Proceedings of the
American Control Conference
San Francisco, California • June 1993

# Learning to Coordinate Control Policies of Hybrid Systems

M.D. Lemmon [*], J. Stiver, P.J. Antsaklis
Dept. of Electrical Eng.
University of Notre Dame
Notre Dame, IN 46556

**Abstract**

Hybrid control systems consist of a discrete event system (DES) supervising the behaviour of a continuous state system (CSS) through the issuance of logical control directives. This paper derives sufficient conditions on the DES/CSS interface which guarantee the existence of a supervisor which transitions the plant through an arbitrary sequence of commanded events. It is further demonstrated that this interface can be learned using an inductive inference protocol which converges after a finite number of updates.

## 1 Introduction

Hybrid systems consist of a continuous-state system (CSS) interfaced to a discrete event system (DES). The resulting system can be seen as consisting of 3 distinct layers. The lowest layer is the continuous-state system or **plant**. The highest layer is a discrete event system or **supervisor**. The middle layer is an interface which facilitates communication between the supervisor and plant. The interface therefore behaves as a 2-way communication port which admits a natural decomposition into two subsystems, one for each communication direction. The subsystem handling control event transformations is called the **actuator**. The subsystem transforming plant states into plant symbols is called the **generator**.

This paper discusses a class of hybrid system in which the interface generator and actuator consist of memoryless mappings [Stiver 1992] [Antsaklis 1993]. The following sections establish sufficient conditions on the interface which guarantee the existence of a supervisor which drives the plant through a connected sequence of events. These conditions provide the basis for an inductive inference protocol which can be used by the system to "learn" the appropriate actuator.

The remainder of this paper is organized as follows. Section 2 discusses the hybrid system under study. Section 3 states the "supervisability" conditions. Section 4 shows how these conditions lead to an inductive inference protocol for automatically learning how to "supervise" the system. Section 5 illustrates the proposed framework for a simple example. Section 6 summarizes the results.

[*]The partial financial support of the National Science Foundation (IRI91-09298) is gratefully acknowledged

## 2 Hybrid Systems

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

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{r}) \quad (1)$$
$$\mathbf{z} = g(\mathbf{x}) \quad (2)$$

where $\mathbf{x} \in \Re^n$, $\mathbf{r} \in \Re^m$, and $\mathbf{z} \in \Re^p$ are the state, control, and observation vectors, respectively. $f : \Re^n \times \Re^m \rightarrow \Re^n$ and $g : \Re^n \rightarrow \Re^p$ are functions.

For the purposes of this work we assume that $\mathbf{z} = \mathbf{x}$ and we assume that the plant is linear in the controls so that

$$\dot{\mathbf{x}} = \sum_{i=1}^{m} r_i f_i(\mathbf{x}) \quad (3)$$

The vector $\mathbf{r} = (r_1, \ldots, r_m)' \in \Re^m$ can be interpreted as "coordinating" a set of control policies represented by the set of $m$ vector fields, $f_i(\mathbf{x})$.

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

$$\tilde{s}[n] = \delta(\tilde{s}[n-1], \tilde{z}[n]) \quad (4)$$
$$\tilde{r}[n] = \phi(\tilde{s}[n]) \quad (5)$$

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

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

$$\mathbf{r} = a(\tilde{r}) = ( \; a_1(\tilde{r}) \quad \cdots \quad a_m(\tilde{r}) \; )' \qquad (6)$$

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

$$\tilde{z} = g(\mathbf{x}) \qquad (7)$$

The plant symbol generating function, $g$, is designed based on an open covering of the state space of the plant. Consider a collection of $p$ open subsets in $\Re^n$ which form an open cover for the plant state space. Let this collection be represented as

$$C = \{ \; c_1 \quad \ldots \quad c_p \; \} \qquad (8)$$

The collection consists of open subsets, each subset is called a <u>covering event</u>. Let the $i$th covering event, $c_i$, be associated with a unique covering symbol, $\tilde{c}_i$. The "alphabet" of covering symbols can therefore be represented as

$$\tilde{C} = \{ \; \tilde{c}_1 \quad \ldots \quad \tilde{c}_p \; \} \qquad (9)$$

These covering symbols are used to define the plant symbols as follows

$$\tilde{z} = g(\mathbf{x}) = \{\tilde{c}_i : \mathbf{x} \in c_i\} \qquad (10)$$

As shown in Equation 10, a plant symbol is a collection of covering symbols which defines a region in the state space. It is convenient to treat this collection as a symbol. A plant symbol is generated only when a new event first occurs. The overall effect is that the state space of the plant is partitioned into a number of regions and each is associated with a unique plant symbol which is generated whenever the state enters that region.

The use of open covers to describe the state space is motivated by several things. The generator, $g$, must take subsets of states onto a "unique" plant symbol. The representation should be "well posed" in the sense that small changes of state or system structure do not result in discontinuous variations in the state's symbolic representation. The well-posedness constraints suggests that events should be open subsets of the state space. The representation must also be complete, meaning every state must be contained within a plant event. These conditions suggest that $g$ should realize an open covering of the state space.

In this paper, the covering events are assumed to have a special form. This form is motivated by the modest computational complexity associated with the evaluation of linear forms. The specific covering events assumed are denoted as

$$C = \{ \; c_1 \quad c_1^+ \quad c_1^- \quad \cdots \quad c_p \quad c_p^+ \quad c_p^- \; \} \qquad (11)$$

where each event is defined by the following open sets in $\Re^n$.

$$\begin{aligned} c_i &= \{\mathbf{x} \in \Re^n : \mathbf{s}_i'\mathbf{x} > \alpha_i\} &(12)\\ c_i^+ &= \{\mathbf{x} \in \Re^n : \mathbf{s}_i'\mathbf{x} > \alpha_i + \beta_i + \delta\} &(13)\\ c_i^- &= \{\mathbf{x} \in \Re^n : \mathbf{s}_i'\mathbf{x} > \alpha_i + \beta_i - \delta\} &(14)\end{aligned}$$

where $\mathbf{s}_i \in \Re^n$ and $\alpha_i, \beta_i, \delta > 0$. Associated with the events $c_i$, $c_i^\pm$ are the covering symbols, $\tilde{c}_i$, $\tilde{c}_i^\pm$.

## 3 Supervisability

In many applications, it is required to drive the plant through a well defined sequence of events. A plant for which this can be done will be said to be "supervisable". This section derives sufficient conditions for a system to be supervisable.

**Definition 1** *Let $C$ be the plant's event covering with associated alphabet $\tilde{C}$. Let the $j$th <u>command event</u> be any set, $u_j$, formed by the intersection of $q$ covering events as shown below $u_j = \bigcap_{i=1}^q c_{j_i}$ where $j_i$ is an integer between 1 and $p$ and $c_{j_i} \in C$.*

The preceding definition implies that the $j$th command event is associated with a set of integers $j_1, \ldots, j_q$ indexing the covering events comprising $u_j$. This set of integers will be called the <u>index set</u>, $I_j$, of the command event $u_j$.

**Definition 2** *Let $C$ be the covering events with associated alphabet $\tilde{C}$. Two command events, $u_1$ and $u_2$ are said to be <u>connected</u> if and only if their index sets, $I_1$ and $I_2$, satisfy the relation, $I_1 \subset I_2$ or $I_2 \subset I_1$.*

**Theorem 1** *Consider a hybrid system with event covering $C$ and covering alphabet $\tilde{C}$. Consider the covering events*

$$\begin{aligned} c^+ &= \{\mathbf{x} \in \Re^n : \mathbf{s}'\mathbf{x} - \alpha - \beta - \delta\} &(15)\\ c^- &= \{\mathbf{x} \in \Re^n : \mathbf{s}'\mathbf{x} - \alpha - \beta + \delta\} &(16)\end{aligned}$$

*If the functional, $V(\mathbf{x}) = (\mathbf{s}'\mathbf{x} - \alpha - \beta)^2$, is a Lyapunov functional for the system*

$$\dot{\mathbf{x}} = \begin{cases} \sum_{j=1}^{m} a_j(\tilde{c}^+) f_i(\mathbf{x}) & \text{if } \mathbf{x} \in c^+ \\ \sum_{j=1}^{m} a_j(\tilde{c}^-) f_i(\mathbf{x}) & \text{if } \mathbf{x} \in c^- \end{cases} \tag{17}$$

*then if $\mathbf{x} \in (c^+ \cup c^-)$ at $t = 0$, there exists a time $T > 0$ such that for all $t > T$, $\mathbf{x} \notin (c^+ \cup c^-)$.*

**Proof:** By assumption, $V(\mathbf{x})$, is a Lyapunov functional so that by theorem 8 of [Utkin 1977] it can be inferred that the complement of $c^+ \cup c^-$ is an attracting invariant set. •

The significance of the preceding theorem is that it suggests a form for the actuator and supervisor which "blocks" the system state out of the event $c^+ \cup c^-$. The following theorem indicates how the "blocking" conditions allow for the system to supervise transitions between two connected events.

**Theorem 2** *Consider a hybrid system with event covering $C$ and alphabet $\tilde{C}$. Let $u_1$ and $u_2$ be 2 connected command events with index sets $I_1$ and $I_2$, respectively. Let the functionals, $V_i(\mathbf{x}) = (\mathbf{s}_i'\mathbf{x} - \alpha_i - \beta_i)^2$, be Lyapunov functionals for the system*

$$\dot{\mathbf{x}} = \begin{cases} \sum_{j=1}^{m} a_j(\tilde{c}_i^+) f_i(\mathbf{x}) & \text{if } \mathbf{x} \in c_i^+ \\ \sum_{j=1}^{m} a_j(\tilde{c}_i^-) f_i(\mathbf{x}) & \text{if } \mathbf{x} \in c_i^- \end{cases} \tag{18}$$

*for all $i \in I_1 \cup I_2$. There exists a supervisor which transitions the plant state from $u_1$ to $u_2$.*

**Proof:** Since $u_1$ and $u_2$ are connected, then either $I_1 \subset I_2$ or $I_2 \subset I_1$. In the first case, $\mathbf{x}$ is already an element of $u_2$ so that only the second case need be considered.

In the second case, the inclusion of $I_1$ in $I_2$ implies that $u_2 \subset u_1$. Let $\tilde{z}$ be the plant symbol issued by the generator (i.e., $\tilde{z} = g(\mathbf{x})$) when $\mathbf{x} \in u_1$. Define the supervisor so that it maps $\tilde{z}$ onto the covering symbol, $\tilde{c}_i^{\pm}$, within the collection $\tilde{z}$ which has the largest index $i$.

If the control symbol issued by the supervisor is $\tilde{c}_i^{\pm}$ then by theorem 1, the plant state will eventually be taken out of the set $c_i^+ \cup c_i^-$. After leaving this set, the subsequent plant symbol issued by the generator will not contain either $\tilde{c}_i^+$ or $\tilde{c}_i^-$. The supervisor then issues a new control symbol, $\tilde{c}_j^+$, which forces the removal of $\tilde{c}_j^+$ and $\tilde{c}_j^-$ from the plant symbol $\tilde{z}$ issued by the generator. Furthermore, the $i$th positive and negative covering symbols cannot be put back into $\tilde{z}$ since this would automatically result in the issuance of a control symbol which takes these symbols out of $\tilde{z}$. Consequently, the proposed supervisor will successively drive the plant state out of all covering sets $c_j^{\pm}$ which have indices in $I_2$. This then means that the plant state must eventually lie in $u_2$. •

Note that the event covering will also define a finite partition of the state space. Let $V$ denote a collection of such "partitioning" events, $v \in V$. The following theorem provides conditions for the existence of a supervisor which transitions the plant between any 2 partition events.

**Theorem 3** *Let $v_0$ and $v_f$ be any two parititon events of $V$. If the assumptions of theorem 2 hold for $i = 1, \ldots, p$, then there exists a supervisor which transitions the plant state from $v_0$ to $v_f$.*

**Proof:** $v_0$ and $v_f$ can be contained within a closed connected set, $M$, which has a finite open cover contained within $C$. Event $v_0$ and $v_f$ can also be defined by subcollections, $\tilde{z}_0$ and $\tilde{z}_f$, of covering symbols issued by the generator. Let $I_0$ and $I_f$ be the corresponding index sets.

If $I_0 \cap I_f$ is not empty, then it is trivial to construct a sequence of command events. This is accomplished by enlarging the collection, $\tilde{z}_0$, with elements from $\tilde{z}_f$ and then removing those symbols of $\tilde{z}_0$ which are not in $\tilde{z}_f$. If $I_0 \cap I_f$ is empty, then the fact that $M$ has a connected open subcover implies that an event can be found which does connect these two events. In this case, the preceding procedure is followed again to construct a sequence of connected command events containing $v_0$ and $v_f$.

Once the sequence of command events is constructed, then theorem 2 implies the existence of a supervisor which transitions the plant state between any 2 connected events in the sequence thereby yielding the supervisability between $v_0$ and $v_f$ asserted in the theorem. •

## 4 Learning to Coordinate by Example

The preceding section showed that the plant can be sequenced through a collection of connected events provided control vectors $\mathbf{r}_i$ ($i = 1, \ldots, p$) can be specified which stabilize the system with regard to the Lypunov functionals, $V_i(\mathbf{x}) = (\mathbf{s}_i'\mathbf{x} - \alpha_i - \beta_i)^2$. This condition implies that for all $\mathbf{x}$,

$$(\mathbf{s}_i'\mathbf{x} - \alpha_i - \beta_i)(\mathbf{s}_i'\dot{\mathbf{x}}) < 0 \tag{19}$$

Without loss of generality assume that $\mathbf{s}_i'\mathbf{x} - \alpha_i - \beta_i > 0$. In light of the supervisor constructed in the proof of theorem 2, this implies that the control symbol to be issued will be $\tilde{c}_i^+$. Using this fact in equation 19 as well as the assumed plant model yields

$$\begin{pmatrix} \mathbf{s}_i'f_1(\mathbf{x}) & \cdots & \mathbf{s}_i'f_m(\mathbf{x}) \end{pmatrix} \begin{pmatrix} r_1 \\ \cdots \\ r_m \end{pmatrix} < 0 \tag{20}$$

where $r_j = a_j(\tilde{c}_i^+)$ for $j = 1, \ldots, m$ and $a_j : \tilde{R} \rightarrow \Re$ are the components of the actuator mapping.

Equation 20 forms a set of linear inequalities. The problem is to define the actuator mappings, $a$, which map control symbols, $\tilde{c}_i^{\pm} \in \tilde{C}$, onto control vectors,$\mathbf{r}_i^{\pm} \in \Re^m$ so that the inequalities of equation 20 are satisfied for all $i$. The solution to this problem is a search procedure for a feasible point for the inequality system. Numerical algorithms for doing such searches are well known. One particularly appealing approach is the central-cut ellipsoid method [Shor 1977]. The advantage of this approach is that it has well understood convergence properties.

The search for the feasible point of inequality system 20 can be easily framed in terms of an inductive inference procedure. Inductive inference is a machine learning algorithm which has found extensive applications in the learning of Boolean functions by example [Angluin 1983] and the proof of polynomial time-complexity for certain linear programming algorithms [Khachiyan 1979]. The ability to reframe the search as an inductive inference protocol immediately suggests that the search can be viewed as a "learning" procedure. This means that the use of such protocols will provide a method by which the hybrid system can learn a set of coordinating control vectors $\mathbf{r}$ by simply observing the plant's behaviour. In other words, for a given set of covering events, these ideas imply that the system can automatically learn those controls which insure that the given events yield a "supervisable" hybrid control system.

The specific inductive protocol proposed in this paper consists of four fundamental components.

1. **Hypothesis:** Form an initial hypothesis. Let $\tilde{R}$ be consist of symbols $\{\tilde{c}_i^{\pm}\}$. Specify an acutator mapping $a : \tilde{R} \rightarrow \Re^m$ such that $a(\tilde{c}_i^{\pm}) = \mathbf{r}_i^{\pm}$ for all $i = 1, \ldots, p$. The hypothesis is that the actuator mapping satisfies the Lyapunov inequalities implied by equation 20.

2. **Experiment:** The information needed to evaluate inequality system 20 is contained in $\mathbf{s}_i' f_j(\mathbf{x})$ for all $i = 1, \ldots, p$. This requires that the individual influence of each control policy, $f_j(\mathbf{x})$, must be measured or estimated. The "experiment" consists of the hybrid system's measurment of these quantities.

3. **Oracle Query:** The information gathered by the experiment is tested to see if it is consistent with the hypothesis. This test is simply the evaluation of inequality 20. The oracle is then a Boolean functional which declares TRUE if the current data satisfies the inequalities and declares FALSE if the current data does not satisfy (i.e. is inconsistent) with the inequalities.

4. **Update Algorithm:** If the oracle detects no inconsistency, then nothing is done. If, however, the oracle detects an inconsistency between the currently collected data and the hypothesis, then that implies the hypothesis is incorrect. The direct application of the central-cut ellipsoid method [Shor 1977] allows the computation of a new hypothesis (actuator mapping) which is consistent with the current and all prior data collected by the experiment.

   More details on the use of this algorithm can be found in [Lemmon 1992].

**Remark 1:** The preceding algorithm makes extensive use of the so-called ellipsoid algorithm. The value of using the ellipsoid updating method is that it can be easily shown to converge after a finite number of updates. If it is known that the supervising control vectors $\mathbf{r}_i^{\pm}$ associated with control symbol $\tilde{c}_i^{\pm}$ lie within an $m$-dimensional ellipsoid of volume $v$, then the learning protocol can be shown [Lemmon 1992] to find the desired control vector after no more than $2m \ln v^{-1}$ updates.

**Remark 2:** The finite time bound determined in remark 1 will also imply polynomial time complexity. Using the fact that the volume of an $m$-dimensional unit sphere is bounded below by $m^{-m}$, it can be shown that the bound cited in remark 1 will scale as $m^2 \ln m \approx m^{2.5}$.

## 5 Example

The hybrid system framework introduced in this paper can be illustrated by a simple example. The event collection, $C$, for a hypothetical second order system is shown in figure 1.

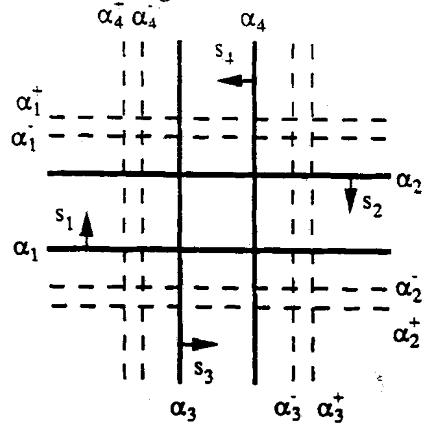

Figure 1: Event Covering for Example

The illustrated event covering consists of twelve events,

$$c_1 \quad c_1^+ \quad c_1^- \quad \cdots \quad c_4 \quad c_4^+ \quad c_4^- \tag{21}$$

where the plant events are characterized by 2-dimensional vectors $\mathbf{s}_i$ and real parameters, $\alpha_i$, $\beta_i$, and $\delta$, for $i = 1, \ldots, 4$.

This section shows how the proposed supervisor transitions the plant from partition event $v_0 = c_1 \cap c_4$ to partition event $v_f = c_2 \cap c_3$. In order to accomplish this task, we must first determine a connected sequence of command events for the initial and final partition events. In this example the command sequence of events will be

$$c_1 \cap c_4 \quad c_4 \quad c_4 \cap c_2 \quad c_2 \quad c_2 \cap c_3 \tag{22}$$

Assume that the plant starts at the location marked "A" in figure 2. In this case, the first symbol issued by the generator will be

$$\tilde{z}_A = \{ \ \tilde{c}_1 \quad \tilde{c}_1^+ \quad \tilde{c}_2^- \quad \tilde{c}_3^- \quad \tilde{c}_4 \quad \tilde{c}_4^+ \ \} \qquad (23)$$

The positive and negative index sets will be $I_A = \{1,2,3,4\}$. The first commanded event is $c_1 \cap c_4$ with index set $\{1,4\}$. Therefore the control symbol issued by the supervisor will be $\tilde{c}_4^+$. This control symbol will drive the plant state out of $c_4^+$ and $c_4^-$ so that the plant state, $\mathbf{x}$, is constrained to the shaded boundary layer marked as "B" in figure 2.

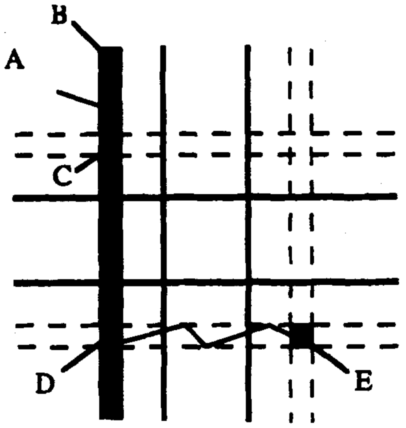

Figure 2: Event Trajectory for Example

With the plant state confined to the boundary layer marked "B", then the generator issues a different event of the form

$$\tilde{z}_B = \{ \ \tilde{c}_1 \quad \tilde{c}_1^+ \quad \tilde{c}_2^- \quad \tilde{c}_3^- \quad \tilde{c}_4 \quad \ \} \qquad (24)$$

The positive and negative index sets will be $I_B^+ = \{1,2,3\}$. The command event is still $c_1 \cap c_4$ so that the control symbol issued by the supervisor will be $\tilde{c}_1^+$. This control symbol drives the plant state out of $c_1^+$ and $c_1^-$ so that the plant state, $\mathbf{x}$, is constrained to the shaded region marked as "C" in figure 2.

Note that the collection of control symbols generated by the supervisor will now be the empty set. This control symbol signals that the supervisor should use the next command event to supervise the plant. The next command event is $c_4$. Which will once again yield the empty set as the control symbol. The following command event, $c_4 \cap c_2$ must therefore be used. This results in the control symbol $\tilde{c}_2^-$ being issued by the supervisor. The resulting action is to drive the plant state into region "D" of figure 2. Once the plant enters region "D", then the final command events $c_2$ and $c_2 \cap c_3$ can be used by the supervisor in the same way. This leads to the issuance of control symbols which drive the plant along another boundary layer connecting events "D" and "E" as shown in figure 2. Event "E" is, of course, the final desired state of the system.

## 6 Summary

This paper builds upon prior work [Stiver 1992] in which the interface between a CSS plant and DES supervisor was consisted of two memoryless mappings between continuous and logical variables. The current work modifies that earlier work in that the generator realizes a finite open cover for the plant's state space which is formed by a family of linear halfspaces. This paper has two principal results concerning this family of hybrid systems. The first result establishes sufficient conditions for the plant to be supervised through a sequence of "command" events. The second result shows that an interface satisfying these supervisability conditions can be learned using an inductive inference protocol after a finite number of updates.

## References

[Angluin 1983] D. Angluin, C.H. Smith, *Computing Surveys*, 15(3):237-269, September 1983.

[Antsaklis 1993] P.J. Antsaklis, M.D. Lemmon, and J.A. Stiver, *Hybrid System Modeling and Event Identification*, Technical Report of the Group for Intedisciplinary Studies of Intelligent Systems, ISIS-93-002, Department of Electrical and Computer Engineering, University of Notre Dame, Notre Dame, IN, January 1993.

[Groetschel 1988] Groetshel, Lovasz, and Schrijver (1988), *Geometric Algorithms and Combinatorial Optimization*, Springer-Verlag, 1988.

[Khachiyan 1979] L.G. Khachiyan (1979), *Soviet Mathematics Doklady*, 20:191-194, 1979.

[Lemmon 1992] M.D. Lemmon, "Ellispoidal Methods for the Estimation of Sliding Mode Domains of Variable Structure Systems", *Proceedings of 1992 Conference on Information Sciences and Systems*, Dept. of Electrical Eng., Princeton University, Princeton, New Jersey, pg 1018-1023.

[Shor 1977] N.Z. Shor (1977), *Cybernetics*, 13:94-96, 1977.

[Stiver 1992] J.A. Stiver, P.J. Antsaklis, "Modeling and Analysis of Hybrid Control Systems", *Proc. 31st Conference on Decision and Control*, 1992.

[Utkin 1977] V.I. Utkin (1977), *IEEE Transactions on Automatic Control*, Vol. AC-22:212-222.