Lecture 2

Hybrid Automata and Executions

Karl Henrik Johansson

Hybrid Automata

Definition 1 (Hybrid Automaton). *A hybrid automaton H is a collection* $H = (V, X, f, \text{Init}, \text{Inv}, \text{Jump})$*:*

Discrete modes *A finite set V is called the discrete modes. It represents the discrete dynamics of H.*

Continuous variables *A finite ordered set $X = \{x_1, \ldots, x_n\}$, $n \geq 0$, of real-valued variables is called the continuous variables. I represents the continuous dynamics.*

Vector field *A function $f : V \times \mathbb{R}^n \to \mathbb{R}^n$ is called the vector field. It defines the continuous flow in each discrete mode $v \in V$ through a differential equation $\dot{x} = f(v, x)$.*

Initial condition *The initial condition is a set* $\text{Init} \subset V \times \mathbb{R}^n$ *that defines the initial state of H. A state of the hybrid automaton is a pair (v, z) that consists of a discrete mode $v \in V$ and a point $z \in \mathbb{R}^n$ being the value of x.*

Invariant condition *The set* $\text{Inv} \subset V \times \mathbb{R}^n$ *is called the invariant condition. As long as the discrete mode of H is in $v \in V$, the state must belong to* Inv.[1]

Jump condition *A set-valued function* $\text{Jump} : V \times \mathbb{R}^n \to P(V \times \mathbb{R}^n)$ *is called the jump condition.*[2] *It specifies if a jump from one discrete mode to another is possible and what new value should be assigned to the continuous variable after the jump.*

We refer to a point $(v, z) \in V \times \mathbb{R}^n$, where z is the valuation of x, as the *state* of H. Sometimes we call v the discrete state and z the continuous state.

A short comment on notation: Note that we distinguish variables from their valuations. The set of variables is a set of symbols, e.g., $X = \{x_1, \ldots, x_n\}$. The set of valuations are all possible values these variables can take, e.g., the set $\mathbb{R}^n$. The valuation itself can be thought of as a map giving a value to a variable. We will use z_i as a generic value for the variable $x_i \in X$. Sometimes we collect the elements of X in a (column) vector and use the notation $x = (x_1, \ldots, x_n)^T$. It

[1] The notion "invariant" for Inv comes from the hybrid system literature in computer science. Note that Inv is in general not invariant in the usual dynamical systems sense.

[2] Here $P(\Omega)$ denotes the *power set* of Ω, i.e., the set of all subsets of Ω.

is common in the control literature to let a variable x also denote the function that is the solution to a differential equation $\dot{x} = f(x)$. We will adopt this slight abuse of notation (as was done above in the definition of the vector field), and, hence, let $x : I \to \mathbb{R}^n$ also denote a function from some interval I to $\mathbb{R}^n$. The component of x is denoted x_i, $i \in \{1, \ldots, n\}$.

Hybrid automata are convenient to visualize as graphs. In order to get a straightforward mapping from the formal definition to a graphical representation, we introduce the following notation based on Definition 1. A hybrid automaton can be represented by a directed graph (V, E) with vertices V and edges

$$E = \{(v, v') \in V \times V : \exists z, z' \in \mathbb{R}^n, (v', z') \in \text{Jump}(v, z)\},$$

also known as the control switches. With each vertex $v \in V$, we associate a set of continuous initial states

$$\text{Init}(v) = \{z \in \mathbb{R}^n : (v, z) \in \text{Init}\},$$

a vector field $f(v, \cdot)$, and an invariant map

$$\text{Inv}(v) = \{z \in \mathbb{R}^n : (v, z) \in \text{Inv}\}.$$

With each edge $e = (v, v') \in E$, we associate a guard condition

$$G(e) = \{z \in \mathbb{R}^n : \exists z' \in \mathbb{R}^n, (v', z') \in \text{Jump}(v, z)\},$$

and a jump map

$$J(e, z) = \{z' \in \mathbb{R}^n : (v', z') \in \text{Jump}(v, z)\}.$$

The guard condition tells for which continuous states it is possible to go from one discrete mode to another and the jump map gives possible continuous states to jump to.

The graphical notation is summarized in Figure 1 with a hybrid automaton having discrete modes $V = \{v_1, v_2\}$, continuous variable $X = \{x\}$, and a single control switch $(v_1, v_2) \in E$. In each vertex of the graph, we specify the discrete mode, the differential equation implied by the vector field, and the invariant condition. On each edge, the guard condition and the jump map is specified. The notation "$:\in$" means that x' is assigned a value from a set-valued map. The initial state $\text{Init}(v)$ may be indicated as an arrow (an edge without a source vertex) pointing at the vertex v with the set $\text{Init}(v)$ specified on it. The expressions "$x \in I(v_1)$","$x \in G(v_1, v_2)$", and "$x' :\in J(v_1, v_2, x)$" should be interpreted such that the conditions should hold when x and x' are replaced by their assigned values $a, a' \in \mathbb{R}^n$, say.

Example 1. Figure 2 shows the hybrid automaton for a bouncing ball in Lecture 1, where $c \in [0, 1]$ and $g > 0$. In the graphical notation we have the following definition of the hybrid automaton:

- $V = \{v\}$ and $X = \{x_1, x_2\}$,

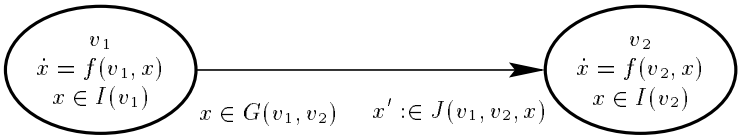

Fig. 1. Hybrid automaton specified in the graphical notation.

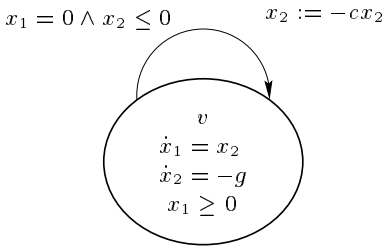

Fig. 2. Hybrid automaton for bouncing ball.

- $f(v, z) = (z_2, -g)^T$,
- $\text{Init}(v) = \{x \in \mathbf{X} : x_1 \geq 0\}$,
- $I(v) = \{z \in \mathbb{R}^2 : z_1 \geq 0\}$,
- $E = \{(v, v)\}$,
- $G(v, v) = \{z \in \mathbb{R}^2 : z_1 = 0 \wedge z_2 \leq 0\}$,
- $J(v, v, (z_1, z_2)) = \{(z_1, -cz_2)\}$.

Using Definition 1, we have

- $\text{Init} = \{v\} \times \{x \in \mathbf{X} : x_1 \geq 0\}$,
- $\text{Inv} = \{v\} \times \{z \in \mathbb{R}^2 : z_1 \geq 0\}$,

and the last three lines reduce into the jump condition

$$\text{Jump}(v, z) = \begin{cases} \{(v', z') \in V \times \mathbb{R}^2 : (v', z_1', z_2') = (v, z_1, -cz_2)\}, & \text{if } z_1 = 0, z_2 \leq 0 \\ \emptyset, & \text{otherwise.} \end{cases}$$

When a continuous variable does not impose a condition on the invariant or guard, we do not necessarily include it in the graph (in Figure 2 x_2 has been dropped from the invariant condition). The same holds if a jump map is the identity for some continuous variable ($x_1 := x_1$ in Figure 2).

Example 2. Consider the water tank system in Figure 3. For $i \in \{1, 2\}$, let x_i denote the volume of water in Tank i and $q_i > 0$ denote the constant flow of water out of Tank i. Let $q_{\text{in}} > 0$ denote the constant flow of water into the system, directed exclusively to either Tank 1 or Tank 2 at each point in time. The objective is to keep the water volumes above r_1 and r_2, respectively (assuming that the water volumes are above r_1 and r_2 initially). This is to be achieved by

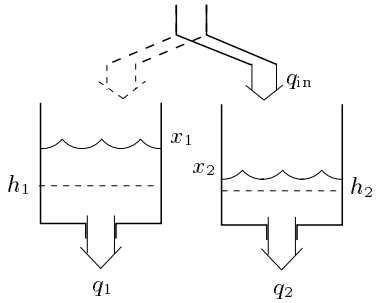

Fig. 3. Water tank control system.

a switched control strategy that switches the inflow to Tank 1 whenever $x_1 \leq r_1$ and to Tank 2 whenever $x_2 \leq r_2$. Let $r_1 = r_2 = 1$ and $(q_1, q_2, q_{\text{in}}) = (2, 3, 4)$. Then, the water tank control system may be described by the hybrid automaton in Figure 4. The formal definition is given by

- $V = \{v_1, v_2\}$ and $X = \{x_1, x_2\}$,
- $f(v_1, z) = (2, -3)$ and $f(v_2, z) = (-2, 1)$,

with either

- $\text{Init} = V \times \{z \in \mathbb{R}^2 : z_1 > 1, z_2 > 1\}$,
- $\text{Inv} = \{(v, z) \in V \times \mathbb{R}^2 : v = v_1, z_2 \geq 1\} \cup \{(v, z) \in V \times \mathbb{R}^2 : v = v_2, z_1 \geq 1\}$,
- $$\text{Jump}(v, z) = \begin{cases} \{(v_1, z)\}, & \text{if } v = v_2, z_2 \leq 1 \\ \{(v_2, z)\}, & \text{if } v = v_1, z_1 \leq 1 \\ \emptyset, & \text{otherwise,} \end{cases}$$

following Definition 1 or, for the graphical notation,

- $\text{Init}(v) = \{z \in \mathbb{R}^2 : z_1 > 1, z_2 > 1\}$ for both $v \in V$,
- $I(v_1) = \{(z \in \mathbb{R}^2 : z_2 \geq 1\}$ and $I(v_2) = \{(z \in \mathbb{R}^2 : z_1 \geq 1\}$,
- $E = \{(v_1, v_2), (v_2, v_1)\}$,
- $G(v_1, v_2) = \{z \in \mathbb{R}^2 : z_2 \leq 1\}$ and $G(v_2, v_1) = \{z \in \mathbb{R}^2 : z_1 \leq 1\}$,
- $J(v_1, v_2, z) = J(v_2, v_1, z) = z$.

Since there is a unique graphical representation for each hybrid automaton, we will in most examples use the corresponding graph as a formal definition.

Hybrid Time Trajectories

Definition 2 (Hybrid Time Trajectory). *A hybrid time trajectory τ is a finite or infinite sequence of intervals $\tau = \{I_i\}_{i=0}^{N}$, such that*

- *$I_i = [\tau_i, \tau_i']$ for $i < N$, and, if $N < \infty$, $I_N = [\tau_N, \tau_N']$ or $I_N = [\tau_N, \tau_N')$,*

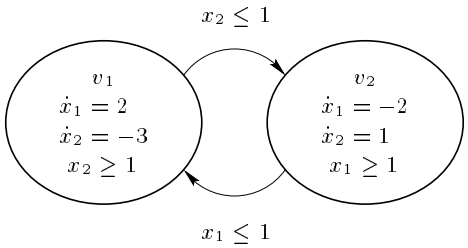

Fig. 4. Hybrid automaton for water tank system.

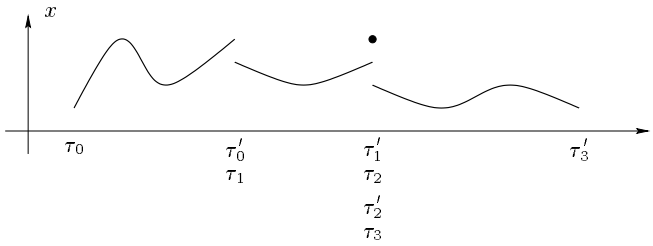

Fig. 5. Hybrid time trajectory $\tau = [\tau_0, \tau_0'][\tau_1, \tau_1'][\tau_2, \tau_2'][\tau_3, \tau_3']$.

– $\tau_i \leq \tau_i' = \tau_{i+1}$ *for* $i \geq 0$.

A hybrid time trajectory is a sequence of intervals of the real line, whose end points overlap. The interpretation is that the end points of the intervals are the times at which discrete transitions take place. Note that $\tau_i = \tau_i'$ is allowed, therefore multiple discrete transitions may take place at the same "time". An illustration of a hybrid time trajectory with four intervals is shown in Figure 5, where jumps in the continuous variable are indicated. At $t = \tau_1' = \tau_2 = \tau_2' = \tau_3$ two jumps take place.

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 $[\tau_N, \infty)$. We denote by $\mathcal{T}$ the set of all hybrid time trajectories and use $t \in \tau$ as shorthand notation for that there exists i such that $t \in I_i$ with $I_i \in \tau$.

We use v and x to also denote the time evolution of the discrete and continuous state, respectively, with a slight abuse of notation (as discussed above). For each $i \in \{1, \ldots, N\}$, they will be defined as functions from the interval I_i to V and $\mathbb{R}^n$, respectively. We use $v : \tau \to V$ and $x : \tau \to \mathbb{R}^n$ as short hand notations for the maps assigning values from V and $\mathbb{R}^n$ to each $t \in \tau$. Note v and x are not functions on the real line, as they assign multiple values to the same $t \in \mathbb{R}$ at $t = \tau_i' = \tau_{i+1}$ for all $i \geq 0$.

Each $\tau \in \mathcal{T}$ is fully ordered by the relation $\prec$ defined by $t_1 \prec t_2$ for $t_1 \in [\tau_i, \tau_i']$ and $t_2 \in [\tau_j, \tau_j']$ if $i < j$, or if $i = j$ and $t_1 < t_2$.

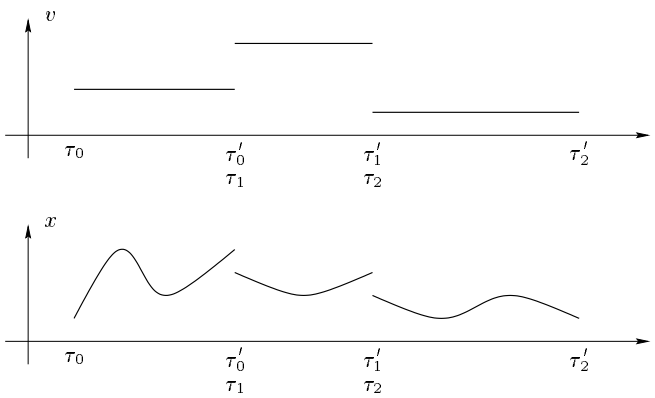

Fig. 6. Example of an execution.

Executions

Next we introduce a concept similar to a solution of a continuous dynamical systems for hybrid automata. This concept is, however, richer the regular solutions, so to distinguish them we introduce the notion of *executions* of hybrid automata.

Definition 3 (Execution). *An execution χ of a hybrid automaton H is a collection $\chi = (\tau, v, x)$ with $\tau \in \mathcal{T}$, $v : \tau \to V$, and $x : \tau \to \mathbb{R}^n$, satisfying*

Initial condition $\big(v(\tau_0), x(\tau_0)\big) \in \text{Init}$,
Continuous evolution *for all i with $\tau_i < \tau_i'$, $v(\cdot)$ is constant and $x(\cdot)$ is a solution to the differential equation $\dot{x}(t) = f\big(v(t), x(t)\big)$ over $[\tau_i, \tau_i']$, and for all $t \in [\tau_i, \tau_i')$, $(v(t), x(t)) \in \text{Inv}$*
Discrete evolution *for all i, $(v(\tau_{i+1}), x(\tau_{i+1})) \in \text{Jump}\big(v(\tau_i'), x(\tau_i')\big)$*

In the graphical representation, the invariant condition corresponds to $x(t) \in I(q(t))$ and jump condition corresponds to $e = (v(\tau_i'), v(\tau_{i+1})) \in E$, $x(\tau_i') \in G(e)$, and $x(\tau_{i+1}) \in J(e, x(\tau_i'))$. Figure 6 illustrates an execution.

We say a hybrid automaton *accepts* an execution χ or not. For an execution $\chi = (\tau, v, x)$, we use $(v_0, x_0) = \big(v(\tau_0), x(\tau_0)\big)$ to denote the initial state of χ. The *execution time* $\tau_\infty(\chi)$ is defined as $\tau_\infty(\chi) = \sum_{i=0}^{N}(\tau_i' - \tau_i)$, where $N+1$ is the number of intervals in the hybrid time trajectory. The argument χ will sometimes be left out.

We say $\chi = (\tau, v, x)$ is a prefix of $\hat{\chi} = (\hat{\tau}, \hat{v}, \hat{x})$ (write $\chi \leq \hat{\chi}$), if $\tau \leq \hat{\tau}$ and $\big(v(t), x(t)\big) = \big(\hat{v}(t), \hat{x}(t)\big)$ for all $t \in \tau$. We say χ is a strict prefix of $\hat{\chi}$ (write $\chi < \hat{\chi}$), if $\chi \leq \hat{\chi}$ and $\chi \neq \hat{\chi}$. An execution is *maximal* if it is not a strict prefix of any other execution.

An execution is *finite* if τ is a finite sequence ending with a compact interval, it is called *infinite* if τ is either an infinite sequence or if $\tau_\infty(\chi) = \infty$, and it

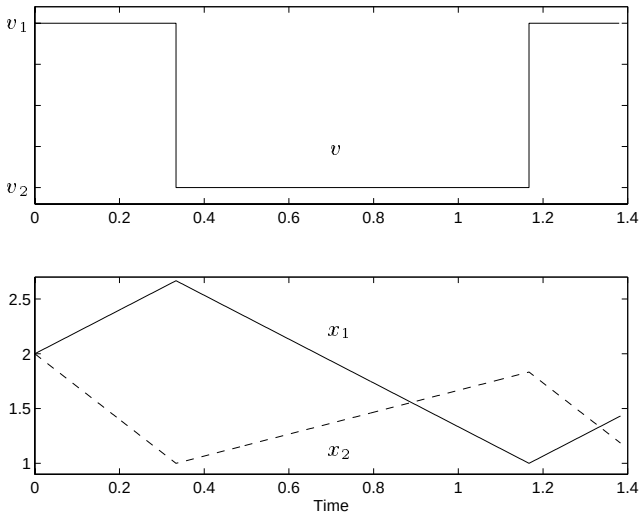

Fig. 7. Example of an execution for the water tank hybrid automaton.

is called *Zeno* if it is infinite but $\tau_\infty(\chi) < \infty$. The execution time of a Zeno execution is also called the Zeno time.

We use $\mathcal{E}_H(v_0, x_0)$ to denote the set of all executions of H with initial condition $(v_0, x_0) \in \text{Init}$, $\mathcal{E}_H^M(v_0, x_0)$ to denote the set of all maximal executions, and $\mathcal{E}_H^\infty(v_0, x_0)$ to denote the set of all infinite executions. We define $\mathcal{E}_H = \bigcup_{(v_0,x_0)\in \text{Init}} \mathcal{E}_H(v_0, x_0)$ and $\mathcal{E}_H^\infty = \bigcup_{(v_0,x_0)\in \text{Init}} \mathcal{E}_H^\infty(v_0, x_0)$. To simplify the notation, we will drop the subscript H whenever the automaton is clear from the context. For all $(v_0, x_0) \in \text{Init}$, we have

$$\mathcal{E}_H^\infty(v_0, x_0) \subset \mathcal{E}_H^M(v_0, x_0) \subset \mathcal{E}_H(v_0, x_0).$$

An execution for the hybrid automaton of the water tank in Example 2 is shown in Figure 7. Figure 8 illustrates the continuous part of an execution for the hybrid automaton of the bouncing ball in Example 1.

Background

The definitions are based on [4, 3, 6]. Some terminology is also borrowed from [2]. The water tank example comes from [1].

References

1. R Alur and T A Henzinger. Modularity for timed and hybrid systems. In A. Mazurkiewicz and J. Winkowski, editors, *CONCUR 97: Concurrency Theory*, Lecture Notes in Computer Science 1243, pages 74–88. Springer-Verlag, 1997.

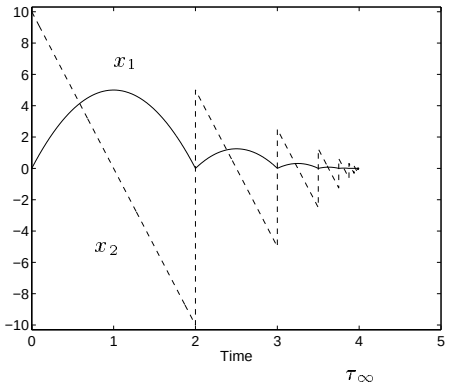

Fig. 8. Example of an execution for the bouncing ball automaton.

2. T.A. Henzinger. The theory of hybrid automata. In *Proceedings of the 11th Annual Symposium on Logic in Computer Science*, pages 278–292. IEEE Computer Society Press, 1996.
3. K. H. Johansson, M. Egerstedt, J. Lygeros, and S. Sastry. On the regularization of Zeno hybrid automata. *System & Control Letters*, 38:141–150, 1999.
4. J. Lygeros, C. Tomlin, and S. Sastry. Controllers for reachability specifications for hybrid systems. *Automatica*, 35(3), March 1999.
5. S. Sastry. *Nonlinear Systems: Analysis, Stability, and Control*. Springer-Verlag, New York, 1999.
6. J Zhang, K H Johansson, J Lygeros, and S Sastry. Dynamical systems revisited: Hybrid systems with Zeno executions. To appear at HSCC'00, 2000.