

Detection of Event Occurrence in Piecewise Linear Hybrid Systems

Antoine Girard

LMC-IMAG

51 rue des mathématiques, 38041 Grenoble Cedex 9, France

e-mail: Antoine.Girard@imag.fr

Abstract: *The detection of the occurrence of events is a central fact in the study of hybrid systems. Even simulation, which is the most simple tool for analysis of hybrid systems, requires an accurate event detection process. Indeed, due to the discontinuity of these systems, an error in this domain can imply a great change in the global behaviour of solutions. Several techniques have been proposed to solve this problem. But these methods use numerical integration schemes and this approximation avoid to certify that an event has effectively occurred. In this paper, we propose a method for piecewise linear hybrid systems which is based on the symbolic expression of the flow. Not only our method allows to compute an approximation of the time at which an event occurs, but also (in most of the cases) gives a time interval where it insures that the event occurs.*

Keywords: Piecewise linear hybrid systems, event detection, symbolic-numeric methods.

1 Introduction

Hybrid systems have become the standard modelisation tool for the systems involving interactions between continuous processes and a discrete automata. The applications are numerous in fields such as avionics, automotive industry or biology.

The subclass of piecewise linear hybrid systems is doubtless one of the most used. Indeed, they have been used for a long time by engineers to approximate non-linear systems. Therefore, many tools have been proposed for the analysis of such systems. Recent work has been realized on global geometric analysis ([6]), reachability analysis ([1]), optimal control and Lyapunov stability ([4])...

A central fact in the definition of hybrid systems is the existence of events which imply some changes in the behaviour of the system. Therefore, a reliable event detection process is needed in various tasks such as simulation or verification. Traditionally, an event occurs when the solution $X(t)$ of a hybrid system verifies a condition $g(X(t)) = 0$ (g is called the guard function). There are numerous cases when classical ODE solvers might fail in detecting these events.

Figure 1 illustrates what may happen if no special process is used for the detection of event occurrence. Most of classical ODE solvers use a uniform discretization of the time to evaluate different values of the solution $X(t)$. A naive algorithm to detect the occurrence of an event is to compute two successive values $X(t_k)$ and $X(t_{k+1})$ of the solution and then to test if the sign of the guard function changed during this interval. Unfortunately, it happens (see figure 1) that the sign of the guard function changes twice during the interval $[t_k, t_{k+1}]$. If the case arises, it becomes impossible to detect the event occurrence. Moreover, due to the discontinuous nature of hybrid systems, a failure to detect the occurrence of an event may cause dramatic changes on the behaviour of the solution.

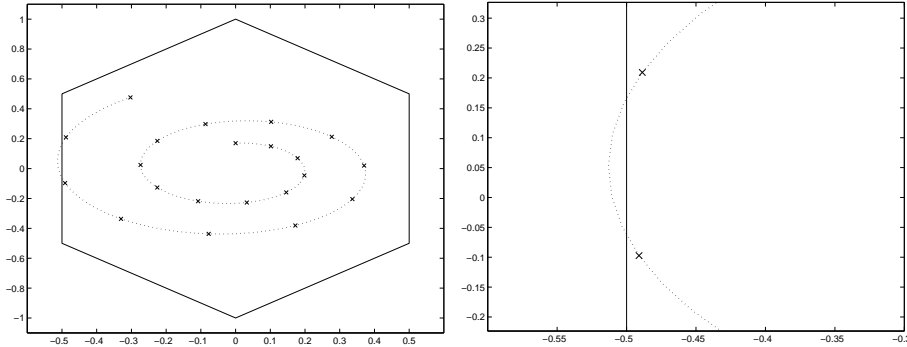


Figure 1: Example of failure of inefficient simulators, the effective solution (dotted line) exits the box but the discretized solution (crosses) does not.

Many techniques have been proposed to solve the problem of event detection. In [7], Shampine et al. proposed a more reliable method to test if an event occurred during the interval $[t_k, t_{k+1}]$. It consists in constructing a polynomial approximation of the guard function on the interval and then to apply a root-finding algorithm to this polynomial. The major drawback of this technique is that it tends to be expensive. More recently, more efficient methods were introduced. Park and Barton [5] combined the idea of polynomial approximation with methods from interval arithmetic to determine intervals where an event may have occurred. Esposito et al. [2] used techniques from control theory to compute adaptive time discretization, slowing down the simulation when an event may occur.

All these techniques are based on approximation scheme for differential equations. Therefore, they have two main drawbacks. First, the accuracy of the detection can obviously not be better than the accuracy of the approximation scheme. Moreover, the time discretization must be refined enough to keep a good accuracy and this may lead to a high number of iterations.

In this paper, we present an algorithm for the detection of events in piecewise linear hybrid systems. It is based on the symbolic expression of the flow and, as a consequence, it allows to get rid of the two limitations presented above.

2 An algorithm to detect event occurrence

The problem of event detection for piecewise linear hybrid systems can be defined as the following. Let us consider the linear initial value problem

$$\dot{X}(t) = AX(t) + b, \quad X(t) \in \mathbb{R}^n, \quad X(0) = X_0$$

where X_0 is inside (possibly on the frontier) of a n -polytope Ω . The problem to solve is

$$\text{Compute the smallest } t^* > 0 \text{ so that } X(t^*) \in \partial\Omega. \quad (1)$$

Let us define some notations in order to formulate this problem in a more tractable way. Let $\{n_1, \dots, n_q\}$ be the unitary vectors orthogonal to the faces $\{F_1, \dots, F_q\}$ of Ω and oriented to the interior of Ω . Then, there exist scalars $\{d_1, \dots, d_q\}$ so that X is a point of Ω if and only if

$$\forall j \in \{1, \dots, q\}, \quad n_j \cdot X - d_j \geq 0.$$

The problem 1 is clearly equivalent to

$$\text{Compute } t^* > 0 \text{ so that } \begin{cases} \forall j \in \{1, \dots, q\}, & \forall t \in]0, t^*[, & n_j \cdot X(t) - d_j > 0 \\ \exists j \in \{1, \dots, q\}, & & n_j \cdot X(t^*) - d_j = 0. \end{cases} \quad (1')$$

For simplicity, we note

$$\text{For } j \in \{1, \dots, q\}, \quad f_j(t) = n_j \cdot X(t) - d_j.$$

The problem 1' consists in computing the smallest strictly positive real so that it is a root of one element of the set of functions $\mathcal{F} = \{f_1, \dots, f_q\}$ (one function for each face of Ω). Therefore, it is equivalent to,

$$\text{Compute } t^* > 0 \text{ so that } \begin{cases} \forall f \in \mathcal{F}, \quad \forall t \in]0, t^*[, & f(t) > 0 \\ \exists f \in \mathcal{F}, & f(t^*) = 0. \end{cases} \quad (1'')$$

In the next section, an algorithm to compute the solution of this problem is detailed. First, a brief overview of the structure of the algorithm will be given. Afterwards, the main points will be detailed.

2.1 Overview of the algorithm

The algorithm presented here uses the fact that under-approximations and over-approximations of the solution of problem 1'' are easily computable. The detail of these computations will be explained in further paragraphs.

Data : The set of function $\mathcal{F} = \{f_1, \dots, f_q\}$
Result : An under-approximation t and an over-approximation T of t^*
repeat
 Compute an over-approximation T of t^* ;
 $\mathcal{T} = \{ \}$;
 for each function f of the set \mathcal{F} **do**
 Assuming $f(t^*) = 0$, compute a hypothetical under-approximation t_f of t^* ;
 if $t_f > T$ **then**
 $f(t^*) \neq 0$, remove f from \mathcal{F}
 else
 $\mathcal{T} = \{\mathcal{T}, t_f\}$
 end
 end
 $t = \min(\mathcal{T})$;
until desired precision reached;

Algorithm 1: Event detection algorithm

At each iteration of algorithm 1, we can point out three important steps.

- The first step consists in computing an over-approximation T of t^* , it is detailed in the next paragraph.
- The second step consists in removing of the set \mathcal{F} useless functions (i.e. these functions can be removed of \mathcal{F} without changing the solution of problem 1'').
For each function $f \in \mathcal{F}$, we assume that $f(t^*) = 0$. This allows to compute a hypothetical under-approximation t_f of t^* (c.f. paragraph 2.3). The hypothetical nature of t_f comes from the fact that we assume that $f(t^*) = 0$. Afterwards we test if t_f is greater than T . If the case arises, it means that t^* is greater than T . This is impossible. Consequently, $f(t^*) \neq 0$. Therefore, we can remove f of the set \mathcal{F} without changing anything to the solution of problem 1''.

- The third step consists in taking the minimum t of all the t_f ($t_f \leq T$) and this gives an under-approximation of t^* . In order to show this result, let us assume that t is greater than t^* , this means that for all f , $f(t^*) \neq 0$ (i.e. $t^* = +\infty$). Consequently, $t = +\infty$. This leads to a contradiction.

Algorithm 1 allows to compute successive under-approximations and over-approximations of the time t^* . In the next paragraphs, these computations are detailed. Results about the convergence of the successive values of t and T to t^* are given.

2.2 Over-approximation of t^*

Before describing the methods of approximation of t^* , we must point out some interesting properties of the functions f of the set \mathcal{F} .

First, note that the derivatives of these functions are simple, indeed

$$\forall j \in \{1, \dots, q\}, f_j'(t) = n_j \cdot X'(t) = n_j \cdot (AX(t) + b).$$

The second usefull fact is that we can compute lower and upper bounds of the second order derivative of the functions on the interval $[0, t^*]$.

Let $\{Y_1, \dots, Y_p\}$ be the smallest set of points of \mathbb{R}^n so that Ω is the convex hull of $\{Y_1, \dots, Y_p\}$. Since on the interval $[0, t^*]$, $X(t)$ is in Ω , there exist $\{\lambda_1(t), \dots, \lambda_p(t)\}$ positive reals so that,

$$X(t) = \sum_{i=1}^{i=p} \lambda_i(t) Y_i, \quad \sum_{i=1}^{i=p} \lambda_i(t) = 1.$$

Consequently, combining with the expression of $f_j^{(2)}(t) = n_j \cdot (A^2X(t) + Ab)$, we have

$$\forall t \in [0, t^*], \begin{cases} f_j^{(2)}(t) \geq m_{f_j} & = \min_{i=1}^{i=p} [n_j \cdot (A^2Y_i + Ab)] \\ f_j^{(2)}(t) \leq M_{f_j} & = \max_{i=1}^{i=p} [n_j \cdot (A^2Y_i + Ab)]. \end{cases}$$

We can now describe the method of over-approximation of t^* . Let t_k be the value of the under-approximation t , T_k the value of the over-approximation T and \mathcal{F}_k the set of functions \mathcal{F} after k iterations of algorithm 1 ($t_0 = 0$, $T_0 = +\infty$, $\mathcal{F}_0 = \{f_1, \dots, f_q\}$).

The polynomial

$$f(t_k) + (t - t_k)f'(t_k) + \frac{(t - t_k)^2}{2}M_f$$

is a second order over-approximation of the function $f(t)$ on the interval $[t_k, t^*]$. Since $f(t)$ is strictly positive on the interval $[t_k, t^*]$, the smallest root of this polynomial is necessarily greater than t^* . Therefore, we define the new value of the variable T

$$T_{k+1} = t_k + \min_{f \in \mathcal{F}_k} (S_f^k)$$

where S_f^k is the smallest strictly positive root of the polynomial

$$f(t_k) + sf'(t_k) + \frac{s^2}{2}M_f.$$

If it has no strictly positive root then we set $S_f^k = +\infty$. T_{k+1} is clearly an over-approximation of t^* . Moreover, we can show that under some assumptions (which will be detailed in the next paragraph) the convergence of the series (T_k) to t^* is quadratic.

2.3 Under-approximation of t^*

We have seen in the overview of the algorithm that the delicate step is the computation of an under-approximation of t^* . It has been shown that it is possible, using a set of hypothetical under-approximations. In this paragraph, we focus on the computation of the latter.

Let us assume that $f(t^*) = 0$. The polynomial

$$f(t_k) + (t - t_k)f'(t_k) + \frac{(t - t_k)^2}{2}m_f$$

is a second order under-approximation of the function $f(t)$ on the interval $[t_k, t^*]$. Since $f(t^*) = 0$, then the value of this polynomial at the point t^* is necessarily negative. Therefore, the polynomial has a root inside the interval $[t_k, t^*]$. We define

$$t_f^k = t_k + s_f^k$$

where s_f^k is the smallest strictly positive root of the polynomial

$$f(t_k) + s_f f'(t_k) + \frac{s_f^2}{2}m_f .$$

If it has no strictly positive root then set $s_f^k = +\infty$. It can be shown ([3]) that t_f^k is a second order under-approximation of t^* .

After computing all the hypothetical under-approximations t_f^k of t^* , we can compute an effective under-approximation t_{k+1} of t^* :

$$t_{k+1} = \min_{f \in \mathcal{F}_k} (t_f^k) .$$

We have the following result of convergence.

Theorem 1 (Convergence of the series (t_k) and (T_k)) [3]

$$\lim_{k \rightarrow \infty} t_k = t^* .$$

If $X(t)$ does not reach the frontier of Ω tangentially,

$$\lim_{k \rightarrow \infty} T_k = t^* ,$$

and both series converge quadratically.

3 Results and Conclusion

For experimentation, we are going to consider the following example.

$$\begin{cases} \dot{x}(t) = 0.1x(t) + y(t) & , x(0) = 0 \\ \dot{y}(t) = -x(t) + 0.1y(t) & , y(0) = 0.17 . \end{cases} \quad (2'')$$

We define the polytope Ω as the convex hull of the set of points $\{(0, 1), (-0.5, 0.5), (-0.5, -0.5), (0, -1), (0.5, -0.5), (0.5, 0.5)\}$. We want to compute the time t^* at which the trajectory $(x(t), y(t))$ exits Ω . The approximate value of t^* computed with the algorithm 1 is 10.8687.

On figure 2, the value of $(x(t), y(t))$ at the times t_k are plotted. Our algorithm allows to use a rough time discretization when the trajectory is far from the faces of Ω . Indeed, only twenty steps of the algorithm are needed to find an interval of size 10^{-10} which contains t^* . Therefore, the mean time step used by the algorithm is 0.5. We can see here one of the great advantage of

using the symbolic expression of the flow rather than a numerical integration scheme. Indeed, any approximate scheme would have needed time step smaller than 0.5 to insure a good accuracy. Consequently, an algorithm using this kind of techniques would have needed much more iterations than our algorithm to compute t^* with the same accuracy. The figure on the right represents the length of the interval containing t^* against the number of iteration. We can see that the convergence is quadratic which matches the theory.

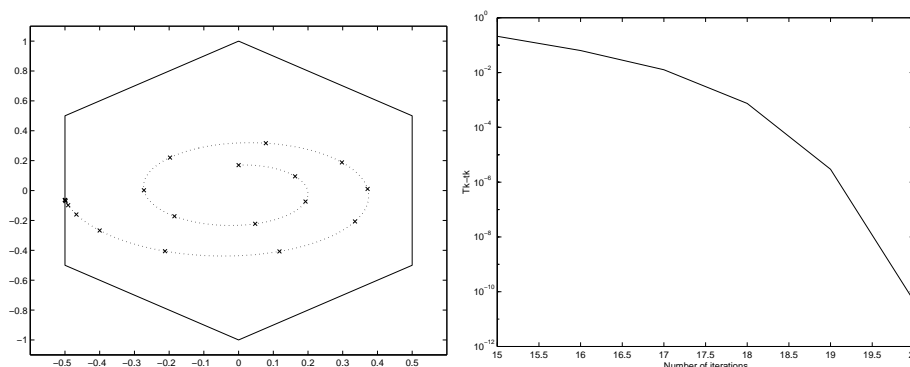


Figure 2: Left: the successive computations of $X(t_k)$ (cross). Right: $|T_k - t_k|$ against the number of iterations.

In this paper, we have presented an algorithm for event detection in piecewise linear hybrid systems. This algorithm allows to compute a convergent series of under-approximations of the solution of problem 1. In most of the case, the series of over-approximations is also convergent and both convergences are quadratic. The use of the symbolic expression of the flow allows to use our algorithm for the verification of hybrid systems.

References

- [1] E. Asarin, O. Bournez, T. Dang, O. Maler (2000). Approximate reachability analysis of piecewise-linear dynamical systems. In *HSCC'00*, no. 1790 in LNCS, pages 21-31, Springer Verlag, 2000.
- [2] J. M. Esposito, V. Kumar, G. J. Pappas (2001). Accurate event detection for simulating hybrid systems. In *HSCC'01*, no. 2034 in LNCS, pages 204-217, Springer Verlag, 2001.
- [3] A. Girard (2002). Computation of the transitions in piecewise linear hybrid systems. Technical report, available from www-lmc.imag.fr/lmc-mosaic/Antoine.Girard/.
- [4] M. Johansson, A. Rantzer (1998). Computation of piecewise quadratic Lyapunov functions for hybrid systems. In *IEEE Transactions in Automatic Control*, vol. 43, no. 4, pages 555-559, 1998.
- [5] T. Park, P. I. Barton (1996). State event location in differential-algebraic models. In *ACM Transactions on Modeling and Computer Simulation*, vol. 6, no. 2, pages 137-165, 1996.
- [6] N. B. O. L. Pettit, P. E. Wellstead (1995). Analysing piecewise linear dynamical systems. In *IEEE Control Systems*, vol. 15, no.5, pages 43-50, 1995.
- [7] L. F. Shampine, I. Gladwell, R. W. Brankin (1991). Reliable solution of special event location problems for ODEs, In *ACM Trans. Math. Soft.*, vol. 17, no.1, pages 11-25, march 1991.