Choice of Directions for the Approximation of Reachable Sets for Hybrid Systems

Xin Chen and Erika Ábrahám

RWTH Aachen University, Germany

Abstract. In this paper we propose an approach to over-approximate the reachable set (with bounded time and number of transitions) of a hybrid system by a finite set of polytopes. The constraints of the polytope are determined by a direction choice method. For the hybrid systems whose (1) continuous dynamics are linear, (2) invariants and guards are defined by linear inequalities, and (3) variable resets are expressed by invertible affine maps, we show that the over-approximations can be computed in polynomial time, and the overestimation can be arbitrarily reduced by decreasing the discretization time step if the continuous dynamics are all deterministic. Some experimental results are also presented to show the effectiveness of our approach.

1 Introduction

Hybrid systems are systems with combined discrete and continuous behavior. Typical examples of hybrid systems are physical systems, continuously evolving over time, controlled by some discrete controller, e.g., a chip or a program. Hybrid systems are often modeled by *hybrid automata*, an extension of discrete transition systems. The discrete locations model the discrete part, e.g., the states of the controller. While control stays in a location, time goes by, and the continuous quantities evolve according to some ordinary differential equations (ODEs) associated with the locations. Location invariants may force the control to move from one location to another, following discrete guarded transitions.

The *verification* of safety-critical hybrid systems is an active research area in computer science as well as in engineering sciences. For the verification of *safety properties* of hybrid systems the main challenge is to compute the set of the reachable states of hybrid automata. In general, this reachability problem is undecidable [1]. Instead of computing the exact reachable set, most approaches compute an *over-approximation.* If the over-approximation does not intersect the unsafe state set, then the system is safe, otherwise we need to refine the approximation.

In this paper, we consider hybrid systems whose continuous dynamics are defined by ODEs of the form $\dot{x} = Ax + Bu$ where A, B are constant matrices and u is an input from a bounded set $\mathcal{U}$. The invariants, transition guards and initial sets are defined by linear inequalities. The initial set should also be bounded. For each transition, the reset of the variables is defined by an invertible affine map. For such hybrid systems, many geometric objects and their representations are proposed as over-approximations of the reachable sets, such as orthogonal polyhedra [2], polyhedra [3], ellipsoids [4], zonotopes [5] and support functions [6]. These objects are generally used in a flowpipe construction manner, decomposing a time interval into small time steps and over-approximating reachability within each small time step by a geometric object. However, the representations of the objects do not have polynomial-time algorithms for all of the necessary operations. Therefore, to reduce the computation time, additional approximations are introduced, with the drawback that the overestimation is no longer reducible by shortening the time step.

Our approach is a new *trade-off* between efficiency and accuracy. We use *polytopes* (bounded polyhedra) for the over-approximations, which are defined by conjunctions of linear constraints. We show that if the continuous dynamics are deterministic, then (1) the computation time is *polynomial*, and (2) the overestimation can be *arbitrarily reduced* by decreasing the time step.

The rest of the paper is structured as follows. In Sect. 2 we introduce hybrid automata and their reachability computation. In Sect. 3 we present our direction choice method to determine over-approximations for reachable sets. After providing experimental results in Sect. 4 we conclude the paper in Sect. 5.

2 Preliminaries

Hybrid automata [7] are a popular modeling formalism for hybrid systems. A *hybrid automaton* is a tuple $HA = (Loc, Var, Flow, Trans, Inv, Init)$. The finite set *Loc* contains the *locations* and *Var* the real-valued *variables*. We also use $x = (x_1, x_2, \ldots, x_d)$ for the variables and $\dot{x} = (\dot{x}_1, \dot{x}_2, \ldots, \dot{x}_d)$ for their first derivatives. A *configuration* is a location-valuation pair $(l, v) \in Loc \times \mathbb{R}^d$. The *continuous dynamics* in location l is defined by $Flow(l)$ which is an ODE of the form $\dot{x} = Ax + Bu$ with A, B constant and $u \in \mathcal{U}$ where $\mathcal{U}$ is bounded. When $\mathcal{U}$ is singleton, we call the continuous dynamics *deterministic*. The set $Trans \subseteq Loc \times 2^{\mathbb{R}^d \times \mathbb{R}^d} \times Loc$ contains *discrete transitions* $a = (l, r, l')$ leading from l to l' and updating the variables according to the *reset* rule r. We call the set $\mathcal{G} = \{v \in \mathbb{R}^d \mid \exists v' \in \mathbb{R}^d.\ (v, v') \in r\}$ the *guard* of a. Given a location l, its *invariant* and *initial* sets are defined by $Inv(l)$ and $Init(l)$ respectively. We consider in the following systems whose invariants and guards can be defined by a conjunction of finitely many linear inequalities $c^T x \leq z$ with $c \in \mathbb{R}^d$, $z \in \mathbb{R}$.

The semantics of a hybrid automaton *HA* distinguishes between:

- *Continuous* evolution or time delay $(l, v) \xrightarrow{t} (l, v')$ is possible if there is a solution f of $Flow(l)$ with $f(0) = v$, $f(t) = v'$ and for all $0 \leq t' \leq t$, $f(t') \in Inv(l)$.
- *Discrete* evolution $(l, v) \xrightarrow{a} (l', v')$ follows a discrete transition $a = (l, r, l') \in Trans$ with $v \in Inv(l)$, $v' \in Inv(l')$ and $(l, l') \in r$.

An *execution* is a chain of continuous and discrete evolutions starting in an initial configuration. A valuation (or *state*) v *reachable* if it appears in an execution.

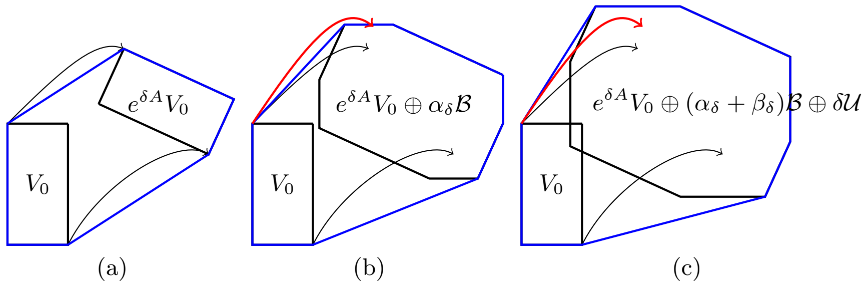

V_0
$e^{\delta A}V_0$
(a)
V_0
$e^{\delta A}V_0 \oplus \alpha_\delta\mathcal{B}$
(b)
V_0
$e^{\delta A}V_0 \oplus (\alpha_\delta + \beta_\delta)\mathcal{B} \oplus \delta\mathcal{U}$
(c)

Fig. 1. The computation of Ω_0

The *reachability problem* is to compute the set of configurations reachable from a location l_0 and an initial set V_0. As this reachability problem is undecidable [1], most algorithms *over-approximate* the reachable set by bounding the duration and the number of evolutions. For the continuous evolution with dynamics $\dot{x} = Ax + Bu$ we can over-approximate the states reachable from V_0 within time T by deviding T into N small steps of length $\delta = T/N$ and constructing the overapproximating flowpipe $\Omega_0, \ldots, \Omega_N$ with $\Omega_{i+1} = e^{\delta A}\Omega_i \oplus \mathcal{V}$ where $\oplus$ is the Minkowski sum with $X \oplus Y = \{x + y \mid x \in X, y \in Y\}$ and $\mathcal{V}$ is a set depending on B and $\mathcal{U}$ only. Each Ω_i is an over-approximation of the states reachable in the time interval $[i\delta, (i+1)\delta]$. The set Ω_0 can be computed by the convex hull of V_0 and a set which is bloated from the set $e^{\delta A}V_0$. In Fig. 1(a), we can see that the convex hull of V_0 and $e^{\delta A}V_0$ does not necessarily cover all of the trajectories even if there is no input. The purpose of bloating $e^{\delta A}V_0$ by the set $\alpha_\delta\mathcal{B}$, where $\mathcal{B}$ is a unit cube, is to include all trajectories from time 0 to δ when the input in zero. When considering inputs from $\mathcal{U}$, e.g. the behavior depicted by the red curve in Fig. 1(b) is still not covered. Hence, we bloat the result with another set $\beta_\delta\mathcal{B}\oplus\delta\mathcal{U}$. The result now includes all trajectories (see Fig. 1(c)). The value of α_δ and β_δ are given by $\alpha_\delta = (e^{\delta\|A\|}-1-\delta\|A\|)\sup_{v\in V_0}\|v\|$ and $\beta_\delta = (e^{\delta\|A\|}-1-\delta\|A\|)\frac{\sup_{u\in\mathcal{U}} B\|u\|}{\|A\|}$ respectively. The remaining Ω_{i+1}, $i{=}0,\ldots,N{-}1$, can be computed by $\Omega_{i+1}{=}e^{\delta A}\Omega_i{\oplus}\beta_\delta\mathcal{B}{\oplus}\delta\mathcal{U}$. If there is an invariant $\mathcal{I}$, we need to replace Ω_0 by $\Omega_0{\cap}\mathcal{I}$ and define $\Omega_{i+1}{=}(e^{\delta A}\Omega_i{\oplus}\beta_\delta\mathcal{B}{\oplus}\delta\mathcal{U}){\cap}\mathcal{I}$. The overestimation generated by this approach can be arbitrarily reduced by decreasing δ [8].

Over-approximating the set reachable via a discrete transition $a = (l, r, l')$, with the reset rule r defined by an invertible affine map $x' = A_r x + b$ and with guard $\mathcal{G}$, is much easier. Assume $\Omega_0, \ldots, \Omega_N$ are the flowpipe over-approximations in l. Then the reachable set after a is over-approximated by the union of the sets $\Omega'_i = A_r(\Omega_i{\cap}\mathcal{G}){+}b$, $i = 0, \ldots, N$. In order not to bring a burden in the following computation, this union is further over-approximated by its convex hull, which is viewed as the initial set in location l'.

3 Choice of the Directions for the Approximations

In this section, we present our approach to over-approximate Ω_i, $i = 0, \ldots, N$ by a polytope whose facet normals are determined by a direction choice method.

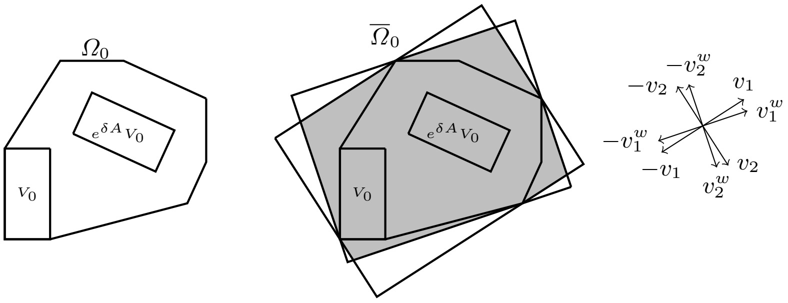
A *polytope* is a bounded polyhedron which can be represented (1) either as a $\mathcal{V}$*-polytope* by the convex hull of its vertices, (2) or as an $\mathcal{H}$*-polytope* by a conjunction $\bigwedge_{1 \leq j \leq n} c_j^T x \leq z_j$ of finitely many linear inequalities, where $\{c_1, \ldots, c_n\}$ is called its *template*.

For $\mathcal{V}$-polytopes, computing the convex hull or Minkowski sum is polynomial-time, but it is not the case for the intersection. For $\mathcal{H}$-polytopes, computing the intersection is polynomial-time, but not the convex hull or Minkowski sum. Therefore, when invariants are involved, for the flowpipe approximation using polytopes none of the representations can be used if we want to keep the computation time polynomial. Furthermore, due to the Minkowski sum in each iteration, the representation size of Ω_i can increase heavily with each iteration. To reduce this size, additional overapproximation of Ω_i is needed.

Remark 1. The sets Ω_i can also be over-approximated by other objects like, e.g., ellipsoids or zonotopes, but since they cannot exactly represent all polytopic sets, it can come to an irreducible overestimation. Another method [6] symbolically represents every Ω_i by its support function. However, the intersection of a support function and a set defined by linear inequalities is hard to compute. Though we can overcome this problem using an additional overapproximation, it again leads to a possibly irreducible overestimation.

In the following we assume that the initial set V_0 and input set $\mathcal{U}$ are $\mathcal{H}$-polytopes. We first suggest a new method to over-approximate Ω_0 by $\overline{\Omega}_0$. Assume a location l with invariant $\mathcal{I}$ and continuous dynamics $\dot{x} = Ax + Bu$, $u \in \mathcal{U}$. Since any polytope can be defined by an intersection of finitely many rectangles, we assume without loss of generality that the initial set V_0 is a rectangle. That means, V_0 can be expressed by $V_0 = QX_0$ where Q is an orthogonal matrix and X_0 is such an axis-aligned rectangle that its length in the i-th dimension is not shorter than that in the j-th dimension for all $i < j$. We use the following procedure to find the template of $\overline{\Omega}_0$:

1. Compute a matrices S_0 and S_δ whose columns are the facet centroids of V_0 and $e^{\delta A} V_0$, respectively.
2. Let $M_0 = (S_0, S_0)$ and $M_\delta = (S_0, S_\delta)$ be the matrices composed by S_0, S_0 and S_0, S_δ respectively.
3. Compute the *covariance matrices* Cov_0 and Cov_δ of M_0 and M_δ, respectively.
4. Compute the *singular value decomposition* for Cov_δ, i.e., $Cov_\delta = U_\delta \Sigma_\delta V_\delta^T$.
5. Compute the singular value decomposition for $Cov_W = QW(Q^T Cov_0 Q)Q^T + (Cov_\delta - Cov_0)$, where $W = diag(w_1, \ldots, w_d)$ with $w_1 > \ldots > w_d$ is a user defined weight matrix. We have $Cov_W = U_W \Sigma_W V_W^T$.
6. Compute the row vector sets $Temp_\delta = \{v^T \mid v \text{ or } -v \text{ is a column in } U_\delta\}$, and $Temp_W = \{v^T \mid v \text{ or } -v \text{ is a column in } U_W\}$.
7. Return $Temp = Temp_\delta \cup Temp_W$.



(a) (b)

Fig. 2. The computation of $\overline{\Omega}_0$

The set *Temp* is the template of $\overline{\Omega}_0$. In Steps 3-5 we apply *principal component analysis (PCA)* [9] and weighted PCA to find the proper directions. The template Temp_δ is the set of proper directions obtained by the standard PCA. For Temp_W, the reason to use the weighted matrix is that we need to differentiate the length of the nonparallel edges of V_0 such that we can ensure the convergence of the template Temp_W to the template of V_0 when δ converges to 0. Finally, the set $\overline{\Omega}_0$ is computed by $\bigwedge_{v\in \mathit{Temp}} v^T x \leq z_v$, where for each $v \in \mathit{Temp}$, the vector z_v is a solution for the following *linear program*:

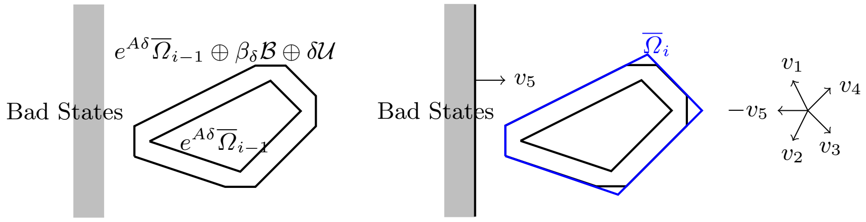
$$\begin{aligned} z = \max(\sup v^T x, \sup v^T y) \;\; s.t.\; & x \in V_0 \;\wedge\; y = y_1 + y_2 + y_3 \;\wedge \\ & y_1 \in e^{\delta A} V_0 \;\wedge\; y_2 \in \delta\mathcal{U} \;\wedge\; (\alpha_\delta + \beta_\delta)\mathcal{B} \end{aligned}$$

In the 2-dimensional example showed in Fig. 2, the vectors $v_1, v_2, -v_1, -v_2 \in \mathit{Temp}$ are computed by the standard PCA, and the vectors $v_1^w, v_2^w, -v_1^w, -v_2^w \in \mathit{Temp}_W$ by the weighted PCA. The over-approximation $\overline{\Omega}_0$ is the light gray region.

Theorem 1. *The set $\overline{\Omega}_0$ is an over-approximation of the reachable set in time $[0, \delta]$, and it converges to the exact reachable set when $\delta \to 0$.*

The first part of the theorem is clear, since $\Omega_0 \subseteq \overline{\Omega}_0$. The second part can be proved via the *matrix perturbation theory* [10].

Next, we turn to the over-approximation for Ω_i, $i = 1, \ldots, N$. Since $\Omega_i = e^{\delta A}\Omega_{i-1} \oplus \beta_\delta \mathcal{B} \oplus \delta\mathcal{U}$, we replace the recurrence relation by $\overline{\Omega}_i = \mathit{Approx}(e^{\delta A}\overline{\Omega}_{i-1} \oplus \beta_\delta \mathcal{B} \oplus \delta\mathcal{U})$. The template of $\overline{\Omega}_i$ is computed by $\mathit{Temp}_i = e^{\delta A}\,\mathit{Temp}_{i-1} \cup S_u$ where Temp_{i-1} is the template of $\overline{\Omega}_{i-1}$, and S_u is a set of all the vectors v such that $(-v)^T x \leq z_v$ is a linear inequality in the definition of the unsafe set. The set S_u is crucial for the safety verification, since it prevents the over-approximation from growing too fast towards the unsafe set. An example is presented in Fig. 3, in which the vectors v_1, v_2, v_3, v_4 are from the template of $e^{\delta A}\overline{\Omega}_{i-1}$, i.e., the set $e^{\delta A}\,\mathit{Temp}_{i-1}$, and $S_u = \{-v_5\}$. Other heuristics to refine the over-approximations by adding linear inequalities are stated in [11].


Bad States
$e^{A\delta}\overline{\Omega}_{i-1} \oplus \beta_\delta \mathcal{B} \oplus \delta \mathcal{U}$
$e^{A\delta}\overline{\Omega}_{i-1}$
Bad States
v_5
$\overline{\Omega}_i$
v_1
v_4
$-v_5$
v_2
v_3

(a) (b)

Fig. 3. The computation of $\overline{\Omega}_i$

Theorem 2. *The set $\overline{\Omega}_i$ is an over-approximation of the reachable set in time $[i\delta, (i+1)\delta]$. The total overestimation of $\overline{\Omega}_0, \ldots, \overline{\Omega}_N$ can be arbitrarily reduced by decreasing δ if $\mathcal{U}$ is singleton.*

Proof. The first part follows from $\Omega_i \subseteq \overline{\Omega}_i$. For the second part, notice that when $\mathcal{U}$ is singleton, there is no bloating from Ω_{i-1} to Ω_i, and the total overestimation only depends on the overestimation in $\overline{\Omega}_0$. As we proved before, the overestimation of $\overline{\Omega}_0$ can be arbitrarily reduced by decreasing δ, thus it is also the case for the total overestimation. □

In each location, first the approximation $\overline{\Omega}_0$ is determined before computing $\overline{\Omega}_i$ for $i = 1, \ldots, N$ until $\overline{\Omega}_{N+1} = \emptyset$ for some $N > 0$. If we use the *interior point method* [12] for solving linear programs, the total computational complexity is polynomial time.

Suppose that we computed $\overline{\Omega}_0, \ldots, \overline{\Omega}_N$ in the location l. As stated in Sect. 2, the reachable set after a discrete transition $a = (l, r, l')$ can be over-approximated by the convex hull of $\overline{\Omega}'_i = A_r(\overline{\Omega}_i \cap \mathcal{G}) + b$, where r is defined by the invertible affine map $x' = A_r x + b$ and $\mathcal{G}$ is the guard. Notice that the set $\overline{\Omega}'_i$ is also an $\mathcal{H}$-polytope and can be computed in polynomial time, since $\overline{\Omega}_i \cap \mathcal{G}$ is an $\mathcal{H}$-polytope whose inequalities are the union of the inequalities of $\overline{\Omega}_i$ and $\mathcal{G}$, and for an $\mathcal{H}$-polytope $P = \bigwedge_{1 \le j \le n} c_j^T x \le z_j$, the polytopes $A_r P + b$ can be computed by $\bigwedge_{1 \le j \le n}(c_j^T A_r^{-1} x \le z_j + c_j^T A_r^{-1} b)$. However, the representation size of the convex hull of $\overline{\Omega}'_0, \ldots, \overline{\Omega}'_N$ can be very large. Therefore, we over-approximate this convex hull by a rectangle Π as follows:

1. For each linear inequality $c^T x \le z$ of $\overline{\Omega}'_i$, we compute a point on the boundary of $\overline{\Omega}'_i$ by solving the linear program: $\sup_x c^T x \; s.t. \; x \in \overline{\Omega}'_i$.
2. Compute the covariance matrix Cov_D for the points, and compute the singular value decomposition of the covariance matrix: $Cov_D = U_D \Sigma_D V_D^T$.
3. Compute the set $Temp_D = \{v \mid v \text{ or } -v \text{ is a column of } U_D\}$.

$Temp_D$ is the template of Π. The role of PCA is to keep the over-approximation and the convex hull of the same dimension. We can also separate the $\overline{\Omega}'_i$s into several groups and over-approximate each of them by a rectangle.

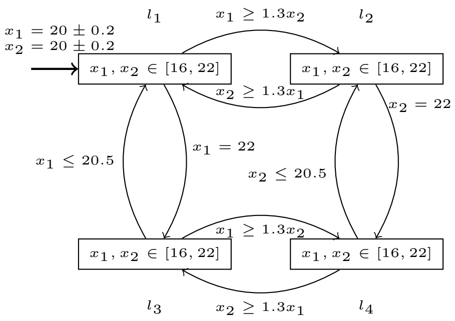


Fig. 4. The hybrid automaton

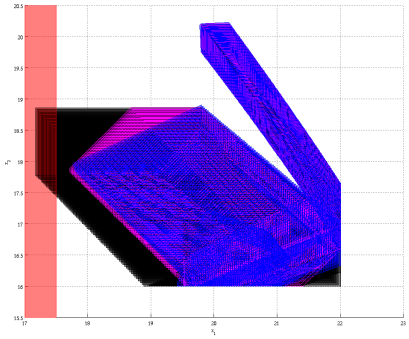


Fig. 5. Results of the experiments

Dimension	5	8	10	15
Octagons	3.97s	21.11s	66.33s	> 600s
Our method	11.74s	24.02s	34.33s	83.85s

Table 1. Average running time of 100 samples

4 Experimental Results

We apply our approach to a room heating benchmark [13]. There are two rooms and one heater. The heater can be turned off and on, and it can be moved from one room to another when some condition is satisfied. The variables x_1 and x_2 of the hybrid automaton model, shown in Fig. 4, represent the temperatures of the rooms. Locations l_1, l_2 model that the heater is on in Room 1 and Room 2 respectively, and l_3, l_4 model analogously the heater turned off. In location l_k, the variable x_i changes according to $\dot{x}_i = 9h_k + 0.3([2.5, 3.5] - x_i) + 0.3(x_j - x_i)$ where $h_1 = h_2 = 1$ and $h_3 = h_4 = 0$. Fig. 5 shows the over-approximations computed for the reachable set in time $[0, 1]$. The red region is the unsafe set. The black region is the over-approximation computed by the support function method with boxes; it has a large overestimation and intersects the unsafe set. The magenta part is produced by the support function method with octagons [14]). The blue region is the result of our method. we can see that our method is very competitive to the support function method with octagon approximations.

We investigated the scalability of the two methods on 100 continuous systems with dynamics $\dot{x} = Ax + Bu$ where A is randomly generated, B is an identity matrix, and $u \in [-0.1, 0.1]^d$. Table 1 lists the average running time[1], from that we conclude that our method has a better scalability.

[1] Platform: Matlab 2010b, Linux, Intel Core i7 2.8 GHz, 4G Memory.

5 Conclusion

We introduced the direction choice method for over-approximating flowpipes and the reachable set after a discrete transition. We find the proper directions via PCA and weighted PCA on a set of well selected samples, such that the computational complexity is polynomial-time. In [15], the authors also use PCA to find the orientations of the rectangular approximations. However, the computation time is not polynomial and by applying their method to the hybrid automata considered in this paper, the overestimation is not reducible by decreasing time step. As future work we will apply the direction choice method to nonlinear continuous dynamics with some adaptations.

References

1. T. A. Henzinger, P. W. Kopke, A. Puri, and P. Varaiya. What's decidable about hybrid automata? In *Proc. of STOC'95*, pages 373–382. ACM, 1995.
2. O. Bournez, O. Maler, and A. Pnueli. Orthogonal polyhedra: Representation and computation. In *Proc. of HSCC'99*, volume 1569 of *LNCS*, pages 46–60. Springer, 2000.
3. A. Chutinan and B. H. Krogh. Computing polyhedral approximations to flow pipes for dynamic systems. In *Proc. of CDC'98*. IEEE Press, 1998.
4. A. B. Kurzhanski and P. Varaiya. On ellipsoidal techniques for reachability analysis. *Optimization Methods and Software*, 17:177–237, 2000.
5. A. Girard. Reachability of uncertain linear systems using zonotopes. In *Proc. of HSCC'05*, volume 3414 of *LNCS*, pages 291–305. Springer, 2005.
6. C. Le Guernic and A. Girard. Reachability analysis of hybrid systems using support functions. In *Proc. of CAV'09*, volume 5643 of *LNCS*, pages 540–554. Springer, 2009.
7. R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138(1):3–34, 1995.
8. C. Le Guernic and A. Girard. Reachability analysis of linear systems using support functions. *Nonlinear Analysis: Hybrid Systems*, 4(2):250 – 262, 2010. IFAC World Congress 2008.
9. I. T. Jolliffe. *Principal Component Analysis.* Springer Series in Statistics. 2002.
10. G. Stewart and J. Sun. *Matrix Perturbation Theory.* Academic Press, 1990.
11. E. Asarin, T. Dang, O. Maler, and R. Testylier. Using redundant constraints for refinement. In *Proc. of ATVA'10*, volume 6252 of *LNCS*, pages 37–51. Springer, 2010.
12. S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.
13. A. Fehnker and F. Ivancic. Benchmarks for hybrid systems verification. In *Proc. of HSCC'04*, volume 2993 of *LNCS*, pages 326–341. Springer, 2004.
14. C. Le Guernic. *Reachability Analysis of Hybrid Systems with Linear Continuous Dynamics.* Ph.d. thesis, Université Joseph Fourier, 2009.
15. O. Stursberg and B. H. Krogh. Efficient representation and computation of reachable sets for hybrid systems. In *Proc. of HSCC'03*, volume 2623 of *LNCS*, pages 482–497. Springer, 2003.