AN INTRODUCTION TO HYBRID SYSTEM MODELING, ANALYSIS, AND CONTROL

JOHN LYGEROS, GEORGE PAPPAS, AND SHANKAR SASTRY

Abstract. Hybrid systems combine discrete event and continuous time dynamics and can serve as models of large scale systems. We provide an overview of modeling, analysis, and controller synthesis techniques for hybrid systems. A modeling language for describing a wide class of hybrid phenomena is first presented, and conditions for existence and uniqueness of executions are provided. We then discuss analysis problems, and characterize a class of hybrid systems for which reachability questions can be verified automatically. Finally, we review a method for designing controllers for hybrid systems with reachability specifications.

1. Introduction

The practice of controlling continuous physical processes by algorithms residing on digital computers has resulted in the paradigm of *hybrid systems*. Hybrid systems involve the interaction of continuous and discrete dynamics in a manner that is expressive enough to model many important applications such as automated highway systems [1, 2], air traffic management systems [3, 4], uninhabited aerial vehicles [5], and manufacturing [6].

In the past few years, researchers from both control theory and theoretical computer science have been developing techniques for the analysis and control of hybrid systems. In this paper, we present an introduction to some fundamental theoretical results that have come out of this major research effort in a consistent and unified framework. We address three main topics: modeling, analysis and control. We first present a formalism for modeling hybrid systems, define a hybrid system execution, and highlight conditions that guarantee existence and uniqueness of executions. We then turn to analysis, and in particular to verification of safety properties for hybrid systems, and try to establish conditions that guarantee that a hybrid system will not wander into an unsafe region of the state space. We present algorithmic approaches for computing the set of states reachable by a hybrid system, based on decidability results for a class of hybrid systems with linear continuous dynamics. Finally, we turn our attention to hybrid systems with inputs, and present a methodology for synthesizing controllers that guarantee by design a desired safety specifications.

Since the goal of this paper is to provide an introduction to the theory of hybrid systems, some of the results are not presented in detail and all the proofs are omitted. Even though our overview follows a certain perspective, many other approaches and issues are highlighted. We provide an extensive list of references, that the interested reader can use as starting point for a deeper study.

2. Modeling

We first address the issue of modeling, and more specifically, the question of existence and uniqueness of solutions. The following discussion is based on [7, 8, 9].

2.1. Notation and Terminology.

Let W be a countable collection of variables and let $\mathbf{W}$ denote its set of valuations, that is the set of all possible assignments of the variables in W. We refer to variables whose set of valuations is countable as *discrete* and to variables whose set of valuations is a subset of a Euclidean space $\mathbb{R}^n$ as *continuous*. We assume that Euclidean space is given the Euclidean metric topology, whereas countable and finite sets are given the discrete topology (all subsets are open). Subsets of a topological space are given the subset topology and products of topological spaces are given the product topology. For a subset U of a topological space we use $\overline{U}$ to denote its closure, U^o its interior, ∂U its boundary, U^c its complement, $|U|$ its cardinality, and 2^U the set of all subsets of U. We use $\wedge$ to denote conjunction, $\vee$ to denote disjunction, $\neg$ to denote negation, $\forall$ to denote the universal quantifier, and $\exists$ to denote the existential quantifier.

As we are interested in hybrid phenomena, that involve both continuous and discrete dynamics, we introduce the notion of a hybrid time trajectory, which will encode the set of times over which the evolution of the system is defined.

Definition 1 (Hybrid Time Trajectory). *A hybrid time trajectory $\tau = \{I_i\}_{i=0}^N$ is a finite or infinite sequence of intervals of the real line, such that*

- $I_i = [\tau_i, \tau_i']$ *for* $i < N$ *and, if* $N < \infty$, $I_N = [\tau_N, \tau_N']$ *or* $I_N = [\tau_N, \tau_N')$;
- *for all* i, $\tau_i \leq \tau_i' = \tau_{i+1}$.

The interpretation is that τ_i are the times at which discrete transitions take place. Notice that discrete transitions are assumed to be instantaneous and that multiple discrete transitions may take place at the same time, since it is possible for $\tau_i = \tau_{i+1}$. Hybrid time trajectories can extend to "infinity" if τ is an infinite sequence or if it is a finite sequence ending with an interval of the form $[\tau_N, \infty)$. Since the dynamical systems we will consider will be time invariant, we assume, without loss of generality, that $\tau_0 = 0$. We denote by $\mathcal{T}$ the set of all hybrid time trajectories. Each $\tau \in \mathcal{T}$ is fully ordered by the relation $\prec$, which for $t \in [\tau_i, \tau_i'] \in \tau$ and $t' \in [\tau_j, \tau_j'] \in \tau$ is defined by $t \prec t'$ if and only if $t < t'$ or $i < j$. For $t \in \mathbb{R}$ and $\tau \in \mathcal{T}$ we use $t \in \tau$ as a shorthand notation for "there exists a j such that $t \in [\tau_j, \tau_j'] \in \tau$". For a topological space $\mathbf{W}$ and $\tau \in \mathcal{T}$, we use $w : \tau \to \mathbf{W}$ as a shorthand notation for a map assigning values from $\mathbf{W}$ to all elements of all intervals of τ. For a collection of variables W, we denote by $Hyb(W)$ the set of *hybrid trajectories* of W, defined as $Hyb(W) = \{(\tau, w) \mid \tau \in \mathcal{T} \text{ and } w : \tau \to \mathbf{W}\}$. We say $\chi = \left(\{I_i\}_{i=0}^N, w\right) \in Hyb(W)$ is a *prefix* of $\chi' = \left(\{I_i'\}_{i=0}^{N'}, w'\right) \in Hyb(W)$ (write $\chi \leq \chi'$) if τ is finite, $N \leq N'$, $I_i = I_i'$ for all $i = 0, \dots, N-1$, $I_N \subseteq I_N'$, and $w(t) = w'(t)$ for all $t \in \tau$. We say χ is a strict prefix of χ' (write $\chi < \chi'$) if $\chi \leq \chi'$ and $\chi \neq \chi'$. The prefix relation defines a partial order on $Hyb(W)$.

2.2. **Basic Definitions.** A hybrid automaton provides a formal way for restricting the set of hybrid trajectories of a collection of discrete and continuous variables.

Definition 2 (Hybrid Automaton). *A hybrid automaton H is a collection $H = (X, V, \text{Init}, f, \text{Inv}, R)$, where*

- X *is a finite collection of* state variables. *We assume* $X = X_D \cup X_C$, *with* $\mathbf{X}_D$ *countable and* $\mathbf{X}_C = \mathbb{R}^n$;
- V *is a finite collection of* input variables. *We assume* $V = V_D \cup V_C$, *with* $\mathbf{V}_D$ *countable and* $\mathbf{V}_C \subseteq \mathbb{R}^m$;
- $\text{Init} \subseteq \mathbf{X}$ *is a set of* initial states;
- $f : \mathbf{X} \times \mathbf{V} \to \mathbf{X_C}$ *is a* vector field, *assumed to be globally Lipschitz in* X_C *and continuous in* V;
- $\text{Inv} \subseteq \mathbf{X} \times \mathbf{V}$ *is an* invariant set;
- $R : \mathbf{X} \times \mathbf{V} \to 2^{\mathbf{X}}$ *is a* reset relation.

We refer to $x \in \mathbf{X}$ as the *state* of H and to $v \in \mathbf{V}$ as the *input* of H. Where necessary, we will use $x = (q, y)$ with $q \in \mathbf{X}_D$ and $y \in \mathbf{X}_C$ to distinguish the continuous from the discrete state. Traditionally, hybrid automata are represented by directed graphs, (Q, E), with vertices $Q = \mathbf{X_D}$ and edges

$$E = \{(q, q') \in Q \times Q \mid (q', y') \in R((q, y), v) \text{ for some } y, y' \in \mathbf{X}_C, v \in \mathbf{V}\}$$

With each vertex $q \in Q$, we associate a set of initial states $Init_q = \{y \in \mathbf{X}_C \mid (q, y) \in Init\}$, a vector field $f_q(y, v) = f((q, y), v)$ and an invariant set $Inv_q = \{(y, v) \in \mathbf{X}_C \times \mathbf{V} \mid ((q, y), v) \in Inv\}$. With each edge $(q, q') \in E$, we associate a guard $Guard_{(q,q')} = \{(y, v) \in \mathbf{X}_C \times \mathbf{V} \mid (q', y') \in R((q, y), v) \text{ for some } y' \in \mathbf{X}_C\}$ and a reset relation $Reset_{(q,q')}(y, v) = \{y' \in \mathbf{X}_C \mid (q', y') \in R((q, y), v)\}$.

Hybrid automata characterize hybrid phenomena in the following sense.

Definition 3 (Execution). *An execution of a hybrid automaton H is a hybrid trajectory, $\chi = (\tau, (q, y), v) \in$ Hyb$(X \cup V)$, with*

- initial condition: $(q(\tau_0), y(\tau_0)) \in \text{Init}$;

- continuous evolution*: for all i with $\tau_i < \tau_i'$, $q(\cdot)$ is constant, $v(\cdot)$ is piecewise continuous and $y(\cdot)$ is a solution*[1] *to the differential equation* $dy/dt = f\big((q,y),v\big)$ *over* $[\tau_i, \tau_i']$, *and for all* $t \in [\tau_i, \tau_i')$, $((q(t),y(t)),v(t)) \in \mathrm{Inv}$;
- discrete evolution*: for all* i, $(q(\tau_{i+1}),y(\tau_{i+1})) \in R((q(\tau_i'),y(\tau_i')),v(\tau_i'))$.

For an execution $\chi = (\tau, x, v)$ we use $x_0 = x(\tau_0)$ to denote the initial state of χ.

Unlike conventional continuous dynamical systems, the interpretation is that a hybrid automaton *accepts* (as opposed to generates) an execution. This difference in perspective allows one to consider hybrid automata that accept no executions for some initial states, accept multiple executions for the same initial state, or do not accept executions over arbitrarily long time horizons. To formalize these notions we introduce the following classification.

Definition 4 (Types of Execution)**.** *An execution* $\chi = (\tau, x, v)$ *is called*

- *finite, if* τ *is a finite sequence ending with a closed interval;*
- *infinite, if* τ *is an infinite sequence, or if* $\sum_{i=0}^{\infty}(\tau_i' - \tau_i) = \infty$;
- *Zeno, if it is infinite but* $\sum_{i=0}^{\infty}(\tau_i' - \tau_i) < \infty$; *and*
- *maximal, if it is not a strict prefix of any other execution of* H.

For an infinite execution we define the *Zeno time* as $\tau_\infty = \sum_{i=0}^{\infty}(\tau_i' - \tau_i)$. Clearly, $\tau_\infty < \infty$ if and only if the execution is Zeno. For $x_0 \in \mathit{Init}$, we use $\mathcal{E}_H(x_0)$ to denote the set of all executions of H with initial condition $x_0 \in \mathbf{X}$, $\mathcal{E}_H^*(x_0)$ to denote the set of all finite executions, $\mathcal{E}_H^M(x_0)$ the set of all maximal executions, and $\mathcal{E}_H^\infty(x_0)$ the set of all infinite executions. It is easy to show that $\mathcal{E}_H(x_0)$ is prefix closed, that is if $\chi \in \mathcal{E}_H(x_0)$, $\chi' \in \mathcal{E}_H(x_0)$ for all $\chi' \leq \chi$. Moreover, $\mathcal{E}_H^\infty(x_0) \subseteq \mathcal{E}_H^M(x_0)$, since an infinite execution can not be a strict prefix of any other execution. We use $\mathcal{E}_H = \cup_{x_0 \in \mathit{Init}} \mathcal{E}_H(x_0)$ to denote the set of all executions of H, and $\mathcal{E}_H|_X$ to denote the projection of $\mathcal{E}_H$ to the state variables (and likewise for finite executions, infinite executions, etc.).

2.3. **Existence and Uniqueness Conditions.** In this section we restrict our attention to *autonomous hybrid automata* (hybrid automata without input variables) and derive conditions under which all executions can be extended to infinite executions and conditions under which this extension can be done uniquely.

Definition 5 (Non-Blocking and Deterministic Automata)**.** *A hybrid automaton* H *is called non-blocking if* $\mathcal{E}_H^\infty(x_0)$ *is non-empty for all* $x_0 \in \mathrm{Init}$. *It is called deterministic if* $\mathcal{E}_H^M(x_0)$ *contains at most one element for all* $x_0 \in \mathrm{Init}$.

To state local existence and uniqueness conditions we need to characterize the set of states that can be reached by H, as well as the set of states from which continuous evolution is possible.

Definition 6 (Reachable State)**.** *A state* $\hat{x} \in \mathbf{X}$ *is called reachable by* H *if there exists a finite execution* $\chi = (\tau, x)$ *with* $\tau = \{[\tau_i, \tau_i']\}_{i=0}^N$ *and* $x(\tau_N') = \hat{x}$.

We use $\mathit{Reach}_H \subseteq \mathbf{X}$ to denote the set of states reachable by H.

Next, for $(\hat{q}, \hat{y}) \in \mathbf{X}$ and some $\epsilon > 0$ small enough[2], consider the solution, $y(\cdot) : [0, \epsilon) \to \mathbf{X}$ to the ordinary differential equation $dy/dt = f(\hat{q}, y)$ with $y(0) = \hat{y}$. To characterize the states from which continuous evolution is impossible, we define the set *Out* $\subseteq \mathbf{X}$ by

$$\mathit{Out} = \{(\hat{q}, \hat{y}) \in \mathbf{X} \mid \forall \epsilon > 0\ \exists t \in [0, \epsilon) \text{ such that } y(t) \notin \mathit{Inv}(\hat{q})\},$$

Note that $\mathit{Inv}^c \subseteq \mathit{Out}$. If *Inv* is a closed set, *Out* may also contain pieces of the boundary of *Inv*.

Using the notions of *Reach* and *Out* one can show the following [9].

Lemma 1. *A hybrid automaton* H *is non-blocking if for all* $x \in \mathrm{Out} \cap \mathrm{Reach}_H$, $R(x) \neq \emptyset$. *A deterministic hybrid automaton is non-blocking if and only if for all* $x \in \mathrm{Out} \cap \mathrm{Reach}_H$, $R(x) \neq \emptyset$.

[1] The solution is considered in the sense of Caratheodory, and is defined over $[\tau_i, \tau_i')$, if τ ends in a right open interval and $i = N$.

[2] Such ϵ exists by the Lipschitz assumption on f in Definition 2.

Lemma 2. *A hybrid automaton is deterministic if and only if for all* $x \in \mathrm{Reach}_H$, $|R(x)| \leq 1$ *and if* $R(x) \neq \emptyset$ *then* $x \in \mathrm{Out}(q)$.

Combining Lemmas 1 and 2 leads to the following.

Theorem 1 (Existence and Uniqueness of Executions). *A hybrid automaton accepts a unique infinite execution for all* $x_0 \in \mathrm{Init}$ *if it satisfies the conditions of Lemmas 1 and 2.*

The conditions of the theorem are sufficient and tight. The lemmas require one to compute the sets *Out* and *Reach*. In [9] it is shown how *Out* can be computed using standard non-linear control tools, if f and the set *Inv* are smooth enough. For non-smooth sets, more powerful results from viability theory [10] can be used to derive existence conditions [11]. Notice that the conditions of the theorem do not require one to compute $Reach_H$ explicitly; it suffices to show that the conditions of the lemmas hold for a set of states that contains $Reach_H$. In [9, 12] methods for systematically computing conservative approximations of $Reach_H$ using deductive techniques are presented.

Finally, note that the theorem provides *local* existence conditions. We still need to determine whether the executions can be extended over arbitrarily long time horizons. The Lipschitz assumption on the vector field f excludes the possibility of finite escape time due to continuous evolution. There is still, however, the possibility of Zeno executions, i.e., executions that take an infinite number of transitions in a finite amount of time, thus preventing time from diverging. This class of executions is studied in more detail in [8, 13], where properties of Zeno executions are established and methods for extending them beyond the (finite) Zeno time are proposed.

2.4. **A Note on Simulation, Modularity and Generality.** Some hybrid modeling formalisms are currently supported by simulation tools. These include SHIFT [14] and OMOLA/OMSIM [15]; for an overview see [16]. Existence of executions for classes of hybrid systems has also been studied in [17, 18]. Besides existence of executions, another modeling issue of interest is modularity [19, 20]. One would like to provide formal operators for composing hybrid models, to be able to build larger systems out of modules and reason about the properties that the composed system inherits from its components. Finally, one would like to make the modeling formalism as general as possible, to be able to capture a wide class of hybrid phenomena[3]. The descriptive power of a number of hybrid models is compared in [21].

3. ANALYSIS

In this section, we turn our attention to verification, that is the question of whether the executions of a hybrid automaton satisfy certain desirable properties. For simplicity, we restrict our attention to autonomous hybrid automata and to a class of properties known as *safety properties* which, as we shall see, are intimately related to reachability. The discussion in this section follows the development in [22, 23, 24, 25].

3.1. **Safety Properties and Bisimulation.** A *property*, P, of a hybrid automaton H is a map:

$$P : \mathcal{H} \rightarrow \{\mathrm{True}, \mathrm{False}\} \tag{1}$$

We say a run $\chi \in \mathcal{E}_H$ *satisfies property* P if $P(\chi) = \mathrm{True}$; we say a hybrid automaton satisfies a property P if $P(\chi) = \mathrm{True}$ for all $\chi \in \mathcal{E}_H$.

Given a set $F \subseteq \mathbf{X}$ we define a *safety property*[4], denoted by $\Box F$, by:

$$\Box F(\chi) = \begin{cases} \mathrm{True} & \text{if } \forall t \in \tau, x(t) \in F \\ \mathrm{False} & \text{otherwise} \end{cases}$$

Clearly, H satisfies a safety property $\Box F$ if and only if $Reach_H \subseteq F$.

In this section we focus on autonomous hybrid systems and algorithmic approaches for computing $Reach_H$. The approach used is based on the notion of bisimulation of transition systems.

[3]The tradeoff of course is that the more general models are usually also more difficult to analyze

[4]Strictly speaking, the class of properties $\Box F$ considered here is a special case of what is known as safety properties in the theoretical computer science literature [26].

Definition 7 (Transition Systems). *A transition system $T = (Q, \Sigma, \rightarrow, Q_O, Q_F)$ consists of a set Q of states, an alphabet Σ of events, a transition relation $\rightarrow \subseteq Q \times \Sigma \times Q$, a set $Q_O \subseteq Q$ of initial states, and a set $Q_F \subseteq Q$ of final states.*

Transition systems can be thought of as abstract graph models with potentially infinite number of states. It is customary to denote a transition $(q_1, \sigma, q_2) \in \rightarrow$ as $q_1 \xrightarrow{\sigma} q_2$. The transition system is *finite* if the cardinality of Q is finite, and it is infinite otherwise. Verification of safety properties for transition systems can be cast in terms of the reachability problem.

Problem 1 (Reachability Problem). *Given a transition system T, is a state $q_f \in Q_F$ reachable from a state $q_o \in Q_O$ by a sequence of transitions?*

The interpretation is that the set of final states encodes an undesirable or unsafe region of the state space. In this context, a hybrid automaton H and safety property $\Box F$ generate a transition system T by setting $Q = \mathbf{X}$, $Q_O = Init$, $Q_F = F^c$, $\Sigma = \{\alpha, \beta\}$, and

- Discrete Transitions: $x \xrightarrow{\alpha} x'$ if and only if $x' \in R(x)$
- Continuous Transitions: $(q, y) \xrightarrow{\beta} (q', y')$ if and only if $q = q'$ and there exists $\delta \geq 0$ and a solution $y : [0, \delta] \longrightarrow \mathbf{X_C}$ to the differential equation $dy/dt = f(q, y)$ with $y(0) = y$, $y(\delta) = y'$ and for all $t \in [0, \delta)$, $(q, y(t)) \in Inv$.

Notice that the continuous transitions, β, are time-abstract transitions, in the sense that the time it takes to reach one state from another is ignored.

A *region* is a subset $P \subseteq Q$. Given $\sigma \in \Sigma$ we define $Pre_\sigma(P)$ of a region P as

$$Pre_\sigma(P) = \{q \in Q \mid \exists p \in P \text{ and } q \xrightarrow{\sigma} p\} \tag{2}$$

Thus $Pre_\sigma(P)$ is the set of states that can reach P with a single σ event. The set of states that can reach P in one step for all σ events is

$$Pre(P) = \{q \in Q \mid \exists \sigma \in \Sigma \; \exists p \in P \text{ and } q \xrightarrow{\sigma} p\} \tag{3}$$

The set of states that can reach P in two steps is simply $Pre(Pre(P))$, and is denoted $Pre^2(P)$. In general, $Pre^i(P)$ consists of states that can reach P in i steps, and

$$Pre^*(P) = \bigcup_{i \in \mathbb{N}} Pre^i(P) \tag{4}$$

is the set of states that are *backward reachable* from P. In this notation, the reachability problem amounts to checking that $Pre^*(Q_F) \cap Q_O \neq \emptyset$. It can be tackled using the following algorithm[5].

Algorithm 1 (Backward Reachability Algorithm)

> **initialization**: $R := Q_F$
> **while** *true* **do**
>   **if** $R \cap Q_O \neq \emptyset$ **return unsafe ; STOP**
>   **if** $Pre(R) \subseteq R$ **return safe ; STOP**
>   **else** $R := R \cup Pre(R)$
> **end while**

If the state space of the transition system is finite, then the algorithm can be implemented and is guaranteed to terminate. If the state space is infinite, however, then there is no guarantee that the algorithm will terminate. Our goal is to find classes of infinite state transition systems for which we can compute the reachable space in a finite number of steps. This is accomplished by reducing the infinite state transition system to a finite quotient system with equivalent reachability properties. To achieve this, the notion of a quotient transition system needs is introduced.

---

[5]In a similar manner, a *forward* reachability algorithm can be constructed

Given an equivalence relation $\sim \subseteq Q \times Q$ on the state space, and a region P, we denote by $P/\sim$ the collection of all equivalence classes which intersect P. Given an equivalence relation $\sim$ on Q, we call a set a $\sim$-block if it is a union of equivalence classes. The *quotient transition system* is defined by $T/\sim = (Q/\sim, \Sigma, \rightarrow_\sim, Q_0/\sim, Q_F/\sim)$ where $Q/\sim$ is the quotient space and for $Q_1,\ Q_2 \in Q/\sim$, $Q_1 \xrightarrow{\sigma}_\sim Q_2$ if and only if there exist $q_1 \in Q_1$ and $q_2 \in Q_2$ such that $q_1 \xrightarrow{\sigma} q_2$.

Definition 8 (Bisimulation)**.** *Let $T = (Q, \Sigma, \rightarrow, Q_O, Q_F)$ be a transition system. The equivalence relation $\sim$ is a bisimulation of T if and only if*

- *Q_O, Q_F are $\sim$-blocks*
- *For all $\sigma \in \Sigma$ and all $\sim$-blocks P, $\mathrm{Pre}_\sigma(P)$ is a $\sim$-block.*

If $\sim$ is a bisimulation, then transition systems T and $T/\sim$ are called *bisimilar*. A bisimulation is called *finite* if the quotient space is finite.

The crucial property of bisimulations is that the intersection of $Pre_\sigma(P)$ and Q, for equivalence classes P, Q, and $\sigma \in \Sigma$, is either the empty set or all of Q. Alternatively, if $\sim$ is a bisimulation, it can be easily shown that if $p \sim q$ then

1. $p \in Q_F$ if and only if $q \in Q_F$, and $p \in Q_O$ if and only if $q \in Q_O$
2. if $p \xrightarrow{\sigma} p'$ then there exists q' such that $q \xrightarrow{\sigma} q'$ and $p' \sim q'$

The above characterization of bisimulations leads to the following theorem.

Theorem 2 (Reachability Equivalence)**.** *Let $T = (Q, \Sigma, \rightarrow, Q_O, Q_F)$ and $T/\sim = (Q/\sim, \Sigma, \rightarrow_\sim, Q_0/\sim, Q_F/\sim)$ be bisimilar transition systems. Then the reachability problems for T and $T/\sim$ are equivalent.*

If T has an infinite state space but we can find a bisimilar but finite transition system $T/\sim$, then checking reachability of T can be equivalently done on $T/\sim$ for which the reachability algorithm is guaranteed to terminate. Therefore, bisimulations are the main tool for obtaining classes of hybrid systems with a *decidable* reachability problem. This philosophy has successfully resulted in various classes of hybrid systems with decidable reachability problem, like timed automata [27] and initialized rectangular automata [28, 29].

A conceptual algorithm that computes a bisimulation of the state space starts with a given transition system T, and computes increasingly finer partitions of the state space Q. If the algorithm terminates, then the resulting quotient transition system is a finite bisimulation.

Algorithm 2 (Bisimulation Algorithm)

> **initialization**: $Q/\sim = \{Q_O, Q_F, Q \setminus (Q_O \cup Q_F)\}$
> **while** $\exists\ P, P' \in Q/\sim$ and $\sigma \in \Sigma$ such that $\emptyset \neq P \cap Pre_\sigma(P') \neq P$ **do**
> $P_1 = P \cap Pre_\sigma(P')$
> $P_2 = P \setminus Pre_\sigma(P')$
> $Q/\sim = (Q/\sim \setminus \{P\}) \cup \{P_1, P_2\}$
> **end while**

In addition to showing that the bisimulation algorithm terminates, decidability requires that each step of the algorithm is *computable*. This means that we must be able to represent sets symbolically, perform boolean operations, check emptiness of a set, and compute $Pre_\sigma(P)$ for any $\sigma \in \Sigma$.

3.2. **Order Minimal and Linear Hybrid Systems.** In this section we provide a brief introduction to mathematical logic and order minimal theories [30], and then use it to construct finite bisimulations of certain classes of hybrid systems.

A *language* is a set of symbols separated into three groups: relations, functions and constants. The sets $\mathcal{L}_0 = \{<, +, -, 0, 1\}$, $\mathcal{L}_R = \{<, +, -, \cdot, 0, 1\}$, and $\mathcal{L}_{\exp} = \{<, +, -, \cdot, 0, 1, \exp\}$ are examples of languages where $<$ (less than) is the relation, $+$ (plus), $-$ (minus), $\cdot$ (product) and exp (exponentiation) are the functions, and 0 (zero) and 1 (one) are the constants.

Consider again a countable collection of variables $W = \{x, y, z, x_0, x_1, \dots\}$. The set of *terms* of a language is inductively defined as follows. A term θ is a variable, a constant, or $F(\theta_1, \dots, \theta_m)$, where F is a m-ary function and θ_i, $i = 1, \dots, m$ are terms. For instance, $x - 2y + 3$ and $x + yz^2 - 1$ are terms of $\mathcal{L}_0$ and $\mathcal{L}_R$, respectively. In other words, terms of $\mathcal{L}_0$ are linear expressions and terms of $\mathcal{L}_R$ are polynomials with integer coefficients. Notice that integers are the only numbers allowed in expressions (they can be obtained by repeatedly adding the constant 1).

The *atomic formulas* of a language are of the form $\theta_1 = \theta_2$, or $R(\theta_1, \dots, \theta_n)$, where θ_i, $i = 1, \dots, n$ are terms and R is an n-ary relation. For example, $xy > 0$ and $x^2 + 1 = 0$ are atomic formulas of $\mathcal{L}_R$. The set of *(first-order) formulas* is recursively defined as follows. An atomic formula ϕ is a formula, and if ϕ_1 and ϕ_2 are formulas and x is a variable, then $\phi_1 \wedge \phi_2$, $\neg\phi_1$, $\forall x : \phi_1$ or $\exists x : \phi_1$ are formulas. Examples of $\mathcal{L}_R$-formulas are $\forall x\ \forall y : xy > 0$, $\exists x : x^2 - 2 = 0$, and $\exists w : xw^2 + yw + z = 0$.

The occurrence of a variable in a formula is *free* if it is not inside the scope of a quantifier; otherwise, it is *bound*. For example, x, y, and z are free and w is bound in the last example. We often write $\phi(x_1, \dots, x_n)$ to indicate that $x_1, \dots, x_n$ are the free variables of the formula ϕ. A *sentence* of $\mathcal{L}_R$ is a formula with no free variables. The first two examples above are sentences, whereas the third one is not.

A *model* of a language consists of a non-empty set S and an interpretation of the relations, functions and constants. For example, $(\mathbb{R}, <, +, -, \cdot, 0, 1)$ and $(\mathbb{Q}, <, +, -, \cdot, 0, 1)$, are *models* of $\mathcal{L}_R$ with the usual interpretation of the all the symbols.

A set $Y \subseteq S^n$ is *definable* in a language if there exists a formula $\phi(x_1, \dots, x_n)$ such that $Y = \{(a_1, \dots, a_n) \in S^n \mid \phi(a_1, \dots, a_n)\}$. For example, over $\mathbb{R}$, the formula $x^2 - 2 = 0$ defines the set $\{\sqrt{2}, -\sqrt{2}\}$. A *theory* is a subset of sentences. Any model of a language defines a theory: *the set of all sentences which hold in the model.* By abuse of notation the theory $(\mathbb{R}, <, +, -, \cdot, 0, 1)$ will refer to the collection of formulas of $\mathcal{L}_R$ which are true in the model (and similarly for other languages). Only models over $\mathbb{R}$ will be considered.

Definition 9 (Order Minimal Theories). *The theory of the reals is called o-minimal ("order minimal") if every definable subset of $\mathbb{R}$ is a finite union of points and intervals (possibly unbounded).*

Examples of o-minimal theories include

1. $\mathcal{R} = (\mathbb{R}, <, +, -, \cdot, 0, 1)$
2. $\mathcal{R}_{\sin} = (\mathbb{R}, <, +, -, \cdot, \sin|_{[-\pi,\pi]}, 0, 1)$
3. $\mathcal{R}_{\exp} = (\mathbb{R}, <, +, -, \cdot, \exp, 0, 1)$
4. $\mathcal{R}_{\sin,\exp} = (\mathbb{R}, <, +, -, \cdot, \sin|_{[-\pi,\pi]}, \exp, 0, 1)$

Based on this notion, a new class of hybrid systems is defined.

Definition 10 (Order Minimal Hybrid Automata). *A hybrid automaton H is said to be o-minimal if*

- $\mathbf{X_C} = \mathbb{R}^n$
- *for each $q \in \mathbf{X_D}$ the flow of f_q is complete (is defined for all time)*
- *for each $q \in \mathbf{X_D}$ all the relevant sets (guards, invariants, initial conditions) and the flow of f_q are definable in the same o-minimal theory of the reals.*
- *for each $(q, q') \in E$, the reset relation* $\mathrm{Reset}_{(q,q')}$ *is a constant set valued map.*

Notice that o-minimal hybrid systems have more restrictive discrete dynamics than timed [27] or rectangular automata [28, 29] but can have much more complicated continuous dynamics. The powerful structure of o-minimal theories, allows us to prove the following theorem [23].

Theorem 3. *Every o-minimal hybrid automaton admits a finite bisimulation. In particular, the bisimulation algorithm, terminates for o-minimal hybrid systems.*

The above result is existential and shows that a finite bisimulation exists for all o-minimal hybrid systems. To make use of this result we have to find classes of hybrid automata that are either o-minimal or that can be transformed into o-minimal hybrid automata, for which each step of the bisimulation algorithm is computable.

Theorem 4 (Linear Hybrid Automata)**.** *Consider an o-minimal hybrid automaton H where all relevant sets (initial set, guards, invariants) are semialgebraic*[6] *and for all $q \in \mathbf{X_D}$, $f_q(y) = A_q y$, where the matrix A_q has rational entries. If*

- *A_q is nilpotent; or,*
- *A_q is diagonalizable and has real, rational eigenvalues; or,*
- *A_q has purely imaginary eigenvalues $i\omega$, with ω rational, and its real Jordan form is block diagonal with 2×2 blocks,*

then the reachability problem for H is decidable.

The proof [24] consists of showing that each step of the bisimulation algorithm can be posed as a quantifier elimination problem in the theory $(\mathbb{R}, +, -, \cdot, <, 0, 1)$, which is known to be decidable [31].

Theorem 4 can be extended to include linear hybrid systems where in each discrete locations the dynamics are of the form $\dot{y} = Ay + Bu$ for various types of inputs u [25].

Theorem 5 (Linear Hybrid Automata with Inputs)**.** *Consider an o-minimal hybrid automaton H where all relevant sets are semialgebraic, and for all $q \in \mathbf{X_D}$ the dynamics are of the form $f_q(y) = A_q y + B_q u$ where the matrices A_q and B_q have rational entries. If*

- *A_q is nilpotent and each entry of u is a polynomial in t; or,*
- *A_q is diagonalizable, has real rational eigenvalues, and each entry of u is of the form $e^{\mu t}$ with μ rational and not an eigenvalue of A_q; or,*
- *A_q has purely imaginary eigenvalues of the form $i\omega$ with ω rational, and the entries in the input u are of the form $\sin(\alpha t)$ or $\cos(\alpha t)$ with α rational and $\alpha \neq \pm\omega$,*

then the reachability problem for the hybrid system H is decidable.

The above results remain valid if the inputs are allowed to be rational, linear combinations of the functions of the corresponding type: exponentials in case of real eigenvalues, and sinusoidal in the case of imaginary eigenvalues. In all cases the same *resonance* restrictions apply on the parameters μ and α.

3.3. **Other Analysis Techniques.** For certain classes of hybrid automata, verification of properties like the ones discussed here may be carried out automatically [27, 32, 29] using computational tools [33, 34, 35]. For more general classes, deductive techniques may be used [19, 12], which involve, for example, invariant assertions proved by induction arguments along the length of the system executions. Even though some of the computations may be carried out automatically using theorem provers [36], most of the burden with deductive proofs falls on the designer. In addition to the class of properties considered here, questions of stability have also been studied for hybrid systems, primarily by extending Lyapunov techniques [37, 38]. Computationally efficient methods using LMIs were developed in [39] for finding piecewise quadratic Lyapunov functions for piecewise linear hybrid dynamics.

4. Control

We now return to hybrid automata with inputs and consider the problem of using these inputs to steer the executions, so that the closed loop system satisfies a given safety property. The discussion in this section follows [7, 40, 41].

4.1. **Control for Reachability Specifications.** Assume that we are given a hybrid automaton H (which we refer to as the *plant*) and we are required to control it using its input variables, so that its executions satisfy a safety property. For the purpose of control the input variables of the plant are partitioned into two classes: *controls* and *disturbances*. We write $V = U \cup D$, where U and D are the sets of control and disturbance inputs respectively. The interpretation is that the controls can be determined using a controller, in an attempt to guide the system, whereas the disturbances are determined by the "environment" and may potentially disrupt

[6]Semialgebraic sets comprise the boolean algebra of sets generated by polynomial equalities and inequalities.

the controller's plans. To prevent u and/or d from "cheating" by blocking the system executions we assume that *Inv* is an open set and that if $(x,(u,d)) \notin \mathit{Inv}$ then $R(x,(u,d)) \neq \emptyset$.

We consider a *controller* for a plant H to be a map:

$$C : \mathcal{E}^*_H|_X \to 2^{\mathbf{U}}$$

For the hybrid automaton H and a controller C we define the set of *closed loop causal executions* by:

$$\mathcal{E}_{H_C} = \{(\tau, x, (u,d)) \in \mathcal{E}_H \mid u(t) \in C(x\downarrow_t) \text{ for all } t \in \tau\}$$

$x \downarrow_t$ is used a shorthand for the finite prefix of (τ, x) ending at $t \in \tau$. A controller is called *memoryless* if for any two finite executions $\chi_1 = (\tau_1, x_1, (u_1, d_1))$ and $\chi_2 = (\tau_2, x_2, (u_2, d_2))$ ending in the same state, $C(x_1) = C(x_2)$. A memoryless controller can be characterized by a feedback map:

$$g : \mathbf{X} \to 2^{\mathbf{U}}$$

An instance of the *controller synthesis problem* consists of a pair, (H, P), of a plant hybrid automaton and a property of that automaton. We sometimes refer to the property P as a *specification*. We say a controller, C, solves the controller synthesis problem if $P(\chi) = \text{True}$ for all $\chi \in \mathcal{E}_{H_C}$. Our goal is to provide solutions to a class of controller synthesis problems, where P is a safety property. For this class of synthesis problems one can show the following [7].

Propostion 1. *A controller synthesis problem $(H, \Box F)$ can be solved by some controller if and only if it can be solved by a memoryless controller.*

Motivated by Proposition 1, we restrict our attention to memoryless controllers for $(H, \Box F)$ synthesis problems.

A subset $W \subseteq \mathbf{X}$ is called *controlled invariant* if there exists a controller that solves the controller synthesis problem $(H, \Box W)$, when $\mathit{Init} = W$; in this case we say that the controller *renders W invariant.* One can show that [7]:

Propostion 2. *A controller that solves the synthesis problem $(H, \Box F)$ exists if and only if there exists a unique, maximal*[7]*, controlled invariant set W with* $\text{Init} \subseteq W \subseteq F$.

Even though controllers that satisfy safety specifications by blocking the system can be excluded by our earlier assumption, a controller may still be able to "cheat" and satisfy the safety property by forcing the system to be Zeno. For this reason, we would like to compute a controller that solves the problem while imposing the minimal amount of restrictions on the controls it allows. Besides making such pathological cases more unlikely, this will also allow us greater flexibility when seeking optimal controllers for lower priority requirements, in a multi-objective setting. Feedback controllers that solve the synthesis problem $(H, \Box F)$ can be partially ordered by defining the relation $g_1 \subseteq g_2$ if and only if $g_1(x) \subseteq g_2(x)$ for all $x \in \mathbf{X}$. We say that a controller that solves $(H, \Box F)$ is *least restrictive* if it is a maximal element in this partial order.

4.2. **Game Theoretic Synthesis Procedure.** To guarantee that the safety property is satisfied despite the action of the disturbance we cast the controller synthesis problem as a zero sum game between u and d. The game can be written in the standard min-max setting by introducing a cost function induced by the discrete metric, $m : \mathbf{X} \times \mathbf{X} \to \mathbb{R}$, with

$$m(x_1, x_2) = \begin{cases} 0 & \text{if } x_1 = x_2 \\ 1 & \text{if } x_1 \neq x_2. \end{cases}$$

The metric induces a map, $M : 2^{\mathbf{X}} \times 2^{\mathbf{X}} \to \mathbb{R}$, by defining $M(W_1, W_2) = \min_{(x_1,x_2) \in W_1 \times W_2} m(x_1, x_2)$. Using M we can define the *cost*, $J : \mathcal{E}_H \to \mathbb{R}$, of an execution by

$$J(\chi) = \min_{t \in \tau} M(x(t), F^c)$$

Notice that $J(\chi) = 1$ if and only if $\Box F(\chi) = \text{True}$.

[7]In the sense of subset inclusion.

Intuitively, u tries to maximize J (prevent the state from leaving F). Because we have no control over the actions of d, we assume that it tries to minimize J (force the state to leave F). Consider the max-min solution, characterized by Bellman's equation:

$$J^*(x_0) = \max_g \min_d \left(\min_{(\tau,x,(u,d)) \in \mathcal{E}_{H_g}(x_0)} J(\tau, x, (u,d)) \right) \tag{5}$$

Note that we restrict our attention to memoryless strategies for u. The minimum over χ removes all nondeterminism. Therefore, provided a solution to this equation can be found, J^* is a well defined function of the initial state x_0. Since we would like all closed loop executions of the system to satisfy the safety specification, we give the disturbance the "benefit of the doubt". Notice, for example, that non-determinism is resolved in favor of the disturbance. Moreover, following the standard game theoretic convention, we assume that the player who appears first in the right hand side of the equation (the controller) has to play first, while the player who appears second (the disturbance) has access to the strategy of the first player, when called upon to make his/her decision.

Using J^* we define a set $W^* \subseteq \mathbf{X}$ by

$$W^* = \{x_0 \in \mathbf{X} | J^*(x_0) = 1\}. \tag{6}$$

One can show that [7]:

Propostion 3. *W^* is the maximal controlled invariant subset of F.*

Therefore, solving the controller synthesis problem involves finding a fixed point for equation (5). Dynamic programming methods can be used to solve this equation for purely discrete [42, 43] and purely continuous [44, 45] systems. More recently, it was shown that these techniques can be extended to classes of hybrid automata (timed automata [46, 47] and rectangular hybrid automata [48]) with discrete inputs. In [41, 49], the following algorithm was proposed to extend this approach to more general classes of hybrid automata.

Given any two sets, $K, L \subseteq X$, define three operators:

$$\begin{array}{rcrcl} Pre_u(K) & = & K \cap \{x^0 \in \mathbf{X} & | & \exists u \in \mathbf{U}\ \forall d \in \mathbf{D}\ \ R(x^0,(u,d)) \subseteq K \wedge (x^0,(u,d)) \notin Inv\} \\ Pre_d(K) & = & K^c \cup \{x^0 \in \mathbf{X} & | & \forall u \in \mathbf{U}\ \exists d \in \mathbf{D}\ \ R(x^0,(u,d)) \cap K^c \neq \emptyset\} \\ Reach(K,L) & = & \{x^0 \in \mathbf{X} & | & \forall u \in \mathcal{U}_g\ \exists d \in \mathcal{D}\ t \geq 0, x(0) = x^0 \wedge x(t) \in K \wedge x(t') \notin L\ \ \forall t' \in [0,t]\} \end{array}$$

In the expression for $Reach(K,L)$, $\mathcal{U}_g$ denotes the set of memoryless strategies for u, $\mathcal{D}$ the set of piecewise continuous functions from the reals to $\mathbf{D}$ and $(x,(u,d))$ a piece of continuous execution. Conceptually, $Pre_u(K)$ is the set of states that are in K, and can be forced by the control to jump back into K, whatever the disturbance does, $Pre_d(K)$ is the set of states that are either outside K, or, whatever the control does, may jump outside of K for some choice of the disturbance, and $Reach(K,L)$ is the set of states that, whatever the control does, can be driven by the disturbance to the set K along a continuous state trajectory that does not pass through L.

Based on these three definitions, the following algorithm can be used to compute the maximal controlled invariant subset of a set F.

Algorithm 3 (Invariant Set Computation)

> **initialization**: $W^0 = F$, $W^{-1} = \emptyset$, $i = 0$.
> **while** $W^i \neq W^{i-1}$ **do**
> $W^{i-1} = W^i \setminus Reach\left(Pre_d(W^i), Pre_u(W^i)\right)$
> $i = i - 1$
> **end while**

Propostion 4. *If the algorithms terminates after a finite number of steps, the fixed point, W^*, is the maximal controlled invariant subset of F.*

4.3. **Reach Set Computation.** Implementation of the above algorithm requires one to be able compute $Pre_u(W)$, $Pre_d(W)$ and $Reach(Pre_d(W), Pre_u(W))$. Computation of Pre_u and Pre_d is conceptually simple (it involves inverting the transition relation R), even though it may be difficult to implement in practice, depending on the method used to encode R. Computation of *Reach* is more conceptually more involved, since none of the standard techniques from optimal control and game theory are directly applicable. Fortunately the computation is simplified somewhat by the fact that it can be performed for each discrete state separately, since is only involves continuous evolution, along which the discrete state remains constant.

Motivated by the "capture set" computation in pursuit-evasion differential games [44, 45], we propose to compute this set using a pair of coupled partial differential equations. For two sets $K, L \subseteq X$, consider an arbitrary discrete state, $q \in \mathbf{X}_D$, and assume that there exist functions $k_q : \mathbf{X} \to \mathbb{R}$ and $l_q : \mathbf{X} \to \mathbb{R}$ differentiable in X_C, such that[8]

$$K = \{(q, y) \in X \mid l_K(q, y) < 0\} \text{ and } L = \{(q, y) \in X \mid l_L(q, y) \leq 0\}.$$

The set *Reach*(K, L) can the be computed using the level sets of the functions $J_K : \mathbf{X} \times \mathbb{R} \to \mathbb{R}$ and $J_L : \mathbf{X} \times \mathbb{R} \to \mathbb{R}$ characterized by the Hamilton-Jacobi partial differential equations

$$\begin{aligned} -\frac{\partial J_K}{\partial t}((q,y),t) &= \min\left\{0, H_K^*\left((q,y), \frac{\partial J_K}{\partial y}((q,y),t), t\right)\right\} \\ -\frac{\partial J_L}{\partial t}((q,y),t) &= \min\left\{0, H_L^*\left((q,y), \frac{\partial J_L}{\partial y}((q,y),t), t\right)\right\} \end{aligned}$$

with the Hamiltonians, H_K^* and H_L^* given by

$$\begin{aligned} H_K^*((q,y),p,t) &= \begin{cases} 0 & \text{if } J_L((q,y),t) \leq 0 \\ \max_{u \in \mathbf{U}} \min_{d \in \mathbf{D}} pf((q,y),(u,d)) & \text{otherwise} \end{cases} \\ H_L^*((q,y),p,t) &= \begin{cases} 0 & \text{if } J_K((q,y),t) \leq 0 \\ \min_{u \in \mathbf{U}} \max_{d \in \mathbf{D}} pf((q,y),(u,d)) & \text{otherwise} \end{cases} \end{aligned}$$

and initial conditions $J_K((q,y),0) = l_K(q,y)$ and $J_L((q,y),0) = l_L(q,y)$. If differentiable solutions to the coupled Hamilton-Jacobi partial differential equations can be found, then the level set

$$\{(q,y) \in x \mid J_K((q,y),t) < 0\}$$

converges to *Reach*(K, L) as t tends to $-\infty$.

Now the problem gets shifted to computing solutions for these partial differential equations. In practice, it is very rare that these equations can be solved analytically. We are currently investigating methods for carrying out the computation numerically, using finite elements, level set propagation methods [50] and viability techniques [51]. Other numerical methods that may be extended to address this problem involve approximating the *Reach* set by polyhedra [52, 53] and ellipsoids [54].

4.4. **Other Control Problems.** Besides reachability, a number of other hybrid control problems have been addressed in the literature. A substantial volume of work has been devoted to optimal control problems, both in a decentralized setting [55] and in a centralized setting, using dynamic programming techniques [56, 57] or extensions of the maximum principle [58]. Methods have also been developed for extending discrete event control techniques to hybrid systems [59] and for systematically decomposing a control task, so that it can be addressed by a hierarchical controller [60, 61, 62].

5. Concluding Remarks

We presented an overview of some of the theoretical issues and results concerning the modeling, analysis, and control of hybrid systems. Even though significant progress has been made in various areas, the field is very young and there are still more questions than answers!

From a modeling perspective, the notion of robustness of executions is important with respect to parameter and initial condition variations, since it affects our confidence in using simulation packages for hybrid systems.

[8]Notice that we are implicitly assuming that the set K is open and the set L is closed. The assumption that the set F is closed, together with continuity assumptions on R need to be imposed, to ensure this is the case at each step of the algorithm.

Another important issue is Zeno executions. Zeno hybrid systems are, in a sense, inadequate models of physical processes. To have confidence in our analysis and design techniques, we must understand how to avoid this modeling over-abstraction.

Both the analysis and the control methods presented here, are based on reachability computations for dynamical and control systems. Reachability computations are very critical, and the research community is focusing on various exact and approximate methods for computing reachable sets. However, many of these computations are either conceptually or computationally complex. A reduction of the complexity of the algorithms is necessary to extend these techniques to large scale systems, which are the motivation for this work. Complexity reduction via hierarchical or distributed designs will force us to better understand hierarchical and distributed systems.

Acknowledgments. The research was supported by ONR under grant N00014-97-1-0946, by DARPA under contract F33615-98-C-3614, and by ARO under grant DAAH04-96-1-0341. The authors would like to thank Magnus Egerstedt, Karl Henrik Johansson, Gerardo Lafferriere, Claire Tomlin, and Sergio Yovine for their contribution to the work presented here. They are also grateful to the students that attended EE291E, the course on hybrid systems at U.C.Berkeley in the Spring of 1999, for their comments and input.

References

[1] Pravin Varaiya, "Smart cars on smart roads: problems of control", *IEEE Transactions on Automatic Control*, vol. AC-38, no. 2, pp. 195–207, 1993.

[2] John Lygeros, Datta N. Godbole, and Shankar Sastry, "Verified hybrid controllers for automated vehicles", *IEEE Transactions on Automatic Control*, vol. 43, no. 4, pp. 522–539, 1998.

[3] S. Sastry, G. Meyer, C. Tomlin, J. Lygeros, D. Godbole, and G. Pappas, "Hybrid systems in air traffic control", in *IEEE Conference on Decision and Control*, 1995, pp. 1478–1483.

[4] Claire Tomlin, George Pappas, and Shankar Sastry, "Conflict resolution for air traffic management: a case study in multi-agent hybrid systems", *IEEE Transactions on Automatic Control*, vol. 43, no. 4, pp. 509–521, 1998.

[5] T.J. Koo, F. Hoffmann, H. Shim, B. Sinopoli, and S. Sastry, "Hybrid control of an autonomous helicopter", in *IFAC Workshop on Motion Control*, Grenoble, September, October 1998, pp. 285–290.

[6] C.G. Cassandras and D.L. Pepyne, "Optimal control of a class of hybrid systems", in *IEEE Conference on Decision and Control*, San Diego, California, December 10-12 1997, pp. 133–138.

[7] John Lygeros, Claire Tomlin, and Shankar Sastry, "Controllers for reachability specifications for hybrid systems", *Automatica*, pp. 349–370, March 1999.

[8] Karl Henrik Johansson, Magnus Egerstedt, John Lygeros, and S. Sastry, "On the regularization of zeno hybrid automata", *Systems and Control Letters*, 1999, (to appear).

[9] John Lygeros, Karl Henrik Johansson S. Sastry, and Magnus Egerstedt, "On the existence of executions of hybrid automata", in *IEEE Conference on Decision and Control*, Phoenix, Arizona, U.S.A., December 7-10 1999.

[10] Jean-Pierre Aubin, *Viability Theory*, Birkhäuser, Boston, 1991.

[11] Jean-Pierre Aubin, John Lygeros, Marc Quincampoix, and Nicolas Seube, "A viability approach to hybrid systems", 1999, (preprint).

[12] Michael Branicky, Ekaterina Dolginova, and Nancy Lynch, "A toolbox for proving and maintaining hybrid specifications", in *Hybrid Systems IV*, A. Nerode P. Antsaklis, W. Kohn and S. Sastry, Eds., number 1273 in LNCS, pp. 18–30. Springer Verlag, 1997.

[13] Karl Henrik Johansson, John Lygeros, S. Sastry, and Magnus Egerstedt, "Simulation of zeno hybrid automata", in *IEEE Conference on Decision and Control*, Phoenix, Arizona, U.S.A., December 7-10 1999.

[14] A. Deshpande, A. Gollu, and L. Semenzato, "The SHIFT programming language for dynamic networks of hybrid automata", *IEEE Transactions on Automatic Control*, vol. 43, no. 4, pp. 584–587, Apr. 1998.

[15] M. Andersson, *Object-Oriented Modeling and Simulation of Hybrid Systems*, PhD thesis, Lund Institute of Technology, Lund, Sweden, December 1994.

[16] P.J. Mosterman, "An overview of hybrid simulation phenomena and their support by simulation packages", in *Hybrid Systems: Computation and Control*, Frits W. Vaandrager and Jan H. van Schuppen, Eds., number 1569 in LNCS, pp. 165–177. Springer Verlag, 1999.

[17] A.J. van der Schaft and H. Schumacher, "Complementarity modeling of hybrid systems", *IEEE Transactions on Automatic Control*, vol. 43, no. 4, pp. 483–490, 1998.

[18] J-I. Imura and A.J. van der Schaft, "Well-posedness of a class of piecewise linear systems with no jumps", in *Hybrid Systems: Computation and Control*, Frits W. Vaandrager and Jan H. van Schuppen, Eds., number 1569 in LNCS, pp. 123–136. Springer Verlag, 1999.

[19] Nancy Lynch, Roberto Segala, Frits Vaandrager, and H.B. Weinberg, "Hybrid I/O automata", in *Hybrid Systems III*, number 1066 in LNCS, pp. 496–510. Springer Verlag, 1996.

[20] R. Alur and T.A. Henzinger, "Reactive modules", in *Proceedings of the 11th Annual Symposium on Logic in Computer Science*. 1996, pp. 207–218, IEEE Computer Society Press.

[21] Michael S. Branicky, *Studies in Hybrid Systems: Modeling, Analysis, and Control*, PhD thesis, Massachusetts Institute of Technology, 1995.

[22] Gerardo Lafferriere, George J. Pappas, and Shankar Sastry, "Reachability analysis of hybrid systems using bisimulations", in *IEEE Conference on Decision and Control*, Tampa, FL, Dec. 1998, pp. 1623–1628.

[23] Gerardo Lafferriere, George J. Pappas, and Shankar Sastry, "O-minimal hybrid systems", *Mathematics of Control, Signals, and Systems*, To appear.

[24] Gerardo Lafferriere, George J. Pappas, and Sergio Yovine, "A new class of decidable hybrid systems", in *Hybrid Systems : Computation and Control*, vol. 1569 of *Lecture Notes in Computer Science*, pp. 137–151. Springer Verlag, 1999.

[25] Gerardo Lafferriere, George J. Pappas, and Sergio Yovine, "Reachability computation for linear hybrid systems", in *Proceedings of the 14th IFAC World Congress*, Beijing, P.R. China, July 1999, vol. E, pp. 7–12.

[26] Z. Manna and A. Pnueli, *Temporal Verification of Reactive Systems: Safety*, Springer-Verlag, New York, 1995.

[27] R. Alur and D. Dill, "A theory of timed automata", *Theoretical Computer Science*, vol. 126, pp. 183–235, 1994.

[28] Anuj Puri and Pravin Varaiya, "Decidability of hybrid systems with rectangular differential inclusions", in *Computer Aided Verification*, 1994, pp. 95–104.

[29] T. Henzinger, P. Kopke, A. Puri, and P. Varaiya, "What's decidable about hybrid automata", in 27^{th} *Annual Symposium on the Theory of Computing, STOC'95*. 1995, pp. 373–382, ACM Press.

[30] L. van den Dries, *Tame Topology and o-minimal structures*, Cambridge University Press, 1998.

[31] A. Tarski, *A decision method for elementary algebra and geometry*, University of California Press, second edition, 1951.

[32] R. Alur, C. Courcoubetis, N. Halbawachs, T. A. Henzinger, P. H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine, "The algorithmic analysis of hybrid systems", *Theoretical Computer Science*, vol. 138, pp. 3–34, 1995.

[33] C. Daws, A. Olivero, and S. Yovine, "Verifying ET-LOTOS programs with KRONOS", in *Proc. 7th. IFIP WG G.1 International Conference of Formal Description Techniques, FORTE'94*, D. Hogrefe and S. Leue, Eds., Bern, Switzerland, Oct. 1994, Formal Description Techniques VII, pp. 227–242, Chapman & Hall.

[34] J. Bengtsson, K.G. Larsen, F. Larsson, P. Petterson, and W. Yi, "UPAAL: a tool suit for automatic verification of real-time systems", in *Hybrid Systems III*, number 1066 in LNCS, pp. 232–243. Springer Verlag, 1996.

[35] T. A. Henzinger, P. H. Ho, and H. Wong Toi, "A user guide to HYTECH", in *TACAS 95: Tools and Algorithms for the Construction and Analysis of Systems*, E. Brinksma, W. Cleaveland, K. Larsen, T. Margaria, and B. Steffen, Eds. 1995, number 1019 in LNCS, pp. 41–71, Springer Verlag.

[36] Myla Archer and Constance Heitmeyer, "Verifying hybrid systems modeled as timed automata: a case study", in *Proceedings of HART97*, Oded Maler, Ed., number 1201 in LNCS, pp. 171–185. Springer Verlag, Berlin, 1997.

[37] Hui Ye, Anthony Michel, and Ling Hou, "Stability theory for hybrid dynamical systems", *IEEE Transactions on Automatic Control*, vol. 43, no. 4, pp. 461–474, 1998.

[38] Michael S. Branicky, "Multiple Lyapunov functions and other analysis tools for switched and hybrid systems", *IEEE Transactions on Automatic Control*, vol. 43, no. 4, pp. 475–482, 1998.

[39] M. Johansson and A. Rantzer, "Computation of piecewise quadratic lyapunov functions for hybrid systems", *IEEE Transactions on Automatic Control*, vol. 43, no. 4, pp. 555–559, 1998.

[40] Claire Tomlin, John Lygeros, and Shankar Sastry, "Synthesizing controllers for nonlinear hybrid systems", in *Hybrid Systems: Computation and Control*, S. Sastry and T.A. Henzinger, Eds., number 1386 in LNCS, pp. 360–373. Springer Verlag, 1998.

[41] John Lygeros, Claire Tomlin, and Shankar Sastry, "On controller synthesis for nonlinear hybrid systems", in *IEEE Conference on Decision and Control*, Tampa, Florida, USA, December 16-18 1998, pp. 2101–2106.

[42] W. Thomas, "On the synthesis of strategies in infinite games", in *Proceedings of STACS 95, Volume 900 of LNCS*, Ernst W. Mayr and Claude Puech, Eds., pp. 1–13. Springer Verlag, Munich, 1995.

[43] P. J. G. Ramadge and W. M. Wonham, "The control of discrete event systems", *Proceedings of the IEEE*, vol. Vol.77, no. 1, pp. 81–98, 1989.

[44] T. Başar and G. J. Olsder, *Dynamic Non-cooperative Game Theory*, Academic Press, second edition, 1995.

[45] Joseph Lewin, *Differential Games*, Springer-Verlag, 1994.

[46] O. Maler, A. Pnueli, and J. Sifakis, "On the synthesis of discrete controllers for timed systems", in *Theoretical Aspects of Computer Science*. 1995, number 900 in LNCS, pp. 229–242, Springer Verlag.

[47] Michael Heymann, Feng Lin, and George Meyer, "Control synthesis for a class of hybrid systems subject to configuration-based safety constraints", in *Hybrid and Real Time Systems*, number 1201 in LNCS, pp. 376–391. Springer Verlag, 1997.

[48] Howard Wong-Toi, "The synthesis of controllers for linear hybrid automata", in *IEEE Conference on Decision and Control*, San Diego, California, USA, December 10-12 1997, pp. 4607–4613.

[49] Claire Tomlin, John Lygeros, and Shankar Sastry, "Computing controllers for nonlinear hybrid systems", in *Hybrid Systems: Computation and Control*, Frits W. Vaandrager and Jan H. van Schuppen, Eds., number 1569 in LNCS, pp. 238–255. Springer Verlag, 1999.

[50] J. A. Sethian, "Theory, algorithms, and applications of level set methods for propagating interfaces", Tech. Rep., Center for Pure and Applied Mathematics (PAM-651), University of California, Berkeley, 1995.

[51] Pierre Cardaliaguet, Marc Quincampoix, and Patrick Saint-Pierre, "Set-valued numerical analysis for optimal control and differential games", in *Stochastic and Differential Games: Theory and Numerical Methods*, Martino Bardi, T.E.S. Raghavan, and T. Parthasarathy, Eds., number 4 in Annals of the International Society of Dynamic Games, pp. 177–247. Birkhäuser, Boston, 1999.

[52] T. Dang and O. Maler, "Reachability analysis via face lifting", in *Hybrid Systems: Computation and Control*, S. Sastry and T.A. Henzinger, Eds., number 1386 in LNCS, pp. 96–109. Springer Verlag, 1998.

[53] M.R. Greenstreet and I. Mitchell, "Integrating projections", in *Hybrid Systems: Computation and Control*, S. Sastry and T.A. Henzinger, Eds., number 1386 in LNCS, pp. 159–174. Springer Verlag, 1998.

[54] A.B. Kurzhanski and P. Varaiya, "Ellipsoidal techniques for reachability analysis", (preprint), 1998.
[55] A. Nerode and W. Kohn, "Multiple agent hybrid control architecture", in *Hybrid Systems*, Robert L. Grossman, Anil Nerode, Anders P. Ravn, and Hans Rischel, Eds., number 736 in LNCS, pp. 297–316. Springer Verlag, New York, 1993.
[56] Michael S. Branicky, Vivek S. Borkar, and Sanjoy K. Mitter, "A unified framework for hybrid control: Model and optimal control theory", *IEEE Transactions on Automatic Control*, vol. 43, no. 1, pp. 31–45, 1998.
[57] A. Bensoussan and J.L. Menaldi, "Hybrid control and dynamic programming", *Dynamics of Continuous, Discrete and Impulsive Systems*, , no. 3, pp. 395–442, 1997.
[58] G. Grammel, "Maximum principle for a hybrid system via singular pertrubations", *SIAM Journal of Control and Optimization*, vol. 37, no. 4, pp. 1162–1175, 1999.
[59] M. Lemmon, J. A. Stiver, and P. J. Antsaklis, "Event identification and intelligent hybrid control", in *Hybrid Systems*, Robert L. Grossman, Anil Nerode, Anders P. Ravn, and Hans Rischel, Eds., number 736 in LNCS, pp. 268–296. Springer Verlag, New York, 1993.
[60] P.E. Caines and Y.J. Wei, "Hierarchical hybrid control systems: A lattice theoretic formulation", *IEEE Transactions on Automatic Control*, vol. 43, no. 4, pp. 501–508, April 1998.
[61] George J. Pappas, Gerardo Lafferriere, and Shankar Sastry, "Hierarchically consistent control systems", *IEEE Transactions on Automatic Control*, To appear.
[62] George J. Pappas, Gerardo Lafferriere, and Shankar Sastry, "Hierarchically consistent control systems", in *IEEE Conference on Decision and Control*, Tampa, FL, Dec. 1998, pp. 4336–4341.

Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, Berkeley, CA 94720, Tel. (510) 642-1857, Fax (510) 642-1341

E-mail address: `lygeros, gpappas, sastry@eecs.berkeley.edu`