A Comprehensive Method for Reachability Analysis of Uncertain Nonlinear Hybrid Systems

Moussa Maiga, Nacim Ramdani, Louise Travé-Massuyès, Christophe Combastel

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAC.2015.2491740, IEEE Transactions on Automatic Control

A comprehensive method for reachability analysis of uncertain nonlinear hybrid systems

Moussa Maïga, Nacim Ramdani, *Member, IEEE,* Louise Travé-Massuyès, *Member, IEEE,* and Christophe Combastel

***Abstract*—Reachability analysis of nonlinear uncertain hybrid systems, i.e. continuous-discrete dynamical systems whose continuous dynamics, guard sets and reset functions are defined by nonlinear functions, can be decomposed in three algorithmic steps: computing the reachable set when the system is in a given operation mode, computing the discrete transitions, i.e. detecting and localizing when (and where) the continuous flowpipe intersects the guard sets, and aggregating the multiple trajectories that result from an uncertain transition once the whole flowpipe has transitioned so that the algorithm can resume. This paper proposes a comprehensive method that provides a nicely integrated solution to the hybrid reachability problem. At the core of the method is the concept of MSPB, i.e. geometrical object obtained as the Minkowski sum of a parallelotope and an axes aligned box. MSPB are a way to control the over-approximation of the Taylor's interval integration method. As they happen to be a specific type of zonotope, they articulate perfectly with the zonotope bounding method that we propose to enclose in an optimal way the set of flowpipe trajectories generated by the transition process. The method is evaluated both theoretically by analysing its complexity and empirically by applying it to well-chosen hybrid nonlinear examples.**

***Index Terms*—Hybrid systems, interval analysis, reachability analysis, uncertain systems, zonotope enclosure.**

I. Introduction

Reachability analysis is a challenging issue involved in many problems, for example in model predictive control [1], [2], nonlinear or optimal control [3]–[5], game theory [6], [7], viability theory [8], estimation [9]–[12] for continuous systems. Applied to hybrid system, it is involved in addressing verification [13]–[16] and synthesis tasks for embedded systems [17]–[19]. Several methods have been developed recently for the explicit computation of reachable sets. For linear systems, they can be differentiated in two classes. The first class of methods compute overapproximations of the reachable sets by using a combination of time discretization, numerical integration and computational geometry. Various representations for the reachable sets such as polytopes [20]–[22], zonotopes [23]–[25] or ellipsoids [26] are used. The second class of methods use hybrid abstractions [27]–[30].

For nonlinear systems, the approaches proposed in the literature compute convergent approximations of the reachable set hence determine as closely as possible the true reachable set. In these methods, backward reachable sets are computed by using level set methods and viscosity solutions to the Hamilton-Jacobi-Isaacs (HJI) partial differential equation [15], [31], or via infinite dimensional linear programming with polynomial hybrid systems [32]. Other approaches try to simplify system dynamics by using conservative linearization [33], or hybridization [34], [35]. Contrary to the latter approaches, the one advocated by [36] interestingly works by relaxing the switching surface to offer a provable convergent numerical integration scheme for hybrid systems, but with no uncertainty. Finally, we find the approaches relying on validated set integration methods based on Interval Taylor Methods (ITMs) [37]–[39]. ITMs have been used for computing the reachable set of nonlinear continuous dynamical systems in the context of hybrid systems verification [37], but no parameter uncertainty was considered. In [40], an ITM was used for rigorous simulation of hybrid systems, and an effective technique was developed to enclose mode switching points as tightly as possible. In [41], an ITM was used for simulating dynamical systems with state-dependent switching characteristics, where the dimension of vectors was small. The lesson learnt from these works is that ITMs provide an interesting solution for uncertain systems but they should be used cautiously. Indeed, when either the initial state or the parameter vector are significantly uncertain, the size of the enclosures given by ITMs blows up after a few integration steps. Constraint propagation techniques combined with interval analysis tools stand as an alternative solution [42], although dealing with dynamic systems is still unmature. The SAT modulo ODE enclosure approach is a step forward in this direction [43]–[45].

In this paper, we address reachability analysis using explicit computation of the set reachable by a hybrid system over a finite time horizon that may encompass several mode transitions. Initial conditions are provided as an initial set and bounded uncertainties are considered for both the model parameters and the inputs. Hybrid reachability computation is decomposed into two steps. The first step consists in computing the reachable set when the system is in a given operation mode. This boils down to computing the reachable set for an uncertain nonlinear continuous dynamical systems, for which

This work is supported by the French National Research Agency under contract ANR 2011 INS 006 MAGIC-SPS (projects.laas.fr/ANR-MAGIC-SPS)

M. Maïga is with Univ. Orléans, INSA-CVL, PRISME, EA 4229, F45072, Orléans, and with CNRS, LAAS, Toulouse, France. mmaiga@laas.fr

N. Ramdani is with Univ. Orléans, INSA-CVL, PRISME, EA 4229, F45072, Orléans, France. nacim.ramdani@univ-orleans.fr

L. Travé-Massuyès is with CNRS, LAAS, University of Toulouse, 31031 Toulouse, France. louise@laas.fr

C. Combastel was with ENSEA, ECS-Lab, EA 3649, 95014 Cergy, France, he is now with the University of Bordeaux and the IMS research lab CNRS UMR5218, Bordeaux, France. christophe.combastel@u-bordeaux.fr.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAC.2015.2491740, IEEE Transactions on Automatic Control

one of the approaches described above can be used. The second step consists in computing the discrete transitions, which requires detecting and localizing in a reliable and conservative way when (and where) the continuous flow-pipe intersects the guard sets [24], [46]. There are few research works that address the latter problem for truly nonlinear hybrid systems, i.e. when the flowpipe is described by nonlinear differential equations and the guards are given by nonlinear functions. Most existing algorithms have been developed for linear hybrid dynamic systems, with two variations. The first type combines a polynomial approximation of the guard condition with algorithms of zero search of a polynomial [47]–[50]. This approach is effective but is not guaranteed. Moreover, it does not take into account the presence of uncertainties in the initial state and model parameters. The second type uses the advantages of zonotopes [24], [25], [51], support functions [52], [53] or polytopes [23], resulting in quite interesting algorithms that scale well with the number of continuous state variables. In the nonlinear case, one may proceed with guaranteed linearization and use the above methods [51], but at the cost of over-approximating the reachable sets. Thus, the most promising approach relies on constraint propagation methods, amongst others, Constraint Satisfaction Problems (CSP) [46], Hybrid Constraint Satisfaction (HCS) [54] and nonlinear differential equation numerical integration. This is directly applicable to nonlinear systems and naturally takes into account the presence of uncertainty. The method that we propose in this paper belongs to this category. [46] proposed a method for solving the flow/guard intersection problem for truly nonlinear hybrid systems. Continuous transitions were addressed via interval Taylor integration methods, and the event detection and localisation problems underlying flow/guard intersection were formulated as a CSP.

In our preliminary work [55], we improved the method in two ways; we implemented Lohner's QR-factorization method [56], i.e. a change of coordinates within the guaranteed numerical set integration method to control the wrapping effect, and solved the CSP at discrete transition steps by making use of a contractor that relies on bisection along one dimension only. In this paper, we push forward the above works in two ways. First, we consolidate and validate with several examples the new contractor. Then, we show how to deal with truly nonlinear reset functions while also keeping track of the change of coordinates. We thus develop an effective and guaranteed *change-of-coordinate-aware* approach to discrete transitions with nonlinear guards and nonlinear reset functions. Interestingly, the combination of the above improvements eventually reduces the overestimation for the whole hybrid flow trajectory. The *guarantee* property means that if an event occurs, our method ensures that it is detected indeed. This derives directly from the use of set computation techniques that combine exhaustive search algorithms for global system solving and verified numerical implementation via interval arithmetics and directed rounding [57]. Quite interestingly, there is no specific conditions required to ensure that discrete transitions are correctly handled.

The second contribution of the paper addresses the blow-up problem in the number of trajectories, that arises from the uncertain transitioning event time. Once the flow-pipe has fully transitioned, the state trajectory tube is decomposed in a multitude of pieces. These must be put together to resume continuous transitions according to the new mode dynamics. This problem is formulated as finding a minimal enclosure of a set of points in an n dimensional space. There is abundant literature on finding parallelotopes enclosing polytopes (or points sets) in two-dimensional and three-dimensional spaces [58]–[60]. In n dimensions, there is no known work which computes a parallelotope with minimum volume [61]. However, enclosing zonotopes which are optimal in the sum of the total length of given generators are presented in [62]. In [63], enclosing parallelotopes in n dimensions have been computed based on a principal component analysis (PCA) of the point set to be enclosed.

In [64], we sketched a method for computing an enclosing zonotope of minimal volume. The approach developed in this paper extends and improves our latter method; now, it can use one among three different size measures (the volume, the segments length or the P-radius) to minimize the size of the enclosing zonotope. In this paper, we also analyze the properties and the computational complexity of the new method and illustrate the impact of the chosen size measure on its performance in curbing the over-approximations. Interestingly, the geometrical transformation introduced by Lohner's QR-factorization method results in manipulating MSPBs, i.e. geometrical objects obtained as the Minkowski sum of a parallelotope and an axes aligned box. Noticing that MSPBs are just a specific type of zonotope, our bounding method nicely integrates with the intra-mode interval continuous integration method.

Finally, our contributions are three-fold. In addition to the ones described above, namely, the *change-of-coordinates-aware* approach to discrete transitions with nonlinear guards and nonlinear reset functions, and the MSPB trajectory merge, the third contribution of this paper resides in the integration of the above methods in a single framework for hybrid reachability. Specifically, the comprehensive interval analysis approaches to validated set integration and constraint satisfaction problems solving are combined and used consistently with theories and tools available for zonotope computations in order to address hybrid nonlinear reachability. The complete hybrid reachability method is evaluated with four well-chosen hybrid nonlinear examples: a mass-spring system, a bouncing ball, a sliding mode control output tracking and a nonlinear hybrid system built from an oscillatory network of transcriptional regulators.

The paper is organized as follows. Sect. II formulates the hybrid reachability problem. Sect. III introduces the set computation tools that are used in the paper. Then Sect IV presents the proposed interval continuous integration method and Sect V is concerned with the guard crossing problem. Sect VI provides the zonotope bounding method for trajectory merge. Properties and complexity analysis are given in Sect VII whereas Sect VIII presents the experiments performed to numerically evaluate the proposed method.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAC.2015.2491740, IEEE Transactions on Automatic Control

II. Hybrid reachability

Dynamical hybrid systems can be represented by a hybrid automaton [20] given by

$$HA = (\mathbb{Q}, \mathbb{D}, \mathbb{P}, \mathbb{F}, \mathrm{Inv}, \Sigma, \Psi, \mathbb{G}, \mathbb{A}), \tag{1}$$

where:

- $\mathbb{Q} = \{q\}$ is a set of locations, i.e. discrete state or modes;
- $\mathbb{D}$ is the definition domain of the continuous variables, $\mathbb{D} \subseteq \mathbb{R}^n$, with dimension n that may depend on q;
- $\mathbb{P} \subseteq \mathbb{R}^{n_p}$ is an uncertainty domain for model parameter vector p;
- $\mathbb{F} = \{f_q\}$ is the set of non-autonomous differential equations characterizing flow transition in mode q, of the form

$$\mathrm{flow}(q): \quad \dot{x}(t) = f_q(x(t), p, t), \tag{2}$$

  where $f_q : \mathbb{D} \times \mathbb{P} \times \mathbb{R}^+ \mapsto \mathbb{D}$ is a nonlinear function assumed sufficiently smooth over $\mathbb{D} \subseteq \mathbb{R}^n$, and $p \in \mathbb{P}$;
- Inv is an optional invariant, which assigns a domain to the continuous state space of each location:

$$\mathrm{Inv}(q): \quad \nu_q(x(t), p, t) < 0, \tag{3}$$

  where inequalities are taken componentwise, $\nu_q : \mathbb{D} \times \mathbb{P} \times \mathbb{R}^+ \mapsto \mathbb{R}^m$ is also nonlinear, and the number m of inequalities may also depend on q;
- Σ is a set of exogenous events;
- $\Psi = \{\rho_e\}_{e \in \mathbb{A}}$ is the set of reset maps, taken as continuous nonlinear functions;
- $\mathbb{G} = \{\gamma_e\}_{e \in \mathbb{A}}$ is the set of guard conditions of the form:

$$\mathrm{guard}(e): \quad \gamma_e(x(t), p, t) = 0; \tag{4}$$

  where $\gamma_e(.) : \mathbb{D} \times \mathbb{P} \times \mathbb{R}^+ \mapsto \mathbb{R}^{m'}$ is a nonlinear continuous function;
- $\mathbb{A} \subseteq \mathbb{Q} \times \mathbb{Q}$ is the set of discrete transitions $\{e = (q \to q')\}$ given by the 5-uple $(q, \mathrm{guard}, \sigma, \rho, q')$, where q and q' represent upstream and downstream locations respectively, $\sigma \in \Sigma$, $\rho_e \in \Psi$, and $\mathrm{guard} \in \mathbb{G}$.

A transition $q \to q'$ occurs when the continuous state flow reaches the guard set, i.e. when the continuous state satisfies condition (4).

Remark 1: In (1), urgent semantics are used, i.e. the guard set triggers a transition as soon as it is hit; this does not usually require the definition of invariants, which are used to force a transition when the switching is non-deterministic. Nevertheless, we keep the definition of invariant to easily specify (with no need for additional transitions) any constraints acting on the continuous state variables.

The set reachable in finite time by system (1-4) is illustrated in Fig. 1. When starting from an initial state vector $x(t_0)$ taken in $\mathbb{X}_0 \subseteq \mathbb{R}^n$, a discrete transition e occurs when the continuous flow intersects the guard set at time t_e. Then, the continuous state vector is reset as $x(t_e^+) = \rho_e(x(t_e^-))$. $\mathbb{X}_e$ is the set of all possible vectors $x(t_e)$ when vector $x(t_0)$ varies in $\mathbb{X}_0$. The reachable set may intersect a forbidden area as shown in the figure. Introducing the new state variable

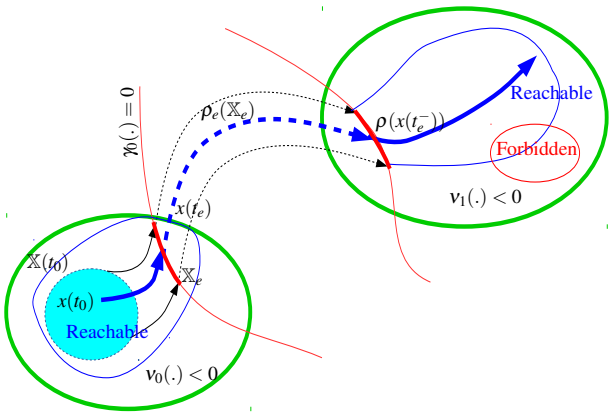


Fig. 1. Set reachable in finite time by hybrid system (1-4)

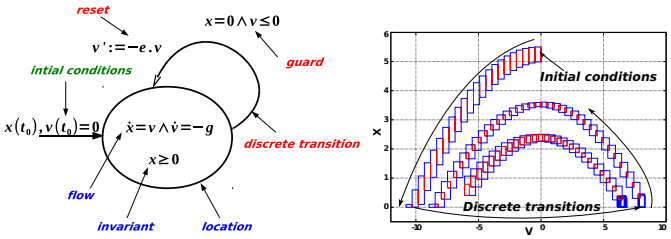


Fig. 2. A 1D bouncing ball modeled as a hybrid automaton, and the set reachable in $x \times v$-space in finite time.

$z(t) = (x(t), p, t)$ with $\dot{z}(t) = (\dot{x}(t), 0, 1)$, and defining its domain $\mathbb{Z} = \mathbb{D} \times \mathbb{P} \times \mathbb{R}^+$, equations (2–4) are rewritten:

$$\mathrm{flow}(q): \quad \dot{z}(t) = f_q(z(t)), \tag{5}$$

$$\mathrm{Inv}(q): \quad \nu_q(z(t)) < 0 \quad \text{and} \tag{6}$$

$$\mathrm{guard}(e): \quad \gamma_e(z(t)) = 0. \tag{7}$$

so that all uncertain quantities are embedded in the state vector. In the following, for sake of simplicity, the dependence on time of z and of the other time-dependent variables is omitted when not ambiguous.

When analyzing hybrid systems, intersections with guard sets that enable discrete transitions may occur, and when a flow-pipe of non-zero size reaches a guard condition, there is a non-empty set of instants during which the constraints are satisfied, leading to a *continuum* of switching times [45]. The problem of set-membership guard crossing can then be divided into three tasks:

- Event detection, i.e. detecting when a guard condition is satisfied ;
- Event localization, i.e. computing the state subset which intersects the guard condition;
- Discrete transition, i.e. computing the image of the latter subset by the reset function.

Fig. 2 illustrates the automaton modeling the hybrid dynamics of a 1D bouncing ball. In this example, there is only one location with invariant $x \geq 0$ and flow equations $\dot{x} = v$, and $\dot{v} = -g$. Position x and velocity v are the continuous time-dependent variables of this system. The ball falls freely until it reaches the ground ($x = 0$). If this occurs with negative velocity, the ball bounces and the reset function applies :

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAC.2015.2491740, IEEE Transactions on Automatic Control

velocity changes direction and the ball looses some of its energy, the reset velocity is $v' = -e.v$, $e \in [0,1]$. e and g are constants. The figure also shows the result of reachability analysis for the bouncing ball and illustrates the interlinkage of flow and discrete transitions.

III. Set computation mathematical tools

In this section, we overview key concepts regarding methods based on interval analysis that we use for finding intersections of the flow with invariants or guards, and for evaluating jump functions. Zonotopes are also introduced because we use particular zonotopes defined as MSPB to limit the over-approximation resulting from enclosing an arbitrary shaped set into a box.

Consider the system of m (in)equalities over $z \in \mathbb{Z}$

$$\mathscr{C} : \wedge_{1\leq i\leq m}(h_i(z) \prec 0), \prec \in \{=,<\}. \tag{8}$$

Inequalities refer to mode invariants whereas equalities are considered when one addresses flow/guard intersection and the evaluation of jump successors. Considering that $\mathbb{Z}$ is the domain of z, the constraint (8) can be viewed as a *numerical constraint satisfaction problem* (CSP) written in compound form as

$$\mathscr{E} := (\mathscr{C}, \mathbb{Z}). \tag{9}$$

Denoting by $\mathscr{S}$ its set of solutions, we have

$$\mathscr{S} = \{z \in \mathbb{Z} \mid \wedge_{1\leq i\leq m}(h_i(z) \prec 0)\}. \tag{10}$$

An enclosure of $\mathscr{S}$ can be computed in a reliable and guaranteed way via branch-and-prune approaches using interval analysis and contractors based on constraint propagation [57].

A. Intervals analysis

A real interval $[u] = [\underline{u}, \overline{u}]$ is a closed and connected subset of $\mathbb{R}$ where $\underline{u}$ represents the lower bound and $\overline{u}$ represents the upper bound. The width of $[u]$ is defined by $\mathrm{wid}([u]) = \overline{u} - \underline{u}$, its midpoint by $\mathrm{mid}([u]) = (\overline{u} + \underline{u})/2$, and its radius by $\mathrm{rad}([u]) = (\overline{u} - \underline{u})/2 = \mathrm{wid}([u])/2$. A interval $[u]$ can be defined by its midpoint and its radius, so $[u] = [\mathrm{mid}([u]) - \mathrm{rad}([u]), \mathrm{mid}([u]) + \mathrm{rad}([u])]$. The unitary interval is $\mathbf{B} = [-1,1]$. The set of all real intervals of $\mathbb{R}$ is denoted $\mathbb{IR}$. Two intervals $[u]$ and $[v]$ are equal if and only if $\underline{u} = \underline{v}$ and $\overline{u} = \overline{v}$. Real arithmetic operations can be extended to intervals [57] and defined as: $\circ \in \{+,-,*,/\}$, $[u] \circ [v] = \{x \circ y \mid x \in [u],\ y \in [v]\}$. An interval vector (or box) $[X]$ is a vector with interval components and may equivalently be seen as a Cartesian product of scalar intervals $[X] = [x_1] \times [x_2] ... \times [x_n]$. The set of $n-$dimensional real interval vectors is denoted by $\mathbb{IR}^n$. A unitary box in $\mathbb{IR}^n$, denoted by $\mathbf{B}^n$, is a box composed by n unitary intervals. An interval matrix is a matrix with interval components. The set of $n \times m$ real interval matrices is denoted by $\mathbb{IR}^{n\times m}$. Classical operations for interval vectors (resp. interval matrices) are direct extensions of the same operations for real vectors (resp. real matrices) [57] Given the function $f : \mathbb{R}^n \to \mathbb{R}^m$, the range of f over an interval vector $[u]$ is given by: $f([u]) = \{f(x) | x \in [u]\}$. The interval function $[f]$ from $\mathbb{IR}^n$ to $\mathbb{IR}^m$ is an inclusion function for f if: $\forall [u] \in \mathbb{IR}^n,\ \ f([u]) \subseteq [f]([u])$. An inclusion function of f can be obtained by replacing each occurrence of a real variable by its corresponding interval and by replacing each standard function by its interval evaluation. Such a function is called the *natural inclusion function*. In practice a function f has not a unique inclusion function, and the overapproximation of a given inclusion function depends on its formal expression.

B. Branch-and-prune algorithms

Consider system (9) and the case when " $\prec$ " is " $=$ ". Constraint satisfaction algorithms work as follows (see also Algorithm 1). Consider a list of candidate boxes, pick the first element in the list, say the box $[z]$, use interval analysis to check whether $[z]$ is consistent or not with the constraints in (8). If there is no $z \in [z]$ that satisfies (8), then discard the box $[z]$. Otherwise, if $[z]$ is too small or if $\max_i \|h_i([z])\|$ is also small enough, then add $[z]$ to the outer solution set. Else, bisect $[z]$ into two sub-boxes and add the two new boxes to the list. The algorithm eventually yields an outer solution set, i.e. a list of solution boxes, composed of boxes for which none of the constraints is violated and that are small enough.

Algorithm 1: Interval-Solve

input : '$\wedge_{1\leq i\leq m} h_i(z) = 0$', $\mathbb{Z}$, ε_1, ε_2
output: list of solution boxes $\overline{\mathscr{S}}$

1 define a running list of boxes $\mathscr{L}$ and initialize it with $[z] = \mathrm{Hull}(\mathbb{Z})$;
3 **while** $\mathscr{L} \neq \emptyset$ **do**
4 pick first box $[z]$ from the list;
5 evaluate $h_i([z])$;
6 **if** $\exists i : 0 \notin h_i([z])$ **then**
7 discard box $[z]$;
8 **else if** $((\|[z]\| < \varepsilon_1) \vee (\max_i \|h_i([z])\| < \varepsilon_2))$ **then**
9 store box $[z]$ in list $\overline{\mathscr{S}}$
10 **else**
11 bisect $[z]$ and store new boxes in $\mathscr{L}$;
12 **end if**
13 **end while**

When " $\prec$ " is " $<$ ", it suffices to replace the test on line 6 by $\exists i : \mathrm{Inf}(h_i([z])) \geq 0$, and use the condition $(\|[z]\| < \varepsilon_1) \vee (\max_i(\overline{h}_i([z])) < \varepsilon_2)$ on line 8. Thresholds ε_1 and ε_2 are tuned by the user. Clearly, this simple algorithm is of exponential complexity but several technical and heuristic improvements make it possible to control the overall computation time and memory storage [57]. Bisection strategies and the possible use of interval narrowing procedures, i.e. contractors, can also be quite efficient [65].

C. Contractors

The idea underlying contractors is to use a function that narrows the size of the box $[z]$ during the branching scheme of algorithm `Interval-Solve` without using bisection. This narrowing can be achieved by an interval narrowing operator, called a contractor for (8) on $[z]$, which we write as

$$[z]' = \texttt{Contractor}(\mathscr{C}, [z]).$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAC.2015.2491740, IEEE Transactions on Automatic Control

This operator removes from $[z]$ subsets that do not contain solutions of (9) and satisfies the following properties:
(*a*) $[z]' \subseteq [z]$, and (*b*) $[z]' \cap \mathscr{S} = [z] \cap \mathscr{S}$, where $\mathscr{S}$ is the solution set defined by (10).

Most contractors use consistency filtering techniques and/or constraint propagation. Interval propagation techniques are based on the interval extension of the local Waltz filtering. Consistency filtering techniques rely on local consistency properties. (see [57], [65] and the references therein).

D. Zonotopes

Enclosing an arbitrary shaped set into a box may result in large over-approximation because the edges of a box are always parallel to the axes of the considered referential. Zonotopes have been shown to limit over-approximation and their properties make them a quite interesting alternative in many situations [66]–[68].

Given a vector $c \in \mathbb{R}^n$ and a matrix $R \in \mathbb{R}^{n\times p}$, a zonotope $\mathtt{Z}$ is the set

$$\mathtt{Z} \equiv c \oplus R\mathbf{B}^p = \{c + Rx : x \in \mathbf{B}^p\}.$$

The vector c is the center of the zonotope, and the matrix R defines the shape of the (centrally symmetric) zonotopic domain. $\mathtt{Z}$ is the Minkowski sum of the m-segments defined as m columns of the matrix R in $\mathbb{R}^{n\times p}$. Zonotopes are characterized by the following properties.

Proposition 3.1: The Minkowski sum of two zonotopes $\mathtt{Z}_1 = c_1 \oplus R_1\mathbf{B}^{p_1} \in \mathbb{R}^n$ and $\mathtt{Z}_2 = c_2 \oplus R_2\mathbf{B}^{p_2} \in \mathbb{R}^n$ is also a zonotope, defined as $\mathtt{Z}_1 + \mathtt{Z}_2 = (c_1 + c_2) \oplus [R_1 \quad R_2]\mathbf{B}^{p_1+p_2}$.

Proposition 3.2: The image of a zonotope $\mathtt{Z} = c \oplus R\mathbf{B}^p \subseteq \mathbb{R}^n$ by a linear mapping L can be computed by a standard matrix product $L\mathtt{Z} = Lc \oplus (LR)\mathbf{B}^p$.

Theorem 1 (Inclusion of a family of zonotopes [66]): Consider a family of zonotopes represented by $\mathtt{Z} = c \oplus [M]\mathbf{B}^p$, where $c \in \mathbb{R}^n$, and $[M] \in \mathbb{IR}^{n\times p}$ is an interval matrix. The family of zonotopes $\mathtt{Z}$ is tightly outer-bounded by the following zonotope

$$\mathtt{Z} \subseteq \Diamond(\mathtt{Z}) \equiv c \oplus J\mathbf{B}^{p+n},$$

where matrix $J \in \mathbb{R}^{n\times(n+p)}$ is defined as

$$J = [mid([M]) \,|\, G]. \tag{11}$$

$[.\,|\,.]$ denotes classical matrix concatenation and $G \in \mathbb{R}^{n\times n}$ is a diagonal matrix that satisfies

$$G_{ii} = \sum_{j=1}^{m} rad([M]_{ij}),\ i = 1,..,n. \tag{12}$$

Theorem 2 (Zonotope extension [66], [68]): Consider a function $F : \mathbb{R}^n \to \mathbb{R}^n$ with continuous derivatives, a zonotope $\mathtt{Z} = c \oplus R\mathbf{B}^p$, and an interval matrix $[M] \in \mathbb{IR}^{n\times p}$. We have $\nabla F(\mathtt{Z})R \subseteq [M] \Rightarrow F(\mathtt{Z}) \subseteq F(c) \oplus \Diamond([M]\mathbf{B}^p)$, where $\Diamond(.)$ is defined in Theorem 1.

IV. Enclosing uncertain non linear continuous flows via interval analysis

In this section, we briefly overview the main ideas underlying non-linear continuous reachability analysis using guaranteed set integration via interval Taylor methods, including the control of the wrapping effect. In the sequel, we focus on reliable numerical methods that deal naturally with non-linear dynamical systems.

A. Interval Taylor based guaranteed set integration

Consider the *uncertain* non-linear dynamical system described by (5)–(7) with $z(t_0) \in \mathbb{Z}_0$ at time $t_0 \geq 0$ and denote by $\mathbb{Z}(t;t_0,\mathbb{Z}_0)$ the set of solutions of (5) at time t originating from each initial condition in $\mathbb{Z}_0$ at t_0. $\mathbb{Z}(t;t_0,\mathbb{Z}_0)$ is abbreviated as $\mathbb{Z}(t)$ when there is no ambiguity.

Define a time grid $t_0 < t_1 < t_2 < \ldots < t_{n_T}$, which does not need to be equally spaced, and assume that the initial domain is an interval vector; i.e. $\mathbb{Z}_0 = [z_0] = [\underline{z}_0, \overline{z}_0]$. Then, guaranteed set integration via interval Taylor methods computes interval vectors $[z_j], j = 1,\ldots,n_T$, that are *guaranteed* to contain the set of solutions $\mathbb{Z}(t_j;t_0,\mathbb{Z}_0)$ of (5) at times t_j, $j = 1,\ldots,n_T$, in three stages:

- verification of the existence and uniqueness of the solution using the Banach fixed point theorem and the Picard-Lindelöf operator [56],
- computation of an a priori enclosure $[\widetilde{z}_j]$ such that $[\widetilde{z}_j] \supseteq \mathbb{Z}(t)$ for all t in $[t_j,t_{j+1}]$. Hence, $[\widetilde{z}_j]$ is indeed an over-approximation of the reachable set over $[t_j,t_{j+1}]$. It can be made as tight as possible in the following stage.
- computation of a tighter enclosure for the set of solutions of (5) at t that can be taken as t_{j+1} or any $t \in [t_j,t_{j+1}]$, not necessarily belonging to the time-grid, using a Taylor series expansion of order k of the solution at t_j, where $[\widetilde{z}_j]$ is used to enclose the remainder term:

$$\mathbb{Z}(t;t_j,[z_j]) \supseteq [z](t;t_j,[z_j]) = [z_j] + \sum_{i=1}^{k-1} (t-t_j)^i \mathtt{f}_q^{[i]}([z_j]) + (t-t_j)^k \mathtt{f}_q^{[k]}([\widetilde{z}_j]), \tag{13}$$

where the $\mathtt{f}_q^{[i]}([z_j])$ are the Taylor coefficients of the solution, which are computed numerically via automatic differentiation.

It is well known that the scheme (13) is width increasing, and thus not suitable for numerical implementation. This scheme suffers from the wrapping effect, which is the over-approximation induced by enclosing a set of any shape in an axis-aligned box. Therefore, effective numerical methods use the mean-value form, matrix preconditioning and linear transformations.

In this paper, we control wrapping using the *mean-value* approach [56]. At each time step t_j, the solution enclosure is computed in the form

$$\mathbb{Z}(t;t_j,[z_j]) = \{v + A(t)r \,|\, v \in [v](t), r \in [r](t)\}, \tag{14}$$

and

$$\mathbb{Z}_{j+1} = \mathbb{Z}(t_{j+1};t_j,[z_j]) = \{v + A_{j+1}r \,|\, v \in [v_{j+1}], r \in [r_{j+1}]\}. \tag{15}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAC.2015.2491740, IEEE Transactions on Automatic Control

The performance of the method relies significantly on the choice of matrices A_j. An effective method introduced by Lohner uses QR-factorization [56].

Remark 2: Eq. (13) is written for any t in $[t_j,t_{j+1}]$. It is an extension of what is classically done for guaranteed set integration, since latter methods aim at computing tight enclosures for time instants taken on the grid. Here, for solving the flow/guard intersection we need an explicit characterization of the solution for any time instant taken between two time grid points. This is summarized in the following proposition.

Proposition 4.1 (Conservative polynomial interpolation): Eq. (13) is a conservative polynomial interpolation, hence acts as an analytical solution for the flow-pipe for t in $[t_j,t_{j+1}]$, since $\mathsf{f}_q^{[k]}([\widetilde{z}_j])$ encloses the remainder of the Taylor series for any t in $[t_j,t_{j+1}]$ [38], [46].

Defining:

$$[\chi](t) \equiv \{[z](t), \hat{z}(t), [v](t), [r](t), A(t)\}, \tag{16}$$

where $\hat{z}(t) := \mathrm{mid}\,([z](t))$, the algorithm $\varphi^{QR}(.)$ given in Table 2 is used in this paper to compute the solution set of (5) at time $t \in [t_j, t_{j+1}]$ [46]. The solution enclosure at time t is given by $[\chi](t) = \varphi^{QR}([\chi_j], t_j, t, [\widetilde{z}_j])$.

Algorithm 2: Algorithm φ^{QR}

input : $[\chi_j]$, t_j, t, $[\widetilde{z}_j]$
output: $[\chi](t)$

$[v](t) := \hat{z}_j + \sum_{i=1}^{k-1} (t-t_j)^i \mathsf{f}_q^{[i]}(\hat{z}_j) + (t-t_j)^k \mathsf{f}_q^{[k]}([\widetilde{z}_j])$;
$[S](t) := \mathbb{I} + \sum_{i=1}^{k-1} (t-t_j)^i \frac{\partial \mathsf{f}_q^{[i]}}{\partial z}([z_j])$;
$[q](t) := ([S](t)A_j)[r_j] + [S](t)([v_j] - \hat{z}_j)$;
$[z](t) := [v](t) + [q](t)$;
obtain $A(t)$ via QR-factorization of $\mathrm{mid}\,([S](t)A_j)$ [56];
$[r](t) := A(t)^{-1}([S](t)A_j)r_j + (A(t)^{-1}[S])([v_j] - \hat{z}_j)$;
$\hat{z}(t) := \mathrm{mid}\,([v](t))$;
$[\chi](t) := \{[z](t), \hat{z}(t), [v](t), [r](t), A(t)\}$;

Proposition 4.2: The solution domain (14) is the Minkowski sum[1] of a parallelotope, i.e. an oriented box, and an aligned box, abbreviated as an MSPB.

$$\mathbb{Z}(t) = A(t)[r](t) \oplus [v](t). \tag{17}$$

An MSPB is a particular zonotope generated by $2n$ line segments (see Fig. 3):

$$\mathbb{Z}(t) = c(t) \oplus R(t)\mathbf{B}^{2n}, \tag{18}$$

where, for all t, the point vector $c(t) \in \mathbb{R}^n$ and the point matrix $R(t) \in \mathbb{R}^{n\times 2n}$ satisfy:

$$c(t) = A(t)\mathrm{mid}\,([r](t)) + \mathrm{mid}\,([v](t)), \tag{19}$$
$$R(t) = [A(t)\mathrm{dr}\,([r](t)) \mid \mathrm{dr}\,([v](t))]. \tag{20}$$

where $\mid$ denotes matrix concatenation, and $\mathrm{dr}\,(.)$ is short for $\mathrm{diag}(\mathrm{rad}\,(.))$, i.e. a diagonal matrix of real numbers each corresponding to the radius of an interval number.

[1]Let $\xi_1, \xi_2 \subset \mathbb{R}^n$, the Minkowski sum of ξ_1 and ξ_2 is: $\xi_1 \oplus \xi_2 = \{s_1 + s_2 | s_1 \in \xi_1, s_2 \in \xi_2\}$.

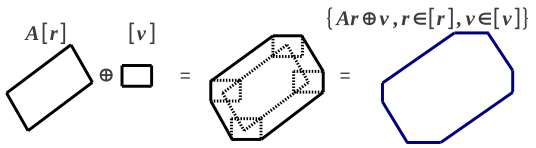


Fig. 3. MSPB

Proof: It is straightforward to see that for all t,

$$\mathbb{Z}(t) = \{A(t)r + v \,|\, v \in [v](t), r \in [r](t)\} = A(t)[r](t) \oplus [v](t).$$

Recall that a centered zonotope $\mathtt{Z}(R)$ generated by a matrix $R \in \mathbb{R}^{n\times p}$ can be defined as the linear image of the unit hypercube $\mathbf{B}^p$ by R, $\mathtt{Z}(R) = \{R\sigma, \sigma \in \mathbf{B}^p\}$, or, alternately, as the Minkowski sum of the generator segments defined by the columns of R [67]. It is then straightforward to see that (18) holds. ■

V. Set-membership guard crossing

A. Event detection and localization

We now show how to compute the geometrical intersection of a continuous flow-pipe with the guard sets[2].

The issue is first to detect if the flow-pipe intersects a guard set, then to compute when and where the intersection occurs, in other words we need to compute the time instants t_e and solution state vector $z(t_e)$ such that (7) is satisfied, i.e. $\gamma_e(z(t_e)) = 0$. Because the flow-pipe has a non-zero volume, there is a continuum of time instants $\mathscr{T}^\star$ when the intersection occurs, and it occurs for state vectors gathered in the set $\mathscr{Z}^\star$. Hence, we need to characterize the set of all such solutions, i.e.

$$\mathscr{T}^\star \times \mathscr{Z}^\star = \{t_e \times z(t_e) \text{ such that } (t_e \in [t_j, t_{j+1}]) \wedge (\gamma_e(z(t_e)) = 0) \wedge (\dot{z}(t) = f_q(z)) \wedge (z(t_j) \in [z]_j)\} \tag{21}$$

Let us assume that an event exists for $t_e \in [t_j, t_{j+1}]$, then the methods described in the sequel are able to detect the existence of such event. Computing the solution set (21) is now an analytical problem since algorithm $\varphi^{QR}(t)$ yields for any t in $[t_j, t_{j+1}]$ an analytical solution for the tube of trajectories over the time interval $[t_j, t_{j+1}]$, hence the method described in section III applies directly. To obtain a tight characterization of (21), we need to partition the search space. Therefore, to curb computational complexity and keep it polynomial time, we further use the guaranteed relaxation introduced in [55]: we use the algorithm `Interval-Solve` presented in section III but bisect only along the single direction of the time variable and use contractors to reduce the solution set at a given time instant.

Let us denote $[t^\star]_l = [\underline{t}^\star, \overline{t}^\star]_l \subseteq [t_j, t_{j+1}]$ a sub-interval over which (7) is satisfied, and $\mathscr{Z}_l^\star$ the set of state vectors for which t exists in $[\underline{t}^\star, \overline{t}^\star]_l$ that satisfy (7). Solution set (21) can then be over-approximated by

$$\mathscr{T}^\star \times \mathscr{Z}^\star \subseteq \bigcup_{l=1}^{L} [\underline{t}^\star, \overline{t}^\star]_l \times \mathscr{Z}_l^\star \tag{22}$$

[2]The same computational methods apply for the intersection with invariant sets.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAC.2015.2491740, IEEE Transactions on Automatic Control

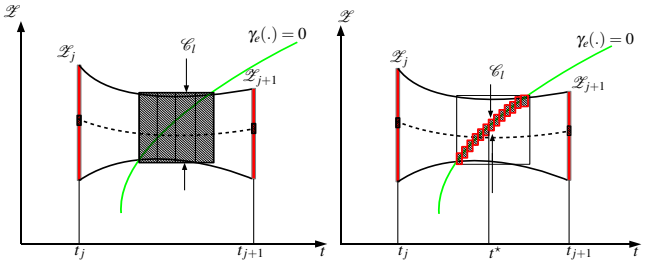


Fig. 4. Illustration of our intersection computation algorithm

where L is the number of solution sub-boxes. Finally, our idea is to contract the flow tube over tiny time slots to compute the smallest solution boxes enclosing the intersection between the tube and the guard set, as depicted in Fig. 4.

Allowing some over-approximation when computing t_e, we say that an event occurs over the subinterval $[\underline{t}^\star,\overline{t}^\star]_l$ if $\text{wid}\,([\underline{t}^\star,\overline{t}^\star]_l)$ is smaller than a given threshold ε_T, as suggested in [46]. Now the question remains about how to compute a tight over-approximation of $\mathscr{Z}_l^\star$. We solve this issue by solving a constraint satisfaction problem as in [55]. This resolution is repeated for $l=1,\ldots,L$. Recall that the enclosure of the solution of (5), which is an Initial Value Problem ODE, can be computed for the small time interval $[t^\star]_l=[\underline{t}^\star,\overline{t}^\star]_l$ in the compound form (16), denoted $[\chi]_l$, using an inclusion function for algorithm $\varphi^{QR}(t)$, which we obtain by using an interval $[t]$ as input. We save the notation $[\chi^\star]_l$ for the compound form characterizing the tight over-approximation of $\mathscr{Z}_l^\star$. We have

$$[\chi]_l=\{[z]_l,\hat{z}_l,[v]_l,[r]_l,A_l^\star\}. \tag{23}$$

Let us assume that $\text{wid}\,([\underline{t}^\star,\overline{t}^\star])\leq\varepsilon_T$. Therefore

$$\begin{aligned}(\exists z\in[z]_l \text{ s.t. } \gamma_e(z)=0)\Rightarrow\\ (\exists v\times r\in[v]_l\times[r]_l \text{ s.t. } \gamma_e(v+A^\star r)=0),\end{aligned} \tag{24}$$

hence, computing the intersection over the time interval $[t^\star]$ boils down to solving the CSP

$$\mathscr{E}_l:=(\mathscr{C}_l,[v]_l\times[r]_l). \tag{25}$$

$$\text{where}\quad \mathscr{C}_l:=(\gamma_e(v+A_l^\star r)=0). \tag{26}$$

Using a contractor, we can obtain a tight over-approximation of $\mathscr{Z}^\star$ as a compound form $[\chi^\star]$, i.e.

$$[v^\star]_l\times[r^\star]_l=\texttt{Contractor}(\mathscr{C}_l,[v]_l\times[r]_l). \tag{27}$$

Here, we use the forward-backward contractor implemented in the `HC4_Revise` method of the IBEX toolbox (www.ibex-lib.org) [65]. In fact, since the guard-set condition (7) is naturally defined in the axis-aligned $z-$space and the trajectory tube is defined as an MSPB, we need to map the solution of (7) naturally characterized in the $z-$space into the MSPB $v\times r-$space; as implied in (24). Doing so, we inevitably introduce over-approximation. To curb the latter, the idea is to combine the solutions of two contractors as now commonly done when building solvers for CSPs (see [57], pp.90). Therefore, constraint (26) is rewritten using redundant constraints as follows

$$\mathscr{C}_l^R:=(\gamma_e(v+A_l^\star r)=0)\wedge(\gamma_e(z)=0)\wedge(z=v+A_l^\star r), \tag{28}$$

whose solution is obtained via

$$[v^\star]_l\times[r^\star]_l\times[z^\star]_l=\texttt{Contractor}(\mathscr{C}_l^R,[v]_l\times[r]_l\times[z]_l). \tag{29}$$

If the solution set for CSP (29) is not empty, we assume that the event $e=q\rightarrow q'$ occurs at $t_e=\underline{t}^\star$ and that $[\chi](t_e^-)=[\chi^\star]_l$, as suggested in [46]. The discrete transition can then be computed from there, using the reset function ρ_e. The performance of the set computation of the reset function is addressed in the next subsection.

B. Set-membership reset mapping

Let us consider a sub-box $\mathscr{Z}_l^\star$, $l\in 1,...,L$, among those defined in (22). After reset, the continuous transition resumes from the set $\mathscr{Z}_l^{\star\prime}=\rho_e(\mathscr{Z}_l^\star)$. Since our continuous reachability tool works with sets characterized as MSPBs, we need to characterize $\mathscr{Z}_l^{\star\prime}$ as an MSPB. The image of a zonotope by a nonlinear mapping is not, in general, a zonotope. But using methods described in [68] and recalled in section III-D, we can compute a tight bounding zonotope for $\mathscr{Z}^{\star\prime}$.

Our method for set-membership reset mapping uses the properties of zonotopes given in section III-D as explained below.

Theorem 3 (Nonlinear reset of an MSPB): Given a solution box in MSPB form $\mathscr{Z}^\star=A[r]+[v]$, the image $\mathscr{Z}^{\star\prime}=\rho(\mathscr{Z}^\star)$ is contained in an MSPB computed as follows

$$\mathscr{Z}^{\star\prime}\subseteq A'[r]'+[v]' \tag{30}$$

where

$$A'=\tilde{J}(:,1:n)\in\mathbb{R}^{n\times n}, \tag{31}$$

$$[r]'=\mathbf{B}^n\subseteq\mathbb{R}^n, \tag{32}$$

$$[v]'=\rho_e(c)+\square(\tilde{J}(:,n+1:3n)\mathbf{B}^{2n})\subseteq\mathbb{R}^n, \tag{33}$$

where $\square(.)$ denotes the convex hull of a set. Matrix $\tilde{J}$ is defined by

$$\tilde{J}=\text{sort}(J),\quad J=[mid([\nabla\rho_e]R)\,|\,G]\in\mathbb{R}^{n\times 3n}, \tag{34}$$

where sort$(.)$ operator sorts the columns of a matrix according to their norm, vector c is defined in (19), matrix $R\in\mathbb{R}^{n\times 2n}$ in (20), and matrix $G\in\mathbb{R}^{n\times n}$ given by (12), with $m=2n$. Here $\tilde{J}(:,1:n)$ denotes the n first column vectors of matrix $\tilde{J}$, and $\tilde{J}(:,n+1:3n)$ the $2n$ last column vectors of matrix $\tilde{J}$.

Proof: From (18), we have $\mathscr{Z}^\star=A[r]+[v]=c\oplus R\mathbf{B}^{2n}$. By theorem 2, we have $\rho_e(\mathscr{Z}^\star)=\rho_e(c)\oplus\Diamond([\nabla\rho_e(\mathscr{Z}^\star)R]\mathbf{B}^{2n})$, then by theorem 1, $\rho_e(\mathscr{Z}^\star)=\rho_e(c)\oplus J\mathbf{B}^{3n}$, where J is defined by (34). After sorting column vectors of J according to their norm, one obtains $\tilde{J}$. It is then straightforward to derive (30)-(34). ■

VI. Enclosing tightly a union of tubes

As stated in the previous section, the partition conducted within the event detection and localization phases yields many solutions boxes which, after a set-membership reset mapping, are the initial state subsets for continuous transitions in the new hybrid mode. Even if our algorithm performs a fast guard crossing at low numerical cost, the computation of all these trajectory sub-tubes increases undesirably the computation

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAC.2015.2491740, IEEE Transactions on Automatic Control

time. One solution consists in merging these sub-tubes into a single tube enclosing their union. Since our method for continuous transitions, i.e. algorithm $\varphi^{QR}(.)$ in algorithm 2 uses MSPBs, a form consistent with the internal representation used by our continuous reachability algorithm, we obviously need to compute the union of these sub-tubes in the form of an MSPB.

The method we propose for merging the set of sub-tubes works in three steps:

- First, it computes the vertices of the convex domain containing the union of the MSPBs describing the sub-tubes;
- Then, it computes a parametrized MSPB enclosing these vertices;
- Finally, it tunes the MSPB parameters in order to optimize the size of the resulting bounding MSPB.

These steps are described in the sequel.

A. Computing the sub-tubes vertices

Let us assume that the hybrid transition under study has been completed and the initial domain for the continuous transition in the new hybrid mode is characterized by the following union of MSPB:

$$\bigcup_{j=1}^{L} \mathscr{Z}_l^{\star\prime} \tag{35}$$

Each of the L MSPB solution domains, $\mathscr{Z}_l^{\star\prime}$, $l=1$ to L, obtained via (30), is characterized as:

$$\mathscr{Z}_l^{\star\prime} = \{A_l r + v \,|\, v \in [v]_l, r \in [r]_l\} = A_l[r]_l + [v]_l = c_l \oplus \mathtt{Z}(R_l), \tag{36}$$

where $\mathtt{Z}(R_l)$ is the centered zonotope generated by matrix $R_l \in \mathbb{R}^{n\times 2n}$, and where c_l and R_l are defined from $[v]_l$ and $[r]_l$ as in (19)-(20).

Definition 1: Let $R \in \mathbb{R}^{n\times p}$ be a matrix and let $\mathtt{Z}(R) = R\mathbf{B}^p \subset \mathbb{R}^n$ be the centered zonotope generated by R. We denote by $\mathtt{Z}^{\pm}(R)$, the set of point-vectors obtained as the linear image by R of the 2^p vertices of the unit hypercube $\mathbf{B}^p$:

$$\mathtt{Z}^{\pm}(R) = \{R\sigma, \sigma \in \{-1,1\}^p\}. \tag{37}$$

Proposition 6.1 (Containment property): Let $\mathscr{C}(.)$ (resp. $\mathscr{V}(.)$) denote the convex hull (resp. the vertices) of any polytopic set. For all $R \in \mathbb{R}^{n\times p}$, we have:

$$\mathscr{V}(\mathtt{Z}(R)) \subseteq \mathtt{Z}^{\pm}(R), \tag{38}$$

$$\mathscr{C}(\mathtt{Z}^{\pm}(R)) = \mathtt{Z}(R), \tag{39}$$

$$p = 2n \;\Rightarrow\; \exists R_g \in \mathbb{R}^{n\times p},$$
$$2card(\mathscr{V}(\mathtt{Z}(R))) \leq card(\mathscr{V}(\mathtt{Z}^{\pm}(R))) \leq 2card(\mathscr{V}(\mathtt{Z}(R_g))) \tag{40}$$

In words, $\mathtt{Z}^{\pm}(R)$ contains all the vertices of $\mathtt{Z}(R)$, $\mathtt{Z}(R)$ is the convex hull of $\mathtt{Z}^{\pm}(R)$, and $\mathtt{Z}^{\pm}(R)$ only contains twice more points than the true number of vertices of a MSPB (i.e. $\mathtt{Z}(R_g)$ with $p=2n$) in the general case (i.e. non degenerate case) yielding equalities instead of inequalities in (40).

Proof: Properties (38)-(40) directly follow from definition 1 and the convexity of zonotopic domains. (40) results from the maximum number of vertices of a p-zonotope in $\mathbb{R}^n$, $2\sum_{i=0}^{n-1}\binom{p-1}{i}$, which equals $2^{(2n-1)}$ when applied to a MSPB ($p=2n$). By comparison, the number $card(\mathtt{Z}^{\pm}(R))$, (37), of potential vertices of a MSPB in $\mathbb{R}^n$ is $2^p = 2^{2n}$, so only twice more than the true number of vertices of a generic MSPB, the ratio being independent on n. ■

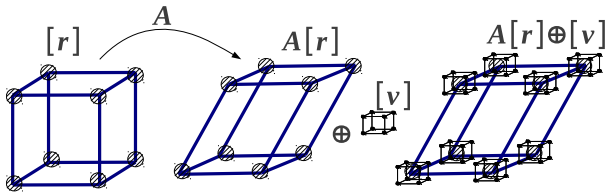


Fig. 5. Building the point-vectors $c_l + \mathtt{Z}^{\pm}(R_l)$

The next theorem formalizes the approach which supports our method for enclosing a set of sub-tubes.

Theorem 4: Let $\mathscr{M}$ be a convex set (e.g. polytope, zonotope, MSPB, etc.). Consider the MSPB $\mathscr{Z}_l^{\star\prime}$ in (36) and remind (19)-(20). Let $\mathscr{P}$ be a set of point-vectors obtained as follows:

$$\mathscr{P} = \bigcup_{j=1}^{L} \left(c_l + \mathtt{Z}^{\pm}(R_l)\right), \tag{41}$$

$$\text{then,} \quad \mathscr{P} \subseteq \mathscr{M} \;\Rightarrow\; \left(\bigcup_{j=1}^{L} \mathscr{Z}_l^{\star\prime}\right) \subseteq \mathscr{M} \tag{42}$$

In words, any convex set $\mathscr{M}$ containing the point-vectors in $\mathscr{P}$ also contains the union of the MSPB domains $\mathscr{Z}_l^{\star\prime}$ for l=1 to L.

Proof: The proof of theorem 4 mainly relies on the polytopic (thus convex) nature of the MSPB domains $\mathscr{Z}_l^{\star\prime} = A_l[r_l] + [v_l] = c_l \oplus \mathtt{Z}(R_l)$ where $c_l \in \mathbb{R}^n$ and $R_l \in \mathbb{R}^{n\times 2n}$. (41) not only provides a constructive way to compute the $m2^{2n}$ point-vectors in $\mathscr{P}$, but also ensures through (38) that $\mathscr{P}$ contains the set $\mathscr{V}$ of all the vertices of all the MSPB domains $\mathscr{Z}_l^{\star\prime}$, $l=1,\ldots,L$. In addition, (39) ensures that any convex set $\mathscr{M}$ satisfying $\mathscr{P} \subseteq \mathscr{M}$ also satisfies the containment property $\mathscr{Z}_l^{\star\prime} \subseteq \mathscr{M}$ for each l=1 to L. ■

Remark 3: Since $\mathtt{Z}^{\pm}([R_1,R_2]) = \mathtt{Z}^{\pm}(R_1) \oplus \mathtt{Z}^{\pm}(R_2)$, the set of point-vectors $\mathscr{P}$ defined in (41) can also be expressed as $\mathscr{P} = \bigcup_{l=1}^{L} \left(A_l[r]_l^{\pm} \oplus [v]_l^{\pm}\right)$ where $[r]_l^{\pm}$ and $[v]_l^{\pm}$ refer to the set of vertices of the n-dimensional axis-aligned boxes $[r]_l$ and $[v]_l$, respectively. For a given value of l, figure 5 illustrates how the set of points $c_l + \mathtt{Z}^{\pm}(R_l)$ can be built from $A_l[r]_l^{\pm} \oplus [v]_l^{\pm}$.

B. Enclosing a point-vector cloud by a zonotope

This subsection describes our generic algorithm `cloud2zonotope` (Algorithm 3) for computing the center $c \in \mathbb{R}^n$ and the shape matrix $R \in \mathbb{R}^{n\times p}$ of a zonotope $c \oplus \mathtt{Z}(R)$ enclosing a cloud X of N point-vectors defined in $\mathbb{R}^n$. For better algorithm readability, vectors are explicitly denoted using arrows $\vec{u}$ in the algorithms and in this subsection.

The algorithm `cloud2zonotope` (algorithm 3), also illustrated in Fig. 6, is mainly based on iterative compressions of the initial cloud X formed by N point-vectors and, jointly, the iterative building of the zonotope enclosure. After centering the cloud (step 5) in the middle of its interval hull (step 4),

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAC.2015.2491740, IEEE Transactions on Automatic Control

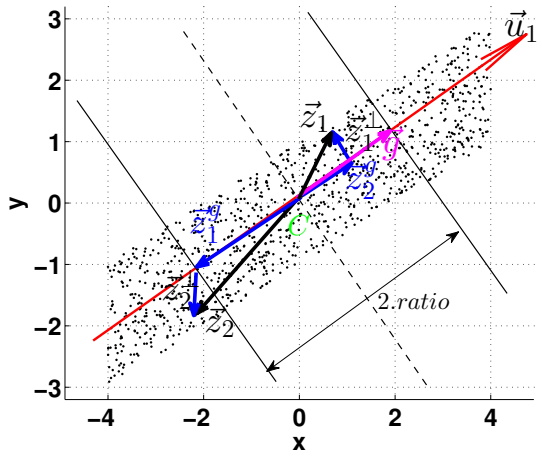
each compression (step 9) is performed according to a vector $\vec{g} \in \mathbb{R}^n$ which further defines one of the generator segments of the resulting zonotope (step 10). At each iteration, $\vec{g}$ is oriented along the principal direction $\vec{u}$ of the current (i.e. partially compressed) cloud (step 7). Moreover, the magnitude of $\vec{g}$ is a fraction of (an approximation of) the largest projection of the points in the (centered) cloud along $\vec{u}$. The fraction used is defined by the scalar compression ratio, denoted *ratio* (*ratio* $\in [0,1]$), and the approximation of the largest projection relies on a very fast computation based on the interval hull of the current cloud which corresponds to the scalar product $\vec{u} \cdot \vec{rad}$ involving only two n-dimensional vectors at step 8.

Each iteration of the while loop in `cloud2zonotope` jointly performs a cloud compression, both in direction and magnitude, as defined by generator $\vec{g}$, and an expansion of the zonotope under construction, which relies on the Minkowski sum with the straight line segment $[-1,+1]\vec{g}$ at step 10. Moreover, the function computing an updated cloud $\underline{X} =$ `compress`$(X,\vec{g})$ from the compression of X in direction $\vec{g}$, described in Algorithms 4, is designed so as to satisfy

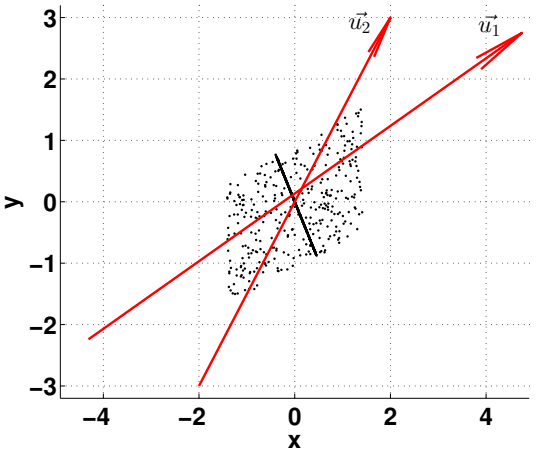
$$X \subset (\underline{X} \oplus \vec{g}[-1,+1]), \tag{43}$$

where matrices X and $\underline{X}$ are each identified to a set of N point-vectors. Note that (43) is a key point to further ensure that the initial cloud is enclosed within the zonotope eventually returned by algorithm `cloud2zonotope`. The joint/iterative "cloud compression and zonotope expansion" process is repeated until one of the two stopping criteria s_1 or s_2 are satisfied (step 11): s_1 is an integer defining the maximum number of allowed iterations, and $s_2 \in [0,1]$ is a positive real number defining a stopping condition for the iterative compressions in the form of a fraction of the initial cloud radius ($\vec{w}$ assigned at step 6). After exiting from the while loop, the interval hull of the residual cloud is computed (step 13) and the corresponding aligned box is summed (Minkowski sum) with the zonotope previously expanded from an initially empty set during the while loop iterations. This results at step 14 into a zonotope $c \oplus \mathrm{Z}(R)$ which is guaranteed to contain all the N point-vectors defined by X when calling `cloud2zonotope`.

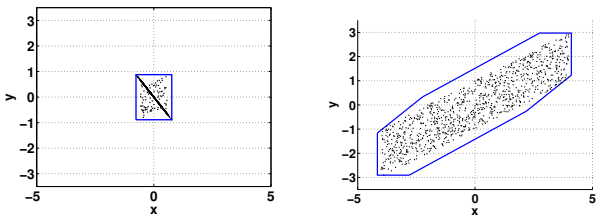
Fig. 6 illustrates the main steps of algorithm 3 `cloud2zonotope` in the case of a 2-dimensional point cloud and an MSPB output. Fig. 6.a shows that our algorithm first centers the point-vector cloud, then compresses the cloud along the first principal direction $\vec{u}_1$. Here one can see point-vectors $\vec{z}_1$ and $\vec{z}_2$ projected onto axis $\vec{u}_1$ while considering the bound defined by *ratio*. Thus, compressing the cloud boils down to keeping only the difference $\vec{z}_i^{\,\delta} = \vec{z}_i - \vec{z}_i^{\,g}$. Fig. 6.b shows the point-vector cloud after the first compression along direction $\vec{u}_1$. Our algorithm finds the next compression direction $\vec{u}_2$ as the first principal direction of the remaining cloud. In order to obtain an MSPB as output set, the number of compression iterations is taken as $n = 2$; Fig. 6.c shows that after $n = 2$ compression iterations, the residual point-vectors are enclosed within an axis-aligned box. Fig. 6.d shows the zonotope, here an MSPB obtained as the Minkowski sum of the axis-aligned box and the parallelotope as defined by the generator segments $\vec{g}_1 = \|\vec{g}_1\|\vec{u}_1$ and $\vec{g}_2 = \|\vec{g}_2\|\vec{u}_2$.



(a) Cloud centering and compression along first principal direction



(b) Finding the second compression direction as the first principal direction of the remaining cloud.



(c) Enclosing the residual point cloud in an axis-aligned box.

(d) Computing the MSPB enclosing the point cloud.

Fig. 6. Main steps for build the enclosing zonotope as performed by Algorithm 3 `cloud2zonotope` in the case of a 2D point cloud and an MSPB output.

C. Building the MSPB from the zonotope enclosure

Using algorithm `cloud2zonotope`, one can tune parameters *ratio* and $s_i, i = 1,2$ introduced in the latter section, in order to obtain a particular zonotope with center $c_{\mathscr{M}}$ and shape matrix $R_{\mathscr{M}} \in \mathbb{R}^{n \times 2n}$ representing the tightest MSPB enclosing the point cloud $\mathscr{P}$ defined in (41). Choosing s_1=n and s_2=0, algorithm `cloud2zonotope` yields an MSPB since there are exactly n compression iterations, hence exactly n generator vectors. Moreover, the remaining cloud is eventually gathered

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAC.2015.2491740, IEEE Transactions on Automatic Control

in an axis-aligned box. The size of the MSPB solely depends on parameter *ratio* that can be tuned to optimize any size-based criterion. The latter tuning acts as a trade-off between the relative weight of the parallelotope and the axis-aligned box in (17). The optimal choice for the parameter *ratio* is then formally given as

$$\underset{ratio\in[0,1]}{ratio^\star} = \arg\min \mu(\mathtt{Z}(R)) \tag{44}$$

where $\mu(.)$ is the size of the zonotope. The optimal choice can be obtained in three ways : volume minimization, segment minimization, or P-*radius* minimization.

1) Volume minimization: The volume of a zonotope [69] $\mathtt{Z}(R) \subset \mathbb{R}^n$, where $R = [r_1, \ldots, r_p] \in \mathbb{R}^{n\times p}$, with $p \geq n$, is given by:

$$Vol(\mathtt{Z}(R)) = 2^n\Big(\sum_{1\leq i_1\leq i_2\leq \ldots i_n\leq p} |det([r_{i_1}, \ldots, r_{i_n}])|\Big) \tag{45}$$

The integers $i_1, \ldots, i_n$ correspond to different ways of choosing n elements amongst p. The sum (45) is thus composed of $\binom{p}{n}$ elements.

2) Segment length minimization: The segment's length of a zonotope $\mathtt{Z} = c \oplus R\mathbf{B}^p \in \mathbb{R}^n$ is given by the sum of squares of the generators of the zonotope $\mathtt{Z}$. Computing the sum of squares of the generators of the zonotope $\mathtt{Z}$ boils down to computing the *Frobenius* norm of R.

$$S_m(\mathtt{Z}(R)) = \sum_{i=1}^{n}\sum_{i=1}^{p} R_{ij}^2 = \|R\|_F^2 = \mathrm{Tr}(R^\top R) \tag{46}$$

where Tr denotes the trace of a square matrix.

3) P-radius minimization: The P-*radius* [11] of a zonotope $\mathtt{Z} = c \oplus R\mathbf{B}^p \in \mathbb{R}^n$ is defined as

$$P_{rad}(\mathtt{Z}(R)) = \max_{x\in\mathtt{Z}} \|x-c\|_P^2 = \max_{x\in\mathtt{Z}} (x-c)^\top P(x-c) \tag{47}$$

where $P = P^\top \geq 0$ is a symmetric and positive definite matrix.

To obtain the solution $ratio^\star$ one can use an iterative algorithm, possibly using derivatives obtained via finite difference. Here, we merely evaluate the size of the MSPB for ten values of *ratio* taken over a grid of values between 0 and 1, and then merely pick up the value which minimises (44).

Summarizing, the union of L MSPB solution domains as defined by (35) is tightly enclosed into a single MSPB of optimized size, and obtained with algorithm `cloud2zonotope` tuned with $ratio$=$ratio^\star$, s_1=n and s_2=0.

VII. Properties and complexity analysis

Our hybrid reachability method alternates an interval Taylor integration method implementing Lohner's QR-factorisation for the continuous expansion of the hybrid system and an original method for guard crossing materialized by transitions.

Proposition 7.1 (Conservative hybrid reachability): The hybrid reachability algorithm provides guaranteed outputs, i.e. the flow-pipe generated by alternating continuous reachability (Algorithm 2: φ^{QR}), flow guard intersection (Algorithm 6: `hybrid-transition`) and trajectory fusion (Algorithm 5: `MSPB`) is guaranteed in the sense that it encloses all the trajectories of the hybrid system *HA* (1) consistent with the

Algorithm 3: Algorithm `cloud2zonotope`

Input : $X := \{\vec{x}_i, i = 1\ldots N\}$, $ratio \in [0,1]$, s_1, s_2
Output: $\vec{c} \in \mathbb{R}^n$, $R \in \mathbb{R}^{n\times p}$

Initialization: $\vec{c} = 0$, $R = \emptyset$, $iter = 0$, $\vec{w} = 0$;
while $(iter < s_1) \vee (\exists i \vec{rad}_i > s_2\vec{w}_i)$ **do**
  update iteration counter. $iter := iter + 1$;
  compute cloud center and radius. $\{\vec{mid}, \vec{rad}\} := \mathtt{minbox}(X)$;
  center point-cloud around *mid*. $X := X - \vec{mid}$;
  store initial radius. **if** $iter = 1$ **then** $\vec{w} := \vec{rad}$;
  find first principal direction (unit vector $\vec{u}$). $(U,S) := svd(X)$, $\vec{u} := U(:,1)$;
  choose generator segment. $\vec{g} := ratio|\vec{u}\cdot\vec{rad}|\cdot\vec{u}$;
  compress cloud along $\vec{g}$ using *ratio*. $X := \mathtt{compress}(X, \vec{g})$;
  update zonotope generators. $\vec{c} := \vec{c} + \vec{mid}$, $R := [R,\ \vec{g}]$
end while
enclose remaining cloud in a box. $\{\vec{mid}, \vec{rad}\} := \mathtt{minbox}(X)$;
update zonotope. $\vec{c} := \vec{c} + \vec{mid}$, $R := [R,\ \mathrm{diag}(\vec{rad})]$;

Algorithm 4: Algorithm `compress`

Function X=`compress`$(X,\ \vec{g})$
  compute compression direction. $\vec{u} = \vec{g}/\|\vec{g}\|$;
  for *each point-vector* $\vec{x}_i$ *in* X **do**
    compute coordinates along $\vec{u}$. $d := \vec{x}_i \cdot \vec{u}$;
    compression along $\vec{u}$. $d := \min(\|\vec{g}\|, \max(-\|\vec{g}\|, d))$;
    update coordinates. $\vec{x}_i = \vec{x}_i - d\vec{u}$;
  end for
return

Algorithm 5: Algorithm `MSPB`

Input : $\mathscr{L}$
Output: $\mathscr{L}$
initialization: $\mathscr{P} = \emptyset$ i.e. empty set of points ;
for $j \leftarrow 1$ **to** $Length(\mathscr{L})$ **do**
  pick up $\mathscr{Z}_l^{\star\prime}$ from $\mathscr{L}$ (i.e. q, A_l, $[r]_l$, $[v]_l$);
  $\mathscr{P} := \mathscr{P} \cup \left(A_l[r]_l^{\pm} \oplus [v]_l^{\pm}\right)$
end for
$X_{\mathscr{P}}$:= matrix representation (n col.) of $\mathscr{P} \subset \mathbb{R}^n$;
for $ratio =: 0, 0.1, \ldots, 0.9, 1$ **do**
  $\{c_{\mathscr{M}}, R_{\mathscr{M}}\} := \mathtt{cloud2zonotope}(X_{\mathscr{P}}, ratio, n, 0)$;
  compute the size of zonotope $\mu(\mathtt{Z}(R_{\mathscr{M}}))$ as in (45);
end for
$r^* := \arg\min \mu(\mathtt{Z}(R_{\mathscr{M}}))$;
extract MSPB attributes of $\mathscr{M}^* := c_{\mathscr{M}^*} \oplus \mathtt{Z}(R_{\mathscr{M}^*})$
obtained with r^*:
$A := R_{\mathscr{M}^*}(:, 1:n)$;
$[r] := [-1, +1]^n$;
$[v] := c_{\mathscr{M}^*} + diag(R_{\mathscr{M}^*}(:, (n+1):(2n)))[-1, +1]^n$;
store MSPB: $\mathscr{L} := (q, A, [r], [v])$ in list $\mathscr{L}$;

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAC.2015.2491740, IEEE Transactions on Automatic Control

Algorithm 6: Algorithm `Hybrid-Transition`

input : $\mathscr{L}_j^F, t_{j+1}, \{\varphi_q^{QR}(), \widetilde{\varphi}_q()\}_{q\in\mathscr{Q}}, \{\gamma_e(), \rho_e()\}_{e\in\mathscr{E}}, \varepsilon_T$
output: $\mathscr{L}_{j+1}^F, \mathscr{L}_{j+1}^R$

Initialization : initialize running frontier list $\mathscr{L} := \mathscr{L}_j^F$;
while $\mathscr{L} \neq \emptyset$ **do**
  pick up $\mathscr{L}$ list element $(q, t_0, [\chi_0])$;
  /* Continuous transition */
  compute continuous expansion over $[t_0, t_{j+1}] \rightarrow [\widetilde{\mathbf{z}}]_j$;
  update reached set list $\mathscr{L}_{j+1}^R \leftarrow (q, t_0, t_{j+1}, [\widetilde{\mathbf{z}}_j])$;
  compute new solution at time $t_{j+1} \rightarrow [\chi_{j+1}]$;
  solve CSP to compute $[\chi'_{j+1}] := [\chi_{j+1}] \cap inv(q)$;
  if $[\chi_{j+1}]' \neq \emptyset$ **then**
    update frontier list $\mathscr{L}_{j+1}^F \leftarrow (q, t_{j+1}, [\chi_{j+1}]')$;
  end if
  if $Length(\mathscr{L}_{j+1}^F > 2)$ **then**
    $\mathscr{L}_{j+1}^F := \mathtt{MSPB}(\mathscr{L}_{j+1}^F)$
  end if
  /* Discrete transition */
  forall the *elements* $e \leftarrow \mathscr{E}$ **do**
    initialize running jump list $\mathscr{L}_e := \{(q, t_0, [\chi_0], t_{j+1})\}$;
    $t_1 := t_{j+1}$;
    while $\mathscr{L}_e \neq \emptyset$ **do**
      compute flow over $[t_0, t_1] \rightarrow [\widetilde{\mathbf{z}}]$;
      if $e = (q, q')$ *exists* **then**
        if $(\gamma_e([\widetilde{\mathbf{z}}]) \ni 0)$ **then**
          $\underline{t}^\star := t_0$; $\overline{t}^\star := t_1$;
          if $((\overline{t}^\star - \underline{t}^\star) \leq \varepsilon_T)$ **then**
            compute flow enclosure over $[\underline{t}^\star, \overline{t}^\star] \rightarrow [\chi]$;
            $[\chi]^\star := \mathtt{HC4\_Revise}([\chi])$;
            if $[\chi]^\star \neq \emptyset$ **then**
              jump and update $\mathscr{L} \leftarrow (q', \underline{t}^\star, \rho_e([\chi]^\star))$;
            end if
          else
            compute solution set at $\underline{t}^\star \rightarrow [\chi]_1$;
            compute solution set at $t_1 := (\underline{t}^\star + \overline{t}^\star)/2 \rightarrow [\chi]_2$;
            update running jump list $\mathscr{L}_e \leftarrow (q, \underline{t}^\star, [\chi]_1, t_1)$;
            update running jump list $\mathscr{L}_e \leftarrow (q, t_1, [\chi]_2, \overline{t}^\star)$;
          end if
        end if
      end if
    end while
  end forall
end while

uncertainty domains of the initial state and the parameter vector given by $\mathbb{X}_0 \subseteq \mathbb{R}^n$ and $\mathbb{P} \subseteq \mathbb{R}^{n_p}$, respectively.

Proof: The proof of proposition 7.1 simply derives from the conservatism of the three algorithms involved proved by Proposition 4.1, Theorem 3, and Theorem 4, and the conservatism of contractors that rely on local consistency properties [65]. ∎

Let us now provide some hints about complexity. The Taylor series method underlying continuous reachability is of polynomial complexity. Denoting k the order of the Taylor series expansion, usually chosen between 10 and 20, the work per step is $O(k^2)$ to compute Taylor coefficients. The computational complexity of the method hence mainly derives from the one of guard crossing, which includes two main steps: flow guard intersection, solved in the form of a CSP, and trajectory fusion, which is solved by the algorithm `MSPB`, a specific kind of zonotope enclosure. It is admitted that solving CSPs, either on discrete or on continuous domains, is in theory NP-hard. However, there have been efforts to develop solving techniques whose practical time complexity is better than the exponential worst case [70]. These improvements rely on technical features that are beyond the scope of this paper.

In the following, we discuss in more details the algorithm for MSPB enclosure, which has been tailored for our use. The MSPB algorithm merges in a single reachability set $\mathscr{M}$, expressed in the form of an MSPB, the possibly many flowpipes whose computation is initiated during the crossing of a guard. The algorithm `cloud2zonotope` makes it possible to build this MSPB (actually an n-dimensional zonotope) from a cloud of points (here, potential vertices) which are known to belong to the union domain to be characterized. Computing the MSPB enclosure involves the generation of a cloud of points and n partial singular value decompositions (SVD). The generation of the cloud of points is $O(2^{2n})$, and each partial SVD computation is $O(Nn^2)$. The cost of enumerating the $N = m2^{2n}$ potential vertices related to a number m of MSPBs can be evaluated as $fl_{PV-Enum}(n,m) \approx mn2^{2n}$ floating-point operations. It can then be shown that the cost of the `MSPB` algorithm can be evaluated as $fl_{MSPB}(n,m) \approx m2^{2n}(40n^3 + 40n^2 + n) + 80n^4$ floating-point operations. The practical cost of the trajectory fusion algorithm therefore comes from the vertex enumeration exponential complexity.

One could have the idea to consider the true (rather than the potential) vertices related to the m MSPBs. However, based on the complexity of vertex enumeration algorithms (e.g. like QuickHull [71]), it can be shown that there is a clear interest in using potential vertices rather than true vertices enumeration: the former method only induces twice more points for a single generic MSPB and this greater number of points is easily balanced by the simplicity of the related enumeration algorithm, though the complexity remains of course exponential in both cases. Though featuring an exponential complexity, the proposed scheme relying on the enumeration of MSPB potential vertices can still compete with other algorithms (even polynomial ones) for problems with moderate continuous state space dimension.

VIII. Numerical evaluation

For the experimentation purpose of this paper, our method for hybrid reachability has been implemented as follows. We use the `Profil/Bias` C++ class for interval computation, the `FADBAD++` package (www.fadbad.com) for automatic differentiation, `AML++` (amlpp.sourceforge.net) and `Armadillo` (arma.sourceforge.net) package for Linear algebra. We use our own implementation of Lohner's method (Algorithm 2) for guaranteed set integration of IVP ODE. Finally, we use the CSP solving techniques as implemented in the `IBEX` C++ library (www.ibex-lib.org). Experiments were conducted on an intel $i5-3470-3.6$GHz-16GB running Linux.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAC.2015.2491740, IEEE Transactions on Automatic Control

TABLE I
(P-*radius*,S_m, VOLUME) VS *ratio* FOR THE MASS-SPRING, FIRST SWITCH

ratio	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
P-*radius*	1.910	1.910	1.910	1.910	1.915	1.929	1.926	1.929	1.936	1.946	1.962
$S_m \times (10^{-3})$	4.646	3.174	2.002	1.823	1.792	2.179	2.329	2.729	3.331	4.149	5.115
Volume $\times(10^{-3})$	8.574	3.417	1.104	1.961	3.122	4.974	5.511	2.445	3.239	4.745	6.577

TABLE II
(P-*radius*,S_m, VOLUME) VS *ratio* FOR THE MASS-SPRING, SECOND SWITCH

ratio	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
P-*radius*	0.710	0.710	0.710	0.710	0.710	0.710	0.710	0.710	0.713	0.725	0.737
$S_m \times (10^{-3})$	8.641	6.116	4.094	3.781	3.950	3.653	4.048	4.769	5.899	7.403	9.122
Volume $\times(10^{-3})$	16.186	7.444	1.295	3.290	6.273	8.838	9.979	5.088	6.383	9.413	13.053

A. Case study 1 : Switched mass-spring system

Consider the switched dynamical system with two modes $q = 1, 2$ and one jump transition $e = 1 \to 2$ obtained by introducing an artificial switching in a mass-spring system. The switching is artificial because continuous dynamics are the same in the two modes. The main idea is to investigate the impact of our guard crossing and trajectory merge algorithms by comparing the results with the original continuous system, i.e. without switching.

The switched system is given by

$$\begin{cases} \text{flow}(1) : f_1(x_1,x_2) = \left(x_2,\ \frac{-k}{m}x_1 - \frac{c}{m}x_2\right) \\ \text{inv}(1) : \nu_1(x_1,x_2) = x_1 - x_2 < 0 \\ \text{flow}(2) : f_2(x_1,x_2) = f_1(x_1,x_2) \\ \text{inv}(2) : \nu_2(x_1,x_2) = -\nu_2(x_1,x_2) < 0 \\ \text{guard}(1) : \gamma_1(x_1,x_2) = x_2 - x_1 = 0 \\ \text{reset}(1) : \rho_1(x_1,x_2) = (\alpha_1 x_1, \alpha_2 x_2) \end{cases} \tag{48}$$

where $\alpha_1 = \alpha_2 = 1$, $k = 4$, $m = 2$, $c = 1.25$. The continuous states are described by two variables (x_1,x_2), where x_1, x_2 respectively represent the position and the velocity of the mass-spring. The initial conditions are given by $x_1 \in [1, 1.1], x_2 \in [-0.63, -0.61]$. Algorithms `Hybrid-Transition` (Algorithm 6) and φ^{QR} (Algorithm 2) were tuned as follows : The time step is chosen constant $h = 0.1$; and the time interval is bisected until a threshold $\varepsilon_T = 0.005$.

Fig. 7 gathers the reachable sets as obtained for the time horizon $[0, 5]$. First the reachable set for the original continuous system is shown, then the one for the switched version without trajectory merge. Then the reachable set as obtained with our trajectory merge algorithm using one of the several criteria proposed in section VI-C: *P*-radius, with *P* taken as identity matrix, volume and segments length. Tables I-II shows the criteria obtained for the first and second switches respectively. The optimal tuning parameter clearly depends on the criterion, though it does not seem to change significantly with the switching. The bottom right graph in Fig. 7 clearly shows that the *P*-radius criterion yields the tightest MSPB, but still with significant over-approximation compared to the enclosure computed for the continuous system (i.e. without switching). Nevertheless, the MSPB obtained with our merge algorithm are always significantly tighter than the enclosure obtained using naive convex interval hull of trajectory tubes.

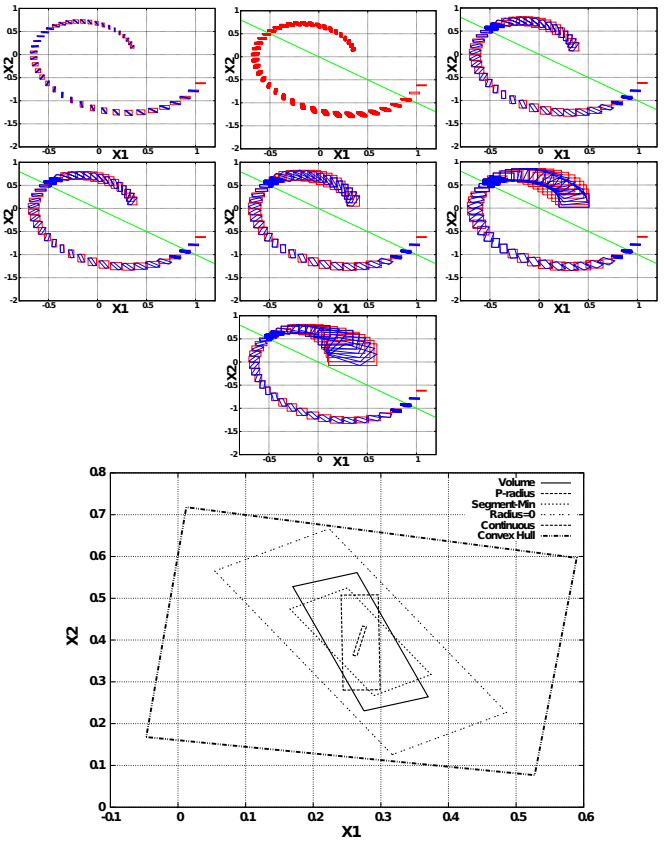


Fig. 7. The frontier, i.e. the solution sets at time-grid points for the Mass-Spring. From left to right, and top to bottom, the continuous version (CPU 0.007*s*), the switched version without trajectory merging (CPU 0.88*s*), the switched version with trajectory merging using P-*radius* criterion (CPU 0.125*s*), using segment length criterion (CPU 0.129*s*), using volume criterion (CPU 0.163*s*), when merging with ratio=0 (CPU 0.202*s*), and using naive convex hull (CPU 0.133*s*). The final graph compares the MSPB solution enclosure obtained at final time for the different experiments.

B. Case study 2 : A ball bouncing on a sinusoidal surface

We consider the uncertain model of a ball that bounces on a sinusoidal surface (modified from [54]). It is described by four variables (p_x, p_y, v_x, v_y), where (p_x, p_y) is ball position in 2D and (v_x, v_y) ball velocity. Figure VIII-B provides the hybrid automaton of the system. Model parameters are set to $g \in [9.8, 9.85]$, $e = 3.5$ and $k \in [0.3, 0.4]$ (all units are S.I). We took a constant integration time step $h = 0.1$ for φ^{QR}

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAC.2015.2491740, IEEE Transactions on Automatic Control

TABLE III
(P-*radius*, S_m, VOLUME) VS *ratio* FOR THE BOUNCING BALL

ratio	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
P-*radius*	4.26	5.19	6.15	5.87	7.37	8.01	8.59	9.83	9.76	10.16	11.07
S_m	4.26	2.10	1.56	1.33	1.58	2.03	2.56	3.25	3.98	4.86	5.95
volume $\times$ (10^{-3})	62.96	24.48	13.05	4.35	14.04	5.52	0.29	26.88	37.78	43.63	50.99

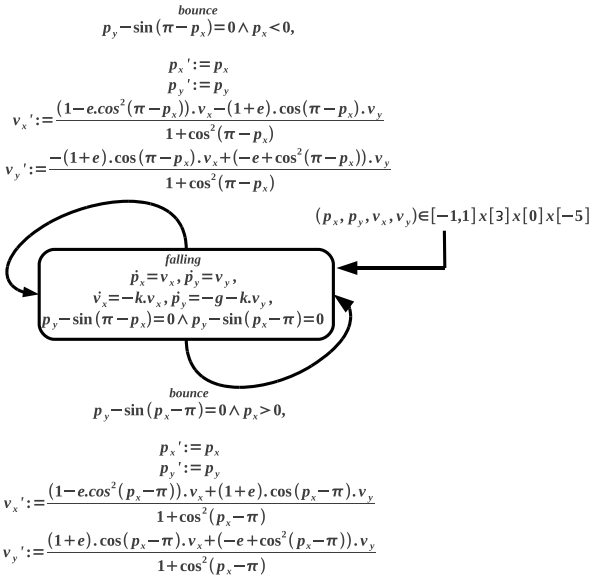


Fig. 8. Case study 2 : A ball bouncing on a sinusoidal surface. Hybrid automaton modeling.

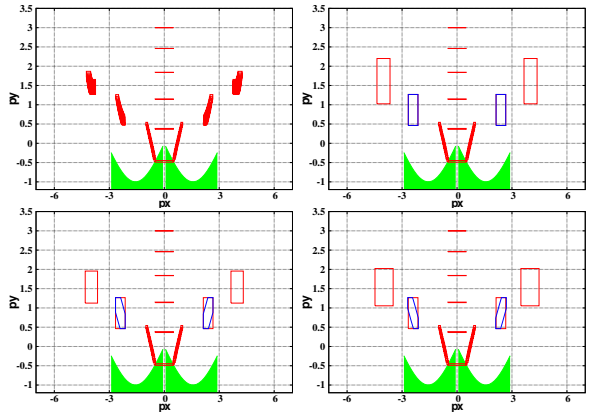


Fig. 9. The frontier, i.e. the solution sets at time-grid points in $px \times py$-space for the bouncing ball. From left to right, the solutions set obtained without trajectory merging (CPU 25.8s), with trajectory merging using the P-radius criterion (CPU 4.1s), using the segment length criterion (CPU 4.1s) and the volume criterion (CPU 4.1s). Here, the left and right flowpipes were labelled hence directly separated.

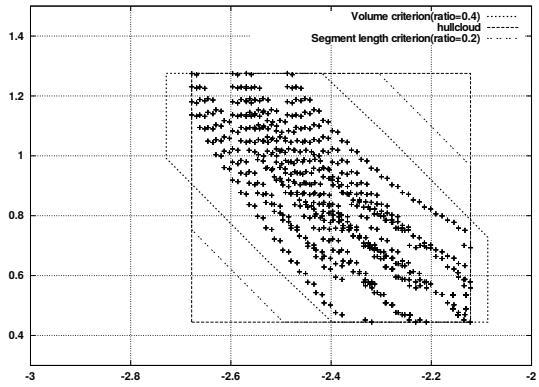


Fig. 10. Capture of the left sub-tubes as captured after the first bounce (Right sub-tubes are the exact symmetrical). Dots show the vertices of the numerous MSPBs obtained after the jump, the unique enclosing MSPB as obtained with each of the three criteria, along with the optimal value for *ratio*.

(Algorithm 2) and set threshold for time interval bisection to $\varepsilon_T = 0.005$ in `Hybrid-Transition` (Algorithm 6). Final time is 0.65s. We run our method with and without trajectory-tubes merging. All results are plotted in Figures 9 and 10. They show that our algorithm can manage two tubes of trajectories simultaneously with non-linear guard conditions. Moreover, we can run the model with large integration time step, which shows the benefits of the interval Taylor method used. Table III gathers the criteria values for the tuning parameter *ratio*. Here again, the optimal value depends on the chosen criteria. Contrary to the first case study, the tightest enclosure is obtained for the segment length criterion.

C. Case study 3 : Sliding mode control

Here, the idea is to investigate the performance of our hybrid reachability approach in situations where the guard includes uncertainty and remains active after reaching it. To this purpose, we consider a very simplified sliding mode control example for the system with state equations $\{\dot{x}_1 = x_2, \dot{x}_2 = u\}$ with desired system output taken as $x_{1des} = y_d \equiv \sin(t)$. The system is modeled as a hybrid system with three modes $q = 1, 2, 3$, each related to the sign of s as depicted in Fig. 11, which also shows the corresponding discrete transitions. Note that the invariant in mode $q = 2$ is defined as $s = 0$ and guard conditions as $s > 0$ or $s < 0$. Because our algorithms provide guaranteed enclosures, the flow-pipe computed for $q = 2$ within the invariant $s = 0$, i.e. a zero-width manifold, always has a non-zero width. This eventually triggers guard conditions, hence discrete transitions towards modes $q = 1$ or $q = 3$. Before proceeding further, let us assume that there are some bounded time-invariant errors $d_1 \in [d_1]$ and $d_2 \in [d_2]$ such that the sliding surface and the control law are given by: $s \in \alpha \cdot (x_1 + [d_1] - y_d) + (x_2 + [d_2] - \dot{y}_d)$ and $u \in \dot{y}_d - \alpha \cdot (\dot{x}_1 + [d_1] - \dot{y}_d) - \eta \text{sign}(s)$. Hence, both the sliding surface, i.e. the guard condition, and the closed-loop state equations become uncertain. Now, remember that our method naturally handles such uncertain hybrid system because the hybrid automaton (1) already considers uncertain parameters in either continuous flow (2), invariant (3) or guard functions (4). All uncertain variables are eventually embedded in the extended state vector and hybrid reachability computations

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAC.2015.2491740, IEEE Transactions on Automatic Control

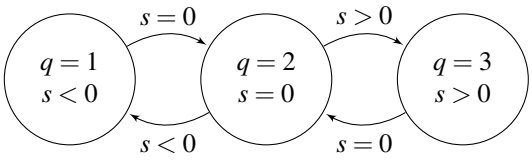


Fig. 11. The hybrid automaton for the sliding mode control under study

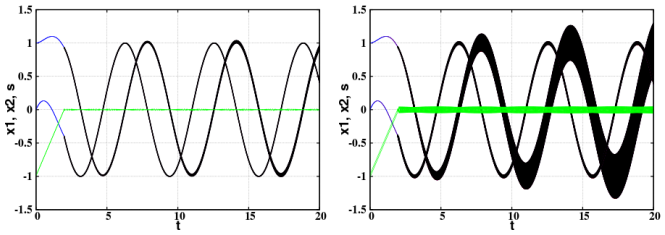


Fig. 12. The reachable set for the sliding mode control trajectory tracking. On the same graph, time-history of x_1-variable (starts at 1), x_2-variable (starts at 0) and s (reaches 0 and stays there). Left: with no uncertainty (34s CPU). Right: with uncertainty (97s CPU).

use equations (5-7). Here, we took a constant integration time step $h = 0.005$s for φ^{QR} (Algorithm 2) and used no time interval bisection in `Hybrid-Transition` (Algorithm 6). $\alpha = 0.01$ and $\eta = 0.5$. Final time is 20s. We run our algorithm with trajectory-tube fusion using the segment-length criterion, and obtained the reachable set plotted in Fig. 12. Notice system trajectories obtained without uncertainty, i.e. $[d_1] = [d_2] = \{0\}$, and the ones with uncertainty taken as $[d_1] = [d_2] = [-0.02, 0.02]$. Starting with $(1,0)$ as initial conditions, the system reaches the uncertain sliding surface at time $t = 2s$ and remains there until simulation ends. It is clear that our algorithm can manage situations where the guard remains active after reaching it, and that it also characterizes efficiently the impact of any uncertainty acting either on the flow or on the guard condition. The thickness that appears in system state trajectories when they reach the sliding surface is induced by the bounded uncertainties $[d_1]$ and $[d_2]$ influencing the sliding surface.

D. Case study 4 : Nonlinear hybrid system of high dimension

The purpose is to investigate the scalability of our trajectory merge approach within our hybrid reachability method. We consider the switched dynamical system with two modes $q = 1, 2$ and one jump transition $e = 1 \to 2$ obtained by introducing an artificial switching in the dynamical model of an oscillatory network of transcriptional regulators with N genes [72]. Let us denote $\vec{x} = (m_1, p_1, \ldots, m_i, p_i, \ldots, m_N, p_N)$ the continuous state vector. The guard condition is given by the nonlinear condition $\gamma_1(\vec{x}) = 0$, with

$$\gamma_1(\vec{x}) = \sum_{i=1}^{i=2N} (x_i - 5)^2 - r^2$$

and $r = 75$, and the continuous dynamics are described by the ODE $\dot{\vec{x}} = f(\vec{x})$ in the form

$$i = 1, \ldots, N \begin{cases} \dot{m}_i &= -m_i + b p_i + \frac{\alpha(t)}{1+p_{i-1}^n} + \alpha_i^0(t) \\ \dot{p}_i &= \kappa(t) m_i - \mu(t) p_i \end{cases} \quad (49)$$

TABLE IV
SCALABILITY OF THE HYBRID REACHABILITY METHOD. COMPUTATION TIMES IN SECONDS (AVERAGE OF 10 RUNS - CASE STUDY 4).

$2N$	Without artificial switching	Hybrid without fusion	Hybrid with fusion
2	0.42	98.4	3.94
4	0.935	337	22.3
6	1.44	740	64.4
10	2.87	796	105
12	3.77	1197	130
14	4.78	1814	347
16	5.91	2076	NA
18	7.16	3460	NA
24	11.75	5113	NA

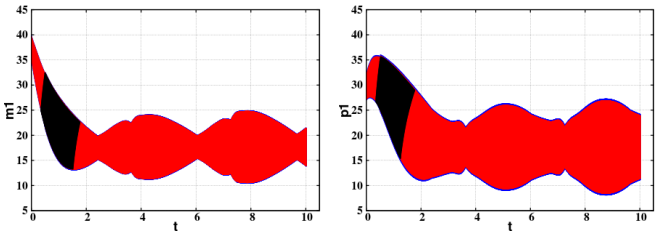


Fig. 13. The reachable set for system (49) with 2N=12. Left, time-history of m_1-variable, Right, time-history of p_1-variable.

with $b = -0.5$, $n = 2$ and $p_0 = p_N$. We assume that the functions $\alpha(t)$, $\kappa(t)$, $\mu(t)$, and $\alpha_i^0(t)$, are unknown but bounded, and the bounds are as follows: for all t, $\alpha(t) \in [0.5, 1.5]$, $\kappa(t) \in [1.9, 2.1]$, $\mu(t) \in [1.9, 2.1]$ and for all i $\alpha_i^0(t) \in [25, 26]$. Initial conditions are taken as: $m_{3k+1}(t_0) \in [35, 40]$, $p_{3k+1}(t_0) \in [27, 32]$, $m_{3k+2}(t_0) \in [30, 35]$, $p_{3k+2}(t_0) \in [25, 30]$, $m_{3k+3}(t_0) \in [40, 45]$, $p_{3k+3}(t_0) \in [32, 37]$, for k=0 to 7. Algorithm φ^{QR} (Algorithm 2) was run with a constant integration time step $h = 0.01$min and algorithm `Hybrid-Transition` (Algorithm 6) was run with no time interval bisection. Final time is 10min. The trajectory merge is done with the segment length criterion. Fig. 13 depicts the reachable sets as obtained with $2N = 12$. Table IV gathers the computation times for settings with $2N$ ranging from 2 to 24. Computations with our fusion algorithm were successful for values of $2N$ up to 14. For larger values of N, the memory size required by the enumeration of potential vertices exceeds the available one. Nevertheless, comparing the simulation runs with and without merging clearly demonstrates the usefulness of our approach since computation times in these settings have been reduced by factor 5 for systems with continuous dimension as large as 14. Comparing the continuous simulation and the hybrid with trajectory merge gives an insight about the overall cost of our method for crossing nonlinear guards. For higher dimensions, further investigations are needed to counteract the impact of the exponential complexity of vertices enumeration in the fusion algorithm.

IX. CONCLUSION AND FUTURE WORKS

We have addressed hybrid reachability analysis of uncertain nonlinear hybrid systems using interval analysis, guaranteed set integration, interval constraint propagation and geometrical tools based on zonotopes. Continuous transitions are addressed

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAC.2015.2491740, IEEE Transactions on Automatic Control

using state-of-the-art methods for guaranteed and validated set integration. Discrete transitions use first a method for reliable event detection and localization that is based on a one-dimensional bisection strategy allied with zonotope-based geometrical tools for domain computation. It then relies on zonotope/MSPB bounding to enclose the flow-pipes obtained after a jump, in a form consistent with our continuous reachability approach. The bounding approach relies on a parameter that is tuned on-the-fly. The method, evaluated on nonlinear hybrid systems, has shown promising performance and reasonable computation times, even under continuous state dimensions greater than ten. Further work will focus on ways to improve the approach's scalability, and detect and merge non-connected clouds of flow-pipes, and also on the possibility to use a paving using MSPB to cover large solution sets.

REFERENCES

[1] R. Gonzalez, M. Fiacchini, T. Alamo, J. Guzman, and F. Rodriguez, "Online robust tube-based MPC for time-varying systems: a practical approach," *International Journal of Control*, vol. 84, no. 6, pp. 1157–1170, 2011.

[2] S. Rakovic, B. Kouvaritakis, M. Cannon, C. Panos, and R. Findeisen, "Parameterized tube model predictive control," *Automatic Control, IEEE Transactions on*, vol. 57, no. 11, pp. 2746–2761, 2012.

[3] J. K. Scott and P. I. Barton, "Bounds on the reachable sets of nonlinear control systems," *Automatica*, vol. 49, no. 1, pp. 93 – 100, 2013.

[4] J. Lygeros, "On reachability and minimum cost optimal control," *Automatica*, vol. 40, no. 6, pp. 917–927, 2004.

[5] J. Lygeros, D. N. Godbole, and S. Sastry, "Verified Hybrid Controllers for Automated Vehicles," *IEEE Transactions on Automatic Control*, vol. 43, no. 4, pp. 522–539, 1998.

[6] C. J. Tomlin, J. Lygeros, and S. Sastry, "A Game Theoretic Approach to Controller Design for Hybrid Systems," *Proceedings of IEEE*, vol. 88, pp. 949–969, Jul. 2000.

[7] L. de Alfaro, T. A. Henzinger, and O. Kupferman, "Concurrent reachability games," *Theor. Comput. Sci.*, vol. 386, no. 3, pp. 188–217, Oct. 2007.

[8] Y. Gao, J. Lygeros, and M. Quincampoix, "The reachability problem for uncertain hybrid systems revisited: A viability theory perspective," in *HSCC*, 2006, pp. 242–256.

[9] N. Meslem, N. Ramdani, and Y. Candau, "Using hybrid automata for set-membership state estimation with uncertain nonlinear continuous-time systems," *Journal of Process Control*, vol. 20, no. 4, pp. 481 – 489, 2010.

[10] E. Benazera and L. Travé-Massuyès, "Set-theoretic estimation of hybrid system configurations," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 39, no. 5, pp. 1277–1291, 2009.

[11] V. T. H. Le, C. Stoica, T. Alamo, E. F. Camacho, and D. Dumur, "Zonotopic guaranteed state estimation for uncertain systems," *Automatica*, vol. 49, no. 11, pp. 3418 – 3424, 2013.

[12] F. Chernousko, "Ellipsoidal state estimation for dynamical systems," *Nonlinear Analysis: Theory, Methods Applications*, vol. 63, no. 57, pp. 872 – 879, 2005.

[13] H. Guéguen, M.-A. Lefebvre, J. Zaytoon, and O.Nasri, "Safety verification and reachability analysis for hybrid systems," *Annual Reviews in Control*, vol. 33, no. 1, pp. 25 – 36, 2009.

[14] E. M. Clarke, A. Fehnker, Z. Han, B. H. Krogh, O. Stursberg, and M. Theobald, "Verification of hybrid systems based on counterexample-guided abstraction refinement," in *TACAS*, 2003, pp. 192–207.

[15] C. J. Tomlin, I. M. Mitchell, A. M. Bayen, and M. Oishi, "Computational techniques for the verification of hybrid systems," *Proceedings of the IEEE*, vol. 91, no. 7, pp. 986–1001, 2003.

[16] A. Aswani, J. Ding, H. Huang, M. Vitus, J. Gillula, P. Bouffard, and C. Tomlin, "Verification and control of hybrid systems using reachability analysis with machine learning," in *HSCC '12*, 2012, pp. 1–2.

[17] A. Donzé, B. Krogh, and A. Rajhans, "Parameter synthesis for hybrid systems with an application to simulink models," in *HSCC*, ser. LNCS, 2009, pp. 165–179.

[18] J. Lygeros, C. Tomlin, and S. Sastry, "On controller synthesis for nonlinear hybrid systems," in *IEEE CDC*, vol. 2, 1998, pp. 2101–2106.

[19] R. Alur, T. Dang, J. Esposito, Y. Hur, F. Ivancic, V. Kumar, P. Mishra, G. Pappas, and O. Sokolsky, "Hierarchical modeling and analysis of embedded systems," *Proceedings of the IEEE*, vol. 91, no. 1, pp. 11–28, 2003.

[20] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine, "The algorithmic analysis of hybrid systems," *Theoretical Computer Science*, vol. 138, pp. 3–34, 1995.

[21] E. Asarin, O. Maler, and A. Pnueli, "Reachability analysis of dynamical systems having piecewise-constant derivatives," *Theoretical Computer Science*, vol. 138, pp. 35–66, 1995.

[22] A. Chutinan and B. Krogh, "Computational techniques for hybrid system verification," *Automatic Control, IEEE Transactions on*, vol. 48, no. 1, pp. 64–75, 2003.

[23] A. Girard, "Reachability of uncertain linear systems using zonotopes," in *HSCC, vol 3414 in LNCS*, 2005, pp. 291–305.

[24] A. Girard and C. L. Guernic, "Zonotope/hyperplane intersection for hybrid systems reachability analysis," in *HSCC*, 2008, pp. 215–228.

[25] M. Althoff and B. H. Krogh, "Avoiding geometric intersection operations in reachability analysis of hybrid systems," in *HSCC*, 2012, pp. 45–54.

[26] A. Kurzhanskiy and P. Varaiya, "Ellipsoidal techniques for reachability analysis of discrete-time linear systems," *Automatic Control, IEEE Transactions on*, vol. 52, no. 1, pp. 26–38, 2007.

[27] L. Doyen, T. Henzinger, and J. Raskin, "Automatic rectangular refinement of affine hybrid systems," in *FORMATS'05, vol. 3829 in LNCS*, 2005, pp. 144–161.

[28] M.-A. Lefebvre and H. Guguen, "Hybrid abstractions of affine systems," *Nonlinear Analysis: Theory, Methods Applications*, vol. 65, no. 6, pp. 1150 – 1167, 2006.

[29] M. Kloetzer and C. Belta, "Reachability analysis of multi-affine systems," in *HSCC, vol. 3927 in LNCS*, 2006, pp. 348–362.

[30] R. Alur, T. Dang, and F. Ivančić, "Predicate abstraction for reachability analysis of hybrid systems," *ACM Trans. Embed. Comput. Syst.*, vol. 5, no. 1, pp. 152–199, Feb. 2006.

[31] I. M. Mitchell, A. M. Bayen, and C. J. Tomlin, "A time-dependant Hamilton-Jacobi formulation of reachable sets for continuous dynamics games," *IEEE Trans. Automatic Control*, vol. 50, no. 7, pp. 947–957, 2005.

[32] V. Shia, R. Vasudevan, R. Bajcsy, and R. Tedrake, "Convex computation of the reachable set for controlled polynomial hybrid systems," in *53rd IEEE Conference on Decision and Control*, 2014, pp. 1499–1506.

[33] M. Althoff, O. Stursberg, and M. Buss, "Reachability analysis of nonlinear systems with uncertain parameters using conservative linearization," in *CDC*, 2008, pp. 4042–4048.

[34] G. Batt, C. Belta, and R. Weiss, "Model checking genetic regulatory networks with parameter uncertainty," in *HSCC, vol. 4416 in LNCS*. Springer-Verlag, 2007, pp. 61–75.

[35] E. Asarin, T. Dang, and A. Girard, "Hybridization methods for the analysis of non-linear systems," *Acta Informatica*, vol. 43, pp. 451–476, 2007.

[36] S. Burden, H. Gonzalez, R. Vasudevan, R. Bajcsy, and S. Sastry, "Numerical integration of hybrid dynamical systems via domain relaxation," in *IEEE CDC*, 2011, pp. 3958–3965.

[37] T. Henzinger, B. Horowitz, R. Majumdar, and H. Wong-Toi, "Beyond HyTech: Hybrid systems analysis using interval numerical methods," in *HSCC, vol. 1790 in LNCS*, 2000, pp. 130–144.

[38] N. Ramdani, N. Meslem, and Y. Candau, "A hybrid bounding method for computing an over-approximation for the reachable space of uncertain nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 54, no. 10, pp. 2352–2364, 2009.

[39] X. Chen, E. Ábrahám, and S. Sankaranarayanan, "Taylor model flowpipe construction for non-linear hybrid systems," in *RTSS*, 2012, pp. 183–192.

[40] N. Nedialkov and M. von Mohrenschildt, "Rigorous simulation of hybrid dynamic systems with symbolic and interval methods," in *Proceedings of the American Control Conference*, vol. 1, 2002, pp. 140–147.

[41] A. Rauh, M. Kletting, H. Aschemann, and E. Hofer, "Interval methods for simulation of dynamical systems with state-dependent switching characteristics," in *2006 IEEE CACSD/CCA/SIC*, 2006, pp. 355–360.

[42] S. Ratschan and Z. She, "Safety verification of hybrid systems by constraint propagation based abstraction refinement," *ACM Transactions in Embedded Computing Systems*, vol. 6, no. 1, 2007.

[43] S. Gulwani and A. Tiwari, "Constraint-based approach for analysis of hybrid systems," in *CAV*, 2008, pp. 190–203.

[44] A. Eggers, M. Fränzle, and C. Herde, "SAT modulo ODE: A direct SAT approach to hybrid systems," in *ATVA*, ser. LNCS, vol. 5311. Springer, 2008, pp. 171–185.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TAC.2015.2491740, IEEE Transactions on Automatic Control

[45] A. Eggers, N. Ramdani, N. S. Nedialkov, and M. Fränzle, "Improving the SAT Modulo ODE approach to hybrid systems analysis by combining different enclosure methods," *Software & Systems Modeling*, vol. 14, pp. 121–148., 2015.

[46] N. Ramdani and N. S. Nedialkov, "Computing reachable sets for uncertain nonlinear hybrid systems using interval constraint-propagation techniques," *Nonlinear Analysis: Hybrid Systems*, vol. 5, no. 2, pp. 149 – 162, 2011.

[47] L. G. Birta, T. I. Oren, and D. L. Kettenis, "A robust procedure for discontinuity handling in continuous system simulation," *Trans. Soc. Comput. Simul. Int.*, vol. 2, no. 3, pp. 189–205, Sep. 1985.

[48] L. F. Shampine, I. Gladwell, and R. W. Brankin, "Reliable solution of special event location problems for ODEs," *ACM transactions on Mathematical Software*, vol. 17, pp. 11–25, 1987.

[49] T. Park and P. I. Barton, "State event location in differential-algebraic models," *ACM Trans. Model. Comput. Simul.*, vol. 6, no. 2, pp. 137–165, 1996.

[50] J. M. Esposito, V. Kumar, and G. J. Pappas, "Accurate event detection for simulating hybrid systems," in *HSCC*, 2001, pp. 204–217.

[51] M. Althoff, O. Stursberg, and M. Buss, "Reachability analysis of nonlinear systems with uncertain parameters using conservative linearization," in *IEEE CDC*, 2008, pp. 4042–4048.

[52] C. Guernic and A. Girard, "Reachability analysis of hybrid systems using support functions," in *Proceedings of CAV*. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 540–554.

[53] G. Frehse and R. Ray, "Flowpipe-guard intersection for reachability computations with support functions," in *IFAC Conf. Analysis and Design of Hybrid Systems (ADHS)*, 2012, pp. 94–101.

[54] D. Ishii, K. Ueda, and H. Hosobe, "Simulation of hybrid systems based on hierarchical interval constraints," in *SimuTools*, 2009, p. 37.

[55] M. Maïga, N. Ramdani, and L. Travé-Massuyès, "A fast method for solving guard set intersection in nonlinear hybrid reachability." in *IEEE CDC*, 2013, pp. 508–513.

[56] N. Nedialkov, K. Jackson, and G. Corliss, "validated solutions of initial value problems for ordinary differential equations," *Applied Mathematics and Computation*, vol. 105, pp. 21–68, 1999.

[57] L. Jaulin, M. Kieffer, O. Didrit, and E. Walter, *Applied Interval Analysis: with examples in parameter and state estimation, robust control and robotics*. London: Springer-Verlag, 2001.

[58] G. Barequet and S. Har-Peled, "Efficiently approximating the minimum-volume bounding box of a point set in three dimensions," *J. Algorithms*, vol. 38, pp. 91–109, 2001.

[59] J. O'Rourke, "Finding minimal enclosing boxes," *International Journal of Computer Information Sciences*, vol. 14, no. 3, pp. 183–199, 1985.

[60] F. Vivien and N. Wicker, "Minimal enclosing parallelepiped in 3d," *Comput. Geom. Theory Appl.*, vol. 29, no. 3, pp. 177–190, Nov. 2004.

[61] M. Althoff, "Reachability analysis and its application to the safety assessment of autonomous cars," Ph.D. dissertation, 2010.

[62] L. J. Guibas, A. Nguyen, and L. Zhang, "Zonotopes as bounding volumes," in *ACM-SIAM Symposium on Discrete Algorithms*, 2003.

[63] O. Stursberg and B. Krogh, "Efficient representation and computation of reachable sets for hybrid systems," in *HSCC*, ser. LNCS, 2003, vol. 2623, pp. 482–497.

[64] M. Maïga, C. Combastel, N. Ramdani, and L. Travé-Massuyès, "Nonlinear hybrid reachability using set integration and zonotope enclosures," in *ECC*, 2014, pp. 234–239.

[65] G. Chabert and L. Jaulin, "Contractor programming," *Artificial Intelligence*, vol. 173, no. 11, pp. 1079 – 1100, 2009.

[66] T. Alamo, J. Bravo, and E. Camacho, "Guaranteed state estimation by zonotopes," *Automatica*, vol. 41, pp. 1035–1043, 2005.

[67] A. Lalami and C. Combastel, "A state bounding algorithm for linear systems with bounded input and bounded slew-rate," *European Control Conference*, 2007.

[68] W. Kühn, "Rigorously computed orbits of dynamical systems without the wrapping effect," *Computing*, vol. 61, no. 1, pp. 47–67, Sep. 1998.

[69] G. M. Ziegler, *Lectures on polytopes*. Springer-Verlag, 1995.

[70] H. Tuy, *Handbook of global optimization, D.C. Optimization: Theory, Methods and Algorithms (pp. 149–216)*. Dordrect: Kluwer Academic Publishers, 1995.

[71] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Trans. on Mathematical Software*, vol. 22, no. 4, pp. 469–483, 1996.

[72] M. B. Elowitz and S. Leibler, "A synthetic oscillatory network of transcriptional regulators," *Nature*, vol. 403, no. 6767, pp. 335–338, 01 2000.

Moussa Maïga received the M.S. degree in automatic control, computer and decision-making systems from Paul Sabatier University, Toulouse, France, in 2011, and the Ph.D. degree in automatic control from the University of Orléans, France, in 2015. He is currently with LAAS - Centre National de Recherche Scientifique (CNRS), University of Toulouse, France. His main research interests include diagnosis, reachability analysis, set-membership state and parameter estimation, for hybrid systems in the bounded-error framework.

Nacim Ramdani received the Engineer degree from Ecole Centrale de Paris, France, in 1990, the Ph.D. degree from the University of Paris-Est Créteil, France, in 1994 and the Habilitation in 2005. Since september 2010, he has been a Professor at the Université of Orléans (IUT de Bourges) and member of the Laboratoire PRISME. From 1996 to 2010, he was Maître de Conférences with the University Paris-Est Créteil. He was affiliated with the LIRMM CNRS Montpellier during 2005-2010 and also on secondment with the INRIA during 2007-2009. He is the co-Chair of the workgroup group on Set Computation Techniques within the French research group on Automatic Control (GDR MACS). His current research interests revolve around modelling, analysis, and estimation of non-linear and hybrid systems with applications to robotics, biomimetics and healthcare. He mainly focuses on interval methods and set computation techniques.

Louise Travé-Massuyès is Research Director of Centre National de la Recherche Scientifique (CNRS), leader of the "Diagnosis and Supervisory Control" (DISCO) Team within the LAAS-CNRS research laboratory, Toulouse, France. Her main research interests are in Dynamic Systems Supervision and Diagnosis with special focus on Qualitative, Model-Based Reasoning methods and data mining. She has been particularly active in bridging the AI and Control Engineering Model-Based Diagnosis fields, as leader of the BRIDGE Task Group of the MONET European Network of Excellence. She has been responsible for several industrial and European projects and published more than 250 papers in scientific journals and international conference proceedings and 4 books. She is coordinator of the Maintenance & Diagnosis Strategic Field within the Aerospace Valley World Competitiveness Cluster, and serves as the contact evaluator for the projects submitted to the French Research Funding Agency. She serves in the Editorial Board of the Artificial Intelligence Journal. She is member of the IFAC Safeprocess Technical Committee.

Christophe Combastel received his M.Sc. degree in Engineering (1997) and his Ph.D. in Control Systems (2000), both from the National Polytechnic Institute of Grenoble (G-INP), France. From 2001 to 2015, he was Associate Professor at ENSEA (Ecole Nationale Supérieure de l'Electronique et de ses Applications, Cergy, France) and member of ECS-Lab (Electronics and Control Systems Laboratory, EA3649). Since 2015, he is with the University of Bordeaux (IUT, electrical engineering dpt) and the IMS research lab (laboratoire de l'Intégration du Matériau au Système, CNRS UMR5218). As a member of the ARIA team in the Control Systems group of IMS, his research interests include interval and set-membership algorithms for integrity control applications ranging from on-line fault diagnosis to verified model-based design, with special emphasis on uncertainty propagation and model-based data fusion.