Stability of Hybrid Automata with Average Dwell Time: An Invariant Approach

Sayan Mitra*
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
200 Technology Square
Cambridge, MA 02139, USA
mitras@csail.mit.edu

Daniel Liberzon**
Coordinated Science Laboratory
University of Illinois at Urbana-Champaign,
1308 W. Main Street
Urbana, IL 61801, USA
liberzon@uiuc.edu

***Abstract*— A formal method based technique is presented for proving the average dwell time property of a hybrid system, which is useful for establishing stability under slow switching. The Hybrid Input/Output Automaton (HIOA) framework of [12] is used as the model for hybrid systems, and it is shown that some known stability theorems from system theory can be adapted to be applied in this framework. The average dwell time property of a given automaton, is formalized as an invariant of a corresponding transformed automaton, such that the former has average dwell time if and only if the latter satisfies the invariant. Formal verification techniques can be used to check this invariance property. In particular, the HIOA framework facilitates invariant proofs by breaking them down into a systematic case analysis for the discrete actions and continuous trajectories. The invariant approach to proving the average dwell time property is illustrated by analyzing the hysteresis switching logic unit of a supervisory control system.**

***Index Terms*— Average dwell time, Stability, Formal methods, Invariant, Hybrid systems, Hybrid I/O automaton.**

I. Introduction

Systems with both discrete and continuous dynamics are called hybrid systems. Computer scientists have concentrated on verification of hybrid systems, and have developed a wide range of techniques for proving saftey properties, from model checking [1], [6], [12] which is automatic but limited to moderate sized linear hybrid systems, to interactive theorem proving [2], [5] which is applicable to larger and more complicated hybrid systems. Control theorists, on the other hand, have viewed hybrid systems as switched systems or as dynamical systems with special boolean variables, and have addressed stability, controllability, and controller synthesis of such systems [18], [10]. The differences in these approaches espoused different terminologies and mathematical models, which has led to a lack of interaction between the two communities and isolated developments.

A platform bridging the gap by allowing both computer scientists and control theorists to apply their techniques in the same modeling framework is desirable. To this end, we introduce the Hybrid Input/Output Automaton (HIOA) [12] to the Control Systems community. HIOA is a mathematical model for developing compositional specifications for a very general class of hybrid systems and it subsumes the class of untimed and timed distributed systems. Hybrid behavior is modeled as an alternating sequence of actions and trajectories; the actions correspond to discrete state transitions and the trajectories capture continuous evolution of the state variables of an automaton. Most of the prior work with HIOA focused on verifying safety of hybrid systems [16], [11], [4]. Owing to this special structure of the HIOA, its safety properties, which are actually its invariants, can be proved inductively by a systematic case analysis of its actions and trajectories.

In this paper we demonstrate how formal methods and the HIOA framework can be useful for proving invariants arising in stability analysis of hybrid systems. First, we show the straightforward adaptation of some known stability theorems from system theory to the HIOA framework. Then, we show that the task of proving the average dwell time property [7] which is used to prove stability of hybrid systems under slow switching, can be reduced to checking a set of invariants. We have chosen the average dwell time property to demonstrate the invariant approach because it decouples the problem of finding the Lyapunov functions (which we assume are given), from the problem of checking that all the executions of the HIOA satisfy certain properties. In general, properties of the executions of an automaton, are harder to prove than invariant properties which are properties of the state. We transform the given HIOA $\mathcal{A}$, to a new HIOA $\mathcal{A}'$ and find a condition $\mathcal{I}$ on the states of $\mathcal{A}'$, such that $\mathcal{A}$ satisfies the average dwell time property if and only if $\mathcal{I}$ is an invariant of $\mathcal{A}'$. This enables us to prove the average dwell time property by checking $\mathcal{I}$ with a suitable formal verification technique. We illustrate our approach by analyzing the stability of the hysteresis switching logic unit in a supervisory control system. In this case study we have proved the invariants by hand, however our long term goal is to develop an integrated system which uses automatic theorem provers to efficiently verify the invariants arising in stability analysis of hybrid systems.

The rest of this paper is organized as follows: In Section II we describe the HIOA model, in Section III we define

*Research supported by AFRL contract number F33615-010C-1850
**Supported by the MURI project:DARPA/AFOSR MURI F49620-02-1-0325 Award

the various notions of stability and restate some known stability theorems in the HIOA framework. In Section IV we define the aforementioned transformations and proceed to formalize the average dwell time property as a set of invariants. In Section V we present the analysis of the hysteresis switching unit of a supervisory control system using our invariant approach. In Section VI we conclude with a synopsis of contributions and future research directions.

II. Mathematical Preliminaries

The hybrid I/O automaton framework [12] evolved from the generalization of the timed I/O automaton model [9] for real time distributed systems. Earlier versions of the model appeared in [13] and [14]. A hybrid I/O automaton models hybrid behavior in terms of discrete transitions and continuous evolution of its state variables.

Let V be the set of variables of automaton $\mathcal{A}$. Each $v \in V$ is associated with a *(static) type* which is the set of values v can assume. A valuation $\mathbf{v}$ for V is a function that associates each variable $v \in V$ to a value in $type(v)$. The set of all valuations of V is denoted by $val(V)$.

A trajectory τ of V is a mapping $\tau : J \rightarrow val(V)$, where J is a left closed interval of time. The domain of τ is the interval J and is denoted by $\tau.dom$. The first time of τ is the infimum of $\tau.dom$, also written as $\tau.ftime$. If $\tau.dom$ is right closed then τ is closed and its limit time is the supremum of $\tau.dom$, also written as $\tau.ltime$.

Each variable $v \in V$ is also associated with a $dynamic$ $type$ (or $dtype$) which is the set of trajectories that v may follow. Dynamic type $dtype(v)$ of a *continuous* (*discrete*) variable v is the pasting closure of continuous (constant) functions from left closed intervals of time to $type(v)$.

A. HIOA Model

A hybrid I/O automaton $\mathcal{A}$ consists of :

1) A set V of variables, partitioned into *internal* X, *input* U, and *output variables* Y. The internal variables are also called *state variables*. The set $W = U \cup Y$ is the set of *external variables*. And, the set $Z \triangleq X \cup Y$ is called the set of *locally controlled or local variables*.
2) A set A of actions , partitioned into *internal* H, *input* I, and *output actions* O.
3) A set of states $Q \subseteq val(X)$,
4) A non-empty set of *start states* $\Theta \subseteq Q$,
5) A set of *discrete transitions* $\mathcal{D} \subseteq Q \times A \times Q$. A transition $(\mathbf{x}, a, \mathbf{x}') \in \mathcal{D}$ is written in short as $\mathbf{x} \stackrel{a}{\rightarrow}_{\mathcal{A}} \mathbf{x}'$. The subscript is sometimes omitted and written as $\mathbf{x} \stackrel{a}{\rightarrow} \mathbf{x}'$ when the automaton $\mathcal{A}$ is clear from the context.
6) A set of *trajectories* $\mathcal{T}$ for V, such that for every trajectory τ in $\mathcal{T}$, and for every $t \in \tau.dom$, $\tau(t).X \in Q$ and $\mathcal{T}$ is closed under prefix, suffix, and concatenation. The first state $\tau(0).X$ of trajectory is denoted by $\tau.fstate$. If $\tau.dom$ is finite then $\tau.lstate = \tau(\tau.ltime).X$.

Further, $\mathcal{A}$ is: (1) *input action enabled*, that is, it cannot block input actions, and (2) *input trajectory enabled*, that is, it accepts any trajectory of the input variables either by allowing time to progress for the entire length of the trajectory or by reacting with some internal action before that.

For this paper we assume that (1) All variables are either discrete or continuous. For a set of variables S, we denote its discrete and continuous subsets by S_d and S_c, and the the corresponding state vectors by $\mathbf{s}_d$ and $\mathbf{s}_c$. And, (2) discrete transitions *do not* change the valuation of the continuous variables, that is, if $\mathbf{x} \stackrel{a}{\rightarrow} \mathbf{x}'$, then $\mathbf{x}.\mathbf{x}_c = \mathbf{x}'.\mathbf{x}_c$.

B. Executions

An *execution fragment* of $\mathcal{A}$ is a finite or infinite sequence of actions and trajectories $\alpha = \tau_0, a_1, \tau_1, a_2 \ldots$, where each $\tau_i \in \mathcal{T}, a_i \in A$, and if τ_i is not the last trajectory in α then τ_i is finite and $\tau_i.lstate \stackrel{a_{i+1}}{\rightarrow} \tau_{i+1}.fstate$. If α is an execution fragment, then we define the first state of α, $\alpha.fstate$ to be $\tau_0.fstate$. An execution fragment α is an *execution* if $\alpha.fstate \in \Theta$. An execution fragment is *closed* if it is a finite sequence and the domain of the final trajectory is a finite closed interval. We say a state of $\mathcal{A}$ is *reachable* if it is the last state of some closed execution of $\mathcal{A}$. An execution fragment α is *reachable* if $\alpha.fstate$ is reachable.

The length of a closed execution fragment is the number of elements (actions and trajectories) in the sequence. The first time $\alpha.ftime$ of an execution fragment α is $\tau_0.ftime$, and if α is closed then its limit time $\alpha.ltime$ is $\tau_n.ltime$, where τ_n is the last trajectory of α. The *duration* of a closed execution fragment is its length in time and is defined as $\alpha.dur = \sum_{i=0}^{n}(\tau_i.ltime - \tau_i.ftime)$. We denote the valuation of the continuous variables X_c at time $t, \alpha.ftime \leq t \leq \alpha.ftime$, in the execution fragment α by $\alpha(t)$. Note that $\alpha(t)$ is uniquely determined because the discrete actions do not alter the valuation of the continuous variables.

C. Invariants

An *invariant property* or simply an *invariant* of $\mathcal{A}$ is a condition on V that remains true in all reachable states of $\mathcal{A}$. The structure of HIOA allows systematic proof of invariants. An invariant $\mathcal{I}$ is either derived from other invariants or proved by induction on the length of a closed execution of $\mathcal{A}$. The induction consists of the following steps:

1) **base step**: $\mathcal{I}(s)$ is true for all $s \in \Theta$,
2) **induction step**: consisting of
    a) discrete part: for every discrete transition $s \stackrel{\pi}{\rightarrow} s'$, $\mathcal{I}(s)$ implies $\mathcal{I}(s')$, and
    b) continuous part: for any closed trajectory $\tau \in \mathcal{T}$, with $\tau.fstate = s$ and $\tau.lstate = s'$, $\mathcal{I}(s)$ implies $\mathcal{I}(s')$.

So, the inductive proof of an invariant breaks down into a set of cases, one for each action and trajectory. This is particularly helpful in organizing large, complex proofs and for automating invariant proofs in a theorem prover.

III. Stability Theorems in HIOA Framework

In this section we define what it means for a HIOA $\mathcal{A}$ to be stable. In this and the following section, we are concerned with hybrid systems with no continuous inputs, and we assume that there exists a family of sufficiently regular[1] functions $f_p : \mathsf{R}^n \to \mathsf{R}^n$, $p \in \mathcal{P}$, such that every trajectory of $\mathcal{A}$ satisfies $\dot{\mathbf{x}}_c = f_p(\mathbf{x}_c)$ for some $p \in \mathcal{P}$, where $\mathcal{P}$ is a finite index set.

A. Stability Definitions

Let us assume that all the subsystems of $\mathcal{A}$ have the origin their the common equilibrium point, that is, $f_p(0) = 0$ for all $p \in \mathcal{P}$. The origin is a *stable* equilibrium point of a HIOA $\mathcal{A}$, in the sense of Lyapunov, if for every $\epsilon > 0$, there exists a $\delta > 0$, such that for every execution α of $\mathcal{A}$, we have

$$|\alpha(0)| \leq \delta \Rightarrow |\alpha(t)| \leq \epsilon \quad \forall t \quad 0 \leq t \leq \alpha.ltime, \tag{1}$$

and we say that $\mathcal{A}$ is *stable*. A HIOA $\mathcal{A}$ is *asymptotically stable* if it is stable and δ can be chosen so that

$$|\alpha(0)| \leq \delta \Rightarrow \alpha(t) \to 0 \quad as \quad t \to \infty \tag{2}$$

If the above condition holds for all δ then $\mathcal{A}$ is *globally asymptotically stable*.

Uniform stability is a concept which guarantees that the stability property in question holds, not just for executions, but for any execution fragment. Therefore, $\mathcal{A}$ is uniformly stable in the sense of Lyapunov, if for every $\epsilon > 0$ there exists a constant $\delta > 0$, such that for any execution fragment α,

$$|\alpha(t_0)| \leq \delta \Rightarrow |\alpha(t)| \leq \epsilon, \forall t_0, t, \ 0 \leq t_0 \leq t \leq \alpha.ltime$$

A HIOA $\mathcal{A}$ is said to be uniformly asymptotically stable if it is uniformly stable and there exists a $\delta > 0$, such that for every $\epsilon > 0$ there exists a T, such that for any execution fragment α,

$$|\alpha(t_0)| \leq \delta \Rightarrow |\alpha(t)| \leq \epsilon, \quad \forall t \geq t_0 + T \tag{3}$$

It is said to be *globally uniformly asymptotically stable* if the above holds for all δ.

All the above stability properties are by definition uniform over executions. We will also make use of the following weaker notion of stability: a given execution is stable (uniformly stable, asymptotically stable, etc.) if the corresponding property is satisfied for this execution.

In the remaining part of this section we show how some known theorems on stability (see, e.g., [10]), can be adapted for the stability analysis of HIOA.

B. Common Lyapunov Function

The basic tool for studying stability of hybrid systems relies on the existence of a single Lyapunov function whose derivative along the trajectories of all the subsystems in P satisfies the suitable inequalities.

[1] Locally Lipschitz

Definition 1. *Given a positive definite continuously differentiable function $V : \mathsf{R}^n \to \mathsf{R}^n$, we say that it is a common Lyapunov function for a HIOA $\mathcal{A}$ if there exists a positive definite continuous function $W : \mathsf{R}^n \to \mathsf{R}^n$, such that we have*

$$\frac{\partial V}{\partial \mathbf{x}_c} f_p(\mathbf{x}_c) \leq -W(\mathbf{x}_c) \quad \forall \mathbf{x}_c, \quad \forall p \in \mathcal{P} \tag{4}$$

Theorem 1. *If a HIOA $\mathcal{A}$ has a radially unbounded common Lyapunov function then $\mathcal{A}$ is globally uniformly asymptotically stable.*

C. Multiple Lyapunov Functions

When a common Lyapunov function for all the subsystems in $\mathcal{P}$ is not known or does not exist, the stability of a HIOA depends on the choice of an execution. Multiple Lyapunov functions [3] is an useful tool for proving stability of a chosen execution. In this case, each subsystem $p \in \mathcal{P}$ is associated with a Lyapunov function V_p, and one attempts to prove the stability of the execution using the continuous decay of the V_p's and the switching logic between the subsystems. In control theory literature [10], [7] the switches between the subsystems $p \in \mathcal{P}$ are defined in terms of a "switching signal" which is a piece-wise constant function $\sigma : [0, \infty) \to \mathcal{P}$. In the HIOA model the switches are defined by the discrete transitions of the automaton, so we define the notion of switching times as follows:

Let $M : \mathcal{T} \to \mathcal{P}$ be a function that gives the index p of the function f_p, which is active over the trajectory τ. Whenever a discrete action a_i occurs such that $M(\tau_{i-1}) \neq M(\tau_i)$, the HIOA $\mathcal{A}$ is said to undergo a *switch*.

Definition 2. *For any execution fragment $\alpha = \tau_0 a_1 \tau_1 \ldots$, an instant of time $t \in \alpha.dom$ is called a switching time if there exists i such that $t = \tau_i.ltime$, and $M(\tau_i) \neq M(\tau_{i+1})$.*

Theorem 2. *Let V_p be a radially unbounded Lyapunov function corresponding to the globally asymptotically stable system $\dot{x} = f_p(x)$ for each $p \in \mathcal{P}$. An execution α of a HIOA $\mathcal{A}$ is globally asymptotically stable if there exists a family of positive definite continuous functions $W_p, p \in \mathcal{P}$ such that, for every pair of successive switching times t, t' in α, and the corresponding trajectories τ_i, τ_j, if $M(\tau_i) = M(\tau_j) = p$ and $M(\tau_k) \neq p, \forall k, \ i < k < j$ then $V_p(\tau_j(t')) - V_p(\tau_i(t)) \leq -W_p(\tau_i(t))$.*

D. Stability Under Slow Switching

It is well known that a switched system is stable if all the individual subsystems are stable and the switching is sufficiently slow, so as to allow the dissipation of the transient effects after each switch. The *dwell time* [17] and the *average dwell time* [7] criteria define restricted classes of switching signals, based on switching speeds, and one can conclude the stability of a system with respect to these restricted classes. In the HIOA framework, the discrete transitions of the automaton define the speed of switching, and so the average dwell time property can be formalized as an invariant of the automaton.

Definition 3. *Let $t_1, t_2, \ldots$ be the switching times of an execution fragment α of a HIOA $\mathcal{A}$. The execution fragment α has a dwell time $\tau_d > 0$ if it satisfies the inequality $t_{i+1} - t_i \geq \tau_d$, for all i. If all reachable execution fragments of $\mathcal{A}$ have dwell times $\geq \tau_d$ then $\mathcal{A}$ has a dwell time τ_d.*

Definition 4. *Let $N(\alpha)$ denote the number of switches over an execution fragment α of a HIOA $\mathcal{A}$. The execution fragment has an average dwell time $\tau_a > 0$ if there exists a positive number N_0 such that:*

$$N(\alpha) \leq N_0 + \frac{\alpha.dur}{\tau_a}. \tag{5}$$

If all reachable execution fragments of $\mathcal{A}$ have average dwell times $\geq \tau_a$ then $\mathcal{A}$ has an average dwell time τ_a.

The following theorem, adapted to the HIOA framework from the results in [7], uses the concept of average dwell time to give a sufficient condition for stability. Since dwell time is a special case of average dwell time with $N_0 = 1$, a separate theorem for dwell time is not necessary.

Theorem 3. *Consider a HIOA $\mathcal{A}$ with its trajectories specified by a family of functions $f_p, p \in \mathcal{P}$. Suppose there exist positive definite, radially unbounded, and continuously differentiable functions $V_p : \mathsf{R}^n \to \mathsf{R}^n$, for each $p \in \mathcal{P}$, and positive numbers λ_0 and μ such that:*

$$\frac{\partial V_p}{\partial \mathbf{x}_c} f_p(\mathbf{x}_c) \leq -\lambda_0 V_p(\mathbf{x}_c), \quad \forall \mathbf{x}_c, \quad \forall p \in \mathcal{P} \tag{6}$$

$$V_p(\mathbf{x}_c) \leq \mu V_q(\mathbf{x}_c), \quad \forall \mathbf{x}_c, \quad \forall p, q \in \mathcal{P}. \tag{7}$$

Then $\mathcal{A}$ is globally uniformly asymptotically stable if it has an average dwell time $\tau_a > \frac{\log \mu}{2\lambda_0}$.

Theorem 3 roughly states that a hybrid system is uniformly stable if the discrete switches are between modes which are individually stable, provided that the switches do not occur *too frequently on the average*. This stability condition effectively allows us to decouple the construction of Lyapunov functions—one for each $p \in \mathcal{P}$, which we assume are known from available methods of system theory—from the problem of checking that every execution of the automaton satisfies Equation (5). In the next section we shall formalize this latter property as an invariant of a corresponding transformed automaton and show how formal methods can be used to verify it.

IV. Average Dwell Time: Invariant Approach

In general, it is harder to prove properties of executions of automata than it is to prove invariants, which are properties of state. Several formal verification techniques have been developed expressly for checking invariants of hybrid automata (see [1], [6], [5], and Chapters 5 and 6 of [18]). So, once we have translated the average dwell time property to a set of invariant properties, we can appeal to the suitable formal verification tool for checking the invariants.

A. Transformed Automaton for Stability Verification

We transform the given HIOA $\mathcal{A}$ to a new HIOA $\mathcal{A}'$ as follows: In addition to all the variables of $\mathcal{A}$, automaton $\mathcal{A}'$ has two new internal variables; a counter Q and a timer t, both initialized 0. The counter Q counts the number of mode switches, and the timer reduces the count by 1 in every τ_a time. For every discrete transition $s \xrightarrow{a}_{\mathcal{A}} s'$ of $\mathcal{A}$, automaton $\mathcal{A}'$ has a corresponding transition $s \xrightarrow{a}_{\mathcal{A}'} s'$, such that $s'.Q = s.Q + 1$. In addition $\mathcal{A}'$ has internal action which occurs every τ_a time and decrements Q by one. Finally, for every trajectory τ' of $\mathcal{A}'$, the projection of τ on the set of continuous variables of $\mathcal{A}$ is a trajectory of $\mathcal{A}$, i.e., $\tau' \downarrow Z_c \in \mathcal{T}_{\mathcal{A}}$, and $\dot{t} = 1$.

Lemma 1. *All closed executions of $\mathcal{A}$ satisfy Equation (5) if and only if $Q \leq N_0$ in all reachable states of $\mathcal{A}'$.*

Proof. Since α is a closed execution of $\mathcal{A}$, we can replace $\alpha.dur$ in Equation (5) with $\alpha.ltime$. For the "if" part, consider a closed execution α of $\mathcal{A}$ and let α' be the "corresponding" execution of $\mathcal{A}'$. Let s' be the last state of α, therefore from the invariant we know that $s'.Q \leq N_0$. From construction of $\mathcal{A}'$ we know that, $N(\alpha) = N(\alpha')$ and $\alpha'.ltime = \alpha.ltime$ and therefore $s'.Q = N(\alpha') - \lfloor \frac{\alpha'.ltime}{\tau_a} \rfloor$. It follows that $N(\alpha) - \frac{\alpha.ltime}{\tau_a} \leq N_0$.

For the "only if" part, consider a reachable state s' of $\mathcal{A}'$. There exists an execution α' such that s' is the last state of α'. Let α be an execution of $\mathcal{A}$ "corresponding" to α'. Since $N(\alpha) \leq N_0 + \lfloor \frac{\alpha.time}{\tau_a} \rfloor$ implies $N(\alpha') \leq N_0 + \lfloor \frac{\alpha'.ltime}{\tau_a} \rfloor$, it follows that $s'.Q \leq N_0$. ☐

Theorem 4. *All executions of $\mathcal{A}$ satisfy Equation (5) if and only if $Q \leq N_0$ in all reachable states of $\mathcal{A}'$.*

Proof. We only have to show that if any execution α of $\mathcal{A}$ violates (5), then there exists a closed execution α' of $\mathcal{A}$ that violates (5) as well. If α is infinite, then there is a closed prefix of α that violates (5). Otherwise, α is finite and open, and the closed prefix of α excluding the last trajectory of α violates (5). ☐

B. Transformed Automaton for Uniform Stability Verification

The above transformation is acceptable for asymptotic stability, but it allows Q to become negative, and then rapidly return to zero, so it does not guarantee uniform stability. For uniform stability we want all reachable execution fragments of $\mathcal{A}$ to satisfy (5).

Consider any reachable execution fragment α of $\mathcal{A}$, with $\alpha.ftime = t_1$, and $\alpha.ltime = t_2$. Let $N(t_2, t_1)$ and $Q(t_2, t_1)$ denote the number of switches and the number of "extra" switches over α with respect to dwell time τ_a, that is, $Q(t_2, t_1) = N(t_2, t_1) - (t_2 - t_1)/\tau_a$. Thus, every reachable execution fragment α of $\mathcal{A}$ satisfies (5), if

$$\begin{aligned} N(t, t_0) = Q(t, 0) + \frac{t}{\tau_a} - Q(t_0, 0) - \frac{t_0}{\tau_a} \quad &\leq \quad N_0 + \frac{t - t_0}{\tau_a} \\ \text{that is,} \quad Q(t, t_0) \quad &\leq \quad N_0, \end{aligned}$$

where $t_0 = \alpha.ftime$, and $t = \alpha.ltime$. So, we introduce an additional variable Q_{min} which stores the magnitude of the smallest value ever attained by Q. Then, for uniform stability we need to show that the total change in Q between any two reachable states is bounded by N_0.

Theorem 5. *All reachable execution fragments of $\mathcal{A}$ satisfy Equation (5), if and only if $Q - Q_{min} \leq N_0$ in all reachable states of $\mathcal{A}'$.*

Proof. For the "if" part, consider a reachable closed execution fragment α of $\mathcal{A}$ which is a part of the execution β, such that $\alpha.fstate = \beta(t_1)$ and $\alpha.lstate = \beta(t_2)$. Let α' and β' be the corresponding execution (fragment) of $\mathcal{A}'$. Based on whether or not Q_{min} changes over the interval $[t_1, t_2]$, we have the following two cases:

If Q_{min} does not change in the interval, then $\beta'(t_1).Q_{min} = \beta'(t_2).Q_{min} = \beta'(t).Q$ for some $t_{min} < t_1$, and $Q(t_2, t_1) = Q(t_2, t_{min}) - Q(t_1, t_{min}) \leq Q(t_2, t_{min})$. Since $\beta'(t_2)$ satisfies the invariant, $Q(t_2, t_{min}) = \beta'(t_2).Q - \beta'(t_2).Q_{min} \leq N_0$ from which we get $Q(t_2, t_1) \leq N_0$.

Otherwise, there exists some $t_{min} \in [t_1, t_2]$, such that $\beta'(t_2).Q_{min} = \beta'(t_{min}).Q < \beta'(t_1).Q_{min}$, and $Q(t_2, t_1) = Q(t_2, t_{min}) + Q(t_{min}, t_1) \leq Q(t_2, t_{min})$. Again, from the invariant property at $\beta'(t_2)$, we get $Q(t_2, t_1) \leq Q(t_2, t_{min}) \leq N_0$.

For the "only if" part, let s' be a reachable state, and ξ' be a closed execution of $\mathcal{A}'$, such that $s' = \xi'.lstate$. Further, let ξ be the corresponding execution of $\mathcal{A}$, and t_0 be the intermediate point where Q attains its minimal value over ξ, that is, $\xi(t).Q_{min} = \xi(t_0).Q$. Since ξ is a reachable execution fragment of $\mathcal{A}$, it satisfies Equation (5), and we have: $N(t, t_0) \leq N_0 + \frac{t - t_0}{\tau_a}$. Rewriting,

$$Q(t, 0) + \frac{t}{\tau_a} - Q(t_0, 0) - \frac{t_0}{\tau_a} \leq N_0 + \frac{t - t_0}{\tau_a}$$

By assumption, $Q(t_0, 0) = \xi'(t).Q_{min} = s'.Q_{min}$, therefore, it follows that $s'.Q - s'.Q_{min} \leq N_0$. □

C. Finding the Required Invariants

For a general HIOA, there are no obvious ways of finding the correct invariants that would lead to the average dwell time property, however, based on some case studies, we provide the following guidelines.

First, for each mode p we identify a variable μ_p that is monotonic; μ_p and μ_q could be the same, for $p \neq q$. Then, we introduce discrete history variables μ_p^i, $i = 1, 2, \ldots$, which record the value of μ_p when the system switched out of mode p for the i^{th} time. Now, we want two invariants of the following forms:

1) $\mu_i^{k+1} \geq f(\mu_i^k)$, for all i, where f is some monotonic increasing function, and
2) $\mu_i^{k+1} \leq g(t)$, where g is some monotonic function of time.

Roughly speaking, the second invariant sets an upper bound on how fast μ_p *can* grow, and the first invariant sets a lower bound on how much μ_p *has* to grow in order to perform a certain number of switches. Together, they bound the minimum time that has to elapse between switches. The case study presented in the next section will illustrate these principles further.

V. Hysteresis Switching

In this section the invariant based technique is applied to a hysteresis switching logic unit which is a subsystem of an adaptive supervisory control system taken from [8] (also Chapter 6 of [10]). Our goal is to prove the average dwell time property of this switching logic, which guarantees stability of the overall supervisory control system (see the above references for details).

An adaptive supervisory controller consists of a family of candidate controllers $u_p, p \in \mathcal{P}$, which correspond to the parametric uncertainty range of the plant in a suitable way. Such a controller structure is particularly useful when the parametric uncertainty is so large that robust control design tools are not applicable. The controller operates in conjunction with a set of on-line estimators that provide *monitoring signals* or time-varying estimates of the unknown parameters of the plant model and at each instant of time, the switching logic unit generates, the index $\sigma(t)$ of the controller to be applied to the plant.

In building the HIOA model, we take as inputs the monitoring signals μ_p and focus on the switching logic unit which implements scale independent hysteresis switching as follows: at an instant of time when controller r is operating, that is, $\sigma = r$ for some $r \in \mathcal{P}$, if there exists a $p \in \mathcal{P}$ such that $\mu_p(1 + h) \leq \mu_r$ for some fixed hysteresis constant h, then the switching logic sets $\sigma = p$ and applies output of controller p to the plant. Below we describe and analyze the HIOA representing this switching logic unit, which we call `HysteresisSwitchingLogic` automaton.

We consider a finite set of continuous, monotonically nondecreasing *monitoring signals* μ_p, $p \in \mathcal{P}$ satisfying:

$$\mu_p(0) \geq C_0 \quad (8)$$

$$\mu_{p^*}(t) \leq C_1 + C_2 e^{2\lambda t}, \text{for some } p^* \in \mathcal{P} \quad (9)$$

where C_0, C_1 and C_2 are positive constants for each $p \in \mathcal{P}$. Equation (8) sets a lower bound on the initial values of all the monitoring signals, and Equation (9) states that there exists some $p^* \in \mathcal{P}$ for which the corresponding monitoring signal satisfies the exponential upper bound.

A. HIOA Specification

In specifying the hysteresis switch as a HIOA (Figure 1), we adopt the style described in [16]. The input, output, and state (internal) variables are declared and initialized in the **variables** section of the code. Each variable's type is listed, after a colon, following its name in the declaration. The variables declared with the preceding **analog** keyword are continuous, the rest are discrete. The monitoring signals μ_p, $p \in \mathcal{P}$, are declared as input variables because they are controlled by an external source. And, the discrete switching signal σ is an output variable because it is visible to the outside word; all other variables are internal and are not visible outside the automaton. The variables c and d count the number of switches and the number of τ_a

periods elapsed; together they define Q, the number of "extra switches". The history variables μ_p^i's store the values of μ_p at the instants when σ becomes equal to p for the i^{th} time. The variable c_p counts the number of intervals in which σ has been equal to p, and the real variable t_p is a reset timer measuring the length of this interval.

The **discrete transitions** section defines the two actions of the automaton, namely $dequeue$ and $switch_p, p \in \mathcal{P}$. An action is *enabled* or in other words, it *can* occur when the condition following the **precondition** keyword is true. The change in the state variables when the action does occur is described by the **effect** part of the transition definition.

The **trajectories** section defines the evolution of the continuous variables in terms of the differential and algebraic equations. The $d(.)$ in the **evolve** section stands for derivative. The stopping condition, in this automaton, is the disjunction of the action preconditions, so it forces the actions to occur whenever they are enabled.

B. Invariant Properties

In this section we prove a sequence of invariants which show that the executions of the `HysteresisSwitchingLogic` automaton satisfy the average dwell time property with $\tau_a = \frac{\log(1+h)}{2\lambda m}$. For simplicity of presentation, we prove the invariants required for asymptotic stability, and not uniform asymptotic stability, and accordingly the average dwell time property we get is over executions and not over execution fragments of the automaton. The invariants closely correspond to those outlined in Section IV-C. The first three invariants lead to Corollary 2, which corresponds to the first invariant of Section IV-C and gives the lower bound on the change in the history variables necessary to perform a certain number of switches. As for the second invariant of Section IV-C, we already have an upper bound on the rate of growth of the monitoring signals from Equations (8) and (9). Putting these two pieces together in Invariant 5, and using Theorem 4 we derive the average dwell time property.

Invariant 1. $Q \leq c - \frac{now}{\tau_a} + 1$.

Proof. Initial states satisfy. The $switch_q$ action does not affect the invariant. Consider the $dequeue$ action $s \xrightarrow{a} s'$. We know $s'.Q = s.Q - 1$ and the right hand side of the inequality does not change, therefore the invariant is preserved. Also, all trajectories preserve the invariant. □

Invariant 2. *For all* $q \in \mathcal{P}$,

(1) $\sigma = q \Rightarrow \forall\ p \in \mathcal{P}, \mu_q \leq (1+h)\mu_p$,

(2) $\sigma = q \wedge c_q > 0 \wedge t_q = 0 \Rightarrow \forall\ p \in \mathcal{P}, \mu_q \leq \mu_p$.

Proof. Part(1): Initial states satisfy. For the induction step we need to consider only discrete transitions $s \xrightarrow{a} s'$, where $a = switch_q$. Let $s.\sigma = r$, we know that $s'.\sigma = q$. By inductive hypothesis $s.\mu_r \leq (1+h)s.\mu_p$, for all $p \in \mathcal{P}$. By precondition of $switch_q$, $(1+h)s.\mu_q \leq s.\mu_r$. By continuity of μ_p's $(1+h)s'.\mu_q \leq s'.\mu_r \leq (1+h)s'.\mu_p$, for all $p \in \mathcal{P}$.

From the above it follows that $s'.\mu_q \leq (1+h)s'.\mu_p$, for all $p \in \mathcal{P}$. The stopping condition of activity $flow$ ensures that the invariant is preserved over all trajectories.

Part(2) : Initial states satisfy the invariant because $q = arg\ min_{p\in\mathcal{P}}\mu_p$. For the induction step, consider a discrete transition $s \xrightarrow{a} s'$, where $a = switch_q$. Let $s.\sigma = r$, we know that $s'.\sigma = q$. From Part (1), $s.\mu_r \leq (1+h)s.\mu_p$, for all $p \in \mathcal{P}$. By precondition of $switch_q$, $(1+h)s.\mu_q \leq s.\mu_r$, and by continuity of μ_p's, $s'.\mu_q \leq s'.\mu_p$, for all $p \in \mathcal{P}$.

We note that the $dequeue$ actions do not alter any of the variables involved in the invariant. Now, consider any trajectory τ. If τ is a point trajectory, then the invariant holds. If τ is not a point trajectory, then the invariant holds vacuously because $\tau.lstate.t_q \neq 0$. □

Corollary 1. *For all* $q, r \in \mathcal{P}$,
$\sigma = r \wedge (1+h)\mu_q \leq \mu_r \Rightarrow (1+h)\mu_q = \mu_r$.

Invariant 3. *For all* $q \in \mathcal{P}$,
$c_q \geq 2 \Rightarrow \mu_q^{c_q} \geq (1+h)\mu_q^{c_q-1}$.

Proof. Initial states have $c_p \leq 1$ for all $p \in \mathcal{P}$. For the induction step we only have to consider discrete transitions of the form $s \xrightarrow{a} s'$, where $a = switch_q$. Let $s.\sigma = r$, $s.c_q = k+1$, that is, a is the $(k+1)^{st}$ $switch_q$ action. From the transition relation, and Corollary 1:

$$s'.\mu_q^{k+1} = s'.\mu_q = (1+h)s'.\mu_r \tag{10}$$

Let s'' be the post state of the k^{th} $switch_q$ action. From Invariant 2 Part(1) $s''.\mu_q = s''.\mu_q^k \leq (1+h)s''.\mu_r$. From monotonicity of μ_r, $s''.\mu_r \leq s'.\mu_r$. Since μ_q^k is not changed after s'',

$$s'.\mu_q^k \leq s'.\mu_r \tag{11}$$

Combining (10) and (11), $s'.\mu_q^{k+1} \geq (1+h)s'.\mu_q^k$. □

Corollary 2. $\forall\ q \in \mathcal{P}, \mu_q^{c_q} \geq (1+h)^{c_q-1}\mu_q^1$.

Invariant 4. *There exists* $q \in \mathcal{P}$ *such that* $c_q \geq \lceil \frac{c-1}{m} \rceil$.

Proof. We observe that, the counter c is incremented every time a $switch_p$ action occurs for any $p \in \mathcal{P}$, and for each $p \in \mathcal{P}$ the counter c_p is incremented when the corresponding $switch_p$ action occurs. Since there are m possible values of p, it follows that at least $\lceil \frac{c-1}{m} \rceil$ out of the c $switch$ actions would correspond to some q in $\mathcal{P}$. □

Invariant 5. *If we set the* $\tau_a = \frac{\log(1+h)}{2\lambda m}$ *then,*

$$Q \leq 2 + m + \frac{m}{\log(1+h)} \log\left(\frac{C_1 + C_2}{C_0}\right) \tag{12}$$

Proof. Consider any reachable state s, from Corollary 2, Invariant 4 and monotonicity of μ_p's we know that there exists q in $\mathcal{P}$ such that $s.\mu_q^{c_q} \geq (1+h)^{\lceil \frac{c-1}{m} \rceil - 1} s.\mu_q^1$. Taking logarithm and rearranging we have,

$$s.c \leq 1 + m + \frac{m}{\log(1+h)} \log\left(\frac{s.\mu_q^{c_q}}{s.\mu_q^1}\right)$$

Let s' be the post state of the c_p^{th} $switch_q$ action, then $s.\mu_q^{c_p} = s'.\mu_q$. From Invariant 2 Part(2) and monotonicity

hybridautomaton HysteresisSwitchingLogic(h:PosReal, $\mathcal{P}$:IndexSet)

variables

input analog μ_p : Real, for each $p \in \mathcal{P}$,
output σ : $\mathcal{P}$, initially $\sigma = arg\ min_{p \in \mathcal{P}} \mu_p$,
internal analog now : Real, initially 0,
internal c, d : Int, initially 0,
internal μ_p^i : Real $\cup \{\perp\}$, for $p \in \mathcal{P}$ and $i \in \{0, 1, 2, \ldots\}$,
initially $\mu_p^0 = \mu_p$, $\mu_\sigma^1 = \mu_\sigma$, otherwise $\mu_p^i = \perp$,
internal c_p : Int, initially 0, for $p \neq \sigma$, and $c_\sigma := 1$
internal t_p : Real, initially 0, for each $p \in \mathcal{P}$,

derived variables

const m : Int $= |\mathcal{P}|$
Q : Int $= c - d$

discrete transitions

switch$_p$ for each $p \in \mathcal{P}$
precondition $(1 + h)\mu_p \leq \mu_\sigma$
effect $\sigma := p$; $c := c + 1$;
$c_p := c_p + 1$; $\mu_p^{c_p} := \mu_p$; $t_p := 0$

dequeue
precondition $now = k\tau_a$
effect $d := d + 1$

trajectories

activity *flow*
evolve $d(now) := 1$; $d(t_p) := 1$
stop at $(\exists p, (1 + h)\mu_p \leq \mu_\sigma) \vee (now = k\tau_a)$

Fig. 1. HIOA specification of the hysteresis switching logic in the supervisory controller

it follows that $s.\mu_q^{c_q} = s'.\mu_q^{c_q} \leq s'.\mu_l \leq s.\mu_l$. It follows that,

$$s.c \leq 1 + m + \frac{m}{\log(1+h)} \log\left(\frac{s.\mu_l}{s.\mu_q^1}\right), \quad \forall l \in \mathcal{P}$$

Form monotonicity and property (8) of the monitoring signals, $\mu_q^1 \geq \mu_q^0 \geq C_0$. Therefore,

$$s.c \leq 1 + m + \frac{m}{\log(1+h)} \log\left(\frac{s.\mu_l}{C_0}\right), \quad \forall\ l \in \mathcal{P}$$

Replacing l with p^* of (9), we get

$$s.c \quad \leq \quad 1 + m + \frac{m}{\log(1+h)} \log\left(\frac{C_1 + C_2 e^{2\lambda\ s.now}}{C_0}\right)$$
$$s.c \quad \leq \quad 1 + m + \frac{m}{\log(1+h)} \log\left(\frac{C_1 + C_2}{C_0}\right) + \frac{2\lambda m\ s.now}{\log(1+h)}$$

Using Invariant 1, and putting $\tau_a = \frac{\log(1+h)}{2\lambda m}$, we get the result. □

Theorem 6. *The* HysteresisSwitchingLogic *automaton has an average dwell time of at least* $\frac{\log(1+h)}{2\lambda m}$.

Proof. Follows from Invariant 5 and Theorem 4. □

VI. Remarks and Future Work

We have introduced the hybrid I/O automaton framework [12] as a common modeling platform in which analysis techniques from both computer science and control theory can be applied. To demonstrate the utility and expressive power of this framework, we have shown how known stability theorems from system theory literature can be adapted and applied in the HIOA framework. Our main contribution is to formalize the average dwell time property of hybrid systems as a set of invariants and thereby making it possible to prove (uniform) stability of hybrid systems under slow switching using formal verification techniques. The suggested method has been illustrated by analyzing the stability of a hysteresis switching logic unit in a supervisory control system.

In this paper we examined internal stability only, however, the HIOA framework explicitly incorporates external variables and is also useful for studying input-output properties of hybrid systems. Further, we have proved the invariants in this paper by hand, but it is possible to mechanize the invariant proofs using theorem provers, and in certain special cases to completely automate the proofs with symbolic model checkers. Of course, substantial work remains to be done both in terms of building software tools and developing the theory, in order to have an efficient and seamless process for stability analysis of hybrid systems, and we plan to pursue these in the future.

VII. Acknowledgments

This work significantly benefited from discussions with Nancy Lynch on the use of formal methods in stability analysis of hybrid systems. We also thank Peter B. Jones for his comments on related case studies.

References

[1] Rajeev Alur, Costas Courcoubetis, Nicolas Halbwachs, Thomas A. Henzinger, P.-H. Ho, Xavier Nicollin, Alfredo Olivero, Joseph Sifakis, and Sergio Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138(1):3–34, 1995.

[2] M. Archer, C. Heitmeyer, and S. Sims. TAME: A PVS interface to simplify proofs for automata models. In *Proceedings of UITP '98*, July 1998.

[3] M. Branicky. Multiple lyapunov functions and other analysis tools for switched and hybrid systems. *IEEE Transactions on Automatic Control*, 43:475–482, 1998.

[4] Ekaterina Dolginova and Nancy Lynch. Safety verification for automated platoon maneuvers: A case study. In *HART'97 (International Workshop on Hybrid and Real-Time Systems)*, volume 1201 of *Lecture Notes in Computer Science series*, Grenoble, France, March 1997. Springer Verlag.

[5] Connie Heitmeyer and Nancy Lynch. The generalized railroad crossing: A case study in formal verification of real-time system. In *Proceedings of the 15th IEEE Real-Time Systems Symposium*, San Juan, Puerto Rico, December 1994. IEEE Computer Society Press.

[6] Thomas Henzinger. The theory of hybrid automata. In *Proceedings of the 11th Annual IEEE Symposium on Logic in Computer Science (LICS '96)*, pages 278–292, New Brunswick, New Jersey, 1996.

[7] J. Hespanha and A. Morse. Stability of switched systems with average dwell-time. In *Proceedings of 38th IEEE Conference on Decision and Control*, pages 2655–2660, 1999.

[8] J.P. Hespanha, D. Liberzon, and A.S. Morse. Hysteresis-based switching algorithms for supervisory control of uncertain systems. *Automatica*, 39:263–272, 2003.

[9] Dilsun K. Kaynar, Nancy Lynch, Roberto Segala, and Frits Vaandrager. Timed I/O automata: A mathematical framework for modeling and analyzing real-time system. In *RTSS 2003: The 24th IEEE International Real-Time Systems Symposium*, Cancun,Mexico, December 2003.

[10] Daniel Liberzon. *Switching in Systems and Control*. Systems and Control: Foundations and Applications. Birkhauser, Boston, June 2003.

[11] Carolos Livadas, John Lygeros, and Nancy A. Lynch. High-level modeling and analysis of TCAS. In *Proceedings of the 20th IEEE Real-Time Systems Symposium (RTSS'99),Phoenix, Arizona*, pages 115–125, December 1999.

[12] Nancy Lynch, Roberto Segala, and Frits Vaandraage. Hybrid I/O automata. *Information and Computation*, 185(1):105–157, August 2003.

[13] Nancy Lynch, Roberto Segala, Frits Vaandrager, and H. B. Weinberg. Hybrid I/O automata. In T. Henzinger R. Alur and E. Sontag, editors, *Hybrid Systems III*, volume 1066 of *Lecture Notes in Computer Science*, New Brunswick, New Jersey, October 1995. Springer-Verlag.

[14] Nancy A. Lynch, Roberto Segala, and Frits W. Vaandrager. Hybrid I/O automata revisited. In M.D. Di Benedetto and A.L. Sangiovanni-Vincentelli, editors, *Proceedings Fourth International Workshop on Hybrid Systems: Computation and Control (HSCC'01)*, Rome, Italy, volume 2034 of *LNCS*, pages 403–417. Springer, March 2001.

[15] Sayan Mitra, Yong Wang, Nancy Lynch, and Eric Feron. Application of hybrid I/O automata in safety verification of pitch controller for model helicopter system. Technical Report MIT-LCS-TR-880, MIT Laboratory for Computer Science, Cambridge, MA 02139, January 2003. http://theory.lcs.mit.edu/mitras/research/.

[16] Sayan Mitra, Yong Wang, Nancy Lynch, and Eric Feron. Safety verification of model helicopter controller using hybrid Input/Output automata. In *HSCC'03, Hybrid System: Computation and Control*, Prague, the Czech Republic, April 3-5 2003. Also, long version in [15].

[17] A. S. Morse. Supervisory control of families of linear set-point controllers, part 1: exact matching. *IEEE Transactions on Automatic Control*, 41:1413–1431, 1996.

[18] A. van der Schaft and H. Schumacher. *An Introduction to Hybrid Dynamical Systems*. Springer, London, 2000.