ON SIMULATION OF ZENO HYBRID SYSTEMS

VIRGINIA ECATERINA OLTEAN

Key words: Hybrid automata, Zeno execution.

The interaction of event-driven with time-driven dynamics in hybrid systems may lead to so-called Zeno behaviour, characterized by accumulation of an infinite number of events within a finite time-interval. Although real systems are not Zeno, hybrid models may exhibit Zeno behaviour, due to modelling abstraction. Understanding this is essential for numerical simulation of hybrid systems, which may become imprecise or time-consuming. In this paper, some basic definitions from the theory of Zeno hybrid systems are firstly overviewed, together with a Simulink model of a Zeno bouncing ball model. For a Zeno water tank model, classic in the literature, a simulation approach based on fixed-step discretization of the continuous state equations is proposed.

1. INTRODUCTION AND MOTIVATION

Hybrid systems combine event-driven with time-driven dynamics and have emerged as an efficient tool in modelling, analysis and design of technical systems, in which computers interact with an analog environment. Despite recent progress, hybrid systems theory still presents open issues [1]. Such a problem is the study of Zeno executions.

In brief, a hybrid system exhibits Zeno behaviour if it executes infinite many transitions in a finite time interval. Although real systems are not Zeno, a hybrid model of a real system may be Zeno, due to modelling abstraction. Understanding when abstraction leads to Zeno behaviour is important for simulation, because without proper analysis, numerical simulations become imprecise or time consuming. Simulations are important also because nowadays it is difficult to draw conclusions about Zeno systems, using available theory. In [2] it is shown that most simulation packages, dedicated to hybrid systems, such as Dymola [3], Omola [4] or SHIFT [5] "get stuck when a large number of discrete transitions take place within a short time interval".

"Politehnica" University of Bucharest, Spl. Independenței 313, sector 6, 77206 Bucharest, E-mail: ecaterina_oltean@yahoo.com

Rev. Roum. Sci. Techn. – Électrotechn. et Énerg., **52**, *2,* p. 229–239, Bucarest, 2007

Zeno hybrid automata have earlier been studied within computer science theory [6]. Motivated by the fact that Zeno hybrid systems reflect the modelling approximations, some general properties of these systems have been recently investigated, in an attempt to classify conditions driving to Zeno behaviour [2], [9], and to deduce possible model transformations, which permit extension beyond the so-called Zeno time [7, 8].

Starting from the hybrid systems analysis reported in [2] and from two classic examples discussed in [7] and [8], this paper is focussed on some technical aspects related to simulation of Zeno behaviour in the MATLAB-Simulink general-purpose environment. Two basic problems have to be treated in the numerical simulation process, among others: detecting when discrete transitions are enabled and avoiding that simulation gets stuck, when simulation time reaches a neighbourhood of the Zeno time. The paper is organized as follows: an overview of Zeno hybrid systems concepts and difficulties related to simulation of a classic Zeno bouncing ball hybrid system are presented in section 2. A proposal for simulation of a Zeno water tank model, based on fixed-step model discretization, is discussed in section 3, followed by concluding remarks.

2. ZENO HYBRID SYSTEMS – A BRIEF OVERVIEW

2.1. BASIC DEFINITIONS

The following definitions review in short some basic concepts presented in [7], [8], [10] and [11].

DEFINITION 1. *A* hybrid automaton *H is a collection* $H=(Q,X,Init,\boldsymbol{f},I,E,G,R)$, *where Q is finite set of discrete variables, X is a finite set of continuous variables, with* $X=\mathbf{R}^n$, $Init\subseteq Q\times X$ *is the set of initial states,* $\boldsymbol{f}:Q\times X\rightarrow TX$ *is a vector field,* $I:Q\rightarrow 2^X$ *is an invariant set for each* $q\in Q$, $E\subset Q\times Q$ *is a set of edges,* $G:E\rightarrow 2^X$ *is a guard of each edge and* $R:E\times X\rightarrow 2^X$ *is a reset map for each edge.*

Remarks. TX denotes the tangent space of *X*. The state of *H* is $(q,\boldsymbol{x})\in Q\times X$. A hybrid automaton can be depicted as directed graph (Q,E), with vertices *Q* and edges *E*. With each vertex $q\in Q$ are associated: (*i*) a set of continuous initial states $Init_q=\{\boldsymbol{x}\in X\,|\,(q,\boldsymbol{x})\in Init\}$, (*ii*) a globally Lipschitz vector field $\boldsymbol{f}_q(x)=\boldsymbol{f}(q,\boldsymbol{x})$ and (*iii*) an invariant set $I(q)$. Also, with each edge $e\in E$ are associated: (*i*) a guard $G(e)$ and (*ii*) a reset relation $R(e,\boldsymbol{x})$.

DEFINITION 2. *A* hybrid time trajectory $\tau = \{I_i\}_{i=0}^{N}$ *is a finite or infinite sequence of intervals, such that* $I_i = [\tau_i, \tau_i^{'}]$ *for* $i < N$ *and either* $I_N = [\tau_N, \tau_N^{'}]$ *or* $I_N = [\tau_N, \tau_N^{'})$ *if* $N < \infty$, *where* $\tau_i \le \tau_i^{'} = \tau_{i+1}$.

Remarks. In a hybrid automaton a discrete transition takes place at the time moments τ_i, and the continuous evolution on non-vanishing intervals $[\tau_i, \tau_i^{'}]$ [8]. $\mathcal{T}$ denotes the set of all hybrid time trajectories.

DEFINITION 3. *An* execution $\mathcal{X}$ *of a hybrid automaton is a collection* $\mathcal{X} = (\tau, q, \boldsymbol{x})$ *with* $\tau \in \mathcal{T}$, $q : \tau \to Q$ *and* $\boldsymbol{x} : \tau \to X$ *satisfying the following conditions: (i)* $(q(\tau_0), \boldsymbol{x}(\tau_0)) \in Init$ *; (ii) for all i with* $\tau_i < \tau_i^{'}$, $q(\cdot)$ *is constant and* $\boldsymbol{x}(\cdot)$ *is a solution to the differential equation* $\dot{\boldsymbol{x}} = \boldsymbol{f}(q, \boldsymbol{x})$ *over* $[\tau_i, \tau_i^{'}]$, *and for all* $t \in [\tau_i, \tau_i^{'})$, $\boldsymbol{x}(t) \in I(q(t))$ *; (iii) for all i,* $e = (q(\tau_i^{'}), q(\tau_{i+1})) \in E$, $\boldsymbol{x}(\tau_i^{'}) \in G(e)$ *and* $\boldsymbol{x}(\tau_{i+1}) \in R(e, \boldsymbol{x}(\tau_i^{'}))$.

An execution is *finite* if τ is a finite sequence ending with a closed interval, it is *infinite* if τ is either an infinite sequence or if $\sum_i (\tau_i^{'} - \tau_i) = \infty$ and it is maximal if it is not a strict prefix of any other execution of *H*.

DEFINITION 4. *Given a hybrid automaton H, an infinite execution started in* $(q_0, x_0) \in Init$ *is called* Zeno *if* $\sum_{i=0}^{\infty} (\tau_i^{'} - \tau_i)$ *is bounded. The time* $\tau_\infty = \sum_{i=0}^{\infty} (\tau_i^{'} - \tau_i)$ *is the* Zeno time. *A hybrid automaton is Zeno if all its infinite executions started in some* $(q_0, x_0) \in Init$ *are Zeno.*

2.2. AN EXAMPLE – THE BOUNCING BALL ZENO HYBRID SYSTEM

The mechanical collision of a bouncing ball illustrates the class of hybrid systems with autonomous jumps [1]. Consider the one-dimensional motion of a unit-mass ball on a vertical axis in gravitational field, with uniform acceleration *g*. The equations of motion are:

$$\dot{x}_1 = x_2 ,\ \dot{x}_2 = -g \text{ , for } x_1 \ge 0 , \tag{1}$$

where x_1 is the height of the ball on the vertical axis and x_2 its velocity. When hitting the ground and assuming downward velocity, the velocity x_2 instantly changes value to $-Kx_2$, where $K \in [0,1)$ is the coefficient of restitution. Equations (1) remain unchanged and the jump rule can be formulated as follows:

$$\text{If } x_1(t) = 0 \text{ and } x_2(t) < 0 \text{ then } x_2(t^+) = -Kx_2(t) . \tag{2}$$

The hybrid automaton with a single location in Fig. 1 models this behaviour.

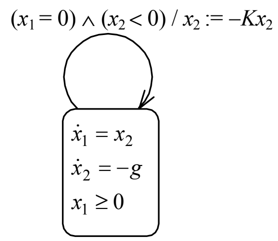


Fig. 1 – A hybrid model of the bouncing ball [1, 7, 11].

At $\tau_0 = 0$, the system starts from the initial continuous state $x_1(0) = x_{10} > 0$ and $x_2(0) = x_{20}$, generating the state trajectory with components $x_2(t) = x_{20} - gt$, $x_1(t) = x_{10} + x_{20}t - gt^2/2$, until at $\tau_0^{'}$ the ball hits the ground, such that

$$x_1(\tau_0^{'}) = 0 \wedge \tau_0^{'} > 0 \Rightarrow \tau_0^{'} = \frac{x_{20} + \sqrt{x_{20}^2 + 2gx_{10}}}{g}, \tag{3}$$

and the speed switches according to (2): $x_2(\tau_1) = -Kx_2(\tau_0^{'}) = K\sqrt{x_{20}^2 + 2gx_{10}}$. The time elapsed till the ball hits the ground is $\Delta_0 = \tau_0^{'} - \tau_0 = \tau_0^{'}$. From $\tau_1 = \tau_0^{'}$, the system evolves with continuous dynamics (1) and with new initial state $x_1(\tau_1)$ and $x_2(\tau_1)$, until, at $\tau_1^{'}$, the ball hits again the ground, after a time $\Delta_1 = \tau_1^{'} - \tau_1$,

$$x_1(\tau_1^{'}) = 0 \wedge \Delta_1 > 0 \Rightarrow \Delta_1 = 2x_2(\tau_1)/g = 2K\sqrt{x_{20}^2 + 2gx_{10}}\,/g\,. \tag{4}$$

The time of the second bounce is

$$\tau_2 = \tau_1^{'} = \tau_1 + \Delta_1 = \frac{x_{20}}{g} + \frac{2\sqrt{x_{20}^2 + 2gx_{10}}}{g}\left(\frac{1}{2} + K\right). \tag{5}$$

Again, the speed switches according to (2). After the k^{th} bounce, the total time elapsed is

$$\tau_k = \sum_{i=0}^{k-1} \Delta_i = \frac{x_{20}}{g} + \frac{2\sqrt{x_{20}^2 + 2gx_{10}}}{g}\left(\frac{1}{2} + \sum_{i=1}^{k-1} K^i\right). \tag{6}$$

Since for $K \in [0, 1)$,

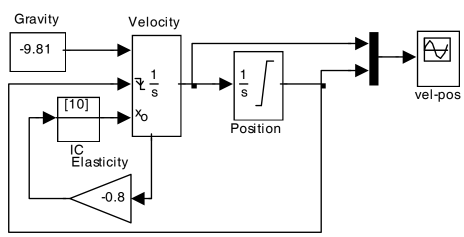
$$\lim_{k\to\infty}\left(\sum_{i=1}^{k-1} K^i\right) = \lim_{k\to\infty} \frac{K - K^{k-1}}{1 - K} = \frac{K}{1 - K}, \tag{9}$$

the Zeno time is

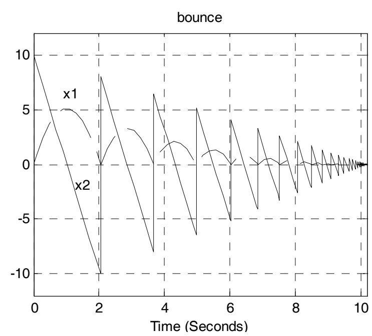
$$\tau_\infty = \frac{x_{20}}{g} + \frac{\sqrt{x_{20}^2 + 2gx_{10}}}{g} \cdot \frac{1+K}{1-K} < \infty \,. \tag{10}$$

Thus, the system executes an infinite number of switching in a finite time interval.

In the Simulink toy example of the bouncing ball model in Fig.2a, the switching rule (2) is implemented by means of a *reset input* of the velocity integrator. Fig.2b depicts a simulation result.



a



b

Fig. 2 – Simulink model of the hybrid automaton in Fig.1 (a) and a simulation experiment with Zeno time $\tau_\infty = 10.1937$ and simulation time $t_{sim} = 10.19$ (b).

The crucial problem is the simulation behaviour in the *vicinity* of Zeno time. If the simulation time is chosen *greater* than the *computed* Zeno time, the simulation gets stuck (Fig. 3a), or peculiar results may be obtained (Fig. 3b).

The hybrid automaton in Fig. 1 is Zeno because it executes an infinite number of transitions in the time interval $(\tau_\infty - \varepsilon, \tau_\infty)$, for any ε, with $\tau_\infty > \varepsilon > 0$. Zeno behaviour results here from the model simplification, in which the real bounce dynamics [11] is replaced by an initial condition reset (2).

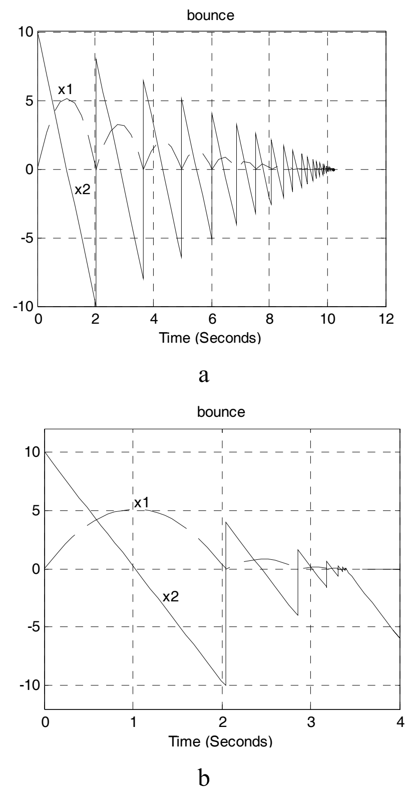


Fig. 3 – Simulation experiments of the Zeno bouncing ball hybrid automaton:
a) $\tau_\infty = 10.1937$ and $t_{sim} = 12$; b) $\tau_\infty = 3.3979$ and $t_{sim} = 4$.

A way to solve Zeno phenomenon is by *regularization* [7, 8]. Given a Zeno hybrid automaton *H*, the idea is to build a family of non-Zeno hybrid automata, H_ε, parameterised by real positive ε, and a continuous map $Q_\varepsilon \times X_\varepsilon \to Q \times X$ which relates the states of each H_ε to the states of *H*, such that H_ε tends to *H*, with $\varepsilon \to 0$, despite the fact that the executions of H_ε are *not* executions of *H*.

For example, *temporal regularization* of the hybrid automaton in Fig.1 can be obtained by introducing a *clock variable* x_3, which counts, i.e. $\dot{x}_3 = 1$, in the ε-vicinity of a bounce moment τ_k, when position x_1 and velocity x_2 are *forced* to remain constant, and which remains constant, *i.e.* $\dot{x}_3 = 0$, "far" from bounce moments. Such a regularized hybrid automaton is presented in [11].

3. A SIMULATION APPROACH OF A WATER TANK HYBRID SYSTEM

The water tank system and the hybrid model depicted in Fig. 4 were initially proposed in [12] and analysed in [7]. x_1 and x_2 denote the water level and v_1 and v_2 are the (constant) water flows out of tank 1 and 2, respectively. w is the (constant) input flow. Assume that $x_1(0) > r_1$, $x_2(0) > r_2$. The control task is to keep the water levels x_1 and x_2 above the limits r_1 and r_2, respectively, by an adequate switching policy. A solution is to switch the inflow w to tank 1, whenever $x_1 \le r_1$, and to tank 2, whenever $x_2 \le r_2$.

At $\tau_0 = 0$, the hybrid automaton starts from initial state (q_0, x_{10}, x_{20}) and evolves generating the trajectory given by $x_1(t) = x_{10} + (w - v_1)t$, $x_2(t) = x_{20} - v_2 t$, until, at $\tau_1 = (x_{20} - r_2)/v_2$, the enabling condition $x_2(\tau_1) = r_2$ for transition $q_0 \to q_1$ is satisfied and the system instantly switches to q_1. The time spent in q_0 is $\Delta_0 = \tau_1 - \tau_0 = (x_{20} - r_2)/v_2$.

Similarly, from τ_1 the system evolves in q_1, with the continuous behaviour $x_1(t) = x_1(t_1) - v_1 t$, $x_2(t) = r_2 + (w - v_2)t$, until, at $\tau_2 = (x_1(\tau_1) - r_1)/v_1$, the enabling condition $x_1(\tau_2) = r_1$ for transition $q_1 \to q_0$ is satisfied, and the system instantly switches back to q_0. The time spent in q_1 is $\Delta_1 = \tau_2 - \tau_1$ and the process continues.

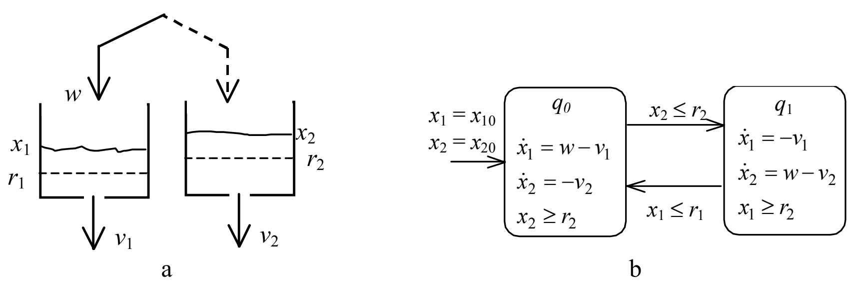


Fig. 4 – Water tank system (a) and associated hybrid automaton (b) [7].

In [7, 11] it is shown that if $\max\{v_x, v_y\} < w < v_x + v_y$, then the water tank automaton is Zeno and the Zeno time of the execution starting in $(q(0), x_0, y_0)$ is

$$\tau_\infty = \frac{x_{10} + x_{20} - r_1 - r_2}{v_1 + v_2 - w}. \tag{11}$$

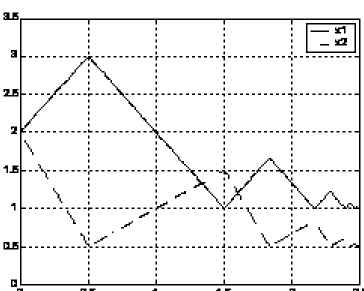
In an attempt to avoid simulation to get stuck in the vicinity of Zeno time, a discrete time approach is proposed. The two continuous integrators in variables x_1 and x_2 are converted into their discrete time versions (using *c2d* routine, for example): $x_{1d}(+1) = x_{1d} + h \cdot u_1$, $x_{2d}(+1) = x_{2d} + h \cdot u_2$, where $h > 0$ is the sampling step. Then the transition condition tested at step *k* is:

if $x_{2d}(+1) \le r_2$ *then* $u_1 = -v_1$, $u_2 = w - v_2$
  else if $x_{1d}(+1) <= r_1$ *then* $u_1 = w - v_1$, $u_2 = -v_2$
end

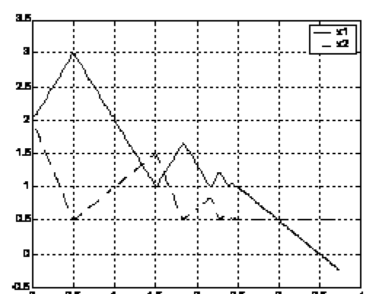
The problem with this approach is the (rather empirical) choice of the sampling step *h*. Simulation was performed for $h = 10^{-3}$ and $x_{10} = x_{20} = 2$, $r_1 = 1$, $r_2 = 0.5$, $w = 4$, $v_1 = 2$, $v_2 = 3$. According to (11), $\tau_\infty = 2.5\text{s}$. Simulation results are depicted in Fig.5a, for simulation time equal to Zeno time, and in Fig.5b for simulation time greater than Zeno time.

The result in Fig.5b can be interpreted as follows: after a time interval given by the Zeno time, in one of the two tanks the water level remains constant, while the other will drain. On the other hand, physical intuition suggests that if the amount of water coming into the system is less than the amount of water getting out, i.e. $w = 4 < v_1 + v_2 = 5$, then at least one of the two tanks will drain.

As already mentioned in previous section, an approach to solve Zeno behaviour is by regularization. In [7], *spatial regularization* of the water tank system is proposed, by introducing a minimum deviation in the continuous state between the discrete transitions.



a



b

Fig. 5 – Simulation experiments of discrete time version of the Zeno water tanks system in Fig. 4, with $\tau_\infty = 2.5$: a) $t_{sim} = \tau_\infty$ and b) $t_{sim} = 1.5\tau_\infty$.

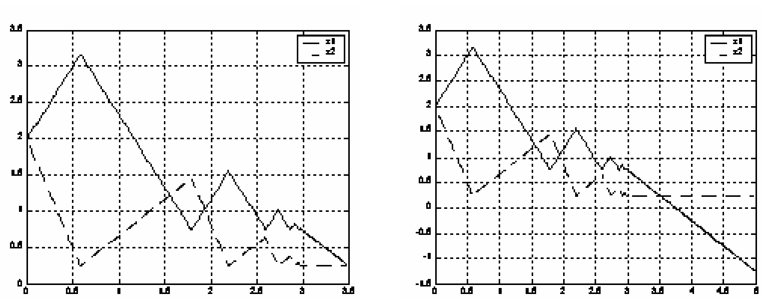
With regard to Fig. 4b, the invariants in the Zeno hybrid automaton are replaced in the regularized hybrid automaton by:

$$I(q_0) = \{\boldsymbol{x} \in X : x_2 \geq r_2 - \varepsilon\},\ I(q_1) = \{\boldsymbol{x} \in X : x_1 \geq r_1 - \varepsilon\}, \qquad (12)$$

and the corresponding guards become

$$G(q_0, q_1) = \{\boldsymbol{x} \in X : x_2 \leq r_2 - \varepsilon\},\ G(q_1, q_0) = \{\boldsymbol{x} \in X : x_1 \leq r_1 - \varepsilon\}. \qquad (13)$$

Simulations of the discrete time version of the regularized hybrid model, as depicted in Fig. 6, were performed using the same *sample time approach* with $\varepsilon = 0.25$ and with same parameter values as in Fig. 5. If the simulation time is shorter (Fig. 6a), the system switches between the two tanks, but for a longer simulation time (Fig. 6b) one the two tanks will drain, as in Fig. 5b, with different new limits and draining moment.



a b

Fig. 6 – Simulations of sampled-time model of the spatial regularized automaton for the water tank system in Fig. 4a: a) $t_{sim} = 1.4\tau_\infty$ and b) $t_{sim} = 2\tau_\infty$.

4. **CONCLUSIONS**

Hybrid systems simulation is basically difficult because it must properly deal with switching among a collection of continuous evolution laws, with eventual resetting maps over initial conditions of differential equations and with testing transition conditions, such as “zero-crossing”, among others.

Far from being only a mathematical curiosity, Zeno behaviour, characterized by infinitely many switching executed in a finite time interval, may arise from model simplifications introduced for analysis and control purposes in practical engineering. In the literature, a classic approach to study and model such situation is based on sliding modes [13].

Zeno executions introduce additional sources of problems in analysis and simulations of hybrid automata. The direct implementation of Zeno models in simulation programs drives to improper simulation evolution: the simulation either gets stuck, or it generates incorrect results. Research efforts made in past decade have lead to the concept of regularization, which basically builds a parameterised family of non-Zeno hybrid models, which converge, as the parameter tends to zero, under certain conditions, to the original Zeno hybrid model [7, 8, 11].

The approach presented in Section 3 is based on conversion of the continuous state equations of the hybrid model to a *discrete time model*, using the *c2d* MATLAB routine with *zero-order hold on the inputs* method. The switching conditions are tested at each discrete time step and simulation does *not* get stuck when the computed Zeno time is reached. Obviously, for same initial conditions, the simulated trajectories do *not* coincide with the trajectories of the Zeno hybrid automaton, but for a simulation step chosen "small enough" – here $h = 10^{-3}$ – the trajectories of the discrete time model reflect adequately the behaviour of the original Zeno hybrid system.

Starting from the *spatial* regularized automaton associated to the Zeno hybrid automaton of the water tank system proposed in [7], the corresponding hybrid model with discrete time state equations was similarly simulated. The results in this second experiment reflect adequately the modified new limits. The continuation of the trajectories obtained in this case has the form of the simulated evolution reported in [8], but which corresponds to *temporal* regularization. This one introduces a *delay* in the time between switching from one tank to another.

Thus, a concluding remark is that, according to simulation results, discretization of continuous state equations of a Zeno hybrid automaton has the effect of a *temporal regularization*.

Further research is needed to analytically investigate the relation between the evolution of the discrete-time hybrid automaton and the original Zeno evolution, as well as the influence of the discretization method. The main difficulty in this approach is related to the adequate choice not only of the discretization method, but also of the discretization step. Additionally, simulation implies a prior model analysis and evaluation of Zeno time.

Received on 21 November 2006

REFERENCES

1. M.S. Branicky, *Introduction to Hybrid Systems*, in: Hristu-Varsakelis D., Levine W.S. (Eds.), *Handbook of Networked and Embedded Control Systems*, Springer Verlag, Birkhäuser, 2005, pp. 91–116.
2. J. Zhang, K.H. Johansson, J. Lygeros, S. Sastry, *Zeno Hybrid Systems*, International Journal of Robust and Nonlinear Control, **11**, *5*, pp. 435–451, 2001.
3. H. Elmqvist, *Dymola – Dynamic Modeling Language*, *User's Manual*, Dynasim AB, Sweden, 1994.

4. S.E. Mattsson, M. Andersson, K.J. Åström, *Object-oriented modeling and simulation*, in: D.A. Linkens (Ed.), *CAD for Control Systems*, Marcel Dekker Inc., New York, 1993, pp. 31–69.
5. A. Deshpande, A Göllü, L. Semenzato, *The SHIFT programming language for dynamic networks of hybrid automata*, IEEE Trans. on AC, **43**, *4*, pp. 584–587, 1998.
6. R. Alur, L.D. Dill, *Automata for modeling real time systems*, in: *ICALP'90*, LNCS **443**, pp. 322-335, Springer Verlag, 1990.
7. K.H. Johansson, M. Egerstedt, J. Lygeros, S. Sastry, *On the regularization of Zeno hybrid automata*, Systems and Control Letters, **38**, pp. 141–150, 1999.
8. K.H. Johansson, L. Lygeros, S. Sastry, M. Egerstedt, *Simulation of Zeno hybrid systems*, IEEE Conference on Decision and Control, Phoenix, AZ, 1999 (Karl Henrik Johansson's homepage http://www.s3.kth.se/~kallej).
9. J. Zhang, K,H. Johansson, J. Lygeros, S. Sastry, *Dynamical systems revisited: Hybrid systems with Zeno executions*, in: B. Krogh, N. Lynch (Eds.), *Hybrid Systems: Computation and Control*, Springer Verlag, 2000, pp. 451–464.
10. J. Lygeros, Claire Tomlin, Sh. Sastry, *Controllers for reachability specifications for hybrid systems*, Automatica, **35**, pp. 349–370, 1999.
11. Claire Tomlin, *Analysis and Control of Hybrid Systems*, Lecture Notes, Spring 2005, http://sun-valley.stanford.edu/~tomlin.
12. R. Alur, T.A. Henzinger, *Modularity for timed and hybrid systems*. In: A. Mazurkiewicz, J. Winkowsky (Eds.), *CONCUR 97: Concurrency Theory*, LNCS **1243**, pp. 74–88, Springer Verlag, 1997.
13. V.I. Utkin, *Sliding Modes in Control Optimization*, Springer Verlag, Berlin, 1992.