Bracketing backward reach sets of a dynamical system

Thomas Le Mézo, Luc Jaulin and Benoît Zerr

ENSTA-Bretagne, LabSTICC UMR CNR 6285, 2 rue François Verny, 29806 Brest, France.

***Abstract*—In this paper, we present a new method for bracketing (*i.e.*, characterizing from inside and from outside) backward reach set of the target region $\mathbb{T}$ of a continuous time dynamical system. The principle of the method is to formalize the problem as a *constraint network*, where the variables are the trajectories (or paths) of the system. The resolution is made possible by using *mazes* which is a set of paths that contain all solutions of the problem. As a result, we will be able to derive a method able to compute a backward reach set for a huge class of systems without any knowledge of a parametric Lyapunov function and without assuming any linearity for our system. The method will be illustrated on several examples.**

***Index Terms*—Abstract interpretation, Backward reach set, Basin of attraction, Stability, Constraint solving, Interval Computation**

I. Introduction

Reachability analysis is a classical approach in control theory Aubin and Frankowska 1990; Blanchini and Miani 2007 and has several applications, for instance *(i)* to validate some properties of cyber-physic systems Konecny et al. 2013; Taha and Duracz 2015, *(ii)* to ensure the safe configuration during the landing Bayen et al. 2007 or the take off Seube, Moitie, and Leitmann 2000 of an airplane or *(iii)* to avoid collisions Desilles, Zidani, and Cruck 2012 with other aircrafts. Reachability analysis allows us to guarantee that the system with a given control law will always reach a target Aubin 2001. In this paper, we deal with a dynamical system $\mathcal{S}$ defined by the following state equation:

$$\mathcal{S} : \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) \tag{1}$$

where $\mathbf{x}(t) \in \mathbb{R}^n$ is the state vector and $\mathbf{f} : \mathbb{R}^n \mapsto \mathbb{R}^n$ is the evolution function of $\mathcal{S}$. Denote by $\boldsymbol{\varphi}$ the flow map of $\mathcal{S}$, *i.e.*, with the initial condition $\mathbf{x}_0 = \mathbf{x}(0)$, the system $\mathcal{S}$ reaches the state $\boldsymbol{\varphi}(t, \mathbf{x}_0)$ at time t. The *backward reach set* $\mathbb{C}$ Aubin 2001Blanchini and Miani 2007 of a target $\mathbb{T} \subset \mathbb{R}^n$ is the set of initial states $\mathbf{x}_0$ from which $\mathcal{S}$ reaches the target $\mathbb{T}$ in a finite time:

$$\mathbb{C} = Bwd\,(\mathbb{T}) = \{\mathbf{x}_0 \mid \exists t \geq 0, \boldsymbol{\varphi}(t, \mathbf{x}_0) \in \mathbb{T}\}\,. \tag{2}$$

Note that $\mathbb{T} \subset \mathbb{C}$. In this paper, we propose a new method to *bracket* $\mathbb{C}$. By bracketing $\mathbb{C}$, we mean computing an inner approximation $\mathbb{C}^-$ and an outer approximation $\mathbb{C}^+$ of $\mathbb{C}$.

Two different types of approaches Mitchell 2007 are used to compute backward reach sets: the Lagrangian and the Eulerian. This classification is taken from the field of fluid mechanics Landau and Lifshitz 1987. In the Lagrangian point of view, the observer follows an individual fluid parcel as it moves through the fluid. In an Eulerian point of view, the observer stays static looking at the fluid moving around.

When we deal with a dynamical system such as (1), the velocity of the fluid corresponds to the evolution function $\mathbf{f}(\mathbf{x}(t))$, and the position of a fluid parcel at time t corresponds to the state $\mathbf{x}(t)$. A Lagrangian approach would require simulations to find states that reach the target. In the literature, the corresponding methods are generally restricted to linear dynamics Asarin, Dang, and Girard 2003; Asarin, Dang, and Girard 2007 where a closed form of the flow $\boldsymbol{\varphi}$ is available. They can also be extended to nonlinear systems if we use guaranteed integration Sandretto and Chapoutot 2016; Wilczak and Zgliczynski 2011, but the resulting method is slow. As shown in Lhommeau, Jaulin, and Hardouin 2011; Monnet, Ninin, and Jaulin 2016 a Lagrangian method requires many bisections with respect to the time line (for the integration of the state equation), but also on the state space (to bracket $\mathbb{C}$). The Eulerian methods are used for nonlinear systems Quincampoix 1992; Gao, Lygeros, and Quincampoix 2006; Kaynama et al. 2012 and try to avoid the integration of the state equation. Most of the corresponding algorithms rely on griding the state space Saint-Pierre 2002. To provide guaranteed results, griding methods require the knowledge of some Lipschitz constant which are rarely available in practice (Saint-Pierre, 1994).

Lyapunov-based methods (Ratschan and She, 2010; Delanoue, Jaulin, and Cottenceau, 2006; Gonzaga, Jungers, and Daafouz, 2012; Barreiro, Aracil, and Pagano, 2002), level-set methods (Mitchell, Bayen, and Tomlin, 2001; Biemond and Michiels, 2014), or barrier functions Bouissou et al. 2014; Esterhuizen and Lévine 2016 can also be considered as Eulerian since they only check the constraints on the state space and do not need to perform any integration through time. Now, these methods require a parametric expression for candidate Lyapunov-like functions She and Xue 2013. When the state-equation is polynomial, due to decidability of the theory of real-closed fields Tarski 1951, there is an algorithm that, for a given polynomial with parametric coefficients, decides whether there exist some consistent instantiations of these parameters. For the resolution of the problem, several methods exist, such as the ones based on sum of squares Parrilo and Lall 2003 or on Linear Matrix Inequalities Henrion, Lasserre, and Lofberg 2009. When the parametric model for the Lyapunov function is non polynomial, interval methods which take into account some quantifier eliminations Ratschan 2002 can also be used.

The main contribution of this paper is to show that an Eulerian approach can be implemented using interval analysis

and abstract interpretation. As a result, we will be able to bracket backward reach sets without any interval integration and without any model for a Lyapunov function. This was not possible before except for some specific cases (Henrion and Korda, 2014) such as when the system is linear Girard 2003.

The paper is organized as follows. Section II proposes a formalization of the backward reach set problem as a constraint network where the variables are paths of $\mathbb{R}^n$. The notion of maze is presented in Section III. A maze is a set of paths that are used to enclose an uncertain path. Section V illustrates the principle of the method on several test-cases taken from the literature and Section VI concludes the paper.

II. Formalization

This section introduces a set-membership based formalism in the context of dynamical systems described by a continuous-time state equation Tornil-Sin et al. 2012; Mazhoud et al. 2012. These corresponding notions will be used to characterize backward reach sets.

We define a *trajectory* as a smooth function $\mathbf{x}(\cdot)$ from $\mathbb{R}^+$ to $\mathbb{R}^n$. The *path* associated with a trajectory $\mathbf{x}(\cdot)$ is the set of all $\mathbf{x}(t) \in \mathbb{R}^n$, $t \geq 0$ Spivak 1965. A path which is consistent with a trajectory satisfying (1) is said to be *feasible*. A path is *valid* if at least one of its points is inside $\mathbb{T}$. Otherwise it is *dead*. A state $\mathbf{x}$ which belongs to a dead path is outside $\mathbb{C}$. Figure 1 provides five paths. All of them satisfy (1) except (v) which makes a loop and thus cannot satisfy a state equation in the form of (1) because of its uniqueness (Khalil, 2002). The path (ii) enters in $\mathbb{T}$ and converges to an equilibrium point. The path (iii) is valid since it enters in $\mathbb{T}$, but because it leaves it later, it contains some subpaths that are dead. The path (iv) corresponds to a limit cycle which is dead since it does not enter in $\mathbb{T}$.

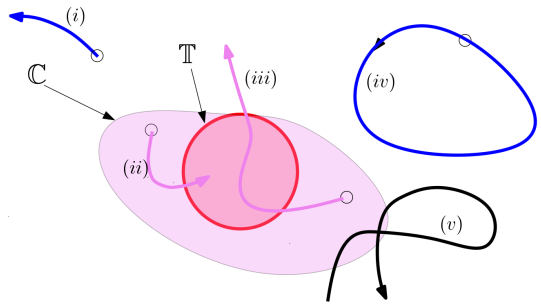

Fig. 1. The paths (i), (iv) are dead, (v) does not satisfy (1) since it loops, and paths (ii), (iii) are valid

Constraint network. We now recall briefly the notion of constraint network that will be used for the formalization of our problem. A *constraint network* (CN) Mackworth 1977 $\mathcal{H}$ is composed of a set of variables $\mathcal{V} = \{x_1, \ldots, x_n\}$, a set of constraints $\mathcal{C} = \{c_1, \ldots, c_m\}$ and a set of domains $\{\mathbb{X}_1, \ldots, \mathbb{X}_n\}$ containing the x_i's. The values for variables x_i of a CN can be symbols, real numbers Araya, Trombettoni, and Neveu 2012, vectors of $\mathbb{R}^n$, and sometimes trajectories (see Le Bars et al. 2012). The constraints can be equations between the variables (such as $x_3 = x_1 + \exp(x_2)$) or differential equations (such as $\dot{x}_3 \cdot \ddot{x}_1 + \exp(x_2) = 0$) and the domains can be intervals, boxes Jaulin et al. 2001, zonotopes Combastel 2005 or tubes Drevelle and Bonnifait 2013. The goal of propagation techniques is to contract as much as possible the domains for the variables of a CN without loosing any solution. The principle is to decompose all constraints of the initial CN into primitive constraints and to call the corresponding contractors Chabert and Jaulin 2009; Collins and Goldsztejn 2008 until no more contraction can be observed. Therefore, constraint propagation is only able to find an outer approximation of the problem. When we want to bracket the solution set from inside and outside, we have to build two complementary constraint networks $\mathcal{H}$ and $\overline{\mathcal{H}}$. Any instantiation of $(x_1, \ldots, x_n)$ is either a solution of $\mathcal{H}$ or a solution of $\overline{\mathcal{H}}$. Applying a constraint propagation alternatively on $\mathcal{H}$ and $\overline{\mathcal{H}}$ will allow us to build an inner and an outer approximation of the solution set.

Backward reach sets as a constraint network. For our problem, to apply a constraint approach, we need to define the constraint network $\mathcal{H}$. The variables are the paths of $\mathbb{R}^n$ which are consistent with $\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t))$. The unique constraint is the following: "the path should cross the target $\mathbb{T}$". And the complementary CN is $\overline{\mathcal{H}}$ has the same variables as $\mathcal{H}$ except that the constraint is now: "the path is dead, *i.e.*, it never reaches the target $\mathbb{T}$". We have now defined the variables and the constraints. It remains to define the domains for paths which is the objective of the next section.

III. Maze

The domains of a CN are in general a finite set (*i.e.*, the variables take their values inside a finite set), or intervals when the variables correspond to real numbers. Domains should be representable in the memory of a computer, and should have a lattice structure in order to allow intersections. In the literature, no type of domains has been proposed to enclose paths except mazes that have been used in (Le Mézo, Jaulin, and Zerr, 2018b) for integrating differential equations, but without a strong formalization. We now propose a formal definition of a maze with a link to lattice theory which will allow us to have some convergence proofs.

Definition. A *maze* $\mathcal{L}$ of $\mathbb{R}^n$ is composed of

- A paving $\mathcal{P}$, i.e., a union of non overlapping boxes which covers $\mathbb{R}^n$.
- A list of nonoverlapping doors between adjacent boxes, i.e., a path is allowed to go from one box to another adjacent box by crossing a door. Formally, a door is defined as a polytope of dimension $n-1$ included in the intersection between two adjacent boxes of dimension n.

A path is said *to belong* to a maze $\mathcal{L}$ if it crosses the boundary between two adjacent boxes of the paving through a door. This set-membership property is illustrated by Figure 2. In all our figures, doors are shown as holes between brown painted walls.

Road. In the computer, a maze can be represented by a list of boxes $[\mathbf{x}](1), [\mathbf{x}](2), \ldots$. Each box $[\mathbf{x}](i)$ is linked with the set $\mathcal{D}(i)$ of all doors it intersects. The pair $\langle [\mathbf{x}](i), \mathcal{D}(i) \rangle$ is called a road. For a given box $[\mathbf{x}]$, the set of possible roads $\langle [\mathbf{x}], \mathcal{D} \rangle$ is a complete lattice with respect to the inclusion. For the largest element $\langle [\mathbf{x}], \top \rangle$, all points of the boundary of $[\mathbf{x}]$

belong to a door. We say that all doors are open or equivalently that there is no wall. For the least element $\langle [\mathbf{x}], \bot \rangle$, $\mathcal{D}$ is empty and we say that all doors are closed. Given a road $\langle [\mathbf{x}], \mathcal{D} \rangle$, we define $\text{CONV}(\mathcal{D})$ as the polytope corresponding to the convex hull of $\mathcal{D}$. In Figure 2 the yellow polytopes correspond to $\text{CONV}(\mathcal{D})$ for each road of the maze.

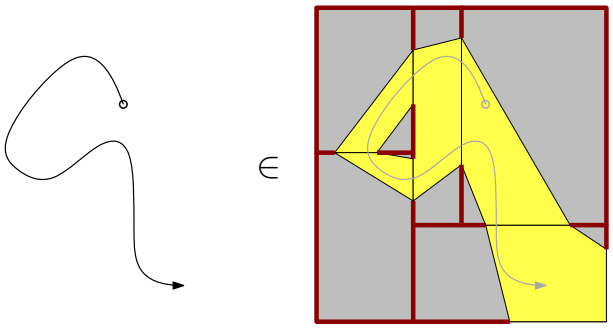

Fig. 2. The path (left) belongs to the maze (right). The maze contains 5 boxes and 6 doors.

Inclusion. Given two mazes $\mathcal{L}_a$ and $\mathcal{L}_b$. We say that $\mathcal{L}_a$ is included in $\mathcal{L}_b$, denoted by $\mathcal{L}_a \subset \mathcal{L}_b$, if (i) the boxes of $\mathcal{L}_a$ are subboxes of the boxes of $\mathcal{L}_b$ and (ii) all paths in $\mathcal{L}_a$ also belong to $\mathcal{L}_b$. An illustration of this definition is given by Figure 3 where the left maze contains less paths than the right maze. Indeed, the left maze $\mathcal{L}_a$ is included in the right maze $\mathcal{L}_b$, since (i) the 5 boxes of $\mathcal{L}_a$ are all subboxes of the 3 boxes of $\mathcal{L}_b$ and (ii) all doors of $\mathcal{L}_a$ are tighter than those of $\mathcal{L}_b$. It is trivial to check that if $\mathcal{L}_a \subset \mathcal{L}_b$ then all paths in $\mathcal{L}_a$ are also in $\mathcal{L}_b$, but there is no equivalence.

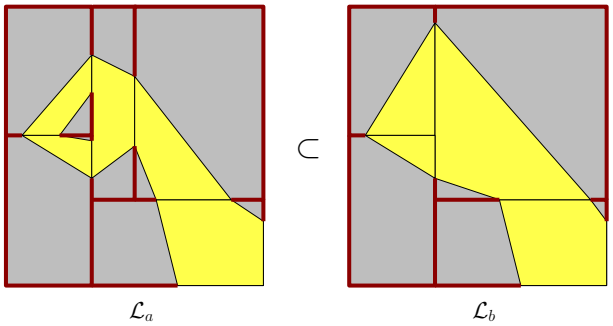

Fig. 3. Inclusion between two mazes

Lattice. Given two mazes $\mathcal{L}_a$ and $\mathcal{L}_b$, we define the *meet* $\mathcal{L}_a \cap \mathcal{L}_b$ as the largest maze (with respect to $\subset$) which is included in both $\mathcal{L}_a$ and $\mathcal{L}_b$. We also define the *join* $\mathcal{L}_a \cup \mathcal{L}_b$ as the smallest maze which contains both $\mathcal{L}_a$ and $\mathcal{L}_b$. It is trivial to check that the set of mazes of $\mathbb{R}^n$ is a lattice.

Door consistency. The road $\langle [\mathbf{x}], \mathcal{D} \rangle$ is said to be door-consistent with the system $\mathcal{S}$ if all paths in $[\mathbf{x}]$ that are consistent with the doors $\mathcal{D}$ and with the state equation (1) remain inside $\text{CONV}(\mathcal{D})$. This property means that inside $[\mathbf{x}]$ a trajectory cannot leave $\text{CONV}(\mathcal{D})$ and then come back. The concept is illustrated by Figure 4. The road (a) is not door-consistent since some trajectories which are consistent with the two doors of $\mathcal{D}$ may leave $\text{CONV}(\mathcal{D})$. The left door of road (a) may be contracted into road (b) to make the road door-consistent. It may also be inflated into roads (c) or (d) to get the door-consistency. We can notice that the inflation proposed by road (c) is more accurate than that of road (d).

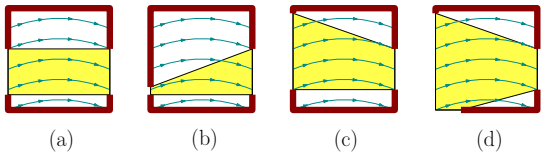

Fig. 4. The roads (b),(c),(d) are door-consistent. The polygon $\text{CONV}(\mathcal{D})$ is painted yellow

Maze. The maze $\mathcal{L}$ can be thus seen as a puzzle, the piece of which are the roads. If $[\mathbf{x}] \in \mathcal{P}$, we define as $\text{DOORS}([\mathbf{x}])$, the union of all doors associated to $[\mathbf{x}]$ and as $\text{WALLS}([\mathbf{x}])$ the set of all points of the boundary of $[\mathbf{x}]$ that are not inside a door of $[\mathbf{x}]$. In our implementation, we do not memorize the polytope associated with the road $\langle [\mathbf{x}](i), \mathcal{D}(i) \rangle$ since it can be obtained geometrically by using as first estimate the outer approximation of the vector field $[\mathbf{f}]([\mathbf{x}])$ in the pave. The yellow polytope corresponds to the set of all point $\mathbf{a} \in [\mathbf{x}]$ such that there exists a ray starting from $\mathbf{a}$ of direction $\mathbf{v} \in [\mathbf{f}]([\mathbf{x}])$ that intersects $\text{DOORS}([\mathbf{x}])$. Figure 5 depicts a maze with three roads: $\langle [\mathbf{x}](1), \{d(1), d(4)\} \rangle$, $\langle [\mathbf{x}](2), \{d(1), d(2)\} \rangle$, $\langle [\mathbf{x}](3), \{d(2), d(3)\} \rangle$. We have $\text{DOORS}([\mathbf{x}](1)) = d(1) \cup d(4)$, $\text{DOORS}([\mathbf{x}](2)) = d(1) \cup d(2)$ and $\text{DOORS}([\mathbf{x}](3)) = d(2) \cup d(3)$.

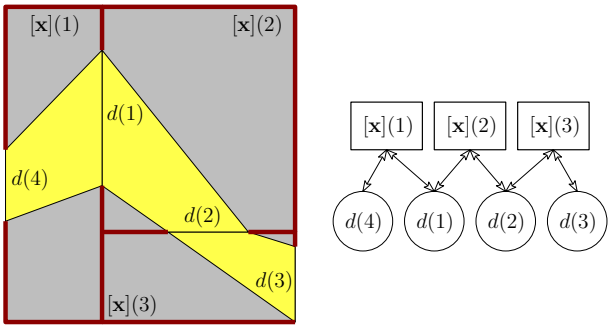

Fig. 5. Left: A maze made with three roads, Right: the machine representation of the links between boxes and doors

IV. METHOD

To bracket the backward reach set $\mathbb{C}$, we propose to build two complementary constraint networks. The first one defines the valid paths (*i.e.*, the paths that satisfy the state equation and that reach the target $\mathbb{T}$) and the second one defines the dead paths (*i.e.*, the paths that satisfy the state equation and that never reach $\mathbb{T}$). Recall that, since the system (1) is deterministic, there exists a one-to-one correspondence between each path and the corresponding initial state. This is why the bisections which will generate the paving will take place in the space $\mathbb{R}^n$ of all initial states and not in the set of paths which has a dimension equal to infinity.

A. Computing an inner approximation

Computing an inner approximation $\mathbb{C}^-$ for $\mathbb{C}$ amounts to computing an outer approximation of the complementary $\overline{\mathbb{C}}$

of $\mathbb{C}$:

$$\overline{\mathbb{C}} = \{\mathbf{x}_0 \mid \forall t \geq 0, \boldsymbol{\varphi}(t, \mathbf{x}_0) \notin \mathbb{T}\}. \quad (3)$$

Since, we deal with paths, we search for all dead paths. For this, we need two contractors: the target contractor and the flow contractor. The goal of these operators is to contract the roads composing the current maze without removing a single dead path.

Target contractor. For each road of the maze, if the corresponding box $[\mathbf{x}]$ is included in $\mathbb{T}$ then we contract the road to $([\mathbf{x}], \bot)$, i.e., we close all doors linked to $[\mathbf{x}]$.

Flow contractor. We contract each road $\langle [\mathbf{x}], \mathcal{D} \rangle$ of the maze with respect to the constraint $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$. To do this, we contract the doors of $\mathcal{D}$ without loosing any point in the set of all $\mathbf{a} \in \text{DOORS}([\mathbf{x}])$ such that there exists a ray starting from $\mathbf{a}$ of direction $\mathbf{v} \in [\mathbf{f}]([\mathbf{x}])$ that does not intersect $\text{WALL}([\mathbf{x}])$.

We can check that contracted road is door-consistent since no trajectory can enter in $\text{CONV}(\mathcal{D})$ (see Figure 4(b)).

An illustration of the corresponding contractor is given by Figure 6. (a) illustrates a road with the vector field $\mathbf{f}(\mathbf{x})$, three doors corresponding to $\mathcal{D}$, and the polytope (yellow) $\text{CONV}(\mathcal{D})$. In (b) an external event comes to contract the right door. The vector field is now represented by the gray cone $[\mathbf{f}]([\mathbf{x}])$ computed using interval computation. The contraction of the bottom and left doors and the updated polytope $\text{CONV}(\mathcal{D})$ are representes in (c).

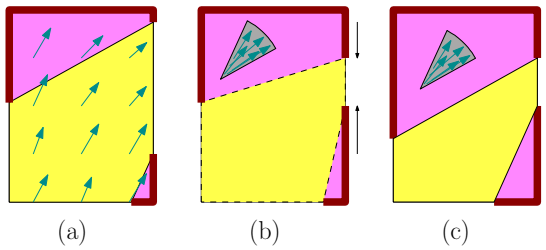

Fig. 6. Illustration of the flow contractor; the road is contracted backward and the eliminated zones are in magenta

Propagation. To compute an inner approximation of $\mathbb{C}$, we apply the two contractors several times, as illustrated by Figure 7. In Sub-figure (1), all doors of all roads are assumed to be open. Since The magenta box in Sub-figure (2) is inside $\mathbb{T}$ (red curve) all corresponding doors are closed. In Sub-figures (3)-(6) the propagation contracts backward the roads without eliminating any dead path, which are all trapped inside the yellow zone representing the current maze.

B. Computing an outer approximation

In the previous section, to compute an inner approximation $\mathbb{C}^-$ for $\mathbb{C}$, we proposed to remove valid paths. The set $\mathbb{T}^-$ contains t the states $\mathbf{x}_0$ that reach $\mathbb{T}$ for at least one $t \geq 0$. In this section, we compute an outer approximation $\mathbb{C}^+$ for $\mathbb{C}$. For this, we need to remove dead paths Le Mézo, Jaulin, and Zerr 2018a. Using a pure constraint approach Desrochers and Jaulin 2016, as it was the case for the inner approximation, is not possible anymore. Indeed, checking that a state $\mathbf{x}_0$ corresponds to a dead path requires to check that it never reaches $\mathbb{T}$ for all $t \geq 0$. The only possibility for this task is to

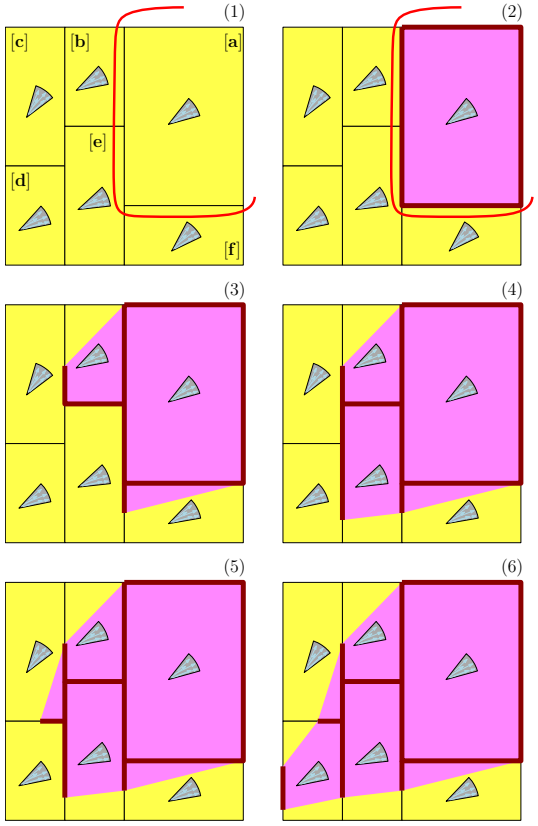

Fig. 7. Inner propagation; at each step, the magenta area inflates and is necessary inside $\mathbb{C}$, even if we did not reach the fixed point yet.

use the inflation principle provided by the theory of abstract interpretation Cousot and Cousot 1977; Goubault and Putot 2006 based on invariant-set theory. The principle is to generate a sequence of nested subsets $\mathbb{X}(k), k \geq 0$ where $\mathbb{X}(0)$ is an outer approximation of $\mathbb{T}$. More precisely, we build $\mathbb{X}(k+1)$ by an inflation of $\mathbb{X}(k)$ by adding all paths (or initial states) that enter inside $\mathbb{X}(k)$. The process terminates at $\bar{k}$ when the fixed point is reached, i.e, when no more path can be added. The set $\mathbb{X}(\bar{k})$ is said to be positive invariant. For this purpose, we will use two inflators: the target inflator and the flow inflator.

Target inflator. For each road $\langle [\mathbf{x}], \mathcal{D} \rangle$ of the maze, if the corresponding box $[\mathbf{x}]$ intersects $\mathbb{T}$ then we inflate the road to $\langle [\mathbf{x}], \top \rangle$, i.e., we open all doors linked to $[\mathbf{x}]$.

Flow inflator. We inflate backward each road $\langle [\mathbf{x}], \mathcal{D} \rangle$ of the maze with respect to the constraint $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$. To do this, we add/inflate doors of $\mathcal{D}$ to enclose the set of all $\mathbf{a} \in \text{WALLS}([\mathbf{x}])$ such that there exists a ray starting from $\mathbf{a}$ of direction $\mathbf{v} \in [\mathbf{f}]([\mathbf{x}])$ that intersects $\text{DOORS}([\mathbf{x}])$.

We can check that inflated roads are door-consistent since no trajectory can enter in $\text{CONV}(\mathcal{D})$ (see Figure 4(c)).

An illustration is given by Figure 8. (a) illustrates a road with the vector field $\mathbf{f}(\mathbf{x})$, two doors corresponding to $\mathcal{D}$, and the yellow polytope $\text{CONV}(\mathcal{D})$. In (b) an external event comes to inflate the right door. The vector field is now represented by the gray cone $[\mathbf{f}]([\mathbf{x}])$. The inflation of the bottom door, the creation of the left door and the updated polytope $\text{CONV}(\mathcal{D})$ are represented in (c).

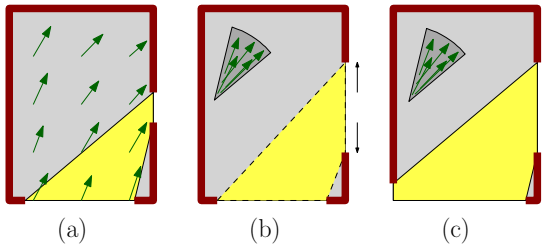

Fig. 8. Illustration of the flow inflation; the road is inflated backward

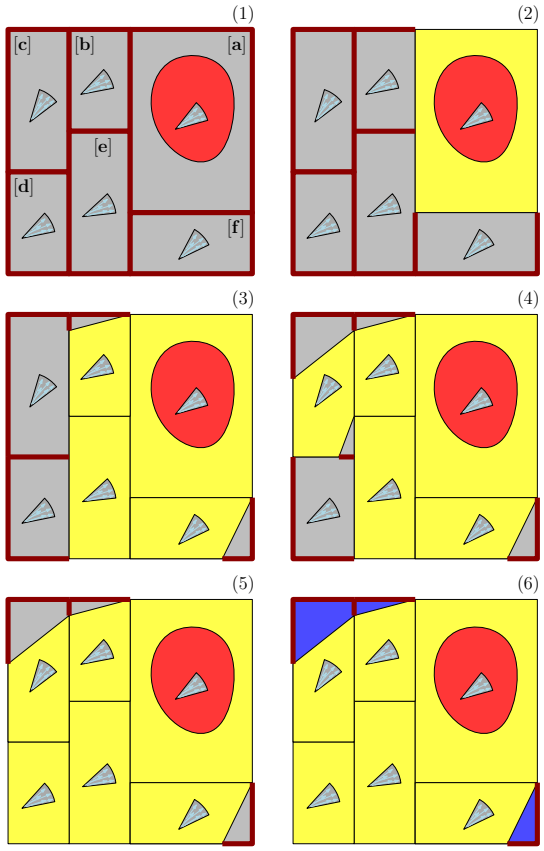

Fig. 9. Outer propagation: Once the fixed-point is reached, we conclude that the blue zone is outside $\mathbb{C}$.

Propagation. The propagation of the inflations is illustrated by Figure 9. In (1), all doors are closed and the red set corresponds to $\mathbb{T}$. In (2) the yellow box is known to intersect $\mathbb{T}$ and the corresponding doors are open. In (3)-(5) the backward propagation takes place and doors open accordingly. Gray areas have not been processed yet. Once the fixed point is reached, we can conclude that the remaining area, is outside $\mathbb{C}$ (painted blue), or equivalently that the yellow area corresponds to an outer approximation $\mathbb{C}^+$ of $\mathbb{C}$.

Remark. Both Flow inflator and Flow contractor operator can be expressed as a CN using geometrical contractors as in (Guyonneau, Lagrange, and Hardouin, 2013) or with a Time-Elapse Operator as in (Halbwachs, Proy, and Roumanoff, 1997).

C. Main algorithm

This section proposes an algorithm to compute an inner and an outer approximation of the backward reach set. It uses the following contraction/inflation procedures.

TargetContract$\left(\mathcal{L}^{in}\right)$ calls the target contractor proposed in Section IV-A. For each $\langle [\mathbf{x}], \mathcal{D}\rangle$ in $\mathcal{L}^{in}$ if $[\mathbf{x}] \subset \mathbb{T}$ then we close the doors, i.e., $\mathcal{D} = \bot$.

FlowContract$\left(\mathcal{L}^{in}\right)$ calls the flow contractor proposed in Section IV-A to shrink the doors without removing a single path consistent with the state equation.

TargetInflate$\left(\mathcal{L}^{in}\right)$ calls the target inflator proposed in Section IV-B. For each $\langle [\mathbf{x}], \mathcal{D}\rangle$ in $\mathcal{L}^{out}$, if $[\mathbf{x}] \cap \mathbb{T} \neq \emptyset$ then, we open the doors, i.e., $\mathcal{D} = \top$

FlowInflate$(\mathcal{L}^{out})$ inflates doors as explained in Section IV-B.

Bisect($[\mathbf{x}]$) cuts a box into two nonoverlapping subboxes. For simplicity, in the algorithm, all doors have to be reconstructed and we do not take advantage of the doors computed before bisection.

1 $\mathcal{P} := \{\mathbb{R}^n\}$;
2 $\mathbb{C}^- := \emptyset$;$\mathcal{L}^{in} = \{\langle [\mathbf{x}], \top\rangle \mid [\mathbf{x}] \in \mathcal{P}\}$
TargetContract$\left(\mathcal{L}^{in}\right)$
Repeat several times
FlowContract$\left(\mathcal{L}^{in}\right)$
For each $\langle [\mathbf{x}], \mathcal{D}\rangle$ in $\mathcal{L}^{in}$
$\mathbb{C}^- = \mathbb{C}^- \cup ([\mathbf{x}] \setminus \textsc{Conv}(\mathcal{D}))$
3 $\mathbb{C}^+ := \emptyset$; $\mathcal{L}^{out} = \{\langle [\mathbf{x}], \bot\rangle \mid [\mathbf{x}] \in \mathcal{P}\}$
TargetInflate$(\mathcal{L}^{out})$
Repeat up to the fixed point
FlowInflate$(\mathcal{L}^{out})$
For each $\langle [\mathbf{x}], \mathcal{D}\rangle$ in $\mathcal{L}^{out}$
$\mathbb{C}^+ = \mathbb{C}^+ \cup \textsc{Conv}(\mathcal{D})$
4 If not accurate enough
Bisect all $[\mathbf{x}]$ in $\mathcal{P}$ s.t. $[\mathbf{x}] \not\subset \mathbb{C}^- \cup \overline{\mathbb{C}^+}$
Go to Step 2
Else return $\mathbb{C}^-$,$\mathbb{C}^+$.

- Step 1 corresponds to the initialization. The paving contains a single box: $[\mathbf{x}] = \mathbb{R}^n$.
- Step 2 proceeds to the inner contractions for each road $\langle [\mathbf{x}](i), \mathcal{D}(i)\rangle$ of the inner maze $\mathcal{L}^{in}$ and add the resulting inner approximation to $\mathbb{C}^-$. The approximation $\mathbb{C}^-$ is initialized to empty and will inflate while the algorithm progresses. Note that reaching a fixed point is not mandatory here contrary to the inflation propagation of Step 3.
- Step 3 initializes $\mathbb{C}^+$ to empty and applies the outer fixed-point procedure. The outer approximation $\mathbb{C}^+$ is built as the union of all $\textsc{Conv}(\mathcal{D})$ of the maze.
- Step 4. If the maze is not accurate enough, all boxes of $\mathcal{P}$ are then bisected and we go to Step 2. The new iteration will compute a more accurate approximation.

Proposition. The algorithm computes the following enclosure

$$\mathbb{C}^- \subset \mathbb{C} \subset \mathbb{C}^+, \tag{4}$$

or using an interval notation, we have $\mathbb{C} \in [\mathbb{C}^-, \mathbb{C}^+]$.

Proof. We first prove the inclusion ($\mathbb{C}^- \subset \mathbb{C}$). For this, we show that any point $\mathbf{c}$ of $\mathbb{C}^-$ has been eliminated by a contractor associated with the complementary set $\overline{\mathbb{C}}$ of

$\mathbb{C}$. Indeed, the point $\mathbf{c}$ belongs to a polytope that has been eliminated at Step 2 either (i) by the target contractor or (ii) by the flow contractor. In case (i), $\mathbf{c} \in \mathbb{T}$ and thus $\mathbf{c} \in \mathbb{C}$. In case (ii), the path with the initial condition $\mathbf{c}$ reaches the target $\mathbb{T}$, or equivalently reaches a zone which has been proved to be inside $\mathbb{C}$. Thus, in both cases, $\mathbf{c} \in \mathbb{C}$, which concludes the first part of the proof.

To prove that $\mathbb{C} \subset \mathbb{C}^+$, we will prove that if a state $\mathbf{c}$ is outside $\mathbb{C}^+$, it is necessary outside $\mathbb{C}$. From the procedure at Step 3, the point $\mathbf{c}$ is outside a positive invariant set $\mathbb{X}(\overline{k})$ which encloses $\mathbb{C}$. Since the system is deterministic, from Blanchini and Miani 2007, we know that no feasible path starting from $\mathbf{c}$ can enter in $\mathbb{X}(\overline{k})$ and thus cannot enter in $\mathbb{C}$. Therefore $\mathbf{c} \notin \mathbb{C}$.

Complexity. Due to the paving approximation with boxes, for a given required accuracy, the worst-case complexity of the algorithm is exponential with respect to the dimension of $\mathbf{x}$. Even if we observed that the polytope approximations and the algorithms used inside the contractors strongly determine the computing time and the degree of accuracy, they do not change the theoretical complexity.

V. Test-cases

We consider here several test-cases in order to illustrate the principle of the method. For all these test-cases, computing times are given for an Intel i5-3320M CPU.

We designed a simple python library that can be found at http://www.ensta-bretagne.fr/lemezo/pyinvariant/pyinvariant.html which implements the algorithms of this paper. The reader can test their own two dimensional examples.

Test-case 1. *Reverse Van der Pol system*. Consider the system described by:

$$\begin{cases} \dot{x}_1 = -x_2 \\ \dot{x}_2 = -\left(1 - x_1^2\right) \cdot x_2 + x_1. \end{cases} \tag{5}$$

The corresponding vector field is depicted in Figure 10.

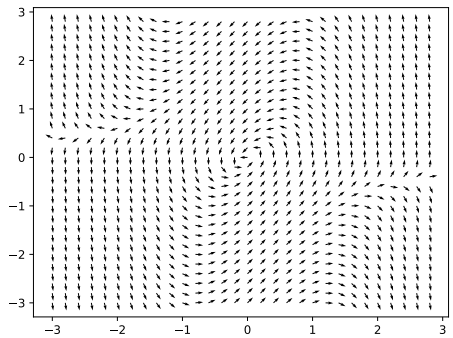

Fig. 10. Vector field of the *reverse* Van der Pol system

If we apply our algorithm with the target $\mathbb{T}_1 = \{\mathbf{x} \mid x_1^2 + x_2^2 \leq (0.4)^2\}$, we get the approximation of Figure 11 in about 0.9 sec and with 12 iterations. The red circle corresponds to the target $\mathbb{T}$. The magenta area is proved to be inside $\mathbb{C}$ whereas the blue area is proved to be outside $\mathbb{C}$. Figure 12 gives also a trend of the evolution, as a function of the iteration, of the computation time and the volume of the uncertainty layer (in yellow) which has not yet been proven to be inside or outside $\mathbb{C}$.

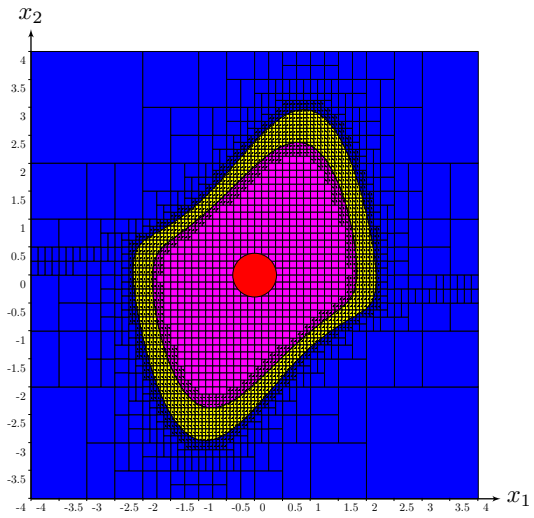

Fig. 11. Bracketing the backward reach set of the target $\mathbb{T}_1$ (the red circle). The frame box is $[-4, 4] \times [-4, 4]$.

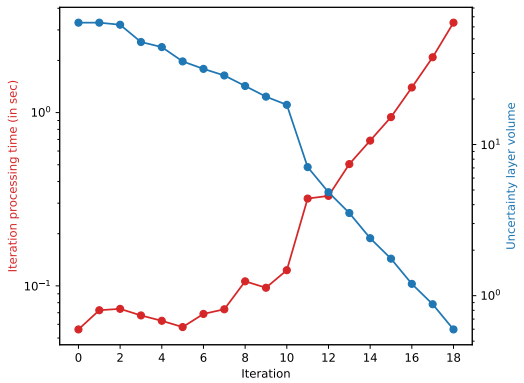

Fig. 12. Computation time and volume of the uncertainty layer at each iteration for the reverse Van Der Pol system in logarithmic scale.

We can also choose a target that is not centered on an equilibrium point. If we take $\mathbb{T}_2 = \{[0.6, 1.4] \times [-0.4, 0.4]\} \cup \{[-1, -0.6] \times [0.3, 0.7]\}$, we obtain Figure 13 in about 3.7 sec and with 14 iterations.

Test-case 2. *Rayleigh system.* Consider the system described by:

$$\begin{cases} \dot{x}_1 = -x_2 \\ \dot{x}_2 = (x_1 + x_2^3 - x_2) \end{cases} \tag{6}$$

and a target $\mathbb{T} = \{\mathbf{x} \mid x_1^2 + x_2^2 \leq (0.4)^2\}$. In about 2.3 sec and with 14 iterations, we get the approximation given by Figure 14.

Test-case 3. Consider the system Bacha, Jerbi, and Braiek 2008 described by the state equation

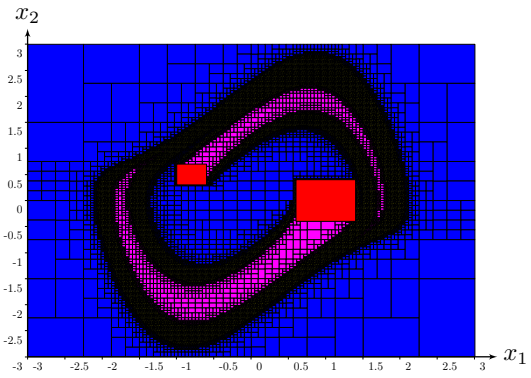

Fig. 13. Bracketing the backward reach set of the target $\mathbb{T}_2$ (red box). The frame box is $[-3,3]\times[-3,3]$.

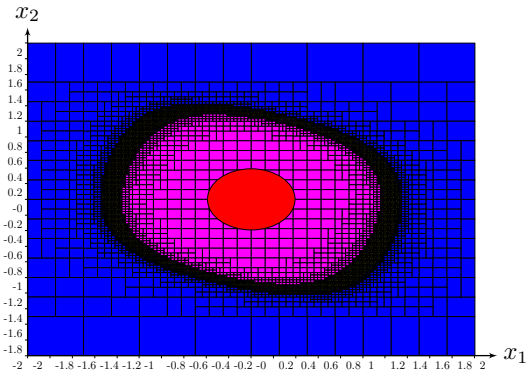

Fig. 14. Bracketing the backward reach set of the target $\mathbb{T}$ (the red circle). The frame box is $[-2,2]\times[-2,2]$.

$$\begin{cases} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -0.5x_2 - \sin(x_1+0.412) + \sin(0.412) \end{cases} \tag{7}$$

where $\mathbb{T} = \{\mathbf{x} \mid x_1^2 + x_2^2 \leq (0.4)^2\}$. In about 2.3 sec and 14 iterations, we get the paving depicted on Figure 15 which is consistent (and more accurate) with the results of Bacha, Jerbi, and Braiek 2008 obtained using a geometrical approach.

Test-case 4. We consider a Dubins car (Dubins, 1957) described by the following state equation

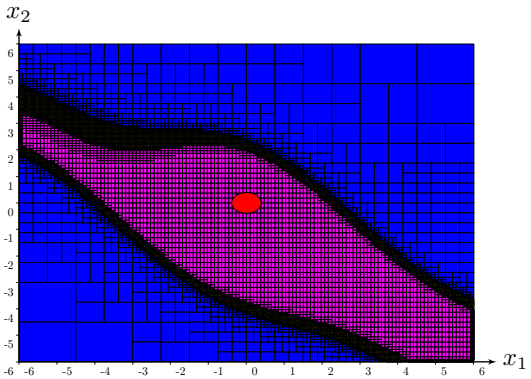

Fig. 15. Bracketing the backward reach set of $\mathbb{T}$ (the red circle). The frame box is $[-6,6]\times[-6,6]$.

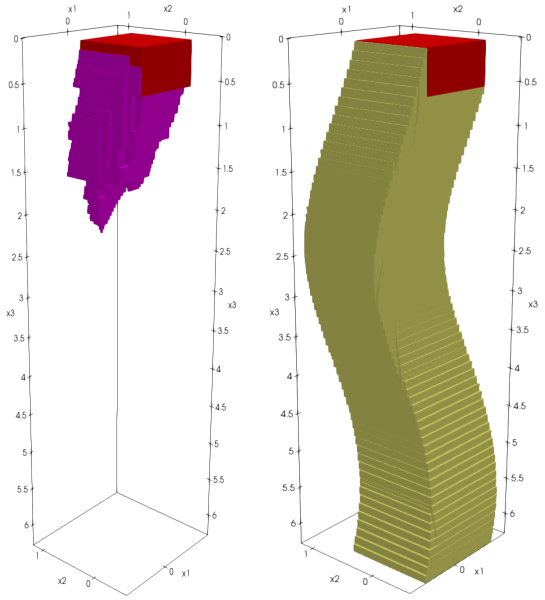

Fig. 16. Bracketing the backward reach set of $\mathbb{T}$ (the red box at the top). The frame box is $[-10,10]\times[-10,10]\times[0,2\pi]$. On the right $\mathbb{C}^-$ is in magenta and on the left $\mathbb{C}^+$ is painted yellow.

$$\begin{cases} \dot{x}_1 &= -0.1\cos(x_3) \\ \dot{x}_2 &= -0.1\sin(x_3) \\ \dot{x}_3 &= -0.3 \end{cases} \tag{8}$$

and the target set $\mathbb{T} = [-0.5,0.5]\times[-0.5,0.5]\times[0,0.5]$. In about 173 sec and with 22 iterations, we get the enclosure illustrated by Figure 16. Note that the poor quality of the approximation and the large computing time shows the difficulty of the approach to deal with problems with a dimension greater than three.

In our implementation, we used the *Parma Polyhedra Library* (PPL) to compute the projections, intersections and convex envelops of polytopes (when $n \geq 3$). Very few libraries are available nowadays and they mainly work with rationals as PPL. The main drawback of such libraries is the fact that they try to compute a convex polyhedral solution with rational numbers, which makes the algorithm inefficient.

Test-case 5. Two-wells system

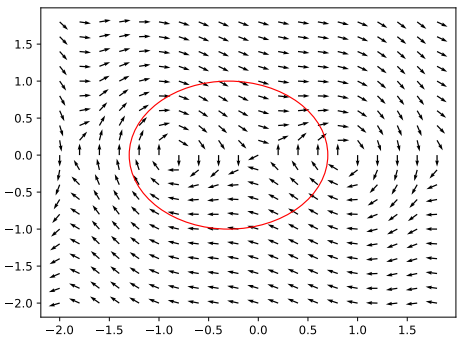

Fig. 17. Vector field of the two-wells system

We define the attraction basin of a set $\mathbb{T}$ as:

$$\mathbb{A} = \{\mathbf{x} | \exists t_1 \geq 0, \forall t \geq t_1, \boldsymbol{\varphi}(t, \mathbf{x}) \in \mathbb{T}\}$$

Since

$$\begin{array}{rc} & \mathbf{x} \in \mathbb{A} \\ \Leftrightarrow & \exists t_1 \geq 0, \forall t \geq t_1, \boldsymbol{\varphi}(t, \mathbf{x}) \in \mathbb{T} \\ \Leftrightarrow & \exists t_1 \geq 0, \neg(\exists t \geq t_1, \boldsymbol{\varphi}(t, \mathbf{x}) \in \overline{\mathbb{T}}) \\ \Leftrightarrow & \exists t_1 \geq 0, \boldsymbol{\varphi}(t_1, \mathbf{x}) \in \overline{Bwd(\overline{\mathbb{T}})} \\ \Leftrightarrow & \mathbf{x} \in \overline{Bwd(\overline{Bwd(\overline{\mathbb{T}})})}, \end{array}$$

we can use our algorithm to compute $\mathbb{A}$.

For the system described by the state equation

$$\begin{cases} \dot{x}_1 = & x_2 \\ \dot{x}_2 = & \frac{8}{25}x_1^5 - \frac{4}{3}x_1^3 + \frac{4}{5}x_1 - \frac{3}{10}x_2 \end{cases} \tag{9}$$

and a target $\mathbb{T} = \{\boldsymbol{x} | (x_1 + 0.3)^2 + (x_2)^2 \leq 1\}$, Figure 18 gives the approximation of $Bwd(\mathbb{T})$. Figure 19 provides an approximation of $\overline{Bwd(\overline{\mathbb{T}})}$ which also corresponds the largest positive invariant set included in $\mathbb{T}$. Figure 20 corresponds to the attraction basin $\overline{Bwd(\overline{Bwd(\overline{\mathbb{T}})})}$. Note that the uncertainty layer (in yellow) is large but could be reduced at the cost of more computation as shown on Figure 21. The cost is exponential as a function of the number of iterations. All these images have been generated in about 45 sec and with 14 iterations.

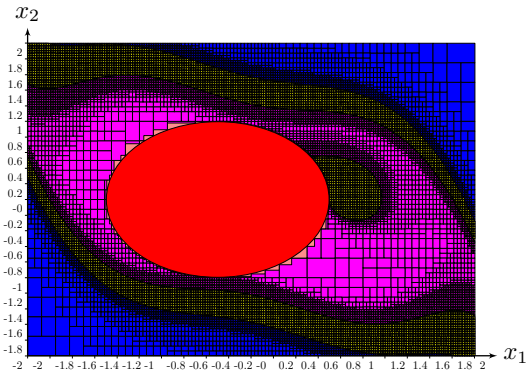

Fig. 18. Backward reach set $Bwd(\mathbb{T})$ of the target $\mathbb{T}$ (the red circle). The frame box is $[-2, 2] \times [-2, 2]$.

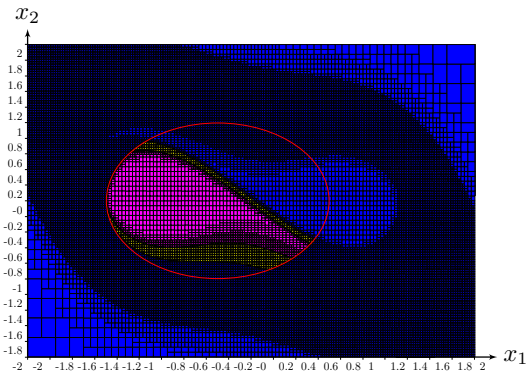

Fig. 19. Largest positive invariant set $\overline{Bwd(\overline{\mathbb{T}})}$ included in $\mathbb{T}$

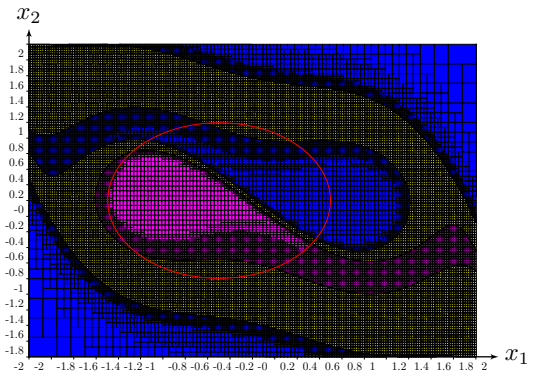

Fig. 20. Characterization of $\mathbb{A} = \overline{Bwd(\overline{Bwd(\overline{\mathbb{T}})})}$, the capture set of $\mathbb{T}$

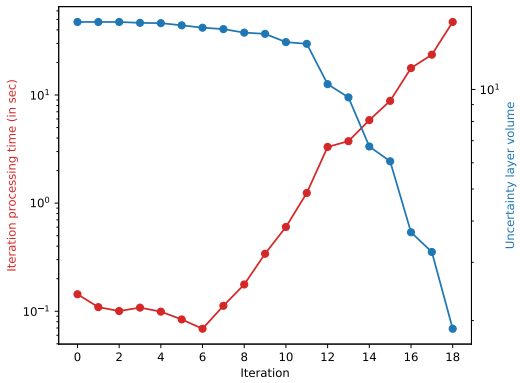

Fig. 21. Computation time and volume of the uncertainty layer at each iteration for the reverse Two-wells system in logarithmic scale.

VI. Conclusion

In this paper, we have proposed a new method able to discretize a dynamical system described by a state equation. The corresponding discrete structure is called a *maze*. It corresponds to a graph with boxes inside each nodes and doors between nodes. Our main motivation for the discretization is to bracket the backward reach set associated to a target $\mathbb{T}$. The inner contraction was made possible using classical tools taken from constraint programming whereas the outer approximation was based on the theory of abstract interpretation and requires fixed point procedures.

Now, due to the fact that a pure Eulerian approach has been developed, no integration method was used, which makes the method much more efficient than those which require bisections with respect to both the state space and the time line.

Since the discretization method generates a structure which is similar to a region graph, a generalization to hybrid systems (Podelski and Wagner, 2006; Doyen, Henzinger, and Raskin, 2005), where the state can jump from one state to another when some time-independent guard conditions are satisfied, could be a straightforward extension of our approach.

ACKNOWLEDGMENT

This work has been supported by the *French Government Defense procurement and technology agency* (DGA).
It also benefited from the support of the project CONTREDO of the French National Research Agency (ANR-16-CE33-0024).

REFERENCES

Araya, I., G. Trombettoni, and B. Neveu (2012). "A Contractor Based on Convex Interval Taylor". In: *Proc. of CPAIOR*. LNCS 7298, Springer, pp. 1–16.

Asarin, E., T. Dang, and A. Girard (2003). "Reachability analysis of nonlinear systems using conservative approximation". In: *Hybrid Systems: Computation and Control*. Springer Berlin Heidelberg, pp. 20–35.

– (2007). "Hybridization methods for the analysis of nonlinear systems". In: *Acta Informatica* 7.43, pp. 451–476.

Aubin, JP. (2001). "Viability Kernels and Capture Basins of Sets Under Differential Inclusions". In: *SIAM Journal on Control and Optimization* 40.3, pp. 853–881. ISSN: 0363-0129. DOI: 10.1137/S036301290036968X. (Visited on 07/24/2015).

Aubin, JP. and H. Frankowska (1990). *Set-Valued Analysis*. Birkhäuser, Boston.

Bacha, A., H. Jerbi, and N.B. Braiek (2008). "On the Estimation of Asymptotic Stability Region of Nonlinear Polynomial Systems: Geometrical Approaches". In: *New Approaches in Automation and Robotics*. Ed. by H. Aschemann. I-Tech Education and Publishing, pp. 55–72.

Barreiro, A., J. Aracil, and D. Pagano (2002). "Detection of attraction domains of non-linear systems using bifurcation analysis and Lyapunov functions". In: *International Journal of Control* 75.5, pp. 314–327. DOI: 10.1080/00207170110110568.

Bayen, AM. et al. (2007). "Aircraft Autolander Safety Analysis Through Optimal Control-Based Reach Set Computation". In: *Journal of Guidance, Control, and Dynamics* 30.1, pp. 68–77. ISSN: 0731-5090. DOI: 10.2514/1.21562.

Biemond, JJB. and W. Michiels (2014). "Estimation of basins of attraction for controlled systems with input saturation and time-delays". In: *IFAC Proceedings Volumes* 47.3, pp. 11006–11011.

Blanchini, F. and S. Miani (2007). *Set-Theoretic Methods in Control*. Springer Science & Business Media. ISBN: 978-0-8176-4606-6.

Bouissou, O. et al. (2014). "Computation of parametric barrier functions for dynamical systems using interval analysis". In: *2014 IEEE 53rd Annual Conference on Decision and Control (CDC)*, pp. 753–758. DOI: 10.1109/CDC.2014.7039472.

Chabert, G. and L. Jaulin (2009). "QUIMPER, A Language for Quick Interval Modelling and Programming in a Bounded-Error Context". In: *Artificial Intelligence* 173, pp. 1079–1100.

Collins, P. and A. Goldsztejn (2008). "The Reach-and-Evolve Algorithm for Reachability Analysis of Nonlinear Dynamical Systems". In: *Electronic Notes in Theoretical Computer Science* 223, pp. 87–102.

Combastel, C. (2005). "A State Bounding Observer for Uncertain Non-linear Continuous-time Systems based on Zonotopes". In: *CDC-ECC '05*.

Cousot, P. and R. Cousot (1977). "Abstract Interpretation: A Unified Lattice Model for Static Analysis of Programs by Construction or Approximation of Fixpoints". In: *Conference Record of the Fourth ACM Symposium on Principles of Programming Languages*. Los Angeles, California, pp. 238–252.

Delanoue, N., L. Jaulin, and B. Cottenceau (2006). "Attraction domain of a nonlinear system using interval analysis". In: *Twelfth International Conference on Principles and Practice of Constraint Programming (IntCP 2006)*. France, Nantes.

Desilles, A., H. Zidani, and E. Cruck (2012). "Collision analysis for an UAV". In: *AIAA Guidance, Navigation, and Control Conference*. American Institute of Aeronautics and Astronautics, pp. 13–16.

Desrochers, B. and L. Jaulin (2016). "A minimal contractor for the polar equation: Application to robot localization". In: *Engineering Applications of Artificial Intelligence* 55, pp. 83 –92. ISSN: 0952-1976. DOI: http://dx.doi.org/10.1016/j.engappai.2016.06.005.

Doyen, L., TA Henzinger, and JF Raskin (2005). "Automatic rectangular refinement of affine hybrid systems". In: *Proceedings of the Third International Conference on Formal Modeling and Analysis of Timed Systems*. Springer, pp. 144–161.

Drevelle, V. and P. Bonnifait (2013). "Localization confidence domains via set inversion on short-term trajectory". In: *IEEE Transactions on Robotics*.

Dubins, L. E. (1957). "On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents". In: *American Journal of Mathematics* 79.3, pp. 497–516.

Esterhuizen, W. and J. Lévine (2016). "Barriers and potentially safe sets in hybrid systems: Pendulum with non-rigid cable". In: *Automatica* 73, pp. 248 –255. ISSN: 0005-1098.

Gao, Y., J. Lygeros, and M. Quincampoix (2006). "The reachability problem for uncertain hybrid systems revisited: a viability theory perspective". In: *Hybrid Systems: Computation and Control*. Ed. by J. Hespanha and A. Tiwari. Springer-Verlag, pp. 242–256.

Girard, A. (2003). "Computation and stability analysis of limit cycles in piecewise linear hybrid systems". In: *1st IFAC Conference on Analysis and Design of Hybrid Systems*, pp. 181–186.

Gonzaga, C., M. Jungers, and J. Daafouz (2012). "Stability analysis and stabilisation of switched nonlinear systems". In: *International Journal of Control* 85.7, pp. 822–829.

Goubault, E. and S. Putot (2006). "Static Analysis of Numerical Algorithms". In: *In Proceedings of SAS 06, LNCS 4134*. Springer-Verlag, pp. 18–34.

Guyonneau, R., S. Lagrange, and L. Hardouin (2013). "A Visibility Information for Multi-Robot Localization". In:

IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).

Halbwachs, Nicolas, Yann-Erick Proy, and Patrick Roumanoff (1997). "Verification of Real-Time Systems using Linear Relation Analysis". In: *Formal Methods in System Design* 11.2, pp. 157–185. ISSN: 1572-8102. DOI: 10.1023/A:1008678014487. URL: https://doi.org/10.1023/A:1008678014487 (visited on 02/20/2019).

Henrion, D. and M. Korda (2014). "Convex computation of the region of attraction of polynomial control systems". In: *IEEE Transactions on Automatic Control* 59.2, pp. 297–312.

Henrion, D., JB. Lasserre, and J. Lofberg (2009). "GloptiPoly 3: moments, optimization and semidefinite programming". In: *Optimization Methods and Software* 24.4–5, pp. 761–779.

Jaulin, L. et al. (2001). *Applied Interval Analysis, with Examples in Parameter and State Estimation, Robust Control and Robotics*. London: Springer-Verlag.

Kaynama, S. et al. (2012). "Computing the Viability Kernel Using Maximal Reachable Sets". In: *Proceedings of the 15th ACM International Conference on Hybrid Systems: Computation and Control*. HSCC '12. New York, NY, USA: ACM, pp. 55–64. ISBN: 978-1-4503-1220-2. DOI: 10.1145/2185632.2185644.

Khalil, H.K. (2002). *Nonlinear Systems, Third Edition*. Prentice Hall. ISBN: 0-13-067389-7.

Konecny, M. et al. (2013). "Enclosing the behavior of a hybrid system up to and beyond a Zeno point". In: *Cyber-Physical Systems, Networks, and Applications (CPSNA)*.

Landau, LD. and EM. Lifshitz (1987). *Fluid Mechanics, Course of Theoretical Physics*. Butterworth-Heinemann.

Le Bars, F. et al. (2012). "State estimation with fleeting data". In: *Automatica* 48.2, pp. 381–387.

Le Mézo, Thomas, Luc Jaulin, and Benoit Zerr (2018a). "An Interval Approach to Compute Invariant Sets". In: *IEEE Transactions on Automatic Control* PP.99, pp. 1–1. ISSN: 0018-9286. DOI: 10.1109/TAC.2017.2685241.

– (2018b). "Bracketing the solutions of an ordinary differential equation with uncertain initial conditions". In: *Applied Mathematics and Computation* 318, pp. 70–79.

Lhommeau, M., L. Jaulin, and L. Hardouin (2011). "Capture Basin Approximation using Interval Analysis". In: *International Journal of Adaptative Control and Signal Processing* 25.3, pp. 264–272.

Mackworth, AK. (1977). "Consistency in Networks of Relations". In: *Artificial Intelligence* 8.1, pp. 99–118.

Mazhoud, I. et al. (2012). "Interval-based global optimization in engineering using model reformulation and constraint propagation". In: *Engineering Applications of Artificial Intelligence* 25.2. Special Section: Local Search Algorithms for Real-World Scheduling and Planning, pp. 404 –417. ISSN: 0952-1976. DOI: http://dx.doi.org/10.1016/j.engappai.2011.10.010.

Mitchell, IM. (2007). "Comparing forward and backward reachability as tools for safety analysis". In: *Hybrid Systems: Computation and Control*. Ed. by A. Bemporad, A. Bicchi, and G. Buttazzo. Springer-Verlag, pp. 428–443.

Mitchell, IM., AM. Bayen, and CJ. Tomlin (2001). "Validating a Hamilton-Jacobi Approximation to Hybrid System Reachable Sets". en. In: *Hybrid Systems: Computation and Control*. Ed. by Maria Domenica Di Benedetto and Alberto Sangiovanni-Vincentelli. Lecture Notes in Computer Science 2034. Springer Berlin Heidelberg, pp. 418–432. ISBN: 978-3-540-41866-5 978-3-540-45351-2. (Visited on 07/24/2015).

Monnet, D., J. Ninin, and L. Jaulin (2016). "Computing an inner and an outer approximation of the viability kernel". In: *Reliable Computing*.

Parrilo, P. and S. Lall (2003). "Semidefinite programming relaxations and algebraic optimization in control". In: *European Journal of Control* 9.2.

Podelski, A. and S. Wagner (2006). "Model checking of hybrid systems: From reachability towards stability". In: *Hybrid Systems: Computation and Control*. Ed. by J. Hespanha and A. Tiwari. Springer-Verlag, pp. 507–521.

Quincampoix, M. (1992). "Differential inclusions and target problems". In: *SIAM journal on control and optimization* 30.2, pp. 324–335.

Ratschan, S. (2002). "Approximate Quantified Constraint Solving by Cylindrical Box Decomposition". In: *Reliable Computing* 8.1, pp. 21–42.

Ratschan, S. and Z. She (2010). "Providing a Basin of Attraction to a Target Region of Polynomial Systems by Computation of Lyapunov-like Functions ". In: *SIAM J. Control and Optimization* 48.7, pp. 4377–4394.

Saint-Pierre, P. (1994). "Approximation of the viability kernel". In: *Applied Mathematics and Optimization* 29.2, pp. 187–209.

– (2002). "Hybrid kernels and capture basins for impulse constrained systems". In: *in Hybrid Systems: Computation and Control*. Ed. by C.J. Tomlin and M.R. Greenstreet. Vol. 2289. Springer-Verlag, pp. 378–392.

Sandretto, JAD. and A. Chapoutot (2016). "Validated Simulation of Differential Algebraic Equations with Runge-Kutta Methods". In: *Reliable Computing* 22.

Seube, N., R. Moitie, and G. Leitmann (2000). "Aircraft Take-Off in Windshear: A Viability Approach". In: *Set-Valued Analysis* 8.1-2, pp. 163–180.

She, Z. and B. Xue (2013). "Computing an invariance kernel with target by computing lyapunov-like functions". In: *IET Control Theory Applications* 7.15, pp. 1932–1940. ISSN: 1751-8644. DOI: 10.1049/iet-cta.2013.0275.

Spivak, M. (1965). *Calculus On Manifolds: A Modern Approach To Classical Theorems Of Advanced Calculus*. Westview Press.

Taha, W. and A. Duracz (2015). "Acumen: An Open-source Testbed for Cyber-Physical Systems Research". In: *CYCLONE'15*.

Tarski, A. (1951). *A Decision Method for Elementary Algebra and Geometry*. Berkeley: Univ. of California Press.

Tornil-Sin, S. et al. (2012). "Robust fault detection of non-linear systems using set-membership state estimation based on constraint satisfaction". In: *Engineering Applications of Artificial Intelligence* 25.1, pp. 1 –10. ISSN: 0952-1976. DOI: http://dx.doi.org/10.1016/j.engappai.2011.07.007.

Wilczak, D. and P. Zgliczynski (2011). “Cr-Lohner algorithm”.
In: *Schedae Informaticae* 20, pp. 9–46.