Robust Controller Synthesis for Hybrid Systems Using Modal Logic*

T. Moor and J.M. Davoren

Research School of Information Sciences and Engineering
Australian National University
Canberra ACT 0200 AUSTRALIA
thomas.moor@anu.edu.au, j.m.davoren@anu.edu.au

Abstract. In this paper, we formulate and robustly solve a quite general class of hybrid controller synthesis problems. The type of controller we investigate is the switching control mechanism of a hybrid automaton (via guard and mode invariant sets), and the robustness result is with respect to variations in the right hand sides of the differential equations that depend continuously on a parameter. We present a novel methodology for controller design and synthesis which uses modal logic as a formalism for reasoning about sets of plant states, and various operators on sets arising from the differential equations and from metric tolerance relations on the state space.

1 Introduction

In general terms, a hybrid system H can be said to satisfy a performance specification *robustly* if every system H' in some nominated *variation class* around H also satisfies that specification. Likewise, a synthesis procedure for a class of control problems can be called robust if the nominal closed-loop hybrid system obtained from the solution controller can be shown to robustly satisfy each of the specifications of the problem, with respect to some nominated variation class. Robustness in hybrid control systems is an under-explored topic. A starting point is given in [10], which proposes a range of variation classes for hybrid automata, including near relatives of those in the present work and its predecessor [6]. Robustness issues for hybrid controller design, for a variety of different control settings and problems, are also investigated in [3, 8, 19, 21].

In this paper we find a robust solution to a rather general switching control problem for hybrid systems. The plant consists of a finite number of continuous systems, given by differential equations over a common state space; the controller steers the plant state by determining when to discretely switch between the various differential equations; and the closed-loop trajectories correspond to those of (a subclass of) the widely accepted hybrid automaton model. In addition to the well-studied classes of *safety* (reachability or invariance) and

* Research partially supported by US Office of Naval Research, Grant N 00014-98-1-0535.

liveness (non-blocking and non-Zeno) performance specifications, we deal with a class of *event sequence specifications*, requiring that trajectories traverse in prescribed sequences through the blocks of a given finite partition of the plant state space. This gives a general-purpose way of specifying the attainment of local goals along hybrid trajectories, and integrating the type of event sequence specifications examined in DES approaches to hybrid systems [5, 12, 17].

In [6], we develop an abstract algorithm which solves this controller synthesis problem for arbitrary differential equations with unique solutions, with a proof of finite termination derived from an assumption of compactness of the sets given in the data of the specifications. In that work, we consider one type of variation class that is motivated by considerations of *sensor and actuator imprecision*, and is obtained by allowing a metric tolerance or "margin of error" around the guard sets and in the reset relations; we have shown that our synthesis procedure is robust with respect to that class. In the present paper, we turn our attention to the more traditional control-theoretic perspective on robustness in terms of *parameter uncertainty*; i.e. variations in the right hand sides of the differential equations that depend continuously on a parameter. While these two variation classes are quite distinct, a key technical tool for both cases are *metric tolerance relations*, which are put to use in different ways.

This paper also demonstrates the flexibility and adaptability of our novel methodology for hybrid controller synthesis based on *modal logic*, first developed in [6, 7]. For our purposes, modal logic is best viewed as a formalism for reasoning about *sets of states* and *operators on sets* arising from relations on the state space. Considered as a family of logics, modal logic includes the temporal logics more commonly used in formal verification of hybrid systems. More precisely, we work with a *polymodal fusion* of several *normal monomodal logics* [20]. The main benefits our methodology are the following.

- Modal logic provides us with a uniform framework for investigating not only the widely used pre- and post-image operators induced by continuous flows, but also operators induced by metric tolerance relations, and the latter are essential in the context of robustness. As distinct from temporal logics, we reason about the *component parts* of hybrid trajectories, and this is essential for synthesis as opposed to analysis of hybrid systems.
- We use modal logic not merely as a convenient notation, but also draw on the power of *deductive proof systems*. In the course of proving the correctness of our synthesis algorithm, we show that certain key modal formulas are formally deducible from the statement of the algorithm together with explicit assumptions; this appeals to the soundness of a suitable Hilbert proof system w.r.t. the Kripke (transition system) semantics. In future work we will employ automated reasoning tools based on the decidability of the logical consequence and validity problems for modal logics, utilising tableaux proof systems [9].
- In our use of modal logic, we make a clean separation between (i) determining *what* sets need to be computed in order to solve the synthesis problem, and (ii) *how and when* such computations can be performed effectively. Issue (i) is resolved by the our synthesis algorithm below. Issue (ii) is essentially the

standard *model checking problem* for hybrid systems, and any model checking tools — either *exact* [1, 2, 14, 15] or *approximate* [4, 13, 16] — can be used to implement our synthesis algorithm.

In this short paper, we restrict our focus to the core ingredients, and to those aspects of the work that are crucial for plant parameter robustness. Consult [6] for a more detailed account of our framework based on modal logic.

The body of the paper is organised as follows. In Section 2, we briefly review hybrid automata, define plant parameter variation classes, and give a key result on parameterised vector fields. In Section 3, we formally state the controller synthesis problem. Section 4 is a terse review of modal logic applied to hybrid systems, and in Section 5, we give our abstract synthesis algorithm, formalised in the language of modal logic. In Section 6, we outline the proof of the main result of robust correctness. The concluding Section 7 includes a brief discussion of effective implementations of the procedure.

2 Hybrid Automata

We work with the standard and widely accepted hybrid automaton model of Alur, Henzinger *et al.* [1, 2].

Definition 1. *A* hybrid automaton *is a system*

$$H = (\, Q, E, X, \{F_q, Inv_q\}_{q\in Q}, \{r_{q,q'}, Grd_{q,q'}\}_{(q,q')\in E}\,), \tag{1}$$

where: Q *is a finite set of* discrete control modes*;* $E \subseteq Q \times Q$ *is the* discrete transition relation*;* $X \subseteq \mathbb{R}^n$ *is the* continuous state space*; for each* $q \in Q$*,* $F_q : X \to \mathbb{R}^n$ *is a vector field, and* $Inv_q \subseteq X$*; and for each* $(q, q') \in E$*,* $r_{q,q'} \subseteq X \times X$ *is a reset relation, and* $Grd_{q,q'} = \mathrm{dom}(r_{q,q'})$*.*

In order to ensure that closed-loop trajectories are well-defined, we assume that the vector fields F_q are locally Lipschitz continuous, and the state space X is open. Then from each initial condition $x_0 \in X$, each differential equation $\dot{x} = F_q(x)$ has a unique maximal integral curve in X on a well defined maximal interval of time $[0,\, T_q(x_0))$, where $T_q(x_0) \in \mathbb{R}^+ \cup \{\infty\}$. We denote this maximal curve by

$$\Phi_q(x_0,\ \cdot\,)\colon [0,\, T_q(x_0)) \to X\,. \tag{2}$$

In the case of $T_q(x_0) < \infty$, it is well known that $\Phi_q(x_0,\ \cdot\,)$ escapes from any *bounded* subset of X at some time less than or equal to $T_q(x_0)$. For the scope of this paper, we can restrict attention to bounded invariant sets Inv_q. Then maximal curves from $x_0 \in Inv_q$ either leave Inv_q within finite time or stay within Inv_q forever with $T_q(x_0) = \infty$. Closed-loop trajectories are then defined as follows.

Definition 2. *A* trajectory *of a hybrid automaton* H *is a finite or infinite sequence* $\eta = (\Delta_i, q_i, \gamma_i)_{i\in I}$ *such that for each* $i \in I$*:*

- *the duration* $\Delta_i \in \mathbb{R}^+ \cup \{\infty\}$*, with* $\Delta_i = \infty$ *only if* I *is finite and* $i = \max(I)$*;*

- *the discrete state* $q_i \in Q$*;*
- *the continuous curve* $\gamma_i : [0, \Delta_i] \to X$ *satisfies* $\gamma_i(t) = \Phi_{q_i}(\gamma_i(0), t)$ *and* $\gamma_i(t) \in Inv_{q_i}$ *for all* $t \in [0, \Delta_i]$*, with the convention that* $[0, \Delta_i]$ *is* $[0, \infty)$ *if* $\Delta_i = \infty$*;*
- *if* $i < \sup(I)$*, then* $(q_i, q_{i+1}) \in E$ *and* $\gamma_i(\Delta_i) \stackrel{r_{q_i,q_{i+1}}}{\longrightarrow} \gamma_{i+1}(0)$*.*

A trajectory will be called: step-infinite *if it makes infinitely many switches;* time-infinite *if the sum over all durations is unbounded; and* full *if it is either step-infinite or time-infinite or else it is* blocked*, in the sense that it cannot be extended to reach any further guard region.*

A broad framework of variation classes for hybrid automata is proposed in [10]. Our interest here is in parameter variations in the vector fields.

Definition 3. *Given a hybrid automaton* H *as in Eq.* (1)*, let* $F_q^v \colon X \to \mathbb{R}^n$ *be a family of vector fields parameterised by the discrete modes* $q \in Q$ *and an uncertainty parameter* $v \in V \subseteq \mathbb{R}^m$*, where* $\mathbf{0} \in V$ *and* $F_q^{\mathbf{0}} \equiv F_q$*. Then*

$$H^v = (\, Q, E, X, \{F_q^v, Inv_q\}_{q \in Q}, \{r_{q,q'}, Grd_{q,q'}\}_{(q,q') \in E}\,), \tag{3}$$

$$\mathcal{H}^\varepsilon = \{H^v |\ ||v|| < \varepsilon\} \tag{4}$$

defines a parameterised variation class around the nominal model $H^{\mathbf{0}} = H$ *with variation bound* ε*.*

In correspondence with the nominal model, we denote the maximal integral curves of the vector field F_q^v by $\Phi_q^v(x_0, \cdot) \colon [0, T_q^v(x_0)) \to X$ where $T_q^v(x_0) \in \mathbb{R}^+ \cup \{\infty\}$. The following assumptions on the vector fields are to ensure that the flow $\Phi_q^v(x_0, t)$ is continuous in v and x_0.

(A0) The parameter set V is open. The vector field $F_q^v(x)$ is continuous in both x and v. Furthermore, $F_q^v(x)$ is locally Lipschitz continuous in x uniformly in v; i.e. there exists a Lipschitz constant which may depend on x but not on v.

In particular, assumption **(A0)** ensures that for any given finite time interval and any given *open tube* around the nominal integral curve $\Phi_q(x_0, t)$, all variations $\Phi_q^v(x_0, t)$ evolve within that tube – provided that the variation is sufficiently small; e.g. [11], Theorem 2.6. In the hybrid setting, we need to examine continuous parameter dependency w.r.t. a given domain D in the state space, rather than w.r.t. a given interval on the time axis. That is, we are interested in the dependency of $\Phi_q^v(x_0, t)$ in v as long as that curve evolves within an invariant set Inv_q. We formalise these ideas in terms of metric tolerance relations, and in so doing, set up the link to modal logics.

Definition 4. *Given a metric* d *on the state space* X*, the* δ-ball $B_\delta(x)$ *of radius* $\delta > 0$ *with centre* $x \in X$ *is defined by*

$$B_\delta(x) \stackrel{\text{def}}{=} \{\, y \in X \mid d(x, y) < \delta \,\}. \tag{5}$$

For a set $A \subseteq X$*, we call the set* $B_\delta(A) \stackrel{\text{def}}{=} \{x \in X |\ B_\delta(x) \cap A \neq \emptyset\}$ *the* δ-expansion *of* A*. We also call the (reflexive and symmetric) relation* $B_\delta \subseteq X \times X$

a metric tolerance relation. *For the scope of this paper, d is assumed to be a metric that induces the standard Euclidean topology on X.*

For a set $A \subseteq X$, let

$$T_q^v(A, x_0) = \sup\{\tau < T_q^v(x_0) \mid (\forall s \in [0, \tau)\,)\, \Phi_q^v(x_0, s) \in A\,\} \tag{6}$$

denote the time at which $\Phi_q^v(x_0, \cdot\,)$ escapes from A, so $T_q^v(x_0) = T_q^v(X, x_0)$.

Proposition 1. *Let D be a compact set with $B_{4\delta}(D) \subseteq X$ for a given metric tolerance $\delta > 0$. Furthermore, assume $T_q(B_{3\delta}(D), x_0) < \infty$ for all $x_0 \in D$. Then there exits a variation bound $\varepsilon > 0$ such that $T_q^v(D, x_0) \leq T_q(B_{2\delta}(D), x_0) \leq T_q^v(x_0)$ and $\Phi_q^v(x_0, t) \in B_{2\delta}(\Phi_q(x_0, t))$ for all $t \leq T_q(B_{2\delta}(D), x_0)$, all $x_0 \in D$ and all v, $||v|| < \varepsilon$.*

Proof. Apply [11], Theorem 2.6, together with a standard compactness argument.

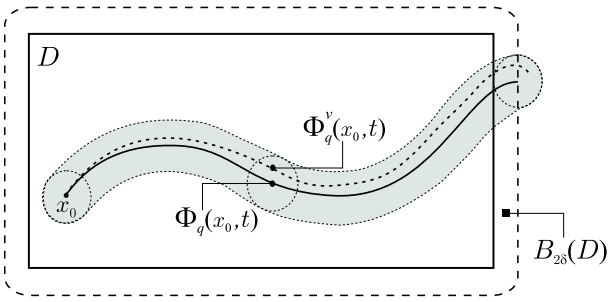

Fig. 1. Illustration of Proposition 1

Figure 1 illustrates a perturbed integral curve lying within a 2δ-tube around the nominal curve from a point $x_0 \in D$, as given by Proposition 1. When a hybrid automaton with bounded invariant sets is designed so that when an integral curve leaves its invariant set, it does so by some uniform minimum distance, Proposition 1 provides an elementary robustness property for this continuous evolution in between any two successive discrete control switches. However, even small variations in the parameter may have the effect that a perturbed trajectory runs into a different guard set than the corresponding nominal trajectory. In turn, such a perturbed trajectory may switch to a different vector field and thus may potentially stray far away from the nominal trajectory. The avoidance of this phenomenon motivates several of the design choices in formulating a robust solution to our target control problem.

3 Control Problem Statement

A hybrid automaton can be seen as the closed-loop feedback system resulting from the inter-connection of a switched continuous plant and a discrete switching controller. See [6] for a more detailed analysis of this control-theoretic content of a hybrid automaton. For the controller synthesis problem under investigation, the plant is given by a finite family of vector fields $F_c\colon X \to \mathbb{R}^n$ indexed by a *control alphabet* $c \in C$. We then ask for a synthesis procedure that constructs a closed-loop hybrid automaton H by building the missing entities that form the switching control mechanism, namely Q, E, Inv_q, and $Grd_{q,q'}$, where the reset relation is required to be *elementary*; i.e.

$$r_{q,q'} = test.Grd_{q,q'} \stackrel{\text{def}}{=} \{\, (x,x') \in X \times X \mid x \in Grd_{q,q'} \text{ and } x' = x \,\}\,. \quad (7)$$

As Q is not known in advance, the synthesis procedure also needs to allocate a particular control $c \in C$ (indexing a vector field) to each discrete mode $q \in Q$.

The control goal is to satisfy the following closed-loop performance specifications.

(S1) *Safety*: given a proscribed set $Bad \subseteq X$, construct a set $Good \subseteq X - Bad$ with the property that every H-trajectory starting in $Good$ always remains outside Bad.

(S2) *Event sequence behaviour with δ-overlaps*: given a finite partition $\{E_k\}_{k\in K}$ of $X - Bad$, a relation $next \subseteq K \times K$, and a metric parameter $\delta > 0$, let $A_k = B_\delta(E_k)$ be the δ-expansion of the partition block E_k, for each $k \in K$; the requirement is that for every full H-trajectory starting in $Good$, whenever it enters one of the sets A_k, it remains there until it crosses into $A_{k'} - A_k$, for some $k' \in next(k)$.

(S3) *Liveness I*: every full H-trajectory starting in $Good$ shall be step-infinite.

(S4) *Liveness II*: every full H-trajectory starting in $Good$ shall be time-infinite.

The specification **(S1)** is the classic form of a safety property, while **(S3)** and **(S4)** are, respectively, the *non-blocking* and the *non-Zeno* forms of liveness properties. The specification **(S2)** prescribes an order of traversal through the δ-expanded partition blocks. Formally, switches from one such block to another are identified as events from the finite alphabet K and **(S2)** requires the closed-loop to generate a sublanguage of $\{(k_i)_{i\in I} \mid \forall i < \sup(I) :\ k_{i+1} \in next(k_i)\}$. The metric tolerance δ ensures that the event sequence specification refers to overlapping regions $A_k \cap A_{k'}$ rather than the common boundaries $bd(E_k) \cap bd(E_{k'})$ of partition blocks. In particular, the overlaps are full dimensional and allow for some "wiggle room" which is essential for our robustness results. A more detailed motivation of **(S2)** is given in [6].

Our synthesis procedure is subject to the following further assumptions.

(A1) The set $X - Bad$ is compact (with respect to the standard Euclidean topology).

(A2) The map $next \subseteq K \times K$ is *total*, so for each $k \in K$, there is at least one $k' \in next(k)$.

(A3) For all $k, k' \in K$ such that $k \xrightarrow{next} k'$, the partition blocks E_k and $E_{k'}$ are contiguous in the sense that $bd(E_k) \cap bd(E_{k'}) \neq \varnothing$.

(A4) For each $k \in K$, the block E_k has a non-empty δ-*contraction*; i.e. the set $\{x \in X \mid B_\delta(x) \subseteq E_k\}$ is non-empty.

(A5) For all $k, k', k'' \in K$ such that $k \xrightarrow{next} k' \xrightarrow{next} k''$, the infimum of the metric distance between points in the set $bd(E_k) \cap bd(E_{k'})$ and points in the set $bd(E_{k'}) \cap bd(E_{k''})$ is at least 3δ.

By **(A1)**, the relevant portion of the state space is required to be compact; this is used in applying Proposition 1 and in proving finite termination of our algorithm. Assumptions **(A2)**, **(A3)** and **(A4)** are non-triviality conditions. The assumption **(A5)** gives a foundation for non-Zeno-ness by ensuring that closed-loop trajectories must traverse some minimum spatial distance when fulfilling the event sequence specification.

4 Modal Logics for Hybrid Systems

This section sets out only the bare details of modal logics and their application to hybrid systems. For a more substantial account, the reader is referred to [7] and also to [6]. The handbook chapter [18] gives a broader introduction to the family of modal and temporal logics.

A *modal signature* is a pair $(\mathrm{Rel}, \mathrm{Prp})$, where Rel is an alphabet of atomic relation labels, and Prp is an alphabet of atomic propositions. The set $\mathcal{L}(\mathrm{Rel}, \mathrm{Prp})$ of *modal formulas* φ of signature $(\mathrm{Rel}, \mathrm{Prp})$ is generated by the grammar:

$$\varphi \ ::= \ \ p \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \langle a \rangle \varphi \tag{8}$$

where $p \in \mathrm{Prp}$ and $a \in \mathrm{Rel}$. The other Boolean connectives are definable, e.g. $\varphi_1 \wedge \varphi_2 \stackrel{\mathrm{def}}{=} \neg(\neg\varphi_1 \vee \neg\varphi_2)$, $\varphi_1 \rightarrow \varphi_2 \stackrel{\mathrm{def}}{=} (\neg\varphi_1 \vee \varphi_2)$, as are the dual modal operators: $[a]\varphi \stackrel{\mathrm{def}}{=} \neg\langle a \rangle \neg\varphi$.

The formal semantics of modal (and temporal) logics are given with respect to *labeled transition systems*, also called *LTS models* or *generalized Kripke models*. An LTS model of signature $(\mathrm{Rel}, \mathrm{Prp})$ is a structure:

$$\mathfrak{M} = (S, \{a^{\mathfrak{M}}\}_{a \in \mathrm{Rel}}, \{[\![p]\!]^{\mathfrak{M}}\}_{p \in \mathrm{Prp}}) \,. \tag{9}$$

where: $S \neq \varnothing$ is the state space, of arbitrary cardinality; for each $a \in \mathrm{Rel}$, $a^{\mathfrak{M}} \subseteq S \times S$ is a relation; and for each $p \in \mathrm{Prp}$, $[\![p]\!]^{\mathfrak{M}} \subseteq S$ is a subset of states. For formulas $\varphi \in \mathcal{L}(\mathrm{Rel}, \mathrm{Prp})$, the *denotation set* $[\![\varphi]\!]^{\mathfrak{M}} \subseteq S$ is defined by induction, starting with the sets $[\![p]\!]^{\mathfrak{M}}$ denoting atomic propositions $p \in \mathrm{Prp}$. For compound formulas:

$$[\![\neg\varphi]\!]^{\mathfrak{M}} \ \stackrel{\mathrm{def}}{=} S - [\![\varphi]\!]^{\mathfrak{M}} \,, \tag{10}$$

$$[\![\, \langle a \rangle \varphi \,]\!]^{\mathfrak{M}} \ \stackrel{\mathrm{def}}{=} \mathrm{Pre}^{\exists}(a^{\mathfrak{M}}) \left([\![\varphi]\!]^{\mathfrak{M}} \right) \quad \text{for } a \in \mathrm{Rel} \,, \tag{11}$$

$$[\![\varphi_1 \vee \varphi_2]\!]^{\mathfrak{M}} \ \stackrel{\mathrm{def}}{=} [\![\varphi_1]\!]^{\mathfrak{M}} \cup [\![\varphi_2]\!]^{\mathfrak{M}} \,, \tag{12}$$

where the *existential pre-image operator* $\mathrm{Pre}^{\exists}(r) : \mathcal{P}(S) \rightarrow \mathcal{P}(S)$ of a relation $r \subseteq S \times S$ is:

$$\mathrm{Pre}^{\exists}(r)(A) \ \stackrel{\mathrm{def}}{=} \ \{x \in S \mid (\exists y \in S)[\, x \xrightarrow{r} y \ \wedge \ y \in A]\} \,. \tag{13}$$

For formulas $\varphi \in \mathcal{L}(\mathrm{Rel}, \mathrm{Prp})$ and models $\mathfrak{M}$ of signature $(\mathrm{Rel}, \mathrm{Prp})$, we say: φ is *satisfied* at state s in $\mathfrak{M}$, written $\mathfrak{M}, s \vDash \varphi$, if $s \in [\![\varphi]\!]^{\mathfrak{M}}$; and φ is *true* in $\mathfrak{M}$, or $\mathfrak{M}$ *satisfies* φ, written $\mathfrak{M} \vDash \varphi$, if $[\![\varphi]\!]^{\mathfrak{M}} = S$.

In encoding the control problem and input data in modal logic, we work in an LTS model $\mathfrak{M}_0$ over the plant state space $S := X \subseteq \mathbb{R}^n$. The set of atomic proposition symbols is $\mathrm{Prp}_0 = \{\mathbf{Bad}\} \cup \{\mathbf{E}_k \mid k \in K\}$, with the self-evident denotation sets. The alphabet Rel_0 of relation symbols will grow dynamically in the course of the synthesis algorithm (but will still be finite, due to finite termination). The relation symbols divide into four sorts, which we indicate by consistently using the same letters, adorned with subscripts and superscripts when needed. We will have relation symbols $\mathbf{e}$ for *evolution relations* and $\mathbf{f}$ for *flow* (or *orbit*) *relations*; symbols $\mathbf{r}$ for *reset relations*; and symbols $\boldsymbol{\delta}$ for *metric tolerance relations.*

Definition 5. *Given a flow* $\Phi : X \times \mathbb{R}^+ \to X$ (*possibly a* partial function) *and any set* $A \subseteq X \subseteq \mathbb{R}^n$*, define a relation* $e(A, \Phi) \subseteq X \times X$ *of* evolution along Φ restricted within A*, by:*

$$x \xrightarrow{e(A,\Phi)} x' \quad \stackrel{\text{def}}{\Leftrightarrow} \quad (\exists t \in \mathbb{R}^+)[\, x' = \Phi(x,t) \ \wedge \ (\forall s \in [0,t])\, \Phi(x,s) \in A \,]\,. \tag{14}$$

The unrestricted orbit relation $f(\Phi) \subseteq X \times X$ *is the special case:* $f(\Phi) = e(X, \Phi)$.

This precisely captures the notion of a hybrid trajectory segment, taking $A = Inv_q$ and $\Phi = \Phi_q$ for each control mode $q \in Q$. For $\mathbf{e}^{\mathfrak{M}_0} = e(A, \Phi)$, a formula $\langle \mathbf{e} \rangle \varphi$ denotes the subset of states in A from which there is a curve along Φ that reaches *some* φ-state, and stays within A at all intermediate points; this is the standard notion of backwards reachability extensively used in the hybrid systems literature. The dual $[\mathbf{e}]$ operator expresses invariance, since $[\mathbf{e}]\varphi$ denotes the set of points *all* of whose e-successors are φ-states. The compound $\mathbf{A} \wedge [\mathbf{e}]\langle \mathbf{e} \rangle \varphi$ denotes the set of states in A all of whose e-successors have a further e-successor which satisfies φ, and so captures the notion of *inevitably* reaching a φ-state. This compound construct is an essential ingredient of our synthesis algorithm, where in addressing the event sequence requirement **(S2)**, we need to identify states that are inevitably driven to certain local goal regions. Figure 2 illustrates the difference between the inevitability formula $\mathbf{A} \wedge [\mathbf{e}]\langle \mathbf{e} \rangle \mathbf{G}$ and the backwards reachability formula $\langle \mathbf{e} \rangle \mathbf{G}$, where $\mathbf{G}$ denotes a local goal.

The reset relations under study are elementary, so $(\mathbf{r}_{q,q'})^{\mathfrak{M}_0} = test.Grd_{q,q'}$. In this case, the modal operators $\langle \mathbf{r}_{q,q'} \rangle$ and $[\mathbf{r}_{q,q'}]$ can be eliminated:

$$\langle \mathbf{r}_{q,q'} \rangle \varphi \ \leftrightarrow \ (\mathbf{Grd}_{q,q'} \wedge \varphi) \qquad \text{and} \qquad [\mathbf{r}_{q,q'}] \varphi \ \leftrightarrow \ (\mathbf{Grd}_{q,q'} \rightarrow \varphi)\,. \tag{15}$$

For metric tolerance relations $\boldsymbol{\delta}^{\mathfrak{M}_0} = B_\delta$, a formula $\langle \boldsymbol{\delta} \rangle \varphi$ denotes the δ-*expansion* of the set of φ-states, since $B_\delta(A) = \mathrm{Pre}^{\exists}(B_\delta)(A)$. The dual box formula $[\boldsymbol{\delta}]\varphi$ denotes the δ-*contraction* of the set of φ-states, meaning the set of points in $[\![\, \varphi \,]\!]^{\mathfrak{M}_0}$ around which one can fit a δ-ball wholly inside $[\![\, \varphi \,]\!]^{\mathfrak{M}_0}$.

An axiomatic Hilbert-style proof system capturing basic properties of the modal operators of evolution, flow and metric tolerance relations is given in [6], Section 5. These axioms also may form a basis for employing automated reasoning tools, e.g. tableaux proof systems [9].

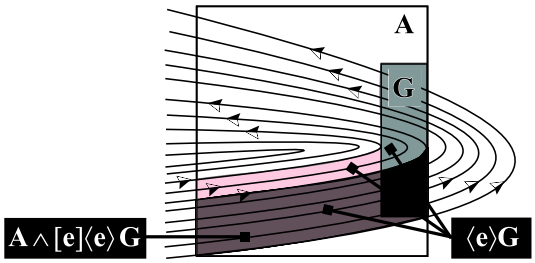

Fig. 2. Denotation of inevitability and backwards reachability formulas

5 Abstract Algorithm of Synthesis Procedure

Our solution to the control problem consists of two parts. First, we strategically construct a finite number of subsets of X, defined in terms of the input data F_c, Bad, E_k, $next$ and δ. Formally, this construction is given as an abstract algorithm where the sets of states are defined by modal logic formulas. The algorithm is a fine-tuned variation of the one presented in [6]. In particular, the proof of finite termination as given in [6] carries over without change. The algorithm may either terminate with failure or indicating success. In the former case it produces some diagnostic output, as described below. In the case of successful termination, the second part of our solution procedure uses the constructed sets of states to assemble our nominal closed-loop hybrid automaton H and the set $Good$. The pair (H, $Good$) then is guaranteed to fulfill the performance specifications **(S1)**-**(S4)**. It is in this second part that the present work departs essentially from [6] and extends the scope of our method to the plant parameter variation class $\mathcal{H}^{\varepsilon}$.

The first part of our procedure is given in Algorithm 1; see [6] for a more detailed exposition including graphical output for a nontrivial example. Given the page constraints on this short paper, we are restricted to a brief discussion of the individual steps of the algorithm. We begin by taking the given metric parameter δ and decomposing it as a sum $\delta = 2\delta_1 + 2\delta_2$, with $\delta_1 > \delta_2 > 0$. Roughly speaking, δ_1 is used as "wiggle room" in order to cope with parameter variations in the vector fields, while δ_2 gives some extra allowance required for an implementation based on approximated evaluation of the modal operators. In the initialisation phase, the formula $\mathbf{Danger}_k$ denotes the states that are dangerous from the viewpoint of the block E_k: the outright Bad states and the relative bad states in blocks $E_{k'}$ with k' not $next$-related to k. The formula $\mathbf{A}^{(0)}_{k,0}$ denotes A_k, the initial δ-expansion of E_k. The formula $\mathbf{Goal}^{(0)}_{k,0}$ denotes the states that are *well inside* $E_{k'}$, in the $(\delta - \delta_2)$-contraction of $E_{k'}$, for some $k' \in next(k)$.

The main routine consists of an outer j-iteration which runs the core routine for successive j and each $k \in K$. The purpose of the core routine is to identify states in $\mathbf{A}^{(j)}_{k,0}$ that can be safely driven into the $\mathbf{Goal}^{(j)}_{k,0}$. The iteration in i is

Algorithm 1 Abstract algorithm for computing sets for synthesis procedure

% INITIALISATION %

$j := 0$ and $i := 0$

FOR ALL $k \in K$ **DO**

  $\mathbf{Danger}_k \stackrel{\text{def}}{=} \langle \boldsymbol{\delta_1} \rangle \mathbf{Bad} \vee \bigvee_{k' \notin next(k) \cup \{k\}} \mathbf{E}_{k'}$ $\mathbf{A}_{k,0}^{(0)} \stackrel{\text{def}}{=} \langle \boldsymbol{\delta} \rangle \mathbf{E}_k$

  $\mathbf{Goal}_{k,0}^{(0)} \stackrel{\text{def}}{=} \bigvee_{k' \in next(k)} [\boldsymbol{\delta} - \boldsymbol{\delta}_2] \mathbf{E}_{k'}$ $\mathbf{Drop}_k^{(0)} \stackrel{\text{def}}{=} \mathbf{A}_{k,0}^{(0)} \wedge \mathbf{Goal}_{k,0}^{(0)}$

% MAIN-ROUTINE %

REPEAT % FOR $j = 0, 1, \ldots$ %

  % CORE-ROUTINE(k, j) for $k \in K$ %

  FOR ALL $k \in K$ **DO**

    REPEAT % FOR $i = 0, 1, \ldots$ %

      FOR ALL $c \in C$ **DO**

        $(\mathbf{e}_{k,i,c}^{(j)})^{\mathfrak{M}_0} \stackrel{\text{def}}{=} e([\![\mathbf{A}_{k,i}^{(j)}]\!]^{\mathfrak{M}_0}, \Phi_c)$

        $\mathbf{Sure}_{k,i,c}^{(j)} \stackrel{\text{def}}{=} \mathbf{A}_{k,i}^{(j)} \wedge [\mathbf{e}_{k,i,c}^{(j)}] \neg \mathbf{Danger}_k$

        $\mathbf{Success}_{k,i,c}^{(j)} \stackrel{\text{def}}{=} \mathbf{A}_{k,i}^{(j)} \wedge [\mathbf{e}_{k,i,c}^{(j)}] \langle \mathbf{e}_{k,i,c}^{(j)} \rangle \big(\mathbf{A}_{k,i}^{(j)} \wedge \mathbf{Goal}_{k,i}^{(j)} \big) \wedge \langle \mathbf{f}_c \rangle \neg \langle \boldsymbol{\delta}_1 \rangle \mathbf{A}_{k,i}^{(j)}$

        $\mathbf{Fine}_{k,i,c}^{(j)} \stackrel{\text{def}}{=} \mathbf{Sure}_{k,i,c}^{(j)} \wedge \mathbf{Success}_{k,i,c}^{(j)}$

      $\mathbf{Goal}_{k,i+1}^{(j)} \stackrel{\text{def}}{=} \mathbf{Goal}_{k,i}^{(j)} \vee \big(\bigvee_{c \in C} [2\boldsymbol{\delta}_1 + \boldsymbol{\delta}_2] \mathbf{Fine}_{k,i,c}^{(j)} \big)$

      $\mathbf{A}_{k,i+1}^{(j)} \stackrel{\text{def}}{=} \mathbf{A}_{k,i}^{(j)} \wedge \neg [2\boldsymbol{\delta}_1 + \boldsymbol{\delta}_2] \mathbf{Goal}_{k,i+1}^{(j)}$

      $i := i + 1$

    UNTIL $\mathfrak{M}_0 \models \neg [\boldsymbol{\delta}_2] \big(\mathbf{Goal}_{k,i}^{(j)} \wedge \neg \mathbf{Goal}_{k,i-1}^{(j)} \big)$

    $last(k, j) := i - 1$

  % j-th ATTEMPT AT *next* COMPATIBILITY %

  FOR ALL $k \in K$ **DO**

    $\mathbf{Pick}_k^{(j)} \stackrel{\text{def}}{=} \bigvee_{k' \in next(k)} \big(\mathbf{A}_{k,0}^{(0)} \wedge \mathbf{Goal}_{k',last(k',j)}^{(j)} \big)$

    $\mathbf{Drop}_k^{(j+1)} \stackrel{\text{def}}{=} \mathbf{Drop}_k^{(j)} \wedge \neg \langle \boldsymbol{\delta}_2 \rangle \big(\mathbf{Drop}_k^{(j)} \wedge \neg \mathbf{Pick}_k^{(j)} \big)$

    $\mathbf{A}_{k,0}^{(j+1)} \stackrel{\text{def}}{=} \mathbf{A}_{k,0}^{(0)}$ $\mathbf{Goal}_{k,0}^{(j+1)} \stackrel{\text{def}}{=} \mathbf{Drop}_k^{(j+1)}$

  $j := j + 1$ and $i := 0$

UNTIL $\mathfrak{M}_0 \models \bigwedge_{k \in K} \big(\mathbf{Drop}_k^{(j-1)} \rightarrow \mathbf{Pick}_k^{(j-1)} \big)$

% FINAL CLEAN-UP %

IF $[\![\mathbf{Drop}_k^{(j-1)}]\!]^{\mathfrak{M}_0} = \varnothing$ for some $k \in K$ **THEN**

  terminate & report incompatibility between k and its *next*-successors

ELSE

  $final := j - 1$

  FOR ALL $k \in K$ and $i \in I_k \stackrel{\text{def}}{=} \{0, \cdots, last(k, final)\}$ and $c \in C$ **DO**

    $\mathbf{A}_{k,i} \stackrel{\text{def}}{=} \mathbf{A}_{k,i}^{(final)}$ $(\mathbf{e}_{k,i,c})^{\mathfrak{M}_0} \stackrel{\text{def}}{=} (\mathbf{e}_{k,i,c}^{(final)})^{\mathfrak{M}_0}$

    $\mathbf{Goal}_{k,i} \stackrel{\text{def}}{=} \mathbf{Goal}_{k,i}^{(final)}$ $\mathbf{Fine}_{k,i,c} \stackrel{\text{def}}{=} \mathbf{Fine}_{k,i,c}^{(final)}$

  terminate with success

with respect to the number of control switches required to achieve this local goal. In the iteration, $\mathbf{Goal}_{k,i}^{(j)}$ accumulates the states that can be driven to the initial $\mathbf{Goal}_{k,0}^{(j)}$ by at most i switches, while $\mathbf{A}_{k,i+1}^{(j)}$ denotes states which have been not resolved so far. The formula $\mathbf{Fine}_{k,i,c}^{(j)}$ identifies the states in $\mathbf{A}_{k,i}^{(j)}$ which can be driven to $\mathbf{Goal}_{k,i}^{(j)}$ using control c, and done so safely by being kept out of $\mathbf{Danger}_k$. Note that the recursive definition of $\mathbf{A}_{k,i+1}^{(j)}$ in line 17 involves the terms $\mathbf{A}_{k,i}^{(j)}$ and $\mathbf{Goal}_{k,i}^{(j)}$ within the scope of an odd number of negations. Thus it cannot be coded as a μ-calculus formula, and in particular the inner i-iteration is essentially different from fixed point iterations of maximal invariant sets as used in game-theoretic approaches to safety problems for hybrid systems [19].

While the core routine works on solving the problem locally, within the individual A_k, the outer j-loop checks that these local solutions can be merged to form a global controller. Within each A_k, the region where the local solution finally "drops-off" states is denoted by $\mathbf{Drop}_k^{(j)}$. The region where such states can be "picked-up" by adjacent local solutions is identified by $\mathbf{Pick}_k^{(j)}$; for compatibility between local solutions, $\mathbf{Pick}_k^{(j)}$ is required to contain $\mathbf{Drop}_k^{(j)}$. If this is not the case, local goals are suitably reduced.

Suppose Algorithm 1 terminates with success. Then, the nominal closed-loop system H and initial states *Good* are defined as:

- $Q := \{\, (k,i,c) \in K \times \mathbb{N} \times C \mid i \in I_k \text{ and } [\![\mathbf{Fine}_{k,i,c}]\!]^{\mathfrak{M}_0} \neq \varnothing \,\}$
- $F_{(k,i,c)} := F_c$ for all $(k,i,c) \in Q$
- for each $q = (k,i,c) \in Q$, set $Inv_q := [\![\langle \boldsymbol{\delta_1} \rangle \mathbf{A}_{k,i}]\!]^{\mathfrak{M}_0} \cap int\big([\![\neg \langle \boldsymbol{\delta_1} \rangle (\mathbf{A}_{k,i} \wedge \mathbf{Goal}_{k,i})]\!]^{\mathfrak{M}_0}\big)$
- $E := \{\, ((k,i,c),(k',i',c')) \in Q \times Q \mid k' \in next(k) \;\text{ or }\; (\, k' = k \;\text{ and }\; i' < i)\}$
- for each $(q,q') = ((k,i,c),(k',i',c')) \in E$ set $Grd_{q,q'} := Inv_q \cap [\![\langle \mathbf{2}\boldsymbol{\delta_1} \rangle (\mathbf{A}_{k,i} \wedge \mathbf{Goal}_{k,i}) \wedge \mathbf{Fine}_{k',i',c'}]\!]^{\mathfrak{M}_0}$
- $r_{q,q'} := test.Grd_{q,q'}$.
- $Good = \cup_{(k,i,c)\in Q}(Inv_{(k,i,c)} \cap [\![\mathbf{Fine}_{k,i,c}]\!]^{\mathfrak{M}_0})$

6 Correctness and Robustness of Synthesis Procedure

Theorem 1. *Let $F_c^v : X \to \mathbb{R}^n$, $c \in C$, $v \in V \subseteq \mathbb{R}^m$ be a finite family of parameter dependent vector fields, where the nominal case is denoted by $F_c = F_c^{\mathbf{0}}$, $c \in C$. For given specification data $Bad \subset X$, $\{E_k\}_{k\in K}$, $next \subseteq K \times K$ and $\delta > 0$, subject to assumptions* **(A0)**–**(A5)**, *suppose Algorithm 1 terminates with success, and H is the nominal closed-loop hybrid automaton as above. Then there exists a parameter bound $\varepsilon > 0$ such that for every H^v in the variation class $\mathcal{H}^\varepsilon$, the pair $(H^v, Good)$ satisfies each of the performance specifications* **(S1)**–**(S4)**.

The proof of Theorem 1 follows the same general line of argumentation as in [6]. In this outline, we focus on the extra challenges of the variation class $\mathcal{H}^\varepsilon$.

We begin by choosing a variation bound $\varepsilon > 0$ such that perturbed integral curves must remain within a δ_1-tube around the nominal curve. This is done by applying Proposition 1 for each $q = (k,i,c) \in Q$, with $D = cl([\![\mathbf{Fine}_q \wedge \mathbf{Inv}_q]\!]^{\mathfrak{M}_0})$

and a metric tolerance of $\frac{1}{2}\delta_1$. From the construction of $\mathbf{Success}_q$ (see Alg. 1, line 14) we conclude that any nominal curve starting in $\mathbf{Fine}_q$ leaves $\langle \boldsymbol{\delta}_1 \rangle \, \mathbf{A}_{k,i}$ via $\mathbf{A}_{k,i} \wedge \mathbf{Goal}_{k,i}$. Then, by the definition of Inv_q, each nominal curve starting in D leaves $B_{\delta_1}(D)$. The requirements of Proposition 1 are fulfilled and we get a variation bound $\varepsilon(q) > 0$ dependent on q. We choose $\varepsilon := \min\{\varepsilon(q) \mid q \in Q\}$ as a witness of the bound claimed by Theorem 1. In what follows, fix an arbitrary $H^v \in \mathcal{H}^\varepsilon$.

The high level strategy is to identify a list of modal logic formulas whose truth in $\mathfrak{M}_0$ provides sufficient conditions for the specifications **(S1)**–**(S4)** to be satisfied by any H^v in $\mathcal{H}^\varepsilon$. The crucial modal formulas **(T1)**–**(T4)** are analogs of those in [6], and are required for each $q = (k, i, c) \in Q$, $(q, q') \in E$ and for the perturbed flow relations $e_q^v := e(Inv_q, \Phi_c^v)$, for $q \in Q$.

$$
\begin{array}{lrcl}
\textbf{(T1)} & \big(\mathbf{Inv}_q \wedge \mathbf{Fine}_q \big) & \rightarrow & [\mathbf{e}_q^v] \, \langle \boldsymbol{\delta}_1 \rangle \, \mathbf{Fine}_q \\
\textbf{(T2)} & \big(\mathbf{Inv}_q \wedge \mathbf{Fine}_q \big) & \rightarrow & [\mathbf{test}.\mathbf{Grd}_{q,q'}] \, \mathbf{Fine}_{q'} \\
\textbf{(T3)} & \big(\mathbf{Inv}_q \wedge \mathbf{Fine}_q \big) & \rightarrow & [\mathbf{e}_q^v] \, \langle \mathbf{e}_q^v \rangle \, \big(\bigvee_{q' \in E(q)} \mathbf{Grd}_{q,q'} \big) \\
\textbf{(T4)} & \big(\mathbf{Inv}_q \wedge \mathbf{Fine}_q \big) & \rightarrow & \langle \mathbf{f}_c^v \rangle \, \neg \mathbf{Inv}_q
\end{array}
$$

In [6], the corresponding formulas are derived directly from the statement of the algorithm together with the explicit assumptions. Here, we are proving correctness of a variant H^v and therefore need to exploit the relationship between the perturbed modal operators and their nominal counterparts in which the algorithm is formalised; that is, $\langle \mathbf{e}_q^v \rangle$ and its relationship to $\langle \mathbf{e}_q \rangle$.

From Proposition 1 and our choice of ε, we can derive the following relational inclusion:

$$
test.Inv_q \circ test.Fine_q \circ e_q^v \;\subseteq\; e_q \circ B_{\delta_1} \,, \tag{16}
$$

where $\circ$ is relational composition, which we write in left-to-right word order. This in turn implies the truth in $\mathfrak{M}_0$ of the formula:

$$
\big(\mathbf{Inv}_q \wedge \mathbf{Fine}_q \wedge [\mathbf{e}_q][\boldsymbol{\delta}_1] \, \varphi \big) \;\rightarrow\; [\mathbf{e}_q^v] \, \varphi \tag{17}
$$

for any $\varphi \in \mathcal{L}(\mathrm{Rel}_0, \mathrm{Prp}_0)$. Then, **(T1)** can be deduced from $\mathbf{Fine}_q \rightarrow [\mathbf{e}_q]\mathbf{Fine}_q$ (see [6], Lemma 7.2) together with formula (17), while **(T2)** is an immediate consequence of the definitions. Formulas **(T1)** and **(T2)** are used to establish the safety specification.

From Proposition 1 and assumption **(A0)**, we can derive the more sophisticated modal fact:

$$
\big(\mathbf{Inv}_q \wedge \mathbf{Fine}_q \big) \;\rightarrow\; \big([\mathbf{e}_q^v] \, \langle \mathbf{e}_q^v \rangle \, \langle \mathbf{2}\boldsymbol{\delta}_1 \rangle \, \mathbf{Goal}_q \;\wedge\; \langle \mathbf{f}_c^v \rangle \, \neg \mathbf{Inv}_q \big) \,. \tag{18}
$$

Formula (18) expresses the essential properties of the construct $\mathbf{Success}_q$ (Alg. 1, line 14) but now referring to the perturbed relations e_q^v rather than the nominal e_q. In particular, we use (18) to deduce **(T3)**, and **(T4)** is an immediate consequence. Formulas **(T3)** and **(T4)** are used to verify step-infinite liveness and the event sequence specification.

Having deduced the modal conditions **(T1)**–**(T4)**, from this point on, we can largely mimic the proof in [6] to establish that H^v and *Good* satisfy each of the specifications **(S1)**–**(S4)**.

7 Discussion and Conclusion

This paper addresses a basic hybrid control problem, namely the design of a switching control mechanism via guard and invariant sets. We use a novel methodology based on modal logic to solve this problem for a significant list of performance specifications, and we do so in a manner that is robust w.r.t. parameter uncertainty in the differential equations.

A significant issue to be investigated in future work is the question of *completeness* of the algorithm; i.e. whether there exists a parameterised plant and specification data such that there is a robust solution to the control problem but the algorithm terminates with failure due to *next* incompatibility. In general one may expect such incompleteness to occur. So the question arises as to what additional conditions on the input data could ensure completeness. A full treatment of this issue necessitates the development of more mathematical tools for analysing the space of all possible solutions to our control problem, leading to an appropriate notion of switching controllability.

As discussed in the introduction, our synthesis algorithm can be implemented on any available model checking tool. There are two main approaches: *exact symbolic computation*, representing sets of states by first-order logic formulas (e.g. [1, 2, 14, 15]), and *approximated representation*, whereby sets are under- or over-approximated as finite unions of cells (e.g. [4, 13, 16]). We have developed a prototype software implementation of our synthesis algorithm based on an approximation using boxes generated by a regular grid, and it is applicable to arbitrary linear differential equations. The software runs on a massively parallel cluster effectively employing 96 CPUs, and has been tested on several non-trivial examples. This work on approximation based model checking is to be presented in a separate paper.

Acknowledgements: We thank our colleagues for useful discussions: Brian Anderson, Thomas Brinsmead, Raj Goré, Gerardo Lafferriere, George Pappas, and Matthew Smith.

References

1. R. Alur, C. Courcoubetis, N. Halbwachs, T.A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138:3–34, 1995.
2. R. Alur, T.A. Henzinger, G. Lafferriere, and G. Pappas. Discrete abstractions of hybrid systems. *Proceedings of the IEEE*, 88:971–984, July 2000.

3. B.D.O. Anderson, F. de Bruyne, S. Dey, and K. Wong. Ensuring robustness in hybrid control systems. Technical report, Dept. Systems Engineering, RSISE, Australian National University, April 1999.
4. E. Asarin, O. Bournez, T. Dang, and O. Maler. Approximate reachability analysis of piecewise-linear dynamical systems. In N. Lynch and B. Krogh, editors, *Hybrid Systems: Computation and Control (HSCC'00)*, LNCS 1790, pages 20–31. Springer-Verlag, 2000.
5. J.E.R. Cury, B.A. Krogh, and T. Niinomi. Synthesis of supervisory controllers for hybrid systems based on approximating automata. *IEEE Transactions on Automatic Control*, 43:564–568, 1998.
6. J.M. Davoren and T. Moor. Logic-based design and synthesis of controllers for hybrid systems. Technical report, Dept. Systems Engineering, RSISE, Australian National University, July 2000. `http://arp.anu.edu.au/~davoren/hybrid_control/hybrid_control.html`, submitted for publication.
7. J.M. Davoren and A. Nerode. Logics for hybrid systems. *Proceedings of the IEEE*, 88:985–1010, July 2000.
8. E. Frazzoli, M.A. Dahleh, and E. Feron. Robust hybrid control for autonomous vehicle motion planning. Technical report, LIDS-P-2468, Laboratory for Information and Decision Systems (LIDS), Massachusetts Institute of Technology, May 2000.
9. R. Goré. Tableaux methods for modal and temporal logics. In M. D'Agostino *et al.*, editors, *Handbook of Tableaux Methods*, pages 297–396. Kluwer, 1999.
10. C. Horn and P.J. Ramadge. Robustness issues for hybrid systems. In *Proceedings of the 34th International Conference on Decision and Control, CDC'95*, pages 1467–1472. IEEE Press, 1995.
11. H.K. Khalil. *Nonlinear Systems*. Prentice-Hall, 1996. Second edition.
12. X. Koutsoukos, P.J. Antsaklis, J.A. Stiver, and M.D. Lemmon. Supervisory control of hybrid systems. *Proceedings of the IEEE*, 88:1026–1049, July 2000.
13. A.B. Kurzhanski and P. Varaiya. Ellipsoidal techniques for reachability analysis. In N. Lynch and B. Krogh, editors, *Hybrid Systems: Computation and Control (HSCC'00)*, LNCS 1790, pages 202–214. Springer-Verlag, 2000.
14. G. Lafferriere, G.J. Pappas, and S. Sastry. O-minimal hybrid systems. *Mathematics of Control, Signals, and Systems*, 13:1–21, 2000.
15. G. Lafferriere, G.J. Pappas, and S. Yovine. A new class of decidable hybrid systems. In F.W. Vaandrager and J.H. van Schuppen, editors, *Hybrid Systems: Computation and Control (HSCC'99)*, LNCS 1569, pages 137–151. Springer-Verlag, 1999.
16. T. Moor and J. Raisch. Discrete control of switched linear systems. In *Proceedings of the European Control Conference 1999*, 1999.
17. T. Moor and J. Raisch. Supervisory control of hybrid systems within a behavioural framework. *Systems and Control Letters*, 38:157–166, 1999.
18. C. Stirling. Modal and temporal logics. In S. Abramsky, D.M. Gabbay, and T. Maibaum, editors, *Handbook of Logic in Computer Science*, volume 2, pages 477–563. Oxford University Press, Clarendon Press, Oxford, 1992.
19. C. Tomlin, J. Lygeros, and S. Sastry. A game-theoretic approach to controller design for hybrid systems. *Proceedings of the IEEE*, 88:949–970, July 2000.
20. F. Wolter. *Decision Problems for Combined Modal Logics*. Habilitationsschrift. Institut für Informatik, Universität Leipzig, 1999.
21. C.A. Yfoulis, A. Muir, P.E. Wellstead, and N.B.O.L. Pettit. Stabilization of orthogonal piecewise linear systems: Robustness analysis and design. In F.W. Vaandrager and J.H. van Schuppen, editors, *Hybrid Systems: Computation and Control (HSCC'99)*, LNCS 1569, pages 256–270. Springer-Verlag, 1999.