Efficient Computation of Reachable Sets of Linear Time-Invariant Systems with Inputs[*]

Antoine Girard[1], Colas Le Guernic[2], and Oded Maler[3]

[1] Department of Electrical and Systems Engineering,
University of Pennsylvania, Philadelphia, PA 19104
`agirard@seas.upenn.edu`,
[2] École Normale Supérieure
45 rue d'Ulm, 75005 Paris, France
`Colas.Le.Guernic@ens.fr`,
[3] VERIMAG
2 avenue de Vignate, 38610 Gières, France
`Oded.Maler@imag.fr`

Abstract. This work is concerned with the problem of computing the set of reachable states for linear time-invariant systems with bounded inputs. Our main contribution is a novel algorithm which improves significantly the computational complexity of reachability analysis. Algorithms to compute over and under-approximations of the reachable sets are proposed as well. These algorithms are not subject to the *wrapping effect* and therefore our approximations are *tight*. We show that these approximations are useful in the context of hybrid systems verification and control synthesis. The performance of a prototype implementation of the algorithm confirms its qualities and gives hope for scaling up verification technology for continuous and hybrid systems.

1 Introduction

Computing reachable states for continuous or hybrid systems subject to bounded disturbances has become a major research issue in hybrid systems [ACH+95], [G96], [CK98], [DM98], [CK03], [GM99], [ABDM00], [BT00], [MT00], [KV00], [D00] [ADG03], [G05], [F05]. One may argue that focusing on this question, which is concerned with *transient* behaviors of dynamical systems, can be seen as a major contribution of computer science to enriching the ensemble of standard questions (stability, controllability) traditionally posed in control [ABD+00], [M02]. For hybrid systems in which the continuous dynamics has constant derivatives in every discrete state, such as timed automata or "linear" hybrid automata, the computation of the reachable states in a continuous phase is simply a matter of linear algebra [ACH+95], [AMP95], [HHW97], [F05]. For systems with a non-trivial continuous dynamics, an approximation of the reachable states is

---

generally computed by a combination of numerical integration and geometrical algorithms [GM99], [CK03], [ABDM00], [D00], [BT00], [KV00] [G05].

As an illustration consider a continuous linear time-invariant system of the form $\dot{x}(t) = Ax(t)$. The computation of the set of states reachable from an initial set I within a time interval $[0, T]$ can be handled as follows. We choose an integration step $r = T/(N + 1)$ and compute a sequence of sets $\Omega_0, \ldots, \Omega_N$ such that Ω_i contains all the states reachable from I within $[ir, (i + 1)r]$ time. The first set of the sequence, Ω_0, can be obtained by bloating the convex hull of the sets I and ΦI where $\Phi = e^{rA}$ (see [CK03], [ABDM00], [D00], [G05]). Then, the other elements of the sequence can be computed from the recurrence relation $\Omega_{i+1} = \Phi\Omega_i$. For obvious reasons, the choice of the representation of the sets Ω_i usually consists of classes of sets closed under linear transformations such as polytopes [CK03], [ABDM00], ellipsoids [KV97], [KV00], [BT00] or zonotopes [G05].

When dealing with continuous linear time-invariant systems with bounded inputs of the form $\dot{x}(t) = Ax(t) + Bu(t)$, where the value of $u(t)$ is constrained in some bounded convex set, a similar algorithm is possible. The computation of the influence of the inputs on the reachable sets can be handled according to two main approaches. The first one uses techniques borrowed from optimal control [V98], [ABDM00], [KV00] to compute for each point on the boundary of Ω_i the input u that transforms it in the most "outward" manner. The second approach consists in computing the reachable set using the autonomous dynamics $\dot{x}(t) = Ax(t)$ and then adding (in the sense of the Minkowski sum) a set which accounts for the influence of the inputs [ADG03], [G05]. The recurrence relation between Ω_i and Ω_{i+1} is then of the form $\Omega_{i+1} = \Phi\Omega_i \oplus U$ where U is a bounded convex set. This is the approach considered in this paper.

The major contribution of this paper is a new implementation scheme for the recurrence relation $\Omega_{i+1} = \Phi\Omega_i \oplus U$ which improves significantly (both theoretically and empirically) the computation of the reachable sets of linear time-invariant (LTI) systems with bounded inputs. A version of this algorithm based on *zonotopes* decisively outperforms related algorithms. In addition, algorithms for the computation of over- and under-approximations of the reachable sets are proposed. These algorithms are not subject to the *wrapping effect* (propagation of approximation errors through the computations [K98], [K99]) and therefore our approximations are *tight* in the sense of [KV00]. In the context of hybrid systems, we show that over- and under-approximations can be computed such that they both intersect the guards if and only if the exact reachable set does. We also show that our under-approximations can be used for control synthesis.

2 Reachability Computations for LTI Systems

We consider the problem of computing an over-approximation of the reachable set of a linear time-invariant system over $\mathbb{R}^d$ with bounded inputs within a bounded time interval. As explained in the introduction, this can be done with arbitrary precision by computing the first N elements of a sequence of sets

defined by a recurrence relation of the form:

$$\Omega_{i+1} = \Phi\Omega_i \oplus U, \ i \in \mathbb{N} \tag{1}$$

where Φ is a $d \times d$ matrix, U is a convex bounded subset of $\mathbb{R}^d$ (not necessarily full dimensional) and $\oplus$ denotes the Minkowski sum. The derivation of this recurrence relation from the continuous-time system is not detailed in the present paper but can be found, for instance, in [ADG03], [G05]. Note that since the system is time-invariant, the matrix Φ and the set of inputs U resulting from time discretization are independent of i.

For representations closed under linear transformation and Minkowski sum such as polytopes or zonotopes, the complexity of Ω_i grows due to the Minkowski sum. As a consequence, the computation of the next element of the sequence becomes more expensive as the cost of the linear transformation is proportional to the complexity of the set to which it is applied. For representations with bounded complexity such as oriented rectangular hulls, ellipsoids or zonotopes with bounded order, the Minkowski sum enforces us to make over-approximations at each step. The propagation of these errors through the computations, known as the *wrapping effect* [K98], [K99], can lead to dramatic over-approximations when considering reachability problems for large time horizons.

For linear time-invariant systems we present an algorithm free of any of these problems. Let us remark that from the recurrence relation (1), we have:

$$\Omega_{i+1} = \Phi^{i+1}\Omega_0 \oplus \Phi^i U \oplus \ldots \oplus U, \ i \in \mathbb{N}.$$

Then, let us define the auxiliary sequences of sets:

$$
\begin{aligned}
X_0 &= \Omega_0, & X_{i+1} &= \Phi X_i, \\
V_0 &= U, & V_{i+1} &= \Phi V_i, \\
S_0 &= \{0\}, & S_{i+1} &= S_i \oplus V_i.
\end{aligned}
\tag{2}
$$

Equivalently, we have

$$X_{i+1} = \Phi^{i+1}\Omega_0, \ V_{i+1} = \Phi^{i+1}U \text{ and } S_{i+1} = \Phi^i U \oplus \ldots \oplus U.$$

Therefore, $\Omega_{i+1} = X_{i+1} \oplus S_{i+1}$ where X_{i+1} is the reachable set of the autonomous system from the set of initial states Ω_0, and S_{i+1} is the reachable set of the system with inputs from the initial set $\{0\}$. Note that the decomposition of the linear transformation and the Minkowski sum in the computation of S_{i+1} is possible only because the system is time-invariant. Algorithm 1 implements the reachable set computation based on the recurrence relations (2).

Let us remark that this algorithm does not depend on the class of sets chosen for the representing of the reachable sets. However, this class has to be closed under linear transformation and Minkowski sum (*e.g.* polytopes, zonotopes). The main advantage of this algorithm is that the linear transformations are applied to sets whose complexity *does not increase* at each iteration and this constitutes a significant improvement over existing algorithmic realizations of the recurrence relation (1). Thus, the time complexity of Algorithm 1 is bounded

Algorithm 1 Reachability of linear time-invariant systems.

Input: The matrix Φ, the sets Ω_0 and U, an integer N.
Output: The first N terms of the sequence defined in equation (1).

1: $X_0 \leftarrow \Omega_0$
2: $V_0 \leftarrow U$
3: $S_0 \leftarrow \{0\}$
4: **for** i from 0 to $N-1$ **do**
5:     $X_{i+1} \leftarrow \Phi X_i$ $\triangleright X_{i+1} = \Phi^{i+1} \Omega_0$
6:     $S_{i+1} \leftarrow S_i \oplus V_i$ $\triangleright S_{i+1} = \Phi^i U \oplus \cdots \oplus U$
7:     $V_{i+1} \leftarrow \Phi V_i$ $\triangleright V_{i+1} = \Phi^{i+1} U$
8:     $\Omega_{i+1} \leftarrow X_{i+1} \oplus S_{i+1}$ $\triangleright \Omega_{i+1} = \Phi^{i+1} \Omega_0 \oplus \Phi^i U \oplus \cdots \oplus U$
9: **end for**
10: **return** $\{\Omega_1, \ldots, \Omega_N\}$

by $\mathcal{O}(N\mathcal{L}(n_{in}) + N\mathcal{K}(n_{out}))$, where $\mathcal{L}$ is the complexity of performing a linear transformation, $\mathcal{K}$ is the complexity of performing a Minkowski sum, n_{in} bounds the size of Ω_0 and U, and n_{out} bounds the size of Ω_N. These parameters depend obviously on the class of sets chosen for the representation.

Due to the Minkowski sum, the size of the output may actually be very large. Hence, for an efficient implementation of Algorithm 1, the class of sets used for the representation of the reachable sets has to satisfy one of the following properties. Either the representation size of the Minkowski sum of two sets equals the representation size of the operands, or the computational complexity of the Minkowski sum is independent of the size of the operands.

General polytopes, for example, do not satisfy any of these requirements. As far as we know, there is no reasonable representation satisfying the first property which is closed under Minkowski sum and linear transformations. The second property is satisfied by the class of zonotopes for which the complexity of Minkowski sum does not depend on the description complexity of the sets. In the following section, the implementation of Algorithm 1 using zonotopes is discussed.

3 Reachability using Zonotopes

The class of zonotopes has already been suggested for efficient reachability computations in [K98], [K99], [G05]. Indeed, zonotopes have a compact representation and are closed under linear transformation and Minkowski sum[4]. A zonotope is defined as the Minkowski sum of a finite set of segments. Equivalently it can be seen as the image of a cube by an affine transformation.

---

[4] Actually, the class of zonotopes is the smallest class of sets closed under linear transformation and Minkowski sum and which contains a connected set with a non-empty interior.

Definition 1 (Zonotope). *A zonotope is a subset of $\mathbb{R}^d$ represented by its center $u \in \mathbb{R}^d$ and its generators $v_1, \dots, v_m \in \mathbb{R}^d$:*

$$(u, \langle v_1, \dots, v_m \rangle) = \left\{ u + \sum_{j=1}^{m} \alpha_j v_j \mid \alpha_j \in [-1, \ 1], \ j = 1, \dots, m \right\}.$$

A zonotope with m generators is said to have order $\frac{m}{d}$.

Each zonotope is a centrally-symmetric convex polytope. Parallelepipeds are zonotopes of order one. A planar zonotope with three generators is depicted in Figure 1. Zonotopes admit a very compact representations relative to their number of vertices or faces. A generic zonotope of order p, though it is encoded by only $pd^2 + d$ numbers, has more than $(2p)^{d-1}/\sqrt{d}$ vertices [Z75]. Hence, zonotopes are perfectly suited for the representation of high dimensional sets.
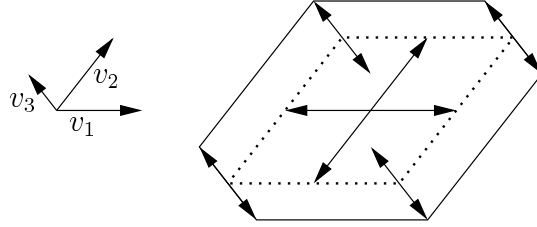

Fig. 1. A planar zonotope with three generators.

The image of a zonotope $Z = (u, \langle v_1, \dots, v_m \rangle)$ under a linear transformation Φ is given by:

$$\Phi Z = (\Phi u, \langle \Phi v_1, \dots, \Phi v_m \rangle).$$

Then, the computational complexity of a linear transformation applied to a zonotope is $\mathcal{O}(p\mathcal{M}(d))$, where $\mathcal{M}(d)$ is the complexity of the multiplication of two $d \times d$ matrices. Using standard matrix multiplication the computational complexity of the linear transformation is[5] $\mathcal{O}(pd^3)$. In comparison, if the zonotope Z was to be represented by its vertices, the linear transformation would require at least $(2p)^{d-1}d^{3/2}$ operations.

The property which really makes zonotopes interesting for the implementation of Algorithm 1 is that their Minkowski sum can be computed in $\mathcal{O}(d)$, independently of the order of the operands. Indeed, the sum of two zonotopes $Z_1 = (u_1, \langle v_1, \dots, v_m \rangle)$ and $Z_2 = (u_2, \langle w_1, \dots, w_n \rangle)$ is

$$Z_1 \oplus Z_2 = (u_1 + u_2, \langle v_1, \dots, v_m, w_1, \dots, w_n \rangle).$$

Hence, the computation of the Minkowski sum consists of summing two vectors and concatenating two lists. Therefore, zonotopes satisfy the requirements

---

[5] Note that, theoretically, the complexity can be further reduced down to $\mathcal{O}(pd^{2.376})$ by using a more sophisticate matrix multiplication algorithm [CW90].

for an efficient implementation of Algorithm 1. Assuming Ω_0 and U are zonotopes of orders p and q, respectively, the time complexity of Algorithm 1 becomes $\mathcal{O}(N(p+q)d^3)$. Moreover, since the Minkowski sum essentially consists of a concatenation of lists, it is not necessary to store the sequence S_i since it can be computed very easily from the sequence V_i. Therefore, the space complexity of a zonotope implementation of Algorithm 1 is $\mathcal{O}(N(p+q)d^2)$.

4 Tight Over-Approximations of the Reachable Sets

The implementation of Algorithm 1 using zonotopes provides for an efficient (in time and in space) computation of the sets $\Omega_1, \ldots, \Omega_N$ defined by (1). Nevertheless, the result of this algorithm, which is typically a set of high-order zonotopes, does not lend itself easily to operations other than linear transformations and Minkowski sum. For example, intersecting a zonotpe with another set, a crucial operation for *hybrid* reachability computation, is very costly as it involves the transformation of the zonotope into a polytopic representation. In this section, we propose an algorithm for computing over-approximations of the sets $\Omega_1, \ldots, \Omega_N$ which are both tight and of low order.

4.1 Interval Hull Approximations

We first consider interval hull over-approximations of the reachable sets. Let Box be a function that maps a set $E \subseteq \mathbb{R}^d$ to its interval hull, that is, to the smallest Cartesian product of intervals containing E. Note that for every $E_1, E_2 \subseteq \mathbb{R}^d$ we have

$$\text{Box}(E_1 \oplus E_2) = \text{Box}(E_1) \oplus \text{Box}(E_2). \tag{3}$$

Algorithm 2 computes the interval hulls of the reachable sets $\Omega_1, \ldots, \Omega_N$.

Algorithm 2 Interval hull approximation of the reachable sets

---

Input: The matrix Φ, the sets Ω_0 and U, and an integer N.
Output: The interval hulls of the N first terms of the sequence defined in (1).
1: $X_0 \leftarrow \Omega_0$
2: $V_0 \leftarrow U$
3: $S_0 \leftarrow \{0\}$
4: **for** i from 0 to $N-1$ **do**
5:     $X_{i+1} \leftarrow \Phi X_i$ $\triangleright X_{i+1} = \Phi^{i+1}\Omega_0$
6:     $S_{i+1} \leftarrow S_i \oplus \text{Box}(V_i)$ $\triangleright S_{i+1} = \text{Box}(\Phi^i U \oplus \cdots \oplus U)$
7:     $V_{i+1} \leftarrow \Phi V_i$ $\triangleright V_{i+1} = \Phi^{i+1}U$
8:     $\overline{\Omega}_{i+1} \leftarrow \text{Box}(X_{i+1}) \oplus S_{i+1}$ $\triangleright \overline{\Omega}_{i+1} = \text{Box}(\Omega_{i+1})$
9: **end for**
10: **return** $\{\overline{\Omega}_1, \ldots, \overline{\Omega}_N\}$

The sequences $X_0, \ldots, X_N$ and $V_0, \ldots, V_N$ are represented as zonotopes which allow to benefit from the low computational complexity of the linear transformations. The sequences $S_0, \ldots, S_N$ and $\overline{\Omega}_1, \ldots, \overline{\Omega}_N$ are represented as interval

products ($2d$ numbers). The computation of the interval hull of a zonotope is particularly easy since the projection of a zonotope on a coordinate axis can be computed by projecting each of its generators on that axis. Then, the time complexity of Algorithm 2 is equivalent to that of Algorithm 1, but its space complexity drops to $\mathcal{O}(Nd + (p + q)d^2)$.

Let us remark that in Algorithm 2, approximations occur only when the function Box is invoked. Note that Box is always applied to *exact* sets and that other operations are computed exactly. Thus, approximation errors do not propagate further through the computations and Algorithm 2 does not suffer from the wrapping effect. Particularly, we have the following result:

Proposition 1. *For all $i \in \{1, \ldots, N\}$, $\overline{\Omega}_i$ is the interval hull of the set Ω_i.*

Proof. From equation (3), we have that

$$\overline{\Omega}_i = \mathrm{Box}(\Phi^i \Omega_0) \oplus \mathrm{Box}(\Phi^{i-1}U) \oplus \ldots \oplus \mathrm{Box}(U)$$
$$= \mathrm{Box}\left(\Phi^i \Omega_0 \oplus \Phi^{i-1}U \oplus \ldots \oplus U\right) = \mathrm{Box}(\Omega_i).$$

■

Thus, each face of $\overline{\Omega}_i$ has at least one common point with the set Ω_i and the over-approximations $\overline{\Omega}_1, \ldots, \overline{\Omega}_N$ computed by Algorithm 2 are *tight* in the sense of [KV00].

Remark 1. Algorithm 2 is not specific to zonotopes and interval products. It can be implemented using any pair of classes of sets, the first of which closed under linear transformation and the second closed under the Minkowski sum and admitting a constant size representation. Then, the function Box has to be replaced by a function that approximates an object from the first class by an object from the second. For accurate over-approximations, this function has to satisfy a property similar to that of equation (3). For instance, we can replace zonotopes by ellipsoids or general polytopes. The choice of the second class is more restricted. In the following, we show that a class of polytopes defined as *intersections of bands* can be used advantageously in the hybrid systems context.

4.2 Guards-oriented Over-Approximations for Hybrid Systems

Let us consider the class of hybrid systems where the continuous dynamics is linear and time-invariant, and where transition guards are specified by hyperplanes:

$$G_e = \left\{x \in \mathbb{R}^d \mid n_e \cdot x = f_e\right\} \text{ where } n_e \in \mathbb{R}^d, \ f_e \in \mathbb{R}.$$

In this section we present a variant of Algorithm 2 whose output can be intersected efficiently with such transition guards. The algorithm computes tight over-approximations of the sets $\Omega_1, \ldots, \Omega_N$ in a class of polytopes defined as intersections of bands.

Definition 2. *Let $\mathcal{S} = \{s_1, \ldots, s_\ell\}$ be a set of vectors. An $\mathcal{S}$-band intersection, represented by two vectors $m, M \in \mathbb{R}^\ell$, is the set defined by:*

$$[m, M]_\mathcal{S} = \left\{ x \in \mathbb{R}^d \mid \ m_i \leq s_i \cdot x \leq M_i, \ i = 1, \ldots, \ell \right\}.$$

Let us remark that interval products constitute a subclass of $\mathcal{S}$-band intersections where $\mathcal{S}$ is the set of coordinate vectors, and that parallelepipeds are obtained when $\mathcal{S}$ is a set of d linearly independent vectors. For a given set of vectors $\mathcal{S}$, it is easy to show that the class of $\mathcal{S}$-band intersections is closed under the Minkowski sum:

$$[m_1, M_1]_\mathcal{S} \oplus [m_2, M_2]_\mathcal{S} = [m_1 + m_2, M_1 + M_2]_\mathcal{S}.$$

To use $\mathcal{S}$-band intersections in Algorithm 2, we need an over-approximation function which maps a zonotope to its smallest enclosing $\mathcal{S}$-band intersection.

Proposition 2. *Let $Z = (u, \langle v_1, \ldots, v_m \rangle)$ be a zonotope, then the $\mathcal{S}$-band intersection $\mathrm{Box}_\mathcal{S}(Z) = [m, M]_\mathcal{S}$ given by*

$$m_i = s_i \cdot u - \sum_{j=1}^m |s_i \cdot v_j|, \ M_i = s_i \cdot u + \sum_{j=1}^m |s_i \cdot v_j|, \ i = 1, \ldots, \ell$$

is an over-approximation of Z. Moreover, each face of $\mathrm{Box}_\mathcal{S}(Z)$ has at least one common point with Z.

Proof. Let $x \in Z$, then for all $i \in \{1, \ldots, \ell\}$,

$$s_i \cdot x = s_i \cdot \left(u + \sum_{j=1}^m \alpha_j v_j \right) = s_i \cdot u + \sum_{j=1}^m \alpha_j s_i \cdot v_j.$$

Since, for all $j \in \{1, \ldots, m\}$, $\alpha_j \in [-1, \ 1]$, $x \in \mathrm{Box}_\mathcal{S}(Z)$. Moreover, let $x_{i,1}$ and $x_{i,2}$ be the elements of Z given by

$$x_{i,1} = u - \sum_{j=1}^m \mathrm{sign}(s_i \cdot v_j) v_j, \ x_{i,2} = u + \sum_{j=1}^m \mathrm{sign}(s_i \cdot v_j) v_j.$$

Then, $s_i \cdot x_{i,1} = m_i$ and $s_i \cdot x_{i,2} = M_i$. ∎

Thus, the function $\mathrm{Box}_\mathcal{S}$ maps a zonotope Z to a tight over-approximation in the class of $\mathcal{S}$-band intersections (see Figure 2). Moreover, it is straightforward to show that $\mathrm{Box}_\mathcal{S}$ satisfies the following property:

$$\mathrm{Box}_\mathcal{S}(Z_1 \oplus Z_2) = \mathrm{Box}_\mathcal{S}(Z_1) \oplus \mathrm{Box}_\mathcal{S}(Z_2).$$

Hence, $\mathcal{S}$-band intersections can replace interval hulls in Algorithm 2. The time complexity of the algorithm becomes $\mathcal{O}(k(p+q)(d^3 + \ell d^2))$ and the space complexity $\mathcal{O}(k\ell + (p+q)d^2 + \ell d)$. Similar to Proposition 1, we can show that the sets computed by Algorithm 2 indeed satisfy $\overline{\Omega}_i = \mathrm{Box}_\mathcal{S}(\Omega_i)$, $i = 1, \ldots, N$.

The following result demonstrates some advantage in using band intersections in Algorithm 2 in the context of hybrid systems verification:
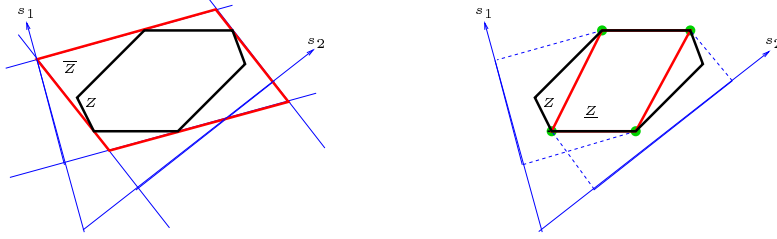
Fig. 2. A zonotope Z, a set of directions $\mathcal{S} = \{s_1, s_2\}$, an over-approximation $\overline{Z} \supset Z$ by an $\mathcal{S}$-band intersection and an under-approximation $\underline{Z} \subset Z$ by the convex-hull of points in Z which are extremal with respect to projections on $\mathcal{S}$.

Proposition 3. *Let $\overline{\Omega}_i$ denote the over-approximation of Ω_i computed by Algorithm 2 using $\mathcal{S}$-band intersections, i.e. $\overline{\Omega}_i = Box_{\mathcal{S}}(\Omega_i)$. If the normal vector n_e to the guard G_e is an element of $\mathcal{S}$ then:*

$$\overline{\Omega}_i \text{ intersects } G_e \Longleftrightarrow \Omega_i \text{ intersects } G_e.$$

Proof. Let $s_j \in \mathcal{S}$, such that $s_j = n_e$. Let us assume that $\overline{\Omega}_i$ intersects G_e. This is equivalent to saying that $m_j \leq f_e \leq M_j$. From Proposition 2, we have that there exist points $x_{j,1}, x_{j,2} \in \Omega_i$ such that $m_j = s_j \cdot x_{j,1}$ and $M_j = s_j \cdot x_{j,2}$. Hence, there exists $x \in \Omega_i$ such that $s_j \cdot x = f_e$ and Ω_i intersects G_e. The other direction of the equivalence is trivial since $\Omega_i \subseteq \overline{\Omega}_i$. ∎

Remark 2. Although Proposition 3 implies that a set computed by a step of Algorithm 2 will intersect a guard exactly when a set computed by Algorithm 1 would (when started from the same set), the corresponding intersections will differ and after the transition, Algorithm 2 will start from a larger set and will generate more behaviors. In other words, the wrapping effect manifests itself during discrete transitions.

5 Tight Under-Approximations and Control Synthesis

When the input U is interpreted as control rather than disturbance, reachability computation can be used to solve controller synthesis problems: find a sequence of input values that drives the system to a desired state while avoiding undesired ones. In this section we show how such control sequences can be extracted from tight *under-approximations* of the reachable sets. Previous work on applying reachability computation to controller synthesis was restricted to synthesizing mode switching conditions for hybrid systems [ABD+00].

5.1 Under-Approximation of the Reachable Sets

Let $\mathcal{S} = \{s_1, \ldots, s_\ell\}$ be a set of vectors. A zonotope $Z = (u, \langle v_1, \ldots, v_m \rangle)$ can be under-approximated by a polytope defined as the convex hull of the finite set

of points corresponding to the extremal points of Z in the directions of $\mathcal{S}$ (see Figure 2). From the proof of Proposition 2, we know that the extremal points of Z in the direction s_i are $x_{i,1} = u - g_i$ and $x_{i,2} = u + g_i$ where:

$$g_i = \sum_{j=1}^{m} \text{sign}(s_i \cdot v_j) v_j.$$

The under-approximation of Z will be denoted by:

$$\underline{Z} = (u, [g_1, \ldots, g_\ell]_{\mathcal{S}}) = \text{ConvexHull}\left(\{u \pm g_i, \ i \in \{1, \ldots, \ell\}\}\right)$$
$$= \left\{u + \sum_{i=1}^{\ell} \alpha_i g_i : \sum_{i=1}^{\ell} |\alpha_i| \leq 1\right\}.$$

Let us remark that the indices have their importance since $u \pm g_i$ are the extremal points in the direction given by s_i. In order to use this under-approximation in a reachability algorithm, we need to express the under-approximation of the Minkowski sum of two zonotopes as a function of the under-approximations of each zonotope.

Lemma 1. *Let us define the following operation*

$$(u, [g_1, \ldots, g_\ell]_{\mathcal{S}}) \boxplus (u', [g_1', \ldots, g_\ell']_{\mathcal{S}}) = (u + u', [g_1 + g_1', \ldots, g_\ell + g_\ell']_{\mathcal{S}}).$$

Then, for two zonotopes Z and Z', we have $\underline{Z \oplus Z'} = \underline{Z} \boxplus \underline{Z'}$.

Proof. The extremal points of $Z \oplus Z'$ in the direction s_i are $u + u' - h_i$ and $u + u' + h_i$ where

$$h_i = \sum_{j=1}^{m} sign(s_i \cdot v_j) v_j + \sum_{j=1}^{m'} sign(s_i \cdot v_j') v_j' = g_i + g_i'.$$

$\blacksquare$

Thus, we can adapt Algorithm 2 to compute under-approximations of the sets $\Omega_1, \ldots, \Omega_N$. This is done by replacing Box by the under-approximation function defined above. Then, from Lemma 1, the output of the algorithm is exactly the sequence $\underline{\Omega}_1, \ldots, \underline{\Omega}_N$. These under-approximations are *tight* since the extremal points of Ω_i in each direction $s_i \in \mathcal{S}$ are vertices of $\underline{\Omega}_i$. Furthermore, it is easy to see that they have the same $\mathcal{S}$-band over-approximation. Then, the following result is straightforward.

Theorem 1. *Let $\overline{\Omega}_i$ denote the $\mathcal{S}$-band over-approximation of Ω_i and let $\underline{\Omega}_i$ denote its under-approximation. If the normal vector n_e to the guard G_e is an element of $\mathcal{S}$ then:*

$$\underline{\Omega}_i \text{ intersects } G_e \Longleftrightarrow \overline{\Omega}_i \text{ intersects } G_e \Longleftrightarrow \Omega_i \text{ intersects } G_e.$$

5.2 Application to Control Synthesis

The under-approximation of Ω_i can be used for control synthesis when U is interpreted as control rather than disturbance. Let y be a point of $\underline{\Omega}_N$ and therefore of Ω_N, we want to determine an initial state $x_0 \in \Omega_0$ and a sequence of inputs $v_0, \ldots, v_{N-1}$ in U, such that the discrete-time system defined by equation (1) reaches y in N steps. Since $y \in \underline{\Omega}_N = (u, [g_1, \ldots, g_\ell]_\mathcal{S})$, it can be written under the form:

$$y = u + \sum_{j=1}^{\ell} \alpha_j g_j, \text{ with } \sum_{j=1}^{\ell} |\alpha_j| \leq 1.$$

If $\ell = d$ and $\underline{\Omega}_k$ is full dimensional, this is equivalent to a change of variable. The under-approximations $\underline{\Phi^N \Omega_0} = (u^N, [g_1^N, \ldots, g_\ell^N]_\mathcal{S})$ and $\underline{\Phi^i U} = (u^i, [g_1^i, \ldots, g_\ell^i]_\mathcal{S})$ ($i \in \{0, \ldots, N-1\}$) are computed by Algorithm 2 while computing $\underline{\Omega}_N$. Let us remark that from Lemma 1, we have $u = u^N + u^{N-1} + \ldots + u^0$ and $g_j = g_j^N + g_j^{N-1} + \ldots + g_j^0$. Then, if Φ is invertible (which is the case if the discrete-time system is achieved by discretization of a continuous-time system), we can choose

$$x_0 = \Phi^{-N}\big(u^N + \sum_{j=1}^{\ell} \alpha_j g_j^N\big),$$

$$v_i = \Phi^{i+1-N}\big(u^{N-1-i} + \sum_{j=1}^{\ell} \alpha_j g_j^{N-1-i}\big), \ i = 0, \ldots, N-1.$$

It is clear that $x_0 \in \Omega_0$, $v_0, \ldots, v_{N-1} \in U$ and, moreover, the sequence $x_{i+1} = \Phi x_i + v_i$ satisfies $x_N = y$.

6 Experimental Results

Algorithms 1 and 2 have been implemented in OCaml [C05]. For the sake of comparison, we have also implemented the zonotope-based reachability algorithm presented in [G05]. This algorithm, which obtained the best accuracy/performance tradeoffs reported so far, computes an over-approximation of the reachable sets using the recurrence relation (1). At each step, in order to avoid computational explosion, it reduces the complexity of the reachable set by over-approximating it by a zonotope of fixed order p. In the following, we refer to this algorithm by Zono-p. Zonotopes and linear algebra operations were implemented in separate modules so that all algorithms use the same subroutines. All computations were performed on a Pentium III 800MHz with 256MB RAM.

6.1 A Five-Dimensional Linear System

As a first benchmark consider the five-dimensional example borrowed from [G05]. Over-approximations of the reachable sets of this system have been computed using Algorithms 1, 2 and Zono-20.

The approximation obtained by Algorithm 1 is always the most accurate because it consists of the exact sequence $\Omega_1, \ldots, \Omega_N$ defined by the recurrence relation (1). For short time horizons, the over-approximations computed by Zono-20 are more accurate than the ones computed by Algorithm 2. However, as we consider longer time horizons, the errors introduced at each step of Zono-20 start propagating through the computations and the wrapping effect becomes too significant to actually say anything interesting about the reachable states of the system. In comparison, the over-approximations obtained by Algorithm 2 are tight and remain accurate even for long time horizons. Moreover, since Algorithm 2 uses interval hull over-approximations, the output of this algorithm is much easier to manipulate than the output of Zono-20 which consists of a sequence of zonotopes of order 20.
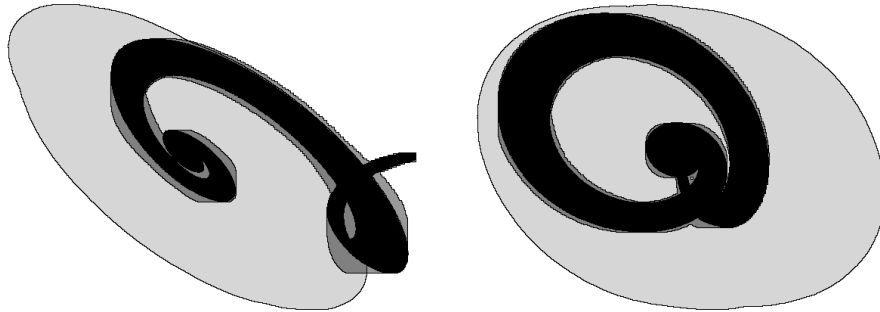

Fig. 3. Reachable states of a five-dimensional linear system after 1000 iterations: projections on coordinates x_1 and x_2 (left), x_4 and x_5 (right). In light gray: set computed by Zono-20 (maximum order allowed for the zonotopes is 20). In dark gray: set computed by Algorithm 2. In black: set computed by Algorithm 1.

Figure 3 shows the over-approximations of the reachable sets obtained by the three algorithms for a long time horizon ($N = 1000$). It is clear that Algorithms 1 and 2 have a much better precision than Algorithm Zono-20, an obvious victim of the wrapping effect. Computation time and memory consumption of the three algorithms for different time horizons N are reported in Table 1. We can see that Algorithms 1 and 2 are fast and require much less memory. Algorithm 2, which computes interval-hulls approximation, is about 100 times faster, and needs 25 times less memory than Algorithm Zono-20, while producing approximations of higher quality.

6.2 High-Dimensional Linear Systems

The three algorithms were also tested on continuous linear time-invariant systems which were randomly generated according to the following procedure: the matrix A was chosen at random and then normalized for the infinity norm and

$N =$	200	400	600	800	1000
Algorithm 1	0.01s	0.02s	0.04s	0.05s	0.07s
Algorithm 2	0.s	0.s	0.01s	0.01s	0.02s
Zono-20	0.34s	0.74s	1.14s	1.46s	2.16s

$N =$	200	400	600	800	1000
Algorithm 1	492KB	737KB	983KB	1.23MB	1.47MB
Algorithm 2	246KB	246KB	246KB	246KB	246KB
Zono-20	1.47MB	2.95MB	4.18MB	5.65MB	6.88MB

Table 1. Time and memory consumptions of reachability computations for a five-dimensional linear system, for different time horizons.

the inputs were chosen bounded for the infinity norm. In [G05], it is explained how the recurrence relation given by equation (1) can be obtained. The discretization time step was $r = 0.01$ and the number of iterations is $N = 100$. Computation times and memory consumptions of the three algorithms for linear systems of several dimensions d are reported in Table 2.

$d =$	5	10	20	50	100	150	200
Algorithm 1	0.0s	0.02s	0.11s	1.11s	8.43s	35.9s	136s
Algorithm 2	0.0s	0.01s	0.07s	0.91s	8.08s	28.8s	131s
Zono-20	0.16s	0.61s	3.32s	22.6s	152s		

$d =$	5	10	20	50	100	150	200
Algorithm 1	246KB	492KB	1.72MB	8.85MB	33.7MB	75.2MB	133MB
Algorithm 2	246KB	246KB	246KB	492KB	983KB	2.21MB	3.69MB
Zono-20	737KB	2.46MB	8.36MB	44.5MB	177MB		

Table 2. Time and memory consumption for $N = 100$ for several linear time-invariant systems of different dimensions

Algorithms 1 and 2 appear to be extremely scalable in terms of both time and space, which confirms the theoretical complexity estimations. Let us remark that using Algorithm 2, we can compute a tight over-approximation of the reachable set of a 100-dimensional system after 100 time steps in less than 10 seconds using less than 1MB memory. To the best of our knowledge, there is no report in the literature of algorithms with similar performances for such high-dimensional systems.

6.3 Varying the Time Step

When the recurrence relation (1) is obtained by discretization of a continuous-time system, we expect that the smaller is the time step, the more accurate will be the over-approximation we compute. However, is not always the case for algorithms suffering from the wrapping effects because reducing the time

steps increases the number of iterations of the reachability algorithm in order to cover the same time interval. As we can see in Figure 4, reducing the time step improves the quality of the over-approximations obtained by Algorithm 2 whereas the over-approximations obtained by Algorithm Zono-5 blows up beyond usefulness.
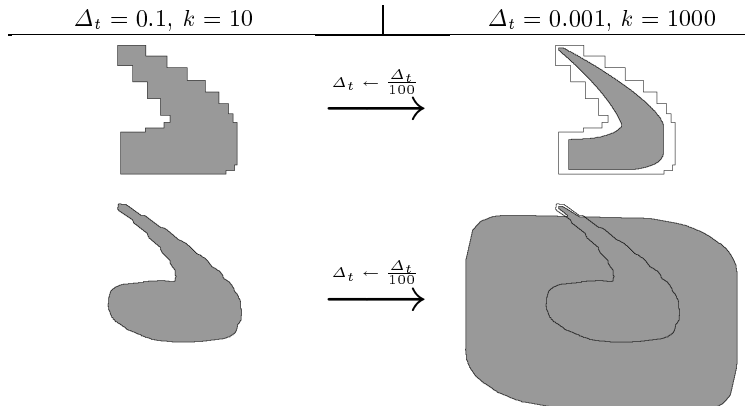

Fig. 4. Reducing the timestep in a 20-dimensional example improves the quality of the over-approximation obtained by Algorithm 2 (top) but increase the *wrapping effect* on the algorithm Zono-5 (bottom).

7 Conclusions

We have presented an extremely-efficient and exact algorithm for computing reachable states of a discrete-time LTI systems, as well as several variants of this algorithm for computing tight over- and under-approximation of these sets, which do not suffer from the wrapping effect. We showed that we can compute over-approximations that facilitate guard intersections, and that our under-approximations can be used to solve control synthesis problems. The prototype implementation of our algorithms has outperfromed previously reported algorithms in terms of execution time, memory consumption and approximation tightness. The implementation will be improved by using efficient linear algebra libraries. Future work will focus on the application to hybrid systems and, in particular, on computing intersections with guards.

References

[ACH⁺95] 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* **138**, 3–34, 1995.

[ABDM00] A. Asarin, O. Bournez, T. Dang, and O. Maler, Approximate Reachability Analysis of Piecewise-linear Dynamical Systems, *HSCC'00*, LNCS 1790, 20–31, Springer, 2000.

[ABD⁺00] E. Asarin, O. Bournez, T. Dang, O. Maler and A. Pnueli, Effective Synthe-
sis of Switching Controllers for Linear Systems, *Proceedings of the IEEE*
88, 1011-1025, 2000.

[ADG03] E. Asarin, T. Dang, and A. Girard, Reachability Analysis of Nonlinear
Systems using Conservative Approximation, *HSCC'03*, LNCS 2623, 20–35,
Springer, 2003.

[AMP95] E. Asarin, O. Maler and A. Pnueli, Reachability Analysis of Dynami-
cal Systems having Piecewise-Constant Derivatives, *Theoretical Computer
Science* **138**, 35–65, 1995.

[BT00] O. Botchkarev and S. Tripakis, Verification of Hybrid Systems with Linear
Differential Inclusions using Ellipsoidal Approximations, *HSCC'00*, LNCS
1790, 73–88, Springer, 2000.

[C05] *The Caml Language webpage*, 2005. `http://caml.inria.fr/` .

[CK98] A. Chutinan and B.H. Krogh, Computing Polyhedral Approximations to
Dynamic Flow Pipes, *CDC'98*, IEEE, 1998.

[CK03] A. Chutinan and B.H. Krogh, Computational Techniques for Hybrid Sys-
tem Verification, *IEEE Trans. on Automatic Control* **48**, 64–75, 2003

[CW90] D. Coppersmith and S. Winograd, Matrix Multiplication via Arithmetic
Progressions, *J. Symbolic Computation* **9**, 251–280, 1990.

[D00] T. Dang, *Verification and Synthesis of Hybrid Systems*, PhD thesis, In-
stitut National Polytechnique de Grenoble, Laboratoire Verimag, 2000.

[DM98] T. Dang and O. Maler, Reachability Analysis via Face Lifting, *HSCC'98*,
LNCS 1386, 96–109, Springer, 1998

[F05] PHAVer: Algorithmic Verification of Hybrid Systems Past HyTech,
HSCC'05, LNCS 3414, 258–273, Springer, 2005.

[G05] A. Girard, Reachability of Uncertain Linear Systems using Zonotopes,
HSCC'05, LNCS 3414, 291–305, Springer, 2005.

[G96] M.R. Greenstreet, Verifying Safety Properties of Differential Equations,
CAV'96, LNCS 1102, 277–287, Springer, 1996.

[GM99] M.R. Greenstreet, and I. Mitchell, Reachability Analysis using Polygonal
Projections, *HSCC'99*, LNCS 1569, 103–116, Springer, 1999.

[HHW97] T.A Henzinger, P.-H. Ho, and H. Wong-Toi, Hytech: A Model Checker
for Hybrid Systems, *Software Tools for Technology Transfer* **1**, 110–122,
1997.

[K98] W. Kühn, Rigorously Computed Orbits of Dynamical Systems without
the Wrapping Rffect. *Computing* **61**, 47–68, 1998.

[K99] W. Kühn, Towards an Optimal Control of the Wrapping Effect, *SCAN
98, Developments in Reliable Computing*, 43–51, Kluwer, 1999.

[KV97] A. Kurzhanski and I. Valyi, *Ellipsoidal Calculus for Estimation and Con-
trol*. Birkhauser, 1997.

[KV00] A.B Kurzhanski and P. Varaiya, Ellipsoidal Techniques for Reachability
Analysis, *HSCC'00*, LNCS 1790, 202–214, Springer, 2000.

[M02] O. Maler, Control from Computer Science, *Annual Reviews in Control* **26**,
175-187, 2002.

[MT00] I. Mitchell and C. Tomlin, Level Set Methods for Computation in Hybrid
Systems, *HSCC'00*, LNCS 1790, 310–323, Springer, 2000.

[V98] P. Varaiya, Reach Set computation using Optimal Control, *KIT Work-
shop, Verimag, Grenoble*, 377–383, 1998.

[Z75] T. Zaslavsky, Facing Up to Arrangements: Face-Count Formulas for Par-
titions of Space by Hyperplanes, *Memoirs of the AMS* **154**, American
Mathematical Society, 1975.