M. D. Lemmon, J. A. Stiver and P. J. Antsaklis, "Event Identification and Intelligent Hybrid Control," Hybrid S ystem s , R L Grossman, A Nerode, A P Ravn, H Rischel Eds, pp 269-296, Lecture Notes in Computer Science, LNCS 736, Springer-Verlag, 1993.

Event Identification and Intelligent Hybrid Control

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

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

Abstract. Hybrid dynamical systems consist of two types of systems, a continuous state system called the *plant* and a discrete event system called the *supervisor*. Since the plant and supervisor are different types of systems, an interface is required to facilitate communication. An important issue in the design of hybrid control systems is the determination of this interface. Essentially, the interface associates logical *symbols* used by the supervisor with nonsymbolic *events* representative of the plant's behaviour. This chapter discusses a method for learning a hybrid system interface where symbols and events are bound in a way which is compatible with the goal of plant stabilization. The method is called *event identification* and provides an on-line method for adapting hybrid dynamical systems in the face of unforseen plant variations.

1 Introduction

Hybrid dynamical systems provide a convenient tool for the analysis and design of supervisory control systems. A supervisory control system arises when a discrete event system is used to supervise the behaviour of a plant by the issuance of logical control directives. The hybrid system framework shown below in figure (1) clearly illustrates this architecture. The specific architecture illustrated below is based on the model used in [Antsaklis 1993] which appears in this volume. The notational conventions adopted in this chapter will be found in [Antsaklis 1993].

In figure (1), note that an interface is included to facilitate communication between the two different types of systems. This interface consists of two subsystems known as the *generator* and *actuator*. The generator transforms the plant's observation vector $\mathbf{z} \in \Re^p$ into a symbol $\tilde{z}$ which is drawn from an alphabet $\tilde{Z}$. The actuator transforms control symbols $\tilde{r}$ output by the supervisor into control vectors $\mathbf{r} \in \Re^m$ which are used by the plant. The control symbols are drawn from an alphabet $\tilde{R}$.

In view of the preceding discussion, it is apparent that the supervisor is used to control the plant. It is also apparent that the supervisor is a symbol manipulation system whose logical symbols have "meanings" grounded in nonsymbolic

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

in Hybrid Systems, Grossman, Nerode, Ravn, Rischel Eds
LNCS 736, Springer-Verlag 1993.

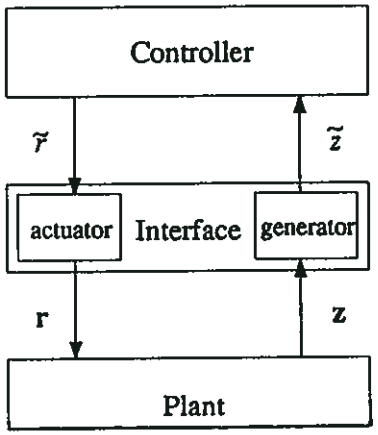

Fig. 1. Hybrid Dynamical System Architecture

external "events". In other words the supervisor's symbols can be assigned interpretations which are generally representative of important categories of plant behaviour. A plant's behaviour refers to its state trajectory, $\mathbf{x}(t)$. Behavioural categories are therefore collections of trajectories with some "common" property. State trajectories of autonomous systems, however, are determined by the initial state. This observation suggests that a class or category of behavioural trajectories can be adequately represented by a subset of the plant's state space. The term *event* is therefore used to denote a distinguished subset of the plant's state space. The term *symbol* is used to denote the logical label associated with an event. The association of a symbol with an event will be referred to as a symbol/event *binding*.

For example, a certain set of temperatures and pressure measurements may be indicative of a potential system failure. In this case, we would like to associate the "nonsymbolic" measurements with a "symbolic" label called "FAILURE". Therefore the supervisor's computations represent the manipulation of abstractions about the plant's current state. The use of such high-level abstractions (representations) of system state to control the system is sometimes called "intelligent" control.

This notion of intelligence, however, is singularly unsatisfying. Note that the action of the controller relies on the prior interpretation assigned to the plant and control symbols. Therefore the "intelligence" of the system lies in the interpretation of these symbols. The "intelligent" choices, however, were made by the human designer, not by the machine. Therefore it is the designer, rather than the machine which is intelligent. This same fundamental argument has been previously leveled against production based inference as a model for human cognition [Searle 1984]. Essentially, it asserts that the "blind" manipulation of symbols is not sufficient to render a system intelligent.

M. D. Lemmon, J. A. Stiver and P. J. Antsaklis, "Event Identification and Intelligent Hybrid Control," Hybrid S ystem s , R L Grossman, A Nerode, A P Ravn, H Rischel Eds, pp 269-296, Lecture Notes in Computer Science, LNCS 736, Springer-Verlag, 1993.

The reduction of the plant to an effective DES plant model, represents one way of designing so-called "intelligent" controllers. This approach to design was discussed briefly in a companion chapter [Antsaklis 1993]. However, this design approach represents the precise disembodiment of controller symbol and event, which was immediately discussed above. In this regard, an approach to supervisory control which assumes a priori symbol/event bindings cannot be considered an "intelligent" control system. Intelligence will only arise when the system is capable of determining its own event/symbol bindings. This requires that any intelligent system solve what may be called the *event identification* problem. The relationship between this "event identification" problem and more traditional issues in artificial intelligence such as the symbol grounding problem [Harnad 1990] is discussed in one of the closing sections of this chapter.

Whether or not the symbolic manipulations of a computational system constitute intelligence can, no doubt, be argued endlessly. There is, however, a much more pragmatic reason for considering such a system undesirable. If we consider those applications for which supervisory control systems are intended, it is immediately apparent that supervision is meant for complex and and unpredictable systems. For such systems, prior plant knowledge or complete plant knowledge may be impossible. This means that "events" which are defined with respect to an assumed plant structure, may change unexpectedly. If this is the case, then it is well within the realm of possiblity for our not-so intelligent supervisor to happily chunk away and produce of stream of nonsensical control symbols. The reason this occurs, of course, is because the supervisor really doesn't understand the significance of the symbols it is manipulating. If we wish to call this un-intelligent processing, that is fine. The end result is the same, however, a system whose autonomy is limited by the designer's initial assignment of symbol bindings. Therefore, a more pragmatic reason for requiring event identification of "intelligent" control systems is that it will undoubtably lead to increased system autonomy. The issue of autonomy in intelligent control was discussed thoroughly in the introduction. It is the need for such autonomy that really motivates the requirement for event identification in hybrid systems. As will be pointed out in one of the closing sections, this ability is also consistent with notions of "intelligence" stemming the symbolic and subsymbolic AI communities [Chalmers 1992].

The preceding discussion therefore indicates that an important problem in hybrid system control is the identification of events. How does one choose events which are consistent with the desired control objectives? Is it possible for the system to identify its own set of "optimal" events. This chapter presents one example of how such event identification can be accomplished. The problem of event identification can be viewed in a variety of contexts. For example, consider a system which has the general architecture shown in figure (1). Assume that the plant uses a collection of control policies, so that the plant's differential equation has the form

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

where $\mathbf{x}$ is the state space and $\mathbf{r}$ is an m-vector of "coordination" coefficients, r_i. The individual vector fields can be seen as "control policies" which are coordinated through the specification of the vector $\mathbf{r}$. In figure (1), it can now be seen that the binding of plant/control symbols with subsets of the state space determines the behaviour of this system. One side of the problem, involves determining plant symbol and event bindings which allow a deterministic or quasideterministic plant DES (see [Antsaklis 1993]). The solution of this problem yields a design for the interface's event generator. A system which can learn a set of bindings consistent with deterministic behaviour will have gone a long way in learning to control itself. Another side of the problem focuses on learning the symbol bindings between the control symbols, $\tilde{r}$, and vectors $\mathbf{r}$. The solution to this problem yields a design for the interface's actuator. System which are capable of forming the event/symbol bindings consistent with with control objectives (i.e. determinism or controllability) will go a long way towards making truly "intelligent" control systems exhibiting a high degree of autonomy.

The following sections provide a specific example of a hybrid system which can automatically learn event/symbol bindings. The example system is a variable structure system and the symbol bindings are learned with regard to invariant sets generated by the plant's dynamics. Early work on this was done in [Lemmon 1992] and refined in [Lemmon 1993a] with regard to the binding of plant symbols. Considerations on the binding of control symbols were discussed in [Lemmon 1993b]. In all of this work it was shown that bindings could be learned in finite time with a sample complexity that scales in a polynomial manner with plant complexity. The remainder of this section is organized as follows. Section (2) discusses the example problem which is referred to in this section as the invariant subspace identification (ISID) problem. Section (3) introduces the learning algorithm. This algorithm consists of two procedures called the oracle and the update procedure. These procedures are derived in sections (4) and (5). The convergence and complexity properties of this learning procedure are discussed in section (6). An example of this algorithm's use is illustrated in section (7). The importance of the following example is that it provides a concrete example of a hybrid system which learns to "identify" its own events in a computationally efficient manner. Some issues and concerns associated with this example are discussed in section (8). The presented algorithm also provides a novel perspective on the relationship between intelligence and control. The central issue in this perspective is the so-called "symbol grounding" problem [Harnad 1990]. This novel perspective on "intelligence" will be discussed in section (9).

2 Invariant Subspace Identification (ISID) Problem

The hybrid system under consideration is assumed to have a very special form. Specifically, it will be assumed that the plant's dynamics are represented by the

M. D. Lemmon, J. A. Stiver and P. J. Antsaklis, "Event Identification and Intelligent Hybrid Control," Hybrid S ystem s , R L Grossman, A Nerode, A P Ravn, H Rischel Eds, pp 269-296, Lecture Notes in Computer Science, LNCS 736, Springer-Verlag, 1993.

following differential equations.

$$\dot{\mathbf{x}} = \sum_{i=1}^{2} r_i f_i(\mathbf{x}) \tag{2}$$

where $\mathbf{x} \in \Re^n$ is the state vector, f_1 and f_2 are smooth mappings from $\Re^n$ onto $\Re^n$. It is also assumed that the vector $\mathbf{r} = (r_1, r_2)^t$ takes on the values of $(0,0)^t$,$(1,0)^t$ or $(0,1)^t$. The resulting plant is therefore a variable structure system [Utkin 1977].

The interface generator for this hybrid system will be formed with respect to two events, c^+ and c^-.

$$c^+ = \{\mathbf{x} \in \Re^n : \mathbf{s}^t\mathbf{x} > -|\alpha|\} \tag{3}$$

$$c^- = \{\mathbf{x} \in \Re^n : \mathbf{s}^t\mathbf{x} < |\alpha|\} \tag{4}$$

where α is a real number and $\mathbf{s}$ is an n-dimensional real vector. These two events form overlapping linear halfspaces and will be called *covering events.* The symbols $\tilde{c}^+$ and $\tilde{c}^-$ are bound with the events c^+ and c^-, respectively. The covering generates three distinct plant events which are represented by the symbols, $\tilde{z}_1$, $\tilde{z}_2$, and $\tilde{z}_3$. The plant state either lies in the deadzone formed by the intersection of c^+ and c^- or else it lies only in one of the halfspaces. Therefore the plant symbols issued by the generator will be either $\tilde{z}_1 = \{\tilde{c}^+, \tilde{c}^-\}$, $\tilde{z}_2 = \{\tilde{c}^+\}$, or $\tilde{z}_3 = \{\tilde{c}^-\}$.

It will be assumed that the supervisor is an identity mapping which simply passes on the plant symbol to the interface actuator. The actuator will then associate each symbol with a control vector $\mathbf{r}$ as follows

$$\mathbf{r} = \begin{cases} (1\,0)^t & \text{if } \{\tilde{c}^+\} \\ (0\,0)^t & \text{if } \{\tilde{c}^+, \tilde{c}^-\} \\ (0\,1)^t & \text{if } \{\tilde{c}^-\} \end{cases} \tag{5}$$

These assumptions for the plant, actuator, generator, and supervisor yield a hybrid system which is, essentially, a variable structure control system. The dynamics of the plant under the supervisor's control are represented by the following set of switching differential equations

$$\dot{\mathbf{x}} = \begin{cases} f_1(\mathbf{x}) & \text{if } \mathbf{s}^t\mathbf{x} < -|\alpha| \\ 0 & \text{if } |\mathbf{s}^t\mathbf{x}| < |\alpha| \\ f_2(\mathbf{x}) & \text{if } \mathbf{s}^t\mathbf{x} > |\alpha| \end{cases} \tag{6}$$

The nature of the system shown in equation (6) is such that the system's structure changes discontinuously as the state crosses over surfaces defined by the equation $\mathbf{s}^t\mathbf{x} = \pm\alpha$. Such a surface is commonly called a *switching surface.*

One objective in variable structure control is to drive the plant state onto the hyperplane, $H_{\mathbf{s}}$, and keep it in the neighborhood of that surface. Define the surface $H_{\mathbf{s}}$ as

$$H_{\mathbf{s}} = \{\mathbf{x} \in \Re^n : \mathbf{s}^t\mathbf{x} = 0\} \tag{7}$$

This is a hyperplane passing through the origin, with normal vector $\mathbf{s}$. The neighborhood of this surface is represented by the set formed by intersecting the two events c^+ and c^-. Since the control objective is to drive the system state into $c^+ \cap c^-$ and keep it there, it is important that this set be an attracting invariant set with respect to the controlled plant's dynamics as shown in equation (6). The $H_{\mathbf{s}}$ which is invariant with respect to the plant's dynamics will be referred to as a *sliding mode.*

An invariant subset with respect to a transformation group $\{\Phi_t\}$ is defined as follows.

Definition 1. The set $H \subset \Re^n$ will be a *Φ-invariant* of the transformation group $\{\Phi_t : \Re^n \to \Re^n\}$ if and only if for any $\mathbf{x} \in H$, $\Phi_t(\mathbf{x}) \in H$ for all $t > 0$.

Of more interest are sets which are attracting invariants of the flow.

Definition 2. The set $H \subset \Re^n$ will be an *attracting Φ-invariant* of the transformation group $\{\Phi_t : \Re^n \to \Re^n\}$ if and only if for any $\mathbf{x} \in H$, there exists a finite $T > 0$ such that $\Phi_t(\mathbf{x}) \in H$ for all $t > T$.

In our example, the transformation groups are the family of transition operators generated by the differential equations (6). These transformations, Φ_t, represent a collection of automorphisms over the state space which are sometimes called the "flow" of the dynamical system.

Unfortunately, not all choices of $\mathbf{s}$ will leave the target event, $c^+ \cap c^-$, invariant. Those hyperplanes which yield invariant target events can be determined directly from the set of vector fields $\{f_1, f_2\}$ representing the system's control policies. Examples of this computation can be found in nonlinear systems theory [Olver 1986]. However, this computation requires explicit equations for these control policies and there are numerous applications where such prior knowledge is unavailable. Uncertainty in the precise form of the control policies can arise from unpredicted variations in the plant's structure. Uncertainty can also arise in highly complex systems where the state space's high dimensionality precludes complete prior knowledge of the distributions. In such situations, it is necessary that the invariants be determined directly from the system's observed behaviour. Since the hybrid system's event covering is defined with respect to these invariants, we see that the problem of finding such invariants is essentially the problem of event identification. In other words, we need to identify a collection of covering events which are invariant with respect to the available control policies f_1 and f_2. This problem is referred to in this chapter as the invariant subspace identification (ISID) problem. The algorithms discussed in the following sections provide one way of solving the ISID problem by direct (active) experimentation.

3 Invariant Subspace Identification Algorithm

Inductive inference is a machine learning protocol in which a system learns by example. It has found significant practical and theoretical uses in learning Boolean

M. D. Lemmon, J. A. Stiver and P. J. Antsaklis, "Event Identification and Intelligent Hybrid Control,"
Hybrid S ystem s , R L Grossman, A Nerode, A P Ravn, H Rischel Eds, pp 269-296, Lecture Notes in
Computer Science, LNCS 736, Springer-Verlag, 1993.

functions by example [Angluin 1983], proving poly-time complexity of linear programming [Khachiyan 1979] and combinatorial optimization [Groetschel 1988] algorithms, developing finite time procedures [Rosenblatt 1962] for training linear classifiers, and estimating sets bounding unknown parameters [Dasgupta 1987]. In this section, an inductive protocol for learning an $n-1$-dimensional invariant subspace of a variable structure system is formally stated.

The inductive protocol developed in this chapter can be seen as consisting of three fundamental components;

- an experiment for generating examples,
- a query to an algorithm called the membership oracle,
- and an update algorithm for modifying the system's current controller (i.e. switching surface).

These components are used to iteratively adjust the system's current estimate for the invariant subspace $H_{\mathbf{S}}$. Figure (2) illustrates the relationship between these three algorithm components.

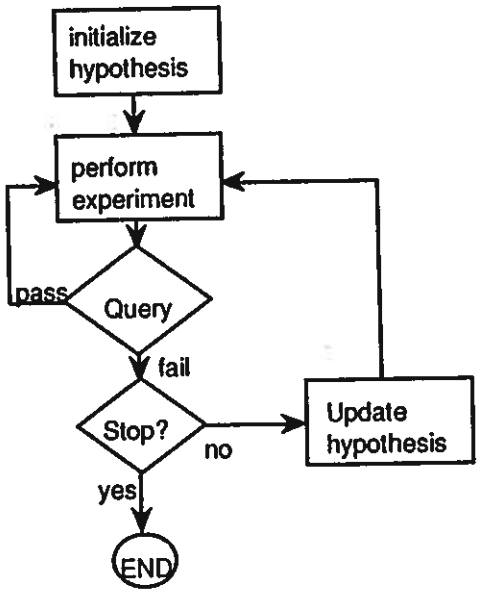

Fig. 2. Flow chart for an inductive inference protocol solving the ISID problem

The algorithm begins by forming an initial hypothesis about the system's sliding mode. This hypothesis takes the form of an n-dimensional vector $\mathbf{s}$ and an n by n symmetric matrix, $\mathbf{Q}$. The vector represents a unit normal to a switching surface, $H_{\mathbf{S}}$, which is hypothesized to be a sliding mode. The matrix represents a convex cone which is known to contain those vectors normal to a sliding modes of the system. Because the matrix, $\mathbf{Q}$, is associated with a convex cone in $\Re^n$, it will have one negative eigenvalue and $n-1$ positive eigenvalues. For the purposes of this section it will therefore be convenient to make the following notational conventions. Let $\mathbf{e}_i$ be the ith eigenvector of $\mathbf{Q}$ and let λ_i be its associated eigenvalue. Assume that the eigenvalues and eigenvectors are ordered so that $\lambda_i > \lambda_{i+1}$ for all i. Define an n by $n-1$ matrix, $\mathbf{E}$, whose columns are the eigenvectors of $\mathbf{Q}$ with positive eigenvalues. This matrix will be called $\mathbf{Q}$'s positive eigenvector matrix. Also form an $n-1$ by $n-1$ diagonal matrix, $\mathbf{L}$, from the positive eigenvalues of $\mathbf{Q}$. This matrix will be called the positive eigenvalue matrix. Both matrices are shown below.

$$\mathbf{E} = \begin{pmatrix} \mathbf{e}_1 & \mathbf{e}_2 & \cdots & \mathbf{e}_{n-1} \end{pmatrix}, \mathbf{L} = \begin{pmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \cdot & \cdot & \cdots & \cdot \\ 0 & 0 & \cdots & \lambda_{n-1} \end{pmatrix}. \tag{8}$$

The normalized eigenvalue matrix will be defined as $\mathbf{R} = \mathbf{L}/|\lambda_n|$.

After forming the initial hypothesis, the algorithm's first component, the experiment, is performed. This component involves the active measurement of the system's state and state velocity. The second algorithm component uses these experimental measurements to make a declaration on the validity of the hypothesis that the switching surface $H_{\mathbf{S}}$ is indeed a sliding mode. The declaration is made by a Boolean functional called the invariance oracle. The oracle's response is either 0 or 1 with a semantic interpretation which depends on the precise form of the Boolean functional. In the applications considered below, a response of 0 has a semantic meaning of TRUE, thereby indicating that the hypothesis *is* consistent with the measured data. A response of 1 has a semantic meaning of MAYBE, thereby indicating that the hypothesis *may not be* consistent with the measured data. If the answer is TRUE then nothing is done. If the answer is MAYBE, however, then the current hypothesis is modified using the algorithm's third component, the update algorithm.

The update algorithm uses a modification of the central-cut ellipsoid method [Shor 1977] to recompute the symmetric matrix $\mathbf{Q}$ and the vector $\mathbf{s}$. In modifying the hypothesis after the oracle's FALSE declaration, the update procedure attempts to generate a new hypothesis which is consistent with prior experimental data. This basic cycle of experiment, query, and update continues until an attracting invariant subspace is found.

The ISID algorithm can now be formally stated.

Invariant Subspace Identification (ISID) Algorithm

1. **Initialize:** Initialize an n by n symmetric matrix, $\mathbf{Q}$, which has $n-1$ positive

M. D. Lemmon, J. A. Stiver and P. J. Antsaklis, "Event Identification and Intelligent Hybrid Control,"
Hybrid S ystem s , R L Grossman, A Nerode, A P Ravn, H Rischel Eds, pp 269-296, Lecture Notes in
Computer Science, LNCS 736, Springer-Verlag, 1993.

eigenvalues and 1 negative eigenvalue such that if $H_{\mathbf{Z}}$ is a sliding mode, then $\mathbf{z}^t\mathbf{Q}\mathbf{z} < 0$. Compute the eigendecomposition of $\mathbf{Q}$.

2. **Form Hypothesis:** Set the system's current switching surface, $\mathbf{s}$, equal to the negative eigenvector, $\mathbf{e}_n$, of $\mathbf{Q}$.
3. **Experiment:** Measure the system's state and state velocity, $\mathbf{x}$ and $\dot{\mathbf{x}}$.
4. **Query:** Compute the invariance oracle's response,

$$I_1(\mathbf{x},\dot{\mathbf{x}},\mathbf{s}) = \begin{cases} 0 \text{ if } (\mathbf{s}^t\mathbf{x})\,(\mathbf{s}^t\dot{\mathbf{x}}) < 0 \\ 1 \text{ otherwise} \end{cases} \tag{9}$$

5. **Update Hypothesis:** If the oracle returns 1 (MAYBE), then recompute $\mathbf{Q}$ using the following equations,

$$\mathbf{c} = \operatorname{sgn}(\mathbf{e}_n^t\mathbf{x})\mathbf{E}^t\dot{\mathbf{x}}, \tag{10}$$

$$\mathbf{b} = \frac{\mathbf{R}^{-1}\mathbf{c}}{\sqrt{\mathbf{c}^t\mathbf{R}^{-1}\mathbf{c}}}, \tag{11}$$

$$\mathbf{a} = -\frac{1}{n}\mathbf{b}, \tag{12}$$

$$\overline{\mathbf{R}}^{-1} = \frac{(n-1)^2}{(n-1)^2-1}\left(\mathbf{R}^{-1} - \frac{2}{n}\mathbf{b}\mathbf{b}^t\right), \tag{13}$$

$$\mathbf{x}_a = \mathbf{E}\mathbf{a} + \mathbf{e}_n, \tag{14}$$

$$\overline{\mathbf{Q}} = (\mathbf{I} - \mathbf{e}_n\mathbf{x}_a^t)\mathbf{E}\overline{\mathbf{R}}\mathbf{E}^t(\mathbf{I} - \mathbf{x}_a\mathbf{e}_n^t), \tag{15}$$

Set $\mathbf{Q}$ equal to $\overline{\mathbf{Q}}$ and recompute the eigendecomposition of $\mathbf{Q}$.

6. If the oracle returns 0 (TRUE), then do nothing.
7. **Loop:** go to step 2.

4 Invariance Oracles

This section derives the oracle used by the ISID algorithm and formally stated in equation (9). The oracle is a Boolean functional which evaluates a sufficient condition for the set $H_{\mathbf{S}}$ to be attracting and invariant.

Consider a set X called the sample set and let M be a measurable subset of X. The membership oracle is defined as follows.

Definition 3. Given a sample set X and a measurable set $M \subset X$, the *membership oracle* for M is a mapping, $O : X \to \{0,1\}$, such that for any $\mathbf{x} \in X$,

$$O(\mathbf{x}) = \begin{cases} 0 \text{ if and only if } \mathbf{x} \in M \\ 1 \text{ if and only if } \mathbf{x} \notin M \end{cases}. \tag{16}$$

The membership or M-oracle can be thought of as a decision machine determining whether or not an example is a member of set M. An example which is an element of M will be called a positive M-example. If M^c is the complement of M, then a positive M^c-example will sometimes be called a negative example.

In this regard, the M-oracle's response 0 or 1 can be interpreted as a TRUE or FALSE declaration, respectively, concerning the membership of the example.

In certain cases, complete membership information may not be practical. It is therefore desirable to consider a weaker form of the M-oracle.

Definition 4. Given a sample set X and a measurable set $M \subset X$, the mapping, $O : X \to \{0,1\}$, is called an *incomplete M-oracle* if there exists another measurable set N such that $M \subseteq N$ and the mapping, O, is an N-oracle.

The incomplete M-oracle is a weaker version of the M-oracle since it only declares that the example is not an element of M. It does not make any declaration about an example's membership in M. In this regard, an incomplete oracle's response of 0 or 1 can be interpreted as a response of MAYBE or FALSE declaration, respectively, on the example's membership in M.

An *invariance oracle* will be a Boolean functional which declares whether or not a given subspace, $H_{\mathbf{S}}$, is attracting and Φ-invariant. Therefore the first step in defining an "invariance" oracle is to determine a test by which invariance can be determined. Sufficient conditions for attracting Φ-invariant sets form the basis of these tests. The following theorem provides a specific example of such a test.

Theorem 5. *Let $\mathbf{s}$ be a given n-dimensional real vector and let $\dot{\mathbf{x}}$ be given by equation (2). If the following condition*

$$(\mathbf{s}^t\mathbf{x})\,(\mathbf{s}^t\dot{\mathbf{x}}) < 0, \tag{17}$$

is satisfied for all $\mathbf{x} \notin H_{\mathbf{S}}$, then the subspace, $H_{\mathbf{S}}$, is an attracting Φ-invariant set.

Proof: Define the functional $V(\mathbf{x}) = \frac{1}{2}(\mathbf{s}^t\mathbf{x})^2$. Clearly, $V > 0$ for all $\mathbf{x}$. By the theorem's assumption, $\dot{V} < 0$, for all $\mathbf{x} \notin H_{\mathbf{S}}$. Therefore by theorem 8 in [Utkin 1977], $H_{\mathbf{S}}$ must be a sliding mode and is therefore an attracting Φ-invariant set of the flow. •

It should be apparent that equation (17) can be recast as a logical function making a declaration about the consistency of the measured state and state velocity with the hypothesis that $H_{\mathbf{S}}$ is a sliding mode. This, then motivates the following definition for an "invariance" oracle.

Definition 6. The Boolean functional, $I_1 : \Re^{3n} \to \{0,1\}$, defined by equation (9) will be called an *invariance oracle.*

Let A denote a subset of $\Re^n$ consisting of those n-dimensional vectors $\mathbf{s}$ for which $H_{\mathbf{S}}$ is attracting and Φ-invariant. This set, A, will be referred to as the set of attracting and invariant subspaces. The following theorem states that the invariance oracle, I_1, is an incomplete A^c-oracle where A^c is the complement of set A.

Theorem 7. *Let A be the set of attracting invariant subspaces. If the function $I_1 : \Re^{3n} \to \{0,1\}$ is an invariance oracle, then it is an incomplete A^c-oracle.*

M. D. Lemmon, J. A. Stiver and P. J. Antsaklis, "Event Identification and Intelligent Hybrid Control,"
Hybrid S ystem s , R L Grossman, A Nerode, A P Ravn, H Rischel Eds, pp 269-296, Lecture Notes in Computer Science, LNCS 736, Springer-Verlag, 1993.

Proof: Let A_1 be a set of n-dimensional vectors $\mathbf{s}$ such that $I_1 = 0$ for any $\mathbf{x}$ and $\dot{\mathbf{x}}$ given by equation (2). By definition (3), I_1 must be an A_1^c-oracle. By theorem 1, any element of A_1 must also be an attracting Φ-invariant set. Therefore $A_1 \subset A$, which implies $A^c \subset A_1^c$. This therefore establishes I_1 as an incomplete A^c-oracle according to definition (4). •

The set A_1 defined in the above proof will be referred to as the set of attracting invariant subspaces which are *declarable* by the invariance oracle, I_1. Note that this set is smaller than A, the set of all attracting invariant subspaces. For this reason, the oracle is incomplete and its response of 0 or 1 is a declaration of TRUE or MAYBE, respectively, concerning the membership of $\mathbf{s}$ in A.

In the remainder of this chapter, the data collection gathered by an experiment will be denoted as $\mathcal{X} = \{\mathbf{x}, \dot{\mathbf{x}}\}$. It is assumed, that these measurements have no measurement noise, so that the oracle's declarations are always correct. An invariance oracle which always makes the correct declaration for a given data collection, $\mathcal{X}$, will be called a *perfect oracle*. In practical oracle realizations, the assumption that the invariance oracle is perfect may be too optimistic due to measurement uncertainty. This realization prompts the definition of an *imperfect oracle* as an oracle whose declarations are incorrect with a given probability. The distinction between perfect and imperfect oracles is critical, because inductive protocols based on oracle queries can fail disasterously with imperfect oracles. The convergence results of section (6) only apply to perfect invariance oracles. Precisely how to manage failures due to imperfect oracles is an important issue for future study. A preliminary indication of how to handle this problem will be discussed in section (8).

5 Ellipsoidal Update Method

The ISID algorithm uses an update procedure which recursively adjusts an estimate for the set A_1 of attracting invariant subspaces declarable by I_1. The proposed updating procedure is therefore a set-estimation algorithm which is closely related to set-membership identification algorithms [Dasgupta 1987] [Deller 1989]. It is also related to analytical techniques used in proving polynomial oracle-time complexity for certain optimization algorithms [Groetschel 1988] [Khachiyan 1979]. The common thread between both of these related areas is the use of the ellipsoid method [Shor 1977] [Bland 1981], which the following discussion also uses to great advantage.

An important property of A_1 (the set of declarable subspaces) is provided in the following lemma.

Lemma 8. *A_1 is a convex cone centered at the origin.*

Proof: Consider a specific collection of measurements as $\mathcal{X} = \{\mathbf{x}, \dot{\mathbf{x}}\}$. Define $C_{\mathcal{X}}$ as

$$C_{\mathcal{X}} = \{\mathbf{s} \in \Re^n : I_1(\mathcal{X}, \mathbf{s}) = 0\}. \tag{18}$$

A_1 will therefore be given by

$$A_1 = \bigcap_{\mathcal{X}} C_{\mathcal{X}}. \tag{19}$$

Since the oracle's response for a given $\mathbf{s}$ is independent of the vector's magnitude, $C_{\mathcal{X}}$ must be a cone centered at the origin. Since $C_{\mathcal{X}}$ is formed by the intersection of two halfspaces (see inequality (17)), it must also be convex. A_1 is therefore the intersection of a collection of convex cones centered at the origin and must therefore be one itself. •

The significance of the preceding lemma is that it suggests A_1 may be well approximated by sets which are themselves convex cones centered at the origin. A particularly convenient selection of approximating cones are the so-called ellipsoidal cones.

An "ellipsoidal cone" cone is defined as follows,

Definition 9. The *ellipsoidal cone*, $C_e(\mathbf{Q})$, is

$$C_e(\mathbf{Q}) = \{\mathbf{s} \in \Re^n : \mathbf{s}^t\mathbf{Q}\mathbf{s} < 0\}, \tag{20}$$

where $\mathbf{Q}$ is an n by n symmetric matrix with $n-1$ positive eigenvalues and one negative eigenvalue.

In the update procedure to be derived below, an ellipsoidal cone, $C_e(\mathbf{Q})$, will be used as an initial estimate for A_1. The current hypothesis is that the subspace normal to the negative eigenvector of $\mathbf{Q}$ is an attracting invariant set. If any data collection, $\mathcal{X}$, results in the oracle, I_1, declaring 1 (MAYBE), the query is said to have *failed*. The information from that failed query can be used to identify a set of subspaces which cannot possibly lie in A_1. This set will be referred to as the "inconsistent" set of subspaces generated by $\mathcal{X}$. The following lemma provides one characterization of these sets.

Lemma 10. *Let $C_e(\mathbf{Q})$ be an ellipsoidal cone with negative eigenvector, $\mathbf{e}_n$. Let $\mathcal{X}$ be a data collection for which a perfect invariance oracle, I_1, declares a failure, $I_1(\mathcal{X}, \mathbf{e}_n) = 1$. If $A_1 \subset C_e(\mathbf{Q})$, then $A_1 \subset C_e(\mathbf{Q}) \cap H(\mathcal{X}, \mathbf{e}_n)$ where*

$$H(\mathcal{X}, \mathbf{e}_n) = \left\{\mathbf{s} \in \Re^n : \mathbf{s}^t\dot{\mathbf{x}} < \operatorname{sgn}(\mathbf{e}_n^t\mathbf{x})\mathbf{e}_n^t\mathbf{x}\right\}. \tag{21}$$

The set $H(\mathcal{X}, \mathbf{e}_n)$ will be called the inconsistent set *generated by $\mathcal{X}$.*

Proof: If a perfect invariance oracle I_1 returns 1 for $\mathcal{X}$ given the subspace represented by $\mathbf{e}_n$, then the following inequality holds.

$$\left(\mathbf{e}_n^t\dot{\mathbf{x}}\right)\left(\mathbf{e}_n^t\dot{\mathbf{x}}\right) > 0. \tag{22}$$

Note that for all $\mathbf{z}$ such that $\mathbf{z}^t\dot{\mathbf{x}} \geq \mathbf{e}_n^t\dot{\mathbf{x}}$, it can be inferred by the comparison principle that $\mathbf{z}^t\mathbf{x} \geq \mathbf{e}_n^t\mathbf{x}$. Similar arguments apply if the inequalities are reversed. Therefore any subspace, $H_{\mathbf{z}}$, such that

$$\mathbf{z}^t\dot{\mathbf{x}} \geq \operatorname{sgn}(\mathbf{e}_n^t\mathbf{x})\mathbf{e}_n^t\dot{\mathbf{x}}, \tag{23}$$

cannot possibly be an attracting invariant set. The collection of such subspaces form the complement of the halfspace $H(\mathcal{X}, \mathbf{e}_n)$ defined in the theorem. Since A_1 is assumed to lie in $C_e(\mathbf{Q})$, it must therefore lie in the intersection of $C_e(\mathbf{Q})$ with $H(\mathcal{X}, \mathbf{e}_n)$. •

M. D. Lemmon, J. A. Stiver and P. J. Antsaklis, "Event Identification and Intelligent Hybrid Control," Hybrid S ystem s , R L Grossman, A Nerode, A P Ravn, H Rischel Eds, pp 269-296, Lecture Notes in Computer Science, LNCS 736, Springer-Verlag, 1993.

The significance of the preceding lemma is that the inconsistent set is an n-dimensional halfspace in $\Re^n$. To discuss this more fully, we first need to introduce the linear varieties of $n-1$-dimensional subspaces.

Definition 11. Let S be an $n-1$-dimensional subspace of $\Re^n$ and let $\mathbf{x}$ be an n-dimensional real vector. The *linear variety* of S generated by $\mathbf{x}$ is the set

$$V(S,\mathbf{x}) = \{\mathbf{s}+\mathbf{x} : \mathbf{s} \in S\}. \tag{24}$$

The following lemma shows that the inconsistent set forms a halfspace in the linear variety, $V(\mathrm{sp}(\mathbf{E}), \mathbf{e}_n)$, where $\mathrm{sp}(\mathbf{E})$ is the span of the $n-1$ positive eigenvectors of $\mathbf{Q}$.

Lemma 12. *Let $C_e(\mathbf{Q})$ be an ellipsoidal cone with negative eigevector, $\mathbf{e}_n$. Let $V(\mathrm{sp}(\mathbf{E}), \mathbf{e}_n)$ be a linear variety of the subspace spanned by the positive eigenvectors of $\mathbf{Q}$. If the inconsistent set $H(\mathcal{X}, \mathbf{e}_n)$ is as defined in lemma (10), then the set $H(\mathcal{X}, \mathbf{e}_n) \cap V(\mathrm{sp}(\mathbf{E}), \mathbf{e}_n)$ is an $n-1$ dimensional halfspace.*

Proof: Any vector $\mathbf{s}$ which lies in $V(\mathrm{sp}(\mathbf{E}), \mathbf{e}_n)$ can be written as

$$\mathbf{s} = \mathbf{E}\mathbf{w} + \mathbf{e}_n \tag{25}$$

where $\mathbf{w}$ is an $n-1$ dimensional real vector. If $\mathrm{sgn}(\mathbf{e}_n^t\mathbf{x}) > 0$ then inserting $\mathbf{s}$ into equation (21) yields

$$\mathbf{w}^t\mathbf{E}^t\dot{\mathbf{x}} + \mathbf{e}_n^t\dot{\mathbf{x}} < \mathbf{e}_n^t\dot{\mathbf{x}} \tag{26}$$

which implies that $\mathbf{w}^t\mathbf{E}^t\dot{\mathbf{x}} < 0$. This, of course, determines a halfspace in the linear variety. A similar equation can be obtained if $\mathrm{sgn}(\mathbf{e}_n^t\mathbf{x}) < 0$. These considerations lead to the following,

$$\mathbf{w}^t\left[\mathrm{sgn}(\mathbf{e}_n^t\mathbf{x})\mathbf{E}^t\dot{\mathbf{x}}\right] < 0 \tag{27}$$

which is an equation for the halfspace in the linear variety generated by the inconsistent set. •

The geometry implied by the preceding lemmas is illustrated in figure (3). The characterization of the ellipsoidal cone and inconsistent sets provided by these lemmas forms the basis for the following theorem. This theorem states the equations used in obtaining a bounding ellipsoidal cone for A_1 from a prior bounding cone and the inconsistent set generated by $\mathcal{X}$. The proof of this theorem is a straightforward application of the central-cut ellipsoid method [Shor 1977].

Theorem 13. *Let $C_e(\mathbf{Q})$ be an ellipsoidal cone with negative eigenvector $\mathbf{e}_n$ such that $A_1 \subset C_e(\mathbf{Q})$. Let $\mathcal{X}$ be a data collection for which $I_1(\mathcal{X}, \mathbf{e}_n) = 1$. There exist ellipsoidal cones, $C_e(\underline{\mathbf{Q}})$ and $C_e(\overline{\mathbf{Q}})$, such that*

$$C_e(\underline{\mathbf{Q}}) \subset H(\mathcal{X}, \mathbf{e}_n) \cap C_e(\mathbf{Q}) \subset C_e(\overline{\mathbf{Q}}). \tag{28}$$

Furthermore if $\mathbf{R} = \mathbf{L}/|\lambda_n|$ where $\mathbf{L}$ is the positive eigenvalue matrix and if $\mathbf{E}$ is the positive eigenvector matrix of $\mathbf{Q}$, then $\overline{\mathbf{Q}}$ is given by equations (10) through (15) and $\underline{\mathbf{Q}}$ is given by equations (10) through (15) where $\underline{\mathbf{R}}^{-1} = \overline{\mathbf{R}}^{-1}/(n-1)^2$ is used in place of $\overline{\mathbf{R}}^{-1}$ in equation (10).

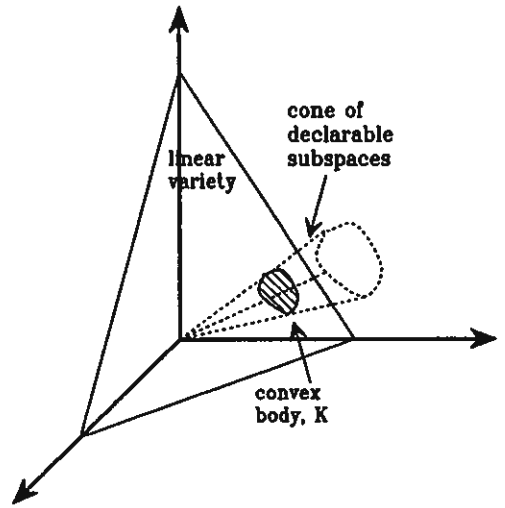

Fig. 3. The set of subspaces, A_M, declarable by a perfect invariance oracle, I_M, forms a convex cone centered at the origin. The intersection of A_M with a linear variety of an $n-1$-dimensional subspace will be a bounded $n-1$-dimensional convex body, K.

Proof: From lemma 2, the intersection of cone $C_e(\mathbf{Q})$ with $V(\mathrm{sp}(\mathbf{E}), \mathbf{e}_n)$ is an ellipsoid of the following form

$$E(\mathbf{R}^{-1}, 0) = \{\mathbf{w} \in \Re^{n-1} : \mathbf{w}^t\mathbf{R}\mathbf{w} < 1\}. \tag{29}$$

From lemma 4, the intersection of the inconsistent set, $H(\mathcal{X}, \mathbf{e}_n)$ and $V(\mathrm{sp}(\mathbf{E}), \mathbf{e}_n)$ will be an $n-1$-dimensional halfspace, H, given by

$$H = \{\mathbf{w} \in \Re^{n-1} : \mathbf{w}^t\mathbf{c} < 0\}, \tag{30}$$

where $\mathbf{c} = \mathrm{sgn}(\mathbf{e}_n^t\mathbf{x})\mathbf{E}^t\dot{\mathbf{x}}$. Therefore the intersection of $C_e(\mathbf{Q})$, $V(\mathrm{sp}(\mathbf{E}), \mathbf{e}_n)$, and $H(\mathcal{X}, \mathbf{e}_n)$ will be an $n-1$-dimensional convex body, K.

It is well known that any bounded convex body can be contained within a unique ellipsoid of minimal volume called the Lowner-John ellipsoid [John 1984]. For convex bodies formed by single cuts of an ellipse, however, the Lowner-John ellipse can be computed in closed form [Groetschel 1988] [Bland 1981]. In particular, let K be the convex body formed by the intersection of an ellipse

$$E(\mathbf{A}, \mathbf{a}) = \left\{\mathbf{x} \in \Re^n : (\mathbf{x}-\mathbf{a})^t\,\mathbf{A}^{-1}\,(\mathbf{x}-\mathbf{a}) \le 1\right\} \tag{31}$$

M. D. Lemmon, J. A. Stiver and P. J. Antsaklis, "Event Identification and Intelligent Hybrid Control," Hybrid S ystem s , R L Grossman, A Nerode, A P Ravn, H Rischel Eds, pp 269-296, Lecture Notes in Computer Science, LNCS 736, Springer-Verlag, 1993.

with a halfspace, $\{\mathbf{x} : \mathbf{c}^t\mathbf{x} < \mathbf{c}^t\mathbf{a}\}$, then the Lowner-John ellipse, $E(\overline{\mathbf{A}}, \overline{\mathbf{a}})$ is given by

$$\overline{\mathbf{a}} = \mathbf{a} - \frac{n^2}{n+1}\mathbf{b}, \tag{32}$$

$$\overline{\mathbf{A}} = \frac{n^2}{n^2-1}\left(\mathbf{A} - \frac{2}{n+1}\mathbf{b}\mathbf{b}^t\right), \tag{33}$$

$$\mathbf{b} = \frac{\mathbf{A}\mathbf{c}}{\sqrt{\mathbf{c}^t\mathbf{A}\mathbf{c}}}. \tag{34}$$

Computing the Lowner-John ellipsoid for $K = E(\mathbf{R}^{-1}, 0) \cap H$ will yield the ellipsoid $E(\overline{\mathbf{R}}^{-1}, \mathbf{a})$ where $\overline{\mathbf{R}}$ and $\mathbf{a}$ are as given in the theorem. Figure (4) illustrates the geometry implied by the central-cut ellipsoid method.

The $n-1$-dimensional Lowner-John ellipsoid generates an n-dimensional ellipsoidal cone. Let $\mathbf{s}$ be any point in the cone generated by the ellipsoid $E(\overline{\mathbf{R}}^{-1}, \mathbf{a})$. There exists an $\alpha \in \Re$ such that $\alpha\mathbf{s}$ is in the linear variety, $V(\mathrm{sp}(\mathbf{E}), \mathbf{e}_n)$. The α for which this is true must satisfy the orthogonality condition,

$$0 = \mathbf{e}_n^t(\alpha\mathbf{s} - \mathbf{e}_n) \tag{35}$$

$$= \alpha\mathbf{e}_n^t\mathbf{s} - 1, \tag{36}$$

which implies that $\alpha = 1/\mathbf{e}_n^t\mathbf{s}$.

Since, $\mathbf{s} = \mathbf{E}\mathbf{w} + \mathbf{e}_n$, the ellipsoid equation for $E(\overline{\mathbf{R}}^{-1}, \mathbf{a})$ is

$$1 > (\mathbf{w}-\mathbf{a})^t\overline{\mathbf{R}}(\mathbf{w}-\mathbf{a}) \tag{37}$$

$$> (\mathbf{s}-\mathbf{x}_a)^t\mathbf{E}^t\overline{\mathbf{R}}\mathbf{E}(\mathbf{s}-\mathbf{x}_a), \tag{38}$$

where $\mathbf{x}_a = \mathbf{E}\mathbf{a} + \mathbf{e}_n$. The vector $\mathbf{s}$ in this equation must, of course, lie in the linear variety generated by $\mathbf{e}_n$, $V(\mathrm{sp}(\mathbf{E}), \mathbf{e}_n)$. From our preceding discussion, any vector in the cone can be pulled back to the variety by appropriate renormalization with α. This then implies that if $\mathbf{s}$ is any vector in the cone, then

$$\left(\frac{\mathbf{s}}{\mathbf{s}^t\mathbf{e}_n} - \mathbf{x}_a\right)^t \mathbf{E}^t\overline{\mathbf{R}}\mathbf{E}\left(\frac{\mathbf{s}}{\mathbf{e}_n^t\mathbf{s}} - \mathbf{x}_a\right) < 1. \tag{39}$$

Multiplying through by $|\mathbf{s}^t\mathbf{e}_n|^2$, we obtain

$$\mathbf{s}^t\left[(\mathbf{I} - \mathbf{e}_n\mathbf{x}_a^t)\,\mathbf{E}^t\overline{\mathbf{R}}\mathbf{E}\,(\mathbf{I} - \mathbf{x}_a\mathbf{e}_n^t) - \mathbf{e}_n\mathbf{e}_n^t\right]\mathbf{s} < 0. \tag{40}$$

This inequality determines an ellipsoidal cone and the term within the square brackets is $\overline{\mathbf{Q}}$.

$\underline{\mathbf{Q}}$ is obtained by noting that if $E(\overline{\mathbf{R}}^{-1}, \mathbf{a})$ is a Lowner-John ellipsoid for K, then $E(\overline{\mathbf{R}}^{-1}/(n-1)^2, \mathbf{a})$ is an ellipsoid contained within K. By repeating the preceding construction with this smaller ellipsoid, the equation for $\underline{\mathbf{Q}}$ is obtained.

•

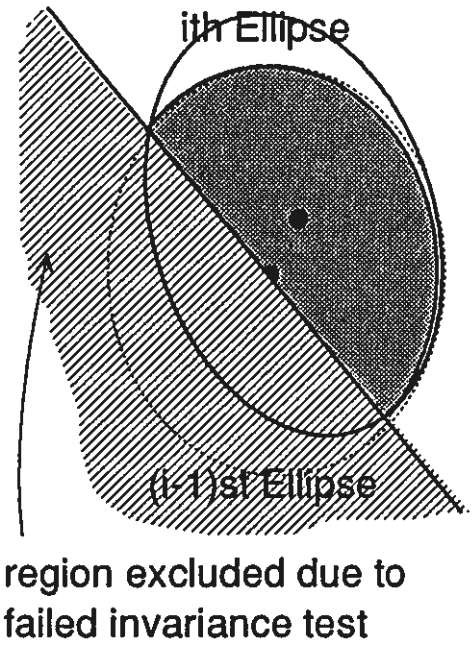

Fig. 4. Lowner-John ellipsoid for convex body formed by a central cut ellipsoid.

6 Convergence and Complexity

This section shows that the ISID algorithm generates a sequence of ellipsoidal cones whose negative eigenvectors must eventually lie in A_1. In particular, it is shown that if A_1 is non-empty then the ISID algorithm must converge after a finite number of MAYBE (1) declarations by the invariance oracle. It is further shown that under certain conditions the convergence time scales as $o(n^{2.5})$ where n is the plant's state space dimension. The section therefore proves that the ISID algorithm has finite oracle-time convergence and polynomial oracle-time complexity where oracle-time is measured by the number of MAYBE declarations made by the invariance oracle.

To prove the convergence of the ISID algorithm requires that there be some measure of the ellipsoidal cone's size or "volume". The set function used to define this volume is given below.

Definition 14. Let $C_e(\mathbf{Q})$ be an ellipsoidal cone and let the eigenvalues of $\mathbf{Q}$

M. D. Lemmon, J. A. Stiver and P. J. Antsaklis, "Event Identification and Intelligent Hybrid Control,"
Hybrid S ystem s , R L Grossman, A Nerode, A P Ravn, H Rischel Eds, pp 269-296, Lecture Notes in
Computer Science, LNCS 736, Springer-Verlag, 1993.

be ordered as $\lambda_i > \lambda_{i+1}$ $(i = 1, \ldots, n)$. The *volume* of cone $C_e(\mathbf{Q})$ is defined to be

$$\text{vol}C_e(\mathbf{Q}) = \sqrt{\prod_{i=1}^{n-1} \frac{|\lambda_n|}{\lambda_i}}. \tag{41}$$

The volume of an ellipsoid, $E(\mathbf{A}, \mathbf{a})$, will be proportional to the square root of the determinant of $\mathbf{A}$. Since the determinant of $\mathbf{A}$ is simply the product of its eigenvalues, it should be clear that the preceding definition is using the volume of the $n-1$-dimensional ellipsoid contained in the linear variety $V(\text{sp}(\mathbf{E}), \mathbf{e}_n)$ as the "volume" of the cone.

The following theorem shows that the ISID algorithm must locate an attracting invariant subspace after a finite number of failed queries to a perfect invariance oracle.

Theorem 15. *Initialize the ISID algorithm with an ellipsoidal cone whose volume is unity and which is known to contain A_1. Let ϵ denote the volume of the smallest ellipsoidal cone containing A_1. If n is the state space dimension, then the ISID algorithm will determine an attracting invariant subspace after no more than $2(n-1)\ln \epsilon^{-1}$ failed queries to a perfect invariance oracle.*

Proof: Consider the ellipsoidal cone $C_e(\mathbf{Q}_i)$ after the ith failed invariance test. Let $\mathbf{E}$ and $\mathbf{L}$ be the positive eigenvector and eigenvalue matrices of $\mathbf{Q}_i$, respectively. The volume of this ellipsoid will be given by

$$\text{vol}C_e(Q_i) = \sqrt{\frac{1}{\prod_{j=1}^{n-1} \lambda_j(\mathbf{R})}}, \tag{42}$$

where $\lambda_j(\mathbf{R})$ is the jth positive eigenvalue of $\mathbf{R}$ and $\mathbf{R} = \mathbf{L}/|\lambda_n|$. Consider the ellipsoidal cone obtained using equations (10) through (15) of section (3). The symmetric matrix characterizing this cone is $\overline{\mathbf{Q}} = \mathbf{X}^t\mathbf{Y}\mathbf{X}$ where

$$\mathbf{X} = \begin{pmatrix} \mathbf{E}^t(\mathbf{I} - \beta \mathbf{e}_a \mathbf{e}_n^t) \\ \mathbf{e}_n^t \end{pmatrix}, \tag{43}$$

$$\mathbf{Y} = \begin{pmatrix} \mathbf{R} & 0 \\ 0 & -1 \end{pmatrix}. \tag{44}$$

where $\beta = ||\mathbf{x_a}||$. Applying the orthogonal transformation,

$$\mathbf{P} = \begin{pmatrix} \mathbf{E}\, \mathbf{e}_n \end{pmatrix}, \tag{45}$$

to $\mathbf{X}$, yields

$$\mathbf{P}^t\mathbf{X}^t = \begin{pmatrix} \mathbf{I} & 0 \\ -\beta \mathbf{e}_a^t \mathbf{E} & 1 \end{pmatrix} \tag{46}$$

where $\beta = ||\mathbf{x}_a||$ and $\beta \mathbf{e}_a = \mathbf{x}_a$. Recall that $\mathbf{x}_a$ is the center of the updated ellipsoid in the linear variety $V(\text{sp}(\mathbf{E}), \mathbf{e}_n)$. For convenience, let $\mathbf{v}^t = -\beta \mathbf{e}_a^t E$.

Since the eigenvalues of $\overline{\mathbf{Q}}$ are unchanged by an orthogonal transformation, the eigenvalues of $\mathbf{P}^t\mathbf{X}^t\mathbf{Y}\mathbf{X}\mathbf{P}$ can be used to compute the volume of $\overline{\mathbf{Q}}$. This transformed matrix has the form

$$\mathbf{P}^t\overline{\mathbf{Q}}\mathbf{P} = \mathbf{P}^t\mathbf{X}^t\mathbf{Y}\mathbf{X}\mathbf{P} \tag{47}$$

$$= \begin{pmatrix} \overline{\mathbf{R}} & \overline{\mathbf{R}}\mathbf{v} \\ \mathbf{v}^t\overline{\mathbf{R}} & \mathbf{v}^t\overline{\mathbf{R}}\mathbf{v} - 1 \end{pmatrix}. \tag{48}$$

Note that $\overline{\mathbf{R}}$ is an $n-1$ by $n-1$ leading principal submatrix of $\mathbf{P}^t\overline{\mathbf{Q}}\mathbf{P}$, so the eigenvalues of the two matrices satisfy the following interlacing property [Golub 1983].

$$\lambda_n(\overline{\mathbf{Q}}) \leq \lambda_{n-1}(\overline{\mathbf{R}}) \leq \lambda_{n-1}(\overline{\mathbf{Q}}) \leq \cdots \leq \lambda_2(\overline{\mathbf{Q}}) \leq \lambda_1(\overline{\mathbf{R}}) \leq \lambda(\overline{\mathbf{Q}}). \tag{49}$$

Since it is known that $\lambda_n(\overline{\mathbf{Q}})$ is negative (by the definition of an ellipsoidal cone), it can be shown that

$$\lambda_n(\mathbf{P}^t\mathbf{X}^t\mathbf{Y}\mathbf{X}\mathbf{P}) \leq \sigma_n^2(\mathbf{P}^t\mathbf{X}^t)\lambda_n(\mathbf{Y}), \tag{50}$$

where $\sigma_n(\mathbf{P}^t\mathbf{X}^t)$ is the smallest singular value of $\mathbf{P}^t\mathbf{X}^t$ and $\lambda_n(\mathbf{Y})$ is the negative eigenvalue of $\mathbf{Y}$ [Golub 1983]. Note that this eigenvalue must be negative one (by construction of $\mathbf{Y}$). Also note that the singular value must satisfy the following inequality for any $\mathbf{x} \in \Re^n$,

$$\sigma_n^2(\mathbf{P}^t\mathbf{X}^t) \leq \frac{\mathbf{x}^t\mathbf{P}^t\mathbf{X}^t\mathbf{X}\mathbf{P}\mathbf{x}}{\mathbf{x}^t\mathbf{x}}. \tag{51}$$

In particular, if we let $\mathbf{x} = (0 \cdots 01)^t$, then the smallest singular value must be less than unity. It can therefore be concluded that $|\lambda_n(\overline{\mathbf{Q}})| < 1$.

With the preceding results, it can be concluded that

$$\text{vol}C_e(\overline{\mathbf{Q}}) = \sqrt{\prod_{j=1}^{n-1} \frac{\lambda_n(\overline{\mathbf{Q}})}{\lambda_j(\overline{\mathbf{Q}})}} \tag{52}$$

$$\leq \sqrt{\prod_{j=1}^{n-1} \frac{1}{\lambda_j(\overline{\mathbf{R}})}} \tag{53}$$

$$\leq e^{-\frac{1}{2(n-1)}} \text{vol}C_e(\mathbf{Q}). \tag{54}$$

Inequality (54) is a consequence of the bound on the absolute value of the negative eigenvalue as well as the interlacing property (Eq. (49)). This inequality is simply the volume of an ellipsoid $E(\overline{\mathbf{R}}^{-1}, \mathbf{a})$. Recall, however, that $\overline{\mathbf{R}}$ is obtained from $\mathbf{R}$ using the central-cut ellipsoid method. The relationship between the volumes of these ellipsoids is given by the last line of the inequality. This last inequality [Groetschel 1988] is

$$\frac{\text{vol}E(\overline{\mathbf{A}}, \overline{\mathbf{a}})}{\text{vol}E(\mathbf{A}, \mathbf{a})} = e^{-1/2n}, \tag{55}$$

M. D. Lemmon, J. A. Stiver and P. J. Antsaklis, "Event Identification and Intelligent Hybrid Control," Hybrid S ystem s , R L Grossman, A Nerode, A P Ravn, H Rischel Eds, pp 269-296, Lecture Notes in Computer Science, LNCS 736, Springer-Verlag, 1993.

which bounds the rate at which ellipsoid volumes decrease when the central-cut ellipsoid method is used.

Since the initial ellipsoidal cone's volume is unity, then the ellipsoidal cone's volume after the Lth failed query must be bounded as follows,

$$\text{vol}C_\epsilon(\mathbf{Q}_L) \leq e^{-\frac{L}{2(n-1)}}. \tag{56}$$

However, $C_\epsilon(\mathbf{Q}_L)$ cannot be smaller than ϵ by assumption, therefore the number of failed queries, L, must satisfy

$$\epsilon \leq e^{-\frac{L}{2(n-1)}}. \tag{57}$$

Rearranging this inequality to extract L shows that the number of failed invariance queries can be no larger than the bound stated by the theorem. •

The following corollary for the preceding theorem establishes the polynomial oracle-time complexity of the ISID algorithm.

Corollary 16. *Assume that A_1 is a set which is contained within an ellipsoidal cone characterized by a matrix, $\mathbf{Q}$, whose normalized positive eigenvalues satisfy the inequality*

$$\frac{|\lambda_n|}{\lambda_i} > \gamma \tag{58}$$

for $1 > \gamma > 0$ and $i = 1, \ldots, n-1$. Under the assumptions of theorem (15), the ISID algorithm will determine an attracting invariant subspace after no more than $2(n-1)^2 \ln(n-1) + (n-1)^2 \ln \gamma^{-1}$ MAYBE declarations by the invariance oracle.

Proof: Because of the constraints on $\mathbf{Q}$, the volume of the smallest bounding ellipsoid will be no greater than $\gamma^{(n-1)/2}(n-1)^{-n+1}$. Inserting this into the bound of theorem (15) yields the asserted result. •

The significance of the preceding corollary is apparent when we consider how such restrictions on the eigenvalues of $\mathbf{Q}$ might arise. In particular, if the ISID algorithm is realized in finite precision arithmetic, then γ is proportional to the least significant bit of the realization. In this regard, the result shows that for finite precision implementations, there is an upper bound on the number of queries which the system can fail before exceeding the realization's precision. In particular, this result then shows that the bound scales as $o(n^{2.5})$ where n is the number of plant states. This result thereby establishes the polynomial oracle-time complexity of the algorithm.

7 Example: AUV Stabilization

This section discusses an application of the ISID algorithm to the stabilization of an autonomous underwater vehicle's (AUV) dive plane dynamics. This problem represents an example of the ISID algorithm's use as an adaptive variable structure control algorithm.

AUV dynamics are highly nonlinear and highly uncertain systems. Nonlinearities arise from hydrodynamic forces, uncompensated buoyancy effects, as well as cross-state dynamical coupling. Uncertainties arise due to environmental effects such as changing current and water conditions as well as poorly known mass and hydrodynamic properties. When an AUV retrieves a large object, for example, the drag and mass of this object may substantially modify the vehicle's buoyancy and drag coefficients. Such changes cannot be accurately modeled beforehand and can therefore have a disasterous effect on the success of the AUV's mission. In these situations, it would be highly desirable to develop an algorithm which can quickly and efficiently relearn the stabilizing controller for the system. The ISID algorithm represents one method for achieving this goal.

The following simulation results illustrate how the ISID algorithm can quickly stabilize an AUV's dive plane dynamics. The simplified equations of motion for vehicle (pitch) angle of attack, θ, in the dive plane as a function of velocity, v, may be written as

$$\ddot{\theta} = K_1\dot{\theta} + K_2\theta|\theta| + K_3\theta|v| + u_\theta, \tag{59}$$

$$\dot{v} = -v + K_4|\theta|v + u_v, \tag{60}$$

where K_1, K_2, K_3, and K_4 are hydrodynamic force coefficients. u_v and u_θ represent control forces applied in the velocity and angle of attack channels, respectively. These equations clearly show how nonlinearities enter the dynamics through the hydrodynamic cross coupling between θ and v. Uncertainty arises from the simple fact that the hydrodynamic coefficients may be poorly known. In general, these coefficients will be complex functions of vehicle geometry, speed, orientation, and water conditions. Consequently, they can never be completely characterized because there are too many degrees of freedom. Figure (5) illustrates the geometry implied by the equations of motion.

Figure (6) illustrates the behaviour of an AUV without active attitude control. The figure shows the 3-d state space trajectory for a vehicle with initial condition $\theta = 1$ and $v = 1$. The commanded state is $\theta = 0$ and $v = 2$. Without active attitude control, $u_\theta = 0$, and $u_v = -v + 2$. In this example, the system is hydrodynamically stable so that natural system damping can be used to eventually null the angle of attack. The figure shows that by using this control strategy, the vehicle exhibits large oscillations in θ and v before settling to the commanded state. For this particular system, the results therefore indicate that the angle of attack should be actively nulled to improve trajectory tracking.

Variable structure control (VSC) has emerged as a powerful technique for controlling AUV's with uncertain dynamics [Yoerger 1985]. In the following simulations, a hierarchical variable structure controller [DeCarlo 1988] with boundary layer was designed. The controls, u_θ and u_v, have the following form

$$u_\theta = \begin{cases} 1 & \text{if } \mathbf{s}_\theta^t\mathbf{x} \leq -\epsilon \\ \frac{\mathbf{s}_\theta^t\mathbf{x}}{\epsilon} & \text{if } -\epsilon < \mathbf{s}_\theta^t\mathbf{x} < \epsilon \\ -1 & \text{if } \mathbf{s}_\theta^t\mathbf{x} > \epsilon \end{cases}, \tag{61}$$

M. D. Lemmon, J. A. Stiver and P. J. Antsaklis, "Event Identification and Intelligent Hybrid Control," Hybrid S ystem s , R L Grossman, A Nerode, A P Ravn, H Rischel Eds, pp 269-296, Lecture Notes in Computer Science, LNCS 736, Springer-Verlag, 1993.

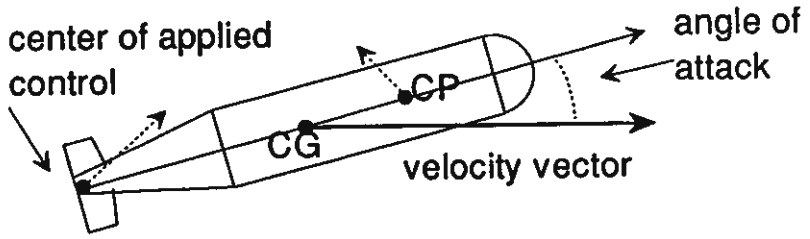

Fig. 5. Autonomous Underwater Vehicle Diveplane Dynamics

$$u_v = \begin{cases} h(\mathbf{s}_\theta^t \mathbf{x}) & \text{if } \mathbf{s}_v^t \mathbf{x} \leq -\epsilon \\ \frac{\mathbf{s}_v^t \mathbf{x}}{\epsilon} & \text{if } -\epsilon < \mathbf{s}_v^t \mathbf{x} < \epsilon \\ -h(\mathbf{s}_\theta^t \mathbf{x}) & \text{if } \mathbf{s}_v^t \mathbf{x} > \epsilon \end{cases}, \tag{62}$$

where $\epsilon > 0$ denotes the width of the boundary layer and $\mathbf{x} = (\theta, \dot{\theta}, v)^t$ is the state vector. The function $h : \Re \to \{0, 1\}$ is assumed to have the following form

$$h(x) = \begin{cases} 0 \text{ if } |x| > \epsilon \\ 1 \text{ otherwise} \end{cases}, \tag{63}$$

and is used to implement a control hierarchy in which the system nulls angle of attack prior to nulling commanded velocity errors. The n-dimensional vectors $\mathbf{s}_\theta$ and $\mathbf{s}_v$ represent hyperplanes called switching surfaces just as was originally shown in equation (2) of the introduction.

The initial design of variable structure controllers can usually be partitioned into two phases. The first phase consists of determining the switching surfaces on which the system trajectories exhibit the desired transient response. The second phase determines the control strategies (gain levels) which insure that the switching surfaces are attracting invariant sets. Such switching surfaces are called sliding modes, since the system state is eventually captured by and slides along the switching surface. The need for adaptive identification of these surfaces arises when the system's structure changes in an unpredictable manner as when the vehicle retrieves a bulky package. In order to preserve system autonomy, the two phase design procedure cannot be followed, since the system's control strategies were fixed at the system's initial design. Consequently, the only part of the controller which can be modified is the switching surface and this modification must be done adaptively on the basis of the system's observed behaviour.

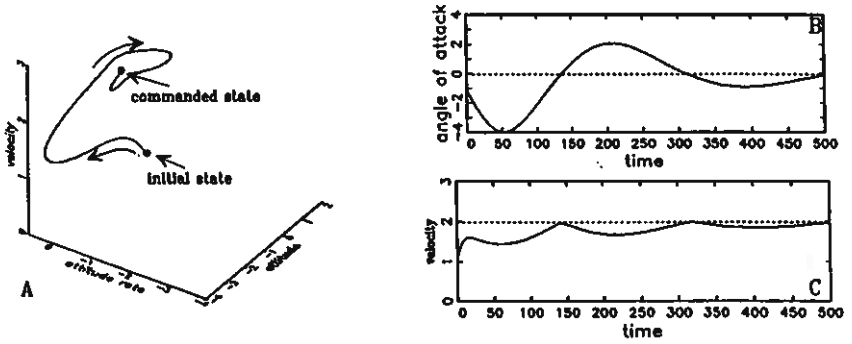

Fig. 6. Simulated AUV dive with no active nulling of angle of attack, θ. **A**: 3-d phase space trajectory, **B**: angle of attack, θ, time history, **C**: velocity, v, time history.

The simulation results shown in figures (7), (8), and (9) illustrate precisely how the ISID algorithm can be used to "relearn" the system's sliding modes. Figure (7) shows the AUV's performance (same initial conditions as shown in figure (6)) with the hierarchical sliding mode controller after a system failure causes the initially chosen switching surfaces to no longer be invariant sets. As can be seen, the sliding controller is actually unstable with the system exhibiting large oscillations in θ. Figures (8) and (9) show the system's behaviour during two "learning" sessions with the ISID algorithm. A learning session involves starting the vehicle at the initial condition and then commanding it over to the desired state. The first learning session is shown in figure (8). This particular example exhibited four adjustments to the sliding surface. On the last adjustment, the sliding condition is satisfied and the system slides easily to the commanded state. Figure (9) shows the system's response during the second training session. In this case, it is clear that learning is complete. There are no readjustments of the sliding surface and the system wastes little effort in bringing the system to its commanded state.

Perhaps the most remarkable thing about this example is the apparent speed with which the sliding surface is learned. In these simulations, only 4 failed invariance tests were required before finding a sliding mode. This low number of failed tests was observed in other simulation runs where the system's initial conditions were randomly varied. When compared with existing methods for learning nonlinear controllers [Narendra 1990] [Barto 1983] [Jacobs 1991], this approach appears to be exceptionally fast.

M. D. Lemmon, J. A. Stiver and P. J. Antsaklis, "Event Identification and Intelligent Hybrid Control," Hybrid S ystem s , R L Grossman, A Nerode, A P Ravn, H Rischel Eds, pp 269-296, Lecture Notes in Computer Science, LNCS 736, Springer-Verlag, 1993.

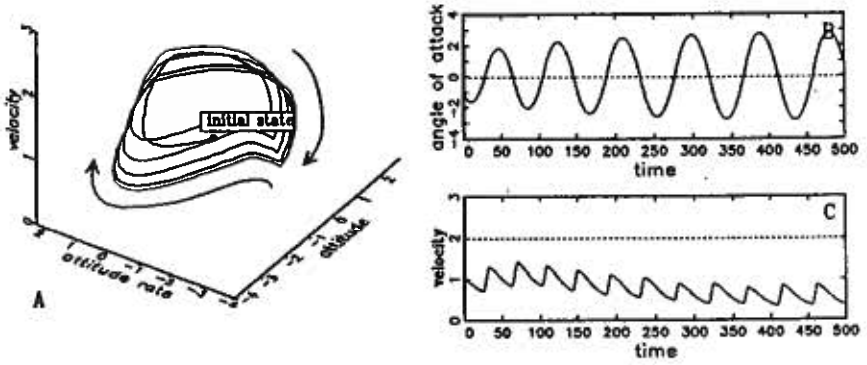

Fig. 7. Simulated AUV dive with hierarchical sliding control in which sliding mode constraints are violated. **A**: 3-d phase space trajectory, **B**: angle of attack, θ, time history, **C**: velocity, v, time history.

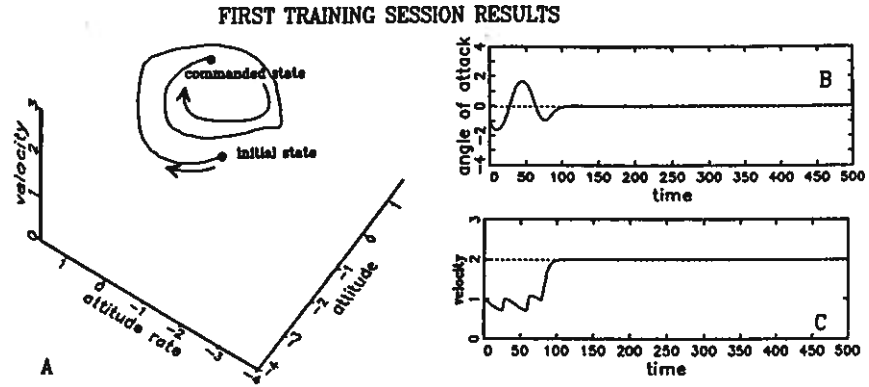

Fig. 8. Simulated AUV dive where ISID algorithm is used to relearn hierarchical sliding mode controller (First Learning Session). **A**: 3-d phase space trajectory, **B**: angle of attack, θ, time history, **C**: velocity, v, time history.

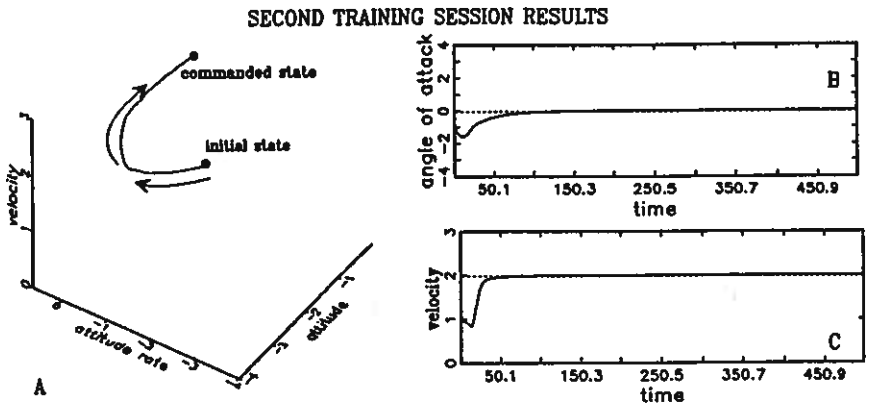

Fig. 9. Simulated AUV dive where ISID algorithm is used to relearn hierarchical sliding mode controller (Second Learning Session). **A**: 3-d phase space trajectory, **B**: angle of attack, θ, time history, **C**: velocity, v, time history.

8 Significant Issues

The final theorem of section (6) is significant for two reasons. First it shows that the invariant subspaces can be located after a *finite* number of failed queries. In sliding mode control, such subspaces are used to stabilize the system as was shown in the preceding example. Therefore, theorem (15) says that a system only needs to perceive itself as "unstable" a finite number of times before system stability is re-established. This result stands in stark contrast to other results [Barto 1983] [Narendra 1990] [Jacobs 1991] where system stability can only be iteratively "learned" after a prohibitively long training period. The second important aspect of the preceding results is that the theorem's bound implies that the algorithm has polynomial time complexity. This means that as systems become more and more complex (i.e. larger state spaces), the time required to learn the system invariants will grow at a modest rate. In other words, the proposed ISID algorithm may represent a practical method for adaptive control and identification of highly complex nonlinear dynamical systems.

It should be noted, however, that these bounds are *not* with respect to system time, but rather with respect to failed oracle time. This is an important distinction for it is quite possible that there may be a long period of time between consecutive oracle declarations of failure. Consequently, convergence of the ISID algorithm can be extremely long in "system" time and may, in fact, never converge at all. At first glance, this observation may seem to cast doubt upon the

M. D. Lemmon, J. A. Stiver and P. J. Antsaklis, "Event Identification and Intelligent Hybrid Control," Hybrid S ystem s , R L Grossman, A Nerode, A P Ravn, H Rischel Eds, pp 269-296, Lecture Notes in Computer Science, LNCS 736, Springer-Verlag, 1993.

value of theorem (15). Upon closer consideration, however, it provides further insight into the method. Recall that the oracle will always declare failures if the system trajectory is diverging from the current subspace, H_S. In other words, if the system is exhibiting "unstable" behaviour, the switching surface is modified. For the times between failures, the system appears to be stable and there is, therefore, no reason to change the switching surfaces. From this viewpoint, it can be seen that the bound of theorem (15) is very meaningful since it is measured with respect to the only quantity of physical interest to the system; the number of times the system "stumbles".

This point should be contrasted to parameter and set-membership identification [Dasgupta 1987] [Deller 1989] algorithms. In these cases, the important measure of parameter convergence is system time (i.e., the total number of experiments), since we are interested in obtaining accurate estimates as quickly as possible. Obviously for the parameter identification problem, the bounds computed by the preceding theorem would be useless unless the time between consecutive failures could be bounded also. That is not the situation, however, in the ISID problem which is primarily an adaptive "control" problem. The fact that these oracle-time bounds are meaningful for the ISID problem is an important point distinguishing this application from other more traditional applications of inductive inference protocols.

Finally, it must be observed that the preceding theorem assumes a perfect invariance oracle. In practice, oracles will not be perfect and the question is then what can be done to minimize the problems generated by an imperfect oracle. The answer is also provided by the preceding theorem. Theorem (15) provides a hard bound on the number of failed oracle queries. If the system generates more failures than implied by the bound, then a failure must either have occured in the oracle or else in the system itself. In either case, the theorem's finite time bound provides a natural criterion for failure detection and the subsequent reinitialization of the identification process. If the rate of oracle failure is known to be small (i.e. failure probability is small), then the natural course of action is to reinitialize the ISID algorithm and try again. The preceding discussion therefore implies the existence of effective and practical methods for dealing with the identification failures caused by imperfect oracles. In particular, if we model an oracle's imperfection as a probabilistic failure rate, then it should be possible to discuss the ISID algorithm's learning abilities within the so-called "probably almost correct" (PAC) framework used by a variety of researchers in the inductive inference community [Valiant 1984]. A full study of techniques for optimally managing the failures introduced by an imperfect oracle is well beyond the scope of the current chapter and represents an important topic for further inquiry.

9 Symbol Grounding and Event Identification

Formal computational systems are often interfaced to the external world. Such "hybrid" systems are used to control or interpret events with that external world.

The example discussed in this chapter is one example of such a hybrid system. Since the supervisor uses high-level abstractions to control the plant, such controllers are often referred to as "intelligent".

As noted in the section's introduction, this notion of intelligence is somewhat limited. If high-level decision making is to constitute intelligence, then this would imply that many symbol systems would be intelligent systems. This notion is, of course, at the heart of symbolic Artificial Intelligence research and it has its detractors. John Searle [Searle 1984] disputed the AI notion of intelligent machines with his now famous Chinese room argument. In this thought experiment it was noted that a prerequisite for "intelligence" is semantic content and that such content is unavailable to a purely symbolic system. For this reason, a computer can never be intelligent thereby debunking the traditional AI assumptions concerning the computational basis of human cognition.

At the heart of Searle's complaint is the notion of a symbol's meaning. This problem is also referred to as the symbol grounding problem [Harnad 1990]. Symbol grounding refers to methods by which symbols of a formal system acquire semantic content or "meaning". Such meaning generally has two interpretations. Meaning can be acquired through the association of symbols with nonsymbolic items in the external world. This acquired meaning is referred to as a symbol's extrinsic meaning. However, symbols also acquire content through internal associations with other symbols. This form of content might be referred to as intrinsic meaning.

An example of extrinsic meaning is seen in the association of the symbolic token "ELEPHANT" with those sensory inputs produced by seeing an elephant. The association of these sensory experiences with the symbolic token is determined by experience. The "meaning" of the symbol is determined by its nonsymbolic associations and is therefore external to the symbol system, itself. Consequently, we refer to such meaning as "extrinsic".

A symbol system, as Searle asserts, is not sufficient for an intelligent machine. Extrinsic meaning simply replaces the symbolic token with nonsymbolic tokens. The system has no real "understanding" of what those tokens signify so that the resulting computation is "meaningless". A good example of this type of "unintelligent" association is seen in intelligent control systems which make extensive use of DES models. In these cases, the "meaning" of the logical constructs is determined in an a priori way by the DES modeler. These intelligent choices for symbol/event bindings therefore imply that it is the modeler, rather than the system, which is intelligent.

In order for a system to be intelligent it must not only use high level abstractions, it must be able to generate them from internally generated construction principles. Symbols which arise in this manner may be grounded in nonsymbolic entities, but the meaning of these entities is determined internally, i.e. with respect to some intrinsic systems principle. In this regard, the symbols of such a system are intrinsically grounded. It is this form of "intrinsic" semantically meaning, which Searle asserted as a prerequisite for intelligence.

Clearly, conventional symbolic AI does not intrinsically ground its symbols.

M. D. Lemmon, J. A. Stiver and P. J. Antsaklis, "Event Identification and Intelligent Hybrid Control," Hybrid S ystem s , R L Grossman, A Nerode, A P Ravn, H Rischel Eds, pp 269-296, Lecture Notes in Computer Science, LNCS 736, Springer-Verlag, 1993.

It has been argued that the more recent connectionist or subsymbolic AI concepts embody some form of internal grounding[Chalmers 1992]. In view of these preceding remarks concerning symbol grounding and intelligence, it might now be appropriate to discuss the preceding ISID algorithm in light of the symbol grounding problem. Does the ISID algorithm produce event/symbol bindings which are "intrinsically" or "extrinsically" grounded. If the bindings are wholly external, then the resulting control system cannot be "intelligent" in the sense proposed by Searle.

In reviewing the modeling framework used in this paper, it is apparent that all plant symbols, $\tilde{z}$, are grounded with respect to a specific subset of the state space. At first glance, one might conclude then that this is an external grounding. However, the true test of external grounding is to see what happens if the event/symbol bindings change. In other words, if we shuffle the associations between symbols and nonsymbolic entities, does the operation of the supervisor change? If the ISID algorithm is not used, then clearly the bindings are unchanged. However, the ISID algorithm uses a computational algorithm (i.e. the invariance oracle) to decide whether or not the current event/symbol bindings satisfy or are consistent with the "internal" principle of control invariance. Therefore, if the initial event/symbol bindings change so that the resulting symbol groundings are inconsistent with the invariance oracle, then the supervisor changes the symbol bindings by redefining the "events". In other words, there is an internally coded principle guiding the symbol grounding process. Under this viewpoint, we can then assert that the ISID algorithm produces symbols with intrinsic semantic content.

The intrinsic content embodied by the invariance oracle is, of course, hardwired into the system. The choice of what this oracle is, represents a choice by the system designer. There can be other oracle structures used, in which different internal event principles are used. Therefore, in some sense, it is still through the ingenuity of the designer that this system appears to have "intelligent" processing. However, the fact remains that this system is endowing its symbols with a meaning which is derived by the system internally. This fact is still true regardless of where that "internally" coded principle came from. In this regard, we could consider the use of the ISID algorithm as resulting in an "intelligent" control system in the sense proposed by J. Searle.

These notions also provide some additional perspective on what is "intelligent" control. Intelligence is often a vaguely defined concept referring to the use of high-level decision making processes in control. In the preceding section, it has been argued that this is not sufficient. Intelligence is not a behaviour, but a property of a system. For intelligence, a system must not only use symbolic abstractions, it must formulate its own symbol bindings with regard to specific internal principles. The ISID algorithm provides a way by which traditional hybrid systems might accomplish this intelligence through event identification.

10 Concluding Remarks

This chapter has shown how hybrid dynamical systems can be used to dichotomize the symbolic and nonsymbolic parts of a supervisory control system. The significance of this dichotomy is that it clearly identifies one of the key challenges facing hybrid supervisory control. This challenge concerns the way in which symbols used by the supervisor are associated with *meaningful* events occuring in the plant. The problem of relating symbols and events has been called *event identification*, the associations are referred to as *symbol/event bindings*. This chapter has presented a method for *learning* event bindings in a way which insures the stabilizability of the plant. This method represents a novel approach to adaptive control in which *inductive inference of controller structure is emphasized over statistical inference of controller parameters*. In this regard, the proposed method provides a radical departure from conventional model reference adaptive control. The advantage of this new approach is that is allows the formulation of learning algorithms which converge to a stabilizing set of bindings after a finite number of updates. Another significant aspect is that this convergence time is bounded in a polynomial manner by plant complexity, thereby recommending this approach as a practical method for the adaptive stabilization of large scale plants. Finally, this new learning algorithm sheds light on the meaning of *intelligent control*. Specifically, the notions developed in this chapter allow the development of a working characterization of intelligent control which is consistent with current viewpoints held by the cognitive psychology and subsymbolic AI communities concerning the computational basis of human cognition.

References

[Angluin 1983] D. Angluin, C.H. Smith, "Inductive Inference: Theory and Methods." *Computing Surveys*, 15(3):237-269, September 1983.

[Antsaklis 1993] P.J. Antsaklis, J.A. Stiver, M.D. Lemmon, "Hybrid System Modeling", this volume.

[Barto 1983] A.G. Barto, R. S. Sutton, and C. W. Anderson), "Neuronlike Elements that can Solve Difficult Learning Control Problems", *IEEE Trans. on Systems, Man, and Cybernetics*, Vol 13:835-846.

[Bland 1981] R.G. Bland, D. Goldfarb, M.J. Todd, "The Ellipsoid Method: a Survey", *Operations Research*, 29:1039-1091, 1981.

[Chalmers 1992] D. J. Chalmers, "Subsymbolic Computation and the Chinese Room", in *The Symbolic and Connectionist Paradigms: closing the gap*, ed: John Dinsmore, Lawrence Erlbaum Associates, pp. 25-48, 1992.

[Dasgupta 1987] Dasgupta and Huang, "Asymptotically Convergent Modified Recrusive Least-Squares with Data-Dependent Updating and Forgetting Factor for Systems with Bounded Noise", *IEEE Trans. on Information Theory*, IT-33(3):383-392.

[DeCarlo 1988] R. A. DeCarlo, S. H. Zak, and GP Matthews (1988), "Variable Structure Control of Nonlinear Multivariable Systems: a Tutorial", *Proceedings of the IEEE*, Vol. 76(3):212-232.

M. D. Lemmon, J. A. Stiver and P. J. Antsaklis, "Event Identification and Intelligent Hybrid Control," Hybrid S ystem s , R L Grossman, A Nerode, A P Ravn, H Rischel Eds, pp 269-296, Lecture Notes in Computer Science, LNCS 736, Springer-Verlag, 1993.

[Deller 1989] J. R. Deller, "Set Membership Identification in Digital Signal Processing", *IEEE ASSP Magazine*, Vol. 6:4-20, 1989.

[Golub 1983] G. Golub, C. Van Loan, *Matrix Computation*, Johns Hopkins University Press, Baltimore, Maryland, 1983

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

[Harnad 1990] S. Harnad, "The Symbol Grounding Problem", *Physica D*, vol. 42, pp. 335-446, 1990.

[Jacobs 1991] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive Mixtures of Local Experts", *Neural Computation*, Vol 3(1):79-87.

[John 1984] John, *Fritz John: Collected Papers (1948)*, Birkhauser, 1984.

[Khachiyan 1979] L. G. Khachiyan, "A Polynomial Algorithm in Learn Program", (english translation), *Soviet Mathematics Doklady*, 20:191-194, 1979.

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

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

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

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

[Olver 1986] P. J. Olver (1986), *Applications of Lie Groups to Differential Equations*, Springer-Verlag, New York, 1986

[Rosenblatt 1962] F. Rosenblatt), *Principles of Neurodynamics*, Spartan books, Washington D.C.

[Searle 1984] J. R. Searle, *Minds, brains, and science*, Cambridge, MA: Harvard University Press, 1984.

[Shor 1977] N. Z. Shor, "Cut-off Method with Space Extension in Convex Programming Problems", (english translation), *Cybernetics*, 13:94-96, 1977.

[Utkin 1977] V.I. Utkin, "Variable Structure Systems with Sliding Modes", *IEEE Transactions on Automatic Control*, Vol. AC-22:212-222.

[Valiant 1984] L. Valiant, "A Theory of the Learnable", *Comm. of ACM*, Vol 27(11):1134-1142

[Yoerger 1985] D. R. Yoerger and J. J. E. Slotine, "Robust Trajectory Control of Underwater Vehicles", *IEEE Journal of Oceanic Engineering*, Vol OE-10(4):462-470.