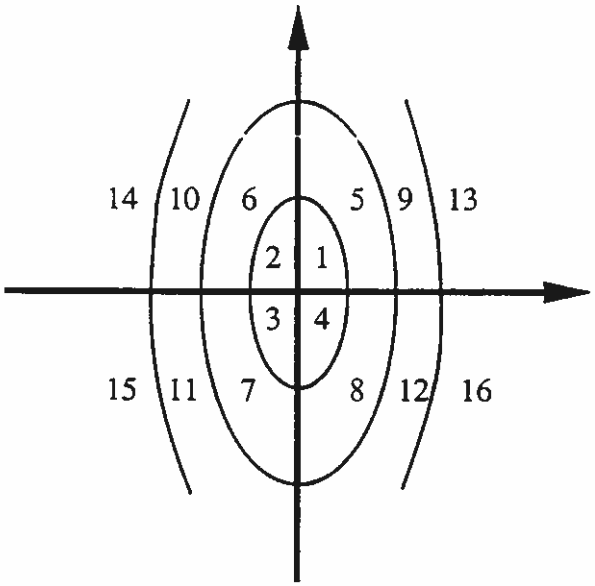

Figure 5.2: Aggregator Regions

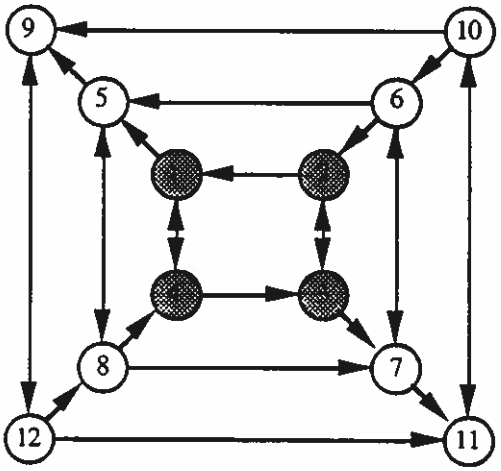

Figure 5.3: Uncontrolled DES Equivalent Plant

$$P_m = \{p_1, p_2, p_3, p_4\} \tag{32}$$

and according to the metric defined in equation (31) we have

$$\rho(p_i, P_m) = \text{int}\left(\frac{i}{4}\right) + 1. \tag{33}$$

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

It should be clear from Figure 5.3, that this DES equivalent plant is not stable. In fact there is no functional, V, which can be created to satisfy the requirements of Theorem 5.1 for a non-empty neighborhood of the invariant set.

Next we consider the entire system which consists of the DES plant described above as well as the DES controller which is intended to stabilize it. The DES controller serves to permit only certain transitions in the DES plant so the combined system can be represented by the DES plant with the forbidden transitions removed. This system is shown in Figure 5.4. Now with the definition

$$V(p_i) = \rho(p_i, P_m) \tag{34}$$

the stability of the combined system can be shown. First conditions (1) and (2) can be satisfied by setting $r_1 = r_2$ and $r_3 = r_4$. To see that condition (3) is satisfied, notice that no state transition leads to a state with a larger valued metric with respect to the invariant set.

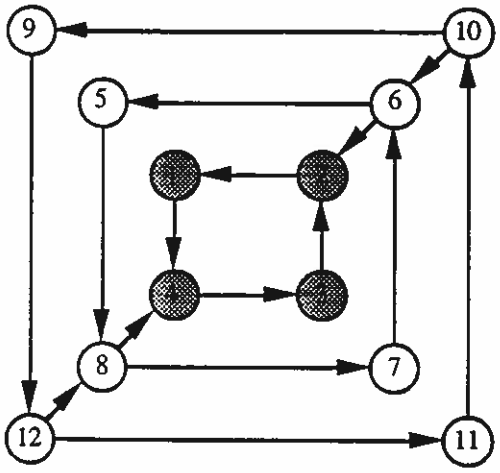

Figure 5.4: Controlled DES Equivalent Plant

It appears that if a DES equivalent plant can be shown to be stable then there exists a relationship between the boundaries of the aggregator regions and the contours of an actual Lyapunov function for the continuous-state plant. If two states in the DES equivalent plant are not the same distance from the invariant set according to the metric, and their corresponding aggregator regions are adjacent, then the boundary between these aggregator regions should lie on a contour of a Lyapunov function for the continuous-state plant. This is the case for the example given, but it is not known how well this condition generalizes.

6. CONCLUDING REMARKS

An important contribution of this work is the development of an explicitly defined model for representing hybrid control systems. There are three chief advantages of this model which distinguish it from others studied. First, the model is defined completely, leaving no gaps which lead to uncertainty in its behavior. One benefit of this is that it allows a hybrid control system, so modeled, to be simulated on a computer using a general program similar to the one used and included in this paper. A second benefit is that tools for analysis can be developed for this model which then apply to any hybrid control system which can be represented.

The second chief advantage of the model is that the interface of the hybrid control system is represented by two simple static maps. This is important for analysis because it is the interface which introduces the difficulty in handling hybrid control systems. Thus it is anticipated that a more simple interface will result in less difficulty. A final advantage of our model is its flexibility; it has been shown to be capable of representing systems with a variety of different types of control action.

The next contribution of this work is to begin the development of some basic concepts and tools to be used in the analysis of hybrid control systems. The DES equivalent plant is intended to represent the behavior of the interface and plant of an HCS in such a way that it reflects the extent to which the controller can achieve its objectives in the system. The definition of determinism and the theorems concerning determinism are used to evaluate the DES equivalent plant. The study of stability in HCS's leads to a greater understanding of the role of the interface in determining the overall behavior of the hybrid control system.

7. REFERENCES

[1] Antsaklis, P. J., Passino, K., Wang, S., "Towards Intelligent Autonomous Control Systems: Architecture and Fundamental Issues", *Journal of Intelligent and Robotic Systems*, pp. 315-342, 1989; also "T", *IEEE Control Systems Magazine*, Vol. 35, No. 6, pp. ,June 1990.

[2] Åstrom, J., Wittenmark, B., *Computer-Controlled Systems*, Prentice-Hall, NJ, 1990.

[3] Benveniste, A., Le Guernic, P., "Hybrid Dynamical Systems Theory and Nonlinear Dynamical Systems Over Finite Fields", *New Trends in Nonlinear Control Theory*, pp. 485-495. Springer-Verlag, NY, 1989.

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

[5] Borgne, M., Benveniste, A., Le Guernic, P., "Polynomial Ideal Theory Methods in Discrete Event and Hybrid Dynamical Systems", Proceedings of the 28th Conference on Decision and Control, pp. 2695-2700, Tampa, FL, December 1989.

[6] Cao, X., Ho, Y., "Models of Discrete Event Dynamic Systems", *IEEE Control Systems Magazine*, pp. 69-76, June 1990.

[7] Carroll, J., Long, D., *Theory of Finite Automata*, Prentice-Hall, N.J., 1989.

[8] Cassandras, C., Strickland, S., "Sample Path Properties of Timed Discrete Event Systems", *Proceedings of the IEEE*, Vol. 77, No. 1, pp. 59 - 71, January 1989.

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

[10] Fishwick, P., Zeigler, B., "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.

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

[14] Passino, K., "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.

[15] Passino, K., Sartori, M., Antsaklis, P.J., "Neural Computing for Numeric-to-Symbolic Conversion in Control Systems", *IEEE Control Systems Magazine*, pp. 44-52, April 1989.

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

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

[18] Peleties, P., DeCarlo, R., "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.

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

[22] Sain, M. K., *Introduction to Algebraic System Theory*, Acedemic Press, NY, 1981.

[23] Sontag, E. D., *Mathematical Control Theory - Deterministic Finite Dimensional Systems*, Springer-Verlag, NY, 1990.

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

[26] Zhong, H.,Wonham, W. M., "On the Consistency of Hierarchical Supervision in Discrete-Event Systems", *IEEE Transactions on Automatic Control*, Vol. 35, No. 10, pp. 1125-1134, October 1990.