Controlled Invariance of Discrete Time Systems[⋆]

René Vidal, Shawn Schaffert, John Lygeros, and Shankar Sastry

Department of Electrical Engineering and Computer Sciences
University of California at Berkeley
Berkeley, CA 94720-1774
Phone: (510) 643 2382, Fax: (510) 642 1341
rvidal,sms,lygeros,sastry@eecs.berkeley.edu

Abstract. An algorithm for computing the maximal controlled invariant set and the least restrictive controller for discrete time systems is proposed. We show how the algorithm can be encoded using quantifier elimination, which leads to a semi-decidability result for definable systems. For discrete time linear systems with all sets specified by linear inequalities, a more efficient implementation is proposed using linear programming and Fourier elimination. If in addition the system is in controllable canonical form, the input is scalar and unbounded, the disturbance is scalar and bounded and the initial set is a rectangle, then the problem is decidable.

1 Introduction

The design of controllers is one of the most active research topics in the area of hybrid systems. Problems that have been addressed include hierarchical control [5, 19], distributed control [18], and optimal control using dynamic programming techniques [3, 4, 20, 23] or extensions of the maximum principle [11]. A substantial research effort has also been directed towards solving control problems with reachability specifications, that is designing controllers that guarantee that the state of the system will remain in a "good" part of the state space. Such control problems turn out to be very important in applications, and are closely related to the computation of the *reachable states* of a hybrid system and to the concept of *controlled invariance*. The proposed solutions extend game theory methods for purely discrete [21, 25] and purely continuous [2, 15] systems to certain classes of hybrid systems: timed automata [13, 17], rectangular hybrid automata [28] and more general hybrid automata [16, 26].

All of these techniques are concerned with hybrid systems whose continuous state evolves in continuous time, according to differential equations or differential inclusions. Unlike conventional continuous dynamical systems, little attention has been devoted to systems where the continuous state evolves in discrete time, according to difference equations. Besides being interesting in its own right, this class of hybrid systems can be used to approximate hybrid systems with

⋆ Research supported by ONR under grant N00014-97-1-0946, by DARPA under contract F33615-98-C-3614, and by ARO under grant MURI DAAH04-96-1-0341.

differential equations. Indeed, most of the techniques that have been proposed for reachability computations for general continuous dynamics involve some form of discretization of the continuous space [8, 12, 26], followed by a reachability computation on the resulting discrete time system.

In Sect. 2, we formulate the problem of controller synthesis for discrete time systems under reachability specifications, introduce the concepts of maximal controlled invariant set and least restrictive controller, propose an algorithm for computing them, and show how the algorithm can be implemented using quantifier elimination. This immediately leads to a semi-decidability result for discrete time systems whose continuous dynamics can be encoded in a decidable theory of the reals. In Sect. 3, we implement the proposed algorithm for discrete time linear systems with all the sets defined by linear inequalities. The implementation is based on a more efficient method for performing quantifier elimination in the theory of linear constraints using linear programming and Fourier elimination. We also show that the problem is decidable when the single-input single-disturbance discrete time linear system is in controllable canonical form, the input is unbounded, and the safe set is a rectangle. Finally, in Sect. 4, we illustrate the proposed method with some examples. For the proofs we refer the reader to [27].

2 Discrete Time Systems and Safety Specifications

2.1 Basic Definitions

Let Y be a countable collection of variables and let $\mathbf{Y}$ denote its set of valuations, that is the set of all possible assignments of these variables. We refer to variables whose set of valuations is countable as *discrete* and to variables whose set of valuations is a subset of a Euclidean space $\mathbb{R}^n$ as *continuous*. For a set $\mathbf{Y}$ we use $\mathbf{Y}^c$ to denote the complement of $\mathbf{Y}$, $2^{\mathbf{Y}}$ to denote the set of all subsets of $\mathbf{Y}$, $\mathbf{Y}^*$ to denote the set of all finite sequences of elements of $\mathbf{Y}$, and $\mathbf{Y}^\omega$ to denote the set of all infinite sequences. Since the dynamical systems we will consider will be time invariant we will use $y = \{y[i]\}_{i=0}^N$ to denote sequences. We use $\wedge$ to denote conjunction, $\vee$ to denote disjunction, $\neg$ to denote negation, $\forall$ to denote the universal quantifier, and $\exists$ to denote the existential quantifier.

Definition 1 (Discrete Time System (DTS)). *A discrete time system is a collection* $H = (X, V, \mathrm{Init}, f)$ *consisting of a finite collection of state variables,* X*, a finite collection of input variables,* V*, a set of initial states,* $\mathrm{Init} \subseteq \mathbf{X}$*, and a reset relation,* $f : \mathbf{X} \times \mathbf{V} \to 2^{\mathbf{X}}$*.*

Definition 2 (Execution of DTS). *A sequence* $\chi = (x, v) \in (\mathbf{X} \times \mathbf{V})^* \cup (\mathbf{X} \times \mathbf{V})^\omega$ *is said to be an execution of the discrete time system* H *if* $x[0] \in Init$*, and for all* $k \geq 0$*,* $x[k+1] \in f(x[k], v[k])$*.*

To ensure that every finite execution can be extended to an infinite execution we assume that $f(x, v) \neq \emptyset$ for all $(x, v) \in \mathbf{X} \times \mathbf{V}$. We call such a DTS *non-blocking.*[1]

[1] The condition is only sufficient. Although it can be refined to be necessary as well, we will not pursue this direction since the emphasis of this paper is controller synthesis.

We denote the set of all executions of H starting at $x_0 \in \mathbf{X}$ as $\mathcal{E}_H(x_0)$, and the set of all executions of H by $\mathcal{E}_H$. Clearly, $\mathcal{E}_H = \bigcup_{x_0 \in Init} \mathcal{E}_H(x_0)$.

Our goal here is to design controllers for DTS. We assume that the input variables are partitioned into two classes, $V = U \cup D$, where U are *control variables*, and D are *disturbance variables*. In this context a controller can be defined as a feedback map.

Definition 3 (Controller). *A controller, C, is a map $C : \mathbf{X}^* \to 2^{\mathbf{U}}$. A controller is called* non-blocking *if $C(x) \neq \emptyset$ for all $x \in \mathbf{X}^*$. A controller is called* memoryless *if for all $x, x' \in \mathbf{X}^*$ ending at the same state we have $C(x) = C(x')$.*

The interpretation is that, given the evolution of the plant state up to now, the controller determines the set of allowable controls for the next transition. With this interpretation in mind, we define the set of *closed loop causal executions* as

$$\mathcal{E}_{H_C} = \{(x, u, d) \in \mathcal{E}_H \mid \forall k \geq 0, u[k] \in C(x{\downarrow_k})\},$$

where $x{\downarrow_k}$ denotes the subsequence of x consisting of its first k elements. Notice that a memoryless controller can be characterized by a map $g : \mathbf{X} \to 2^{\mathbf{U}}$, and its set of closed loop causal executions is simply

$$\mathcal{E}_{H_g} = \{(x, u, d) \in \mathcal{E}_H \mid \forall k \geq 0, u[k] \in g(x[k])\}.$$

Our goal is to use controllers to steer the executions of the plant, so that they satisfy certain desirable properties. In this paper we will restrict our attention to a class of properties known as *safety properties*: Given a set $F \subseteq \mathbf{X}$, we would like to find a non-blocking controller that ensures that the state stays in F for ever. We will say that a controller C *solves the problem* $(H, \Box F)$, if and only if C is non-blocking and for all $(x, u, d) \in \mathcal{E}_{H_C}$, $x[k] \in F$ for all $k \geq 0$. If such a controller exists we say that the problem $(H, \Box F)$ *can be solved.*

Even though safety properties are not the only properties of interest[2], they turn out to be very useful in applications. Many important problems, such as absence of collisions in transportation systems, mutual exclusion in distributed algorithms, etc., can be naturally encoded as safety properties. Fortunately, it can be shown that for this class of properties one can, without loss of generality, restrict attention to memoryless controllers.

Proposition 1. *The problem $(H, \Box F)$ can be solved if and only if it can be solved by a memoryless controller.*

Motivated by Proposition 1, we restrict our attention to memoryless controllers from now on.

2.2 Controlled Invariant Sets and Least Restrictive Controllers

The concept of controlled invariance turns out to be fundamental for the design of controllers for safety specifications [16]. Roughly speaking, a set of states,

[2] Other important properties are liveness properties (ensuring that the state eventually reaches a certain set, visits a set infinitely often, etc.), stability, optimality, etc.

W, is called controlled invariant if there exists a controller that ensures that all executions starting somewhere in W remain in W for ever. More formally:

Definition 4 (Controlled invariant set). *A set $W \subseteq \mathbf{X}$ is called a controlled invariant set of H if there exists a non-blocking controller that solves the problem $(H', \Box W)$, where $H' = (X, V, W, f)$ (the same as H, but with $Init' = W$).*

We say that the controller that solves the problem $(H', \Box W)$ *renders the set W invariant.* Also, given a set $F \subseteq \mathbf{X}$, a set $W \subseteq F$ is called a *maximal controlled invariant subset of F*, if it is controlled invariant and it is not a proper subset of any other controlled invariant subset of F. The following lemma establishes the uniqueness of the maximal controlled invariant set.

Lemma 1. *The problem $(H, \Box F)$ can be solved if and only if there exists a unique maximal controlled invariant set, $\hat{W}$, with* $\text{Init} \subseteq \hat{W} \subseteq F$.

A useful and intuitive characterization of the concept of controlled invariance can be given in terms of the operator $\text{Pre} : 2^{\mathbf{X}} \to 2^{\mathbf{X}}$ defined by

$$\text{Pre}(W) = \{x \in W \mid \exists u \in \mathbf{U}\ \forall d \in \mathbf{D},\ f(x,u,d) \cap W^c = \emptyset\}.$$

The following properties of the operator Pre are easy to establish and will be useful in the subsequent discussion.

Proposition 2. *The operator* Pre *has the following properties:*

1. Pre *is contracting, that is for all $W \subseteq \mathbf{X}$,* $\text{Pre}(W) \subseteq W$;
2. Pre *is monotone, that is for all $W, W' \subseteq \mathbf{X}$ with $W \subseteq W'$,* $\text{Pre}(W) \subseteq \text{Pre}(W')$; *and,*
3. *A set $W \subseteq \mathbf{X}$ is controlled invariant if and only if it is a fixed point of* Pre, *that is if and only if* $\text{Pre}(W) = W$.

Many memoryless controllers may be able to solve a particular problem. Controllers that impose less restrictions on the inputs they allow are in a sense better than controllers that impose more restrictions. For example, controllers that impose fewer restrictions allow more freedom if additional safety specifications are imposed, or if one is asked to optimize the performance of the (safe) closed loop system with respect to other objectives. To quantify this intuitive notion we introduce a partial order on the space of memoryless controllers. We write $g_1 \preceq g_2$ if for all $x \in \mathbf{X}$, $g_1(x) \subseteq g_2(x)$.

Definition 5 (Least restrictive controller). *A memoryless controller $g : \mathbf{X} \to 2^{\mathbf{U}}$ that solves the problem (H, F) is called least restrictive if it is maximal among the controllers that solve $(H, \Box F)$ in the partial order defined by $\preceq$.*

Lemma 2. *A controller that renders a set W invariant exists if and only if a unique least restrictive controller that renders W invariant exists.*

Notice that the least restrictive controller that renders a set W invariant must, by definition, allow $\hat{g}(x) = \mathbf{U}$ for all $x \notin W$. Summarizing Lemmas 1 and 2 we have the following:

Theorem 1. *The problem $(H, \Box F)$ can be solved if and only if there exists:*

1. *a unique maximal controlled invariant set $\hat{W}$ with $\text{Init} \subseteq \hat{W} \subseteq F$, and*
2. *a unique least restrictive controller, $\hat{g}$, that renders $\hat{W}$ invariant.*

Motivated by Theorem 1 we state the controlled invariance problem more formally.

Problem 1 (Controlled Invariance Problem (CIP)) *Given a DTS and a set $F \subseteq \mathbf{X}$ compute the maximal controlled invariant subset of F, $\hat{W}$, the least restrictive controller, $\hat{g}$, that renders $\hat{W}$ invariant, and test whether $\text{Init} \subseteq \hat{W}$.*

2.3 Computation of $\hat{W}$ and $\hat{g}$

We first present a conceptual algorithm for solving the CIP for general DTS. Even though there is no straightforward way of implementing this algorithm in the general case, in subsequent sections we show how this can be done for special classes of DTS.

Algorithm 1 (Controlled Invariance Algorithm)

```
initialization: $W^0 = F$, $W^{-1} = \mathbf{X}$, $l = 0$
while $W^{l-1} \cap (W^l)^c \neq \emptyset$ do
    $W^{l+1} = \text{Pre}(W^l)$
    $l = l + 1$
end while
set $\hat{W} = \bigcap_{l \geq 0} W^l$
set $\hat{g}(x) = \begin{cases} \left\{u \in \mathbf{U} \mid \forall d \in \mathbf{D},\ f(x,u,d) \cap (\hat{W})^c = \emptyset\right\} & x \in \hat{W} \\ \mathbf{U} & x \notin \hat{W} \end{cases}$
```

Theorem 2. *$\hat{W}$ is the maximal controlled invariant subset of F and $\hat{g}$ is the least restrictive controller that renders $\hat{W}$ invariant.*

To implement the controlled invariance algorithm one needs to be able to (1) encode sets of states, perform intersection and complementation, and test for emptiness, (2) compute the Pre of a set, and (3) guarantee that a fixed point is reached after a finite number of iterations. For classes of DTS for which 1 and 2 are satisfied we say that the CIP is *semi-decidable*; if all three conditions are satisfied we say that the CIP is *decidable*. As an example, consider *finite state machines* (FSM), that is the class of DTS for which $\mathbf{X}$, $\mathbf{U}$ and $\mathbf{D}$ are finite. In this case, one can encode sets of states, perform intersection, complementation, test for emptiness and compute Pre by enumeration (or other more efficient representations). Moreover, by the monotonicity of W^l and the fact that $\mathbf{X}$ is finite, the algorithm is guaranteed to terminate in a finite number of steps. Therefore, the CIP is decidable for finite state machines.

In subsequent sections we show how the computation can be performed for DTS with state and input taking values on a Euclidean space and transition relations given by certain classes of functions of the state and input.

2.4 CIP for Definable Discrete Time Systems

In this section we consider the case where all the sets involved in the CIP can be expressed by means of a logic formula that belongs to the language of a certain logic theory. For example, we denote by Lin($\mathbb{R}$) the theory of linear constraints and by OF($\mathbb{R}$) the theory of polynomial constraints.

For some theories, it is possible to determine the sentences that belong to the theory. The Tarski-Seidenberg decision procedure provides a way of doing this for OF($\mathbb{R}$). It can be shown that OF($\mathbb{R}$) is decidable [22, 24], in other words, there exists a computational procedure that after a finite number of steps determines whether an $\mathcal{R}$-sentence belongs to OF($\mathbb{R}$) or not. The decision procedure is based on quantifier elimination, an algorithm that converts a formula $\phi(x_1, \ldots, x_n)$ to an equivalent quantifier free formula. Notice that this provides a method for testing emptiness. A set $Y = \{(x_1, \ldots, x_n) \mid \phi(x_1, \ldots, x_n)\}$ is empty if and only if the sentence $\exists x_1 \ldots \exists x_n \mid \phi(x_1, \ldots, x_n)$ is equivalent to false.

To relate this to the problem at hand, we restrict our attention to CIP which are "definable" in an appropriate theory.

Definition 6 (Definable CIP). *A CIP, $(H, \Box F)$, is definable in a theory if $\mathbf{X} = \mathbb{R}^n$, $\mathbf{U} \subseteq \mathbb{R}^{n_u}$, $\mathbf{D} \subseteq \mathbb{R}^{n_d}$ and the sets $\mathbf{U}$, $\mathbf{D}$,* Init*, $f(x, u, d)$ for all $x \in \mathbf{X}$, $u \in \mathbf{U}$ and $d \in \mathbf{D}$, and F are definable in the theory.*

If $(H, \Box F)$ and W^l are definable in OF($\mathbb{R}$), then

$$\psi^l(x) \equiv \exists u \, \forall d \, \forall x' \mid [x \in W^l] \wedge [u \in \mathbf{U}] \wedge [(d \notin \mathbf{D}) \vee (x' \notin f(x, u, d)) \vee (x' \in W^l)] \quad (1)$$

is a first order formula in the corresponding language. Therefore, each step of the controlled invariance algorithm involves eliminating the quantifiers in (1) to obtain a quantifier free formula defining W^{l+1}. The fact that OF($\mathbb{R}$) is decidable immediately leads to the following:

Theorem 3. *The class of CIP definable in* OF($\mathbb{R}$) *is semi-decidable.*

Moreover, if $(H, \Box F)$ is definable in OF($\mathbb{R}$) and W is a controlled invariant set also definable in OF($\mathbb{R}$), then the set $\{(x, u) \mid \forall d \in \mathbf{D} \;\; \forall x' \in f(x, u, d), \;\; x' \in W\}$ describing the least restrictive controller that renders W invariant is also definable in OF($\mathbb{R}$). Furthermore, quantifier elimination can be performed in this formula, to obtain an explicit expression for the least restrictive controller. Finally, the question $W \cap \text{Init}^c = \emptyset$ can be decided. Therefore, if the algorithm happens to terminate in a finite number of steps, the CIP can be completely solved.

Although different methods have been proposed for performing quantifier elimination in OF($\mathbb{R}$) [1, 22, 24], and the process can be automated using symbolic tools [9], the quantifier elimination procedure is in general hard, both in theory and in practice, since the solvability may be doubly exponential [14]. For the theory Lin($\mathbb{R}$), a somewhat more efficient implementation can be derived using techniques from linear algebra and linear programming. The next section shows how quantifier elimination in the theory Lin($\mathbb{R}$) can be performed more efficiently for the formula (1) used in the controlled invariance algorithm.

3 CIP for Discrete Time Linear Systems

A *linear CIP* (LCIP) consists of

- a Linear DTS (LDTS), i.e. a DTS with $\mathbf{X} = \mathbb{R}^n$, $\mathbf{U} = \{u \in \mathbb{R}^{n_u} \mid Eu \leq \eta\} \subseteq \mathbb{R}^{n_u}$, $\mathbf{D} = \{d \in \mathbb{R}^{n_d} \mid Gd \leq \gamma\} \subseteq \mathbb{R}^{n_d}$, Init $= \{x \in \mathbf{X} \mid Jx \leq \theta\}$ and a reset relation given by $f(x,u,d) = \{Ax+Bu+Cd\}$, where $A \in \mathbb{Q}^{n\times n}$, $B \in \mathbb{Q}^{n \times n_u}$, $C \in \mathbb{Q}^{n\times n_d}$, $E \in \mathbb{Q}^{m_u \times n_u}$, $G \in \mathbb{Q}^{m_d \times n_d}$, $\eta \in \mathbb{Q}^{m_u}$, $\gamma \in \mathbb{Q}^{m_d}$, $J \in \mathbb{Q}^{n \times m_i}$ and $\theta \in \mathbb{Q}^{m_i}$ with m_u, m_d and m_i being the number of constraints on the control, disturbance and initial conditions, respectively; and,
- a set $F = \{x \in \mathbb{R}^n \mid Mx \leq \beta\}$ where $M \in \mathbb{Q}^{m\times n}$, $\beta \in \mathbb{Q}^m$ and m is the number of constraints on the state.

Notice that LDTS are non-blocking and deterministic, in the sense that for every state x and every input (u,d) there exists a unique next state. Since the sets F, $\mathbf{U}$ and $\mathbf{D}$ are all convex polygons, and the dynamics f are given by a linear map, the LCIP is definable in the theory Lin($\mathbb{R}$), and therefore, according to the discussion in Sect. 2.4, it is semi-decidable. We assume that the sets F and $\mathbf{U}$ can be either bounded or unbounded, but $\mathbf{D}$ is bounded[3].

For the LCIP it turns out that, after the l-th iteration, the set W^l can be described by m^l linear constraints as $\{x \in \mathbb{R}^n \mid M^l x \leq \beta^l\}$, that is, W^l remains a convex polygon. Obviously, $m^0 = m$, $M^0 = M$ and $\beta^0 = \beta$. Letting $\hat{A}^l = M^l A$, $\hat{B}^l = M^l B$ and $\hat{C}^l = M^l C$, (1) becomes

$$\psi^l(x) \equiv [M^l x \leq \beta^l] \wedge [\exists u \,|\, (Eu \leq \eta) \wedge (\forall d \,|\, (Gd > \gamma) \vee (\hat{A}^l x + \hat{B}^l u + \hat{C}^l d \leq \beta^l))].$$

Thus, in each step of the algorithm, we need to be able to eliminate variables u and d from the inner formulae, intersect the new constraints with the old ones and check if the new set is empty. Notice that not all of the new constraints generated by quantifier elimination may be necessary to define the set W^{l+1}. Also, some of the old constraints may become redundant after adding the new ones. Hence we need to check the redundancy of the constraints when doing the intersection.

3.1 Quantifier Elimination

We first perform quantifier elimination on d over the formula

$$\phi^l(x,u) \equiv \forall d \;\mid (Gd > \gamma) \vee (\hat{A}^l x + \hat{B}^l u + \hat{C}^l d \leq \beta^l)\,.$$

Let $\hat{a}_i^T$, $\hat{b}_i^T$ and $\hat{c}_i^T$ be the i-th row of $\hat{A}^l$, $\hat{B}^l$ and $\hat{C}^l$, respectively. Then, parsing ϕ^l leads to

$$\phi^l(x,u) \;\equiv\; \forall d \;\mid\; \bigwedge_{i=1}^{m^l} (Gd > \gamma) \vee (\hat{c}_i^T d \leq \beta_i^l - \hat{a}_i^T x - \hat{b}_i^T u).$$

Consider $\delta : \mathbb{R}^{m^l \times n_d} \to \mathbb{R}^{m^l}$ defined by $\delta_i(\hat{C}^l) = \max\limits_{d:Gd\leq\gamma} (\hat{c}_i^T d)$ for $i = 1,\ldots,m^l$.

[3] The theoretical discussion can be extended to unbounded $\mathbf{D}$ sets, but the computational implementation is somewhat more involved.

Proposition 3. *$\phi^l(x,u)$ is equivalent to $\varphi^l(x,u) \equiv \hat{A}^l x + \hat{B}^l u \le \beta^l - \delta(\hat{C}^l)$.*

Therefore, the elimination of the $\forall$ quantifier can be done by solving a finite collection of linear programming problems. Since we have assumed that $\mathbf{D}$ is bounded, such an optimization problem is guaranteed to have a solution, and hence $\delta(\cdot)$ is well defined. Since $\delta(\cdot)$ is applied to each row of $\hat{C}^l$, in the sequel we will use $\delta_i(\hat{C}^l)$ and $\delta(\hat{c}_i^T)$ interchangeably. Notice that, strictly speaking, $\delta(\cdot)$ is not part Lin($\mathbb{R}$), but we use it as a shorthand for the constant obtained by solving the linear programs.

Next, we perform quantifier elimination on u over the formula

$$\phi^l(x) \equiv \exists u \mid (Eu \le \eta) \wedge (\hat{A}^l x + \hat{B}^l u \le \beta^l - \delta(\hat{C}^l)). \tag{2}$$

We will discuss two methods to eliminate u. The first is known as *Fourier Elimination* [10], and the second, attributed to Cernikov [6], is an application of Farkas Lemma on duality [7].

For the first method, assume we want to eliminate u_1 first. Let e_i be the i-th unit vector in $\mathbb{R}^{m^l+m_u}$,

$$H^l = \begin{pmatrix} \hat{B}^l \\ E \end{pmatrix} \quad \text{and} \quad \xi^l(x) = \begin{pmatrix} \beta^l - \delta(\hat{C}^l) - \hat{A}^l x \\ \eta \end{pmatrix}.$$

Thus $\phi^l(x)$ is equivalent to $\exists u \mid H^l u \le \xi^l(x)$. Also define $P^l = \{p \mid H^l_{p1} > 0\}$, $Q^l = \{q \mid H^l_{q1} < 0\}$ and $R^l = \{r \mid H^l_{r1} = 0\}$, where H^l_{ij} refers to the i,j element of the matrix H^l. Then $\phi^l(x)$ is equivalent to

$$\exists u \mid \bigwedge_{p\in P^l} \bigwedge_{q \in Q^l} \left[\frac{1}{H^l_{q1}}(\xi^l_q(x) - \sum_{j=2}^{m} H^l_{qj} u_j) \le u_1 \le \frac{1}{H^l_{p1}}(\xi^l_p(x) - \sum_{j=2}^{m} H^l_{pj} u_j) \right] \wedge \bigwedge_{r \in R^l} \left[0 \le (\xi^l_r(x) - \sum_{j=2}^{m} H^l_{rj} u_j) \right].$$

Hence, after the elimination of u_1 we obtain

$$\exists u \mid \bigwedge_{p\in P^l} \bigwedge_{q\in Q^l \cup R^l} (H^l_{p1} \ -H^l_{q1}) \begin{pmatrix} \hat{e}_q^T \\ \hat{e}_p^T \end{pmatrix} \begin{pmatrix} \hat{A}^l x \\ 0 \end{pmatrix} \le (H^l_{p1} \ -H^l_{q1}) \begin{pmatrix} \hat{e}_q^T \\ \hat{e}_p^T \end{pmatrix} \begin{pmatrix} \beta^l - \delta(\hat{C}^l) \\ \eta \end{pmatrix} - (H^l_{p1} \ -H^l_{q1}) \begin{pmatrix} \sum_{j=2}^m H^l_{qj} u_j \\ \sum_{j=2}^m H^l_{pj} u_j \end{pmatrix}. \tag{3}$$

Therefore, the elimination of the $\exists$ quantifier is performed by taking nonnegative linear combinations of all pairs of constraints so as to cancel the quantified variable. Note that if all the coefficients of the quantified variable are positive (negative), then ϕ^l is true, and we need not to eliminate the remaining variables. Otherwise, after u_1 has been eliminated, we apply the same procedure to the constraints in (3), so as to eliminate $u_2, \dots, u_{n_u}$. Since the procedure is based on nonnegative row operations, it is clear that

$$\phi^l(x) \equiv \Lambda^l \begin{pmatrix} \hat{A}^l x \\ 0 \end{pmatrix} \le \Lambda^l \begin{pmatrix} \beta^l - \delta(\hat{C}^l) \\ \eta \end{pmatrix} \equiv (\tilde{M}^l x \le \tilde{\beta}^l) \wedge (0 \le \Lambda_2^l \eta) , \qquad (4)$$

where $\Lambda^l = [\Lambda_1^l \ \Lambda_2^l] \in \mathbb{Q}^{\tilde{m}^l \times (m^l + m_u)}$ is a matrix with nonnegative entries such that $\Lambda^l H^l = 0$, $\tilde{m}^l$ is the number of new constraints obtained through quantifier elimination, $\tilde{M}^l = \Lambda_1^l \hat{A}^l \in \mathbb{Q}^{\tilde{m}^l \times n}$ and $\tilde{\beta}^l = \Lambda_1^l(\beta^l - \delta(\hat{C}^l)) \in \mathbb{Q}^{\tilde{m}^l}$. Notice that if the condition $\Lambda_2^l \eta \ge 0$ is violated, then $\hat{W} = \emptyset$. Otherwise, we just need to add the new constraints $\tilde{M}^l x \le \tilde{\beta}^l$ to the original set W^l.

Although *Fourier Elimination* is attractive because of its simplicity, it is quite inefficient. In general, it generates many new constraints in the intermediate steps, and in the worst case the method is exponential. This difficulty can be partially remedied since many of the inequalities are likely to be redundant [7].

An alternative method [6] computes the rows of Λ^l directly as the extreme points of the set $\{\lambda^l \in \mathbb{R}^{m+m_u} \mid {\lambda^l}^T H^l = 0 \wedge \lambda^l \ge 0 \wedge \sum \lambda_i^l = 1\}$, where the last constraint is added to ensure that the set is a polytope. Although the extreme points method is better than Fourier elimination, because it eliminates the costly intermediate steps, the computation of the extreme points is still costly and also generates a lot of redundant constraints. A more efficient method [14] uses a generalized linear programming formulation and an on-line convex hull construction to obtain an incremental inner approximation of the set defined by ϕ^l. The method considerably reduces the number of constraints defining the resulting set.

3.2 Intersection, Emptiness and Redundancy

Provided that $\Lambda_2^l \eta \ge 0$, the quantifier elimination procedure presented above computes the set of states $\tilde{W}^l \equiv \{x \mid \tilde{M}^l x \le \tilde{\beta}^l\}$ that can be forced by u to transition into W^l. To obtain W^{l+1}, such a set must be intersected with W^l. Since both sets are convex, the intersection can be carried out by simply appending $\tilde{M}^l$ and $\tilde{\beta}^l$ to M^l and β^l, respectively. However, this method of performing the intersection is likely to lead to a description of the set which is larger than necessary since many of the constraints may be redundant. Algorithm 2 is aimed at checking the emptiness of the intersection and then eliminate redundant constraints. In the algorithm, [] denotes an empty matrix, $\mathbf{1} = (1 \dots 1)^T \in \mathbb{Q}^{\tilde{m}^l + m^l}$, and ${m_i'}^T$ and β_i' are the i-th rows of $M_0' = \begin{pmatrix} \tilde{M}^l \\ M^l \end{pmatrix}$ and $\beta_0' = \begin{pmatrix} \tilde{\beta}^l \\ \beta^l \end{pmatrix}$, respectively. Initially, $M' = M_0'$ and $\beta' = \beta_0'$.

The idea behind the algorithm is that $W^l \cap \tilde{W}^l \neq \emptyset$ if and only if $\exists x | M'x \le \beta'$, which is equivalent to saying that $\min\{t \mid M'x \le \beta' + \mathbf{1}t\} \le 0$. Afterwards, if the problem $\max\{{m_i'}^T x \mid M'x \le \beta'\}$ is feasible, and the constraint ${m_i'}^T x \le \beta_i'$ is not redundant, then the optimal value of the problem is β_i'. Moreover, if the non-redundant constraint ${m_i'}^T x \le \beta_i'$ is removed from the optimization problem, then the new optimal value m^* satisfies $m^* > \beta_i'$.

Algorithm 2 (Emptiness and Redundancy Algorithm)

```
**initialization** $M' = M'_0$, $\beta' = \beta'_0$, $M^{l+1} = []$, $\beta^{l+1} = []$.
$m^* = \min\{t \mid M'x \leq \beta' + \mathbf{1}t\}$
**if** $m^* > 0$ **or** $\Lambda_2^l \eta \not\geq 0$ **then**
    $\hat{W} = \emptyset$, **terminate** controlled invariance algorithm
**else**
    **for** $i = 1$ **to** $\tilde{m}^l + m^l$ **do**
        remove ${m'_i}^T$ from $M'$ and $\beta'_i$ from $\beta'$
        $m^* = \max\{{m'_i}^T x \mid M'x \leq \beta'\}$
        **if** $m^* > \beta'_i$ **then**
            add ${m'_i}^T$ to $M^{l+1}$ and $M'$,
            add $\beta'_i$ to $\beta^{l+1}$ and $\beta'$
        **end if**
    **end for**
**end if**
**if** $M^{l+1} = M^l$ and $\beta^{l+1} = \beta^l$ **then**
    $\hat{W} = W^l$, **terminate** controlled invariance algorithm
**end if**
```

The controlled invariance algorithm terminates if the redundancy algorithm concludes that either $\Lambda_2^l \eta \not\geq 0$ or $W^l \cap \tilde{W}^l = \emptyset$ (in which case $\hat{W} = \emptyset$), or if all the new constraints are redundant (in which case $W^l = W^{l+1} = \hat{W}$)[4]. Otherwise, upon termination of the redundancy algorithm, the process is repeated for W^{l+1}. An obvious optimization of the code involves terminating both algorithms if after all new constraints in $\tilde{M}^l x \leq \tilde{\beta}^l$ have been tested, M^{l+1} and β^{l+1} are still empty. Notice that for all l the set W^l is a convex polygon as claimed. Summarizing:

Theorem 4. *The LCIP is semi-decidable.*

In the next section we study situations where the algorithm is guaranteed to terminate in a finite number of steps. In Sect. 4, we will provide and example which actually converges after an infinite number of iterations.

3.3 Decidable Special Cases

We first summarize some of the observations made so far about situations where the algorithm terminates in a finite number of steps.

Proposition 4. *For an LCIP with* $\mathbf{U} = \mathbb{R}^{n_u}$*, if either one of the columns of* MB *is componentwise positive (negative), or if* $rank(MB) = \min\{m, n\}$*, the algorithm terminates in a finite number of steps.*

[4] Note that any redundant constraint in the original description of F will be eliminated the first time the redundancy algorithm is invoked by the controlled invariance algorithm.

Next, we limit our attention to the case $F = [\alpha_1, \beta_1] \times \ldots \times [\alpha_n, \beta_n] \subset \mathbb{R}^n$ with $\alpha_i \leq \beta_i$ and $[\alpha_i, \beta_i] \subset \mathbb{R}, i = 1 \ldots n$, $u \in \mathbb{R}$, and $d \in [d_1, d_2] \subset \mathbb{R}$. To remind ourselves of the fact that u and d are scalar, we use b and c instead of B and C. We also assume that (A, b) is in controllable canonical form, that is

$$x[k+1] = \begin{pmatrix} 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ \vdots & & & \ddots & & \vdots \\ 0 & & & & & 1 \\ a_{n1} & a_{n2} & & \cdots & & a_{nn} \end{pmatrix} x[k] + \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix} u[k] + \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_{n-1} \\ c_n \end{pmatrix} d[k]. \quad (5)$$

In this case $\psi^1(x)$ is equivalent to

$$\exists u \mid \bigwedge_{j=1}^{n} (\alpha_j \leq x_j \leq \beta_j) \wedge \bigwedge_{j=2}^{n} (\alpha_{j-1} - \delta(-c_{j-1}) \leq x_j \leq \beta_{j-1} - \delta(c_{j-1})) \wedge$$
$$\left(\alpha_n - \sum_{j=1}^{n} a_{nj} x_j - \delta(-c_n) \leq u \leq \beta_n - \sum_{j=1}^{n} a_{nj} x_j - \delta(c_n) \right). \quad (6)$$

From the last expression, it is clear that given $x_1 \in [\alpha_1, \beta_1]$, x_j exists if and only if $\alpha_j^1 = \max(\alpha_j, \alpha_{j-1} - \delta(-c_{j-1})) \leq \min(\beta_j, \beta_{j-1} - \delta(c_{j-1})) = \beta_j^1$, $j = 2 \ldots n$, and u exists if and only if $\alpha_n - \delta(-c_n) \leq \beta_n - \delta(c_n)$. It is straightforward to see that in the l-th iteration $(0 \leq l \leq n)$ W^l is defined by:

$$W^l = [\alpha_1^0, \beta_1^0] \times \ldots \times [\alpha_{l+1}^l, \beta_{l+1}^l] \times [\alpha_{l+2}^l, \beta_{l+2}^l] \times \ldots \times [\alpha_n^l, \beta_n^l],$$

where $\alpha_j^l = \max(\alpha_j^{l-1}, \alpha_{j-1}^{l-1} - \delta(c_{j-1}))$, and $\beta_j^l = \min(\beta_j^{l-1}, \beta_{j-1}^{l-1} - \delta(c_{j-1}))$, for $2 \leq l+1 \leq j \leq n$, with $\alpha_j^0 = \alpha_j$ and $\beta_j^0 = \beta_j$, for $1 \leq j \leq n$.

This means that after n iterations, the maximal controlled invariant set remains unchanged, and the least restrictive controller is given by the last constraint in (6), but with α_n and β_n replaced by α_n^{n-1} and β_n^{n-1}, respectively. This result can be summarized as follows:

Lemma 3. *Given system (5) with $F = [\alpha_1, \beta_1] \times \ldots \times [\alpha_n, \beta_n] \subset \mathbb{R}^n$, $\mathbf{U} = \mathbb{R}$ and $\mathbf{D} = [d_1, d_2] \subset \mathbb{R}$, the solution to the CIP, obtained after at most n iterations of the algorithm, is given by:*

$$\hat{W} = \begin{cases} \left\{ x \mid \bigwedge_{j=1}^{n} \alpha_j^{j-1} \leq x_j \leq \beta_j^{j-1} \right\} & \text{if } \bigwedge_{j=2}^{n} \left(\alpha_j^{j-1} \leq \beta_j^{j-1} \right) \wedge \left(|c_n| \leq \frac{\beta_n^{n-1} - \alpha_n^{n-1}}{d_2 - d_1} \right) \\ \emptyset & \text{otherwise} \end{cases}$$

$$\hat{g}(x) = \begin{cases} \left\{ u \mid \alpha_n^{n-1} - \delta(-c_n) \leq u + \sum_{j=1}^{n} a_{nj} x_j \leq \beta_n^{n-1} - \delta(c_n) \right\} & \text{if } x \in \hat{W} \\ \mathbf{U} & \text{otherwise} \end{cases}$$

Theorem 5. *For systems of the form (5) with $F = [\alpha_1, \beta_1] \times \ldots \times [\alpha_n, \beta_n] \subset \mathbb{R}^n$, $\mathbf{U} = \mathbb{R}$ and $\mathbf{D} = [d_1, d_2] \subset \mathbb{R}$, the LCIP is decidable.*

The conditions of Theorem 5 for decidability are somewhat demanding. If, for example, u is bounded, that is, $\mathbf{U} = [u_1, u_2] \subset \mathbb{R}$, then the new constraints added to x during each iteration may change the bounds on x to a non-rectangular polyhedron. In this case, the CIP is no longer decidable, and the system falls into the more general class of systems described at the beginning of the section. We conjecture that the LCIP is decidable in a much more general setting, using a completely different algorithm that exploits the stabilizability of the pairs (A, B) and (A, C) and the observability of the pair (A, M).

4 Experimental Results

The algorithm proposed in Sect. 3 was implemented in MATLAB. Here, we present two examples that were solved using this implementation. The first example is also worked out analytically to complete the semi-decidability result.

Example 1. The LDTS is defined by $\mathbf{U} = \mathbb{R}$, $\mathbf{D} = [-1, 1]$,

$$A = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}, B = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, C = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, M = \begin{pmatrix} 1 & 1 \\ -1 & -3 \\ 1 & -1 \\ -3 & 1 \end{pmatrix}, \text{ and } \beta = \begin{pmatrix} 100 \\ -50 \\ 100 \\ -50 \end{pmatrix}.$$

It is straightforward to see that the only new constraint added in the l-th iteration is $[0\ m_l]x \le \beta_l$, where $m_l = -10 \cdot 3^{l-1}$, and $\beta_l = -210 - 265(3^{l-1} - 1)$. Therefore after an infinite number of iterations, $\hat{W}$ and $\hat{g}(x)$ converge to

$$\hat{W} = \left\{ x \mid \begin{pmatrix} M \\ 0 \ -2 \end{pmatrix} x \le \begin{pmatrix} \beta \\ -53 \end{pmatrix} \right\}$$

$$\hat{g}(x) = \begin{cases} \{u \in \mathbf{U} \mid u \ge \max(18 - x_1 - \frac{4x_2}{3}, -100 - x_1, -\frac{55}{2} - x_1 - x_2) \\ \qquad u \le \min(98 - x_1 - 2x_2, -52 - x_1 + 2x_2)\} & \text{if } x \in \hat{W} \\ \mathbf{U} & \text{else} \end{cases}$$

Example 2. The LDTS is defined by

$$A = \begin{pmatrix} -1 & -8 & -1 \\ 1 & -4 & -1 \\ -5 & -3 & -1 \end{pmatrix}, B = \begin{pmatrix} 2 & 1 \\ 4 & 1 \\ 1 & -1 \end{pmatrix}, C = \begin{pmatrix} 3 & 2 & 1 \\ 2 & 1 & 7 \\ 1 & 2 & 1 \end{pmatrix}, E = \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ 0 & 1 \\ 0 & -1 \end{pmatrix}, \eta = \begin{pmatrix} 1000 \\ 1000 \\ 1000 \\ 1000 \end{pmatrix},$$

$$M = \begin{pmatrix} 3 & 1 & 0 \\ -1 & 3 & 0 \\ 1 & -1 & 0 \\ -1 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & -1 \end{pmatrix}, \beta = \begin{pmatrix} 100 \\ 100 \\ 100 \\ 100 \\ 100 \\ 100 \end{pmatrix}, G = \begin{pmatrix} 1 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & -1 \end{pmatrix} \text{ and } \gamma = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}.$$

Using MATLAB, this example converges in two iterations. Information about the intermediate calculations of each iteration is shown in Table 1.

Table 1. Results of Example 2

Iteration	1	2
Number of LP problems for quantifier elimination on d	6	10
Number of constraints on (x, u) before elimination of u	10	14
Number of new constraints on x after elimination of u	281	614
Number of new non-redundant constraints on x	4	0
Total number of constraints on x after iteration	10	10

5 Conclusions and Future Work

We showed that the problem of computing the maximal controlled invariant set and the least restrictive controller for discrete time systems is well posed and proposed a general algorithm for carrying out the computation. We then specialized the algorithm to discrete time linear systems with convex polygonal constraints, and showed how it can be implemented using linear programming and Fourier elimination. The decidability of the problem was also analyzed, and some simple, but interesting cases were found to be decidable.

We are currently working on sufficient conditions under which the problem is decidable. So far, it seems that the decidability property is not only dependent on the system itself, but also on the initial set, as shown by Example 1. Another topic of further research, is the application of these algorithm to discrete time hybrid systems, where some states and inputs take values in finite sets, while others in subsets of a Euclidean space. It is easy to show how this class of systems is a special case of the more general class of DTS. Therefore, all the conclusions of Sect. 2 directly extend to them. Unfortunately the implementation of the controlled invariance algorithm is more complicated, even in the case where the continuous state evolves according to a linear difference equation.

References

[1] D.S. Arnon, G.E. Collins, and S. McCallum. Cylindrical algebraic decomposition I: the basic algorithm. *SIAM Journal on Computing*, 13(4):865–877, 1984.

[2] T. Başar and G. J. Olsder. *Dynamic Non-cooperative Game Theory.* Academic Press, 2nd edition, 1995.

[3] A. Bensoussan and J.L. Menaldi. Hybrid control and dynamic programming. *Dynamics of Continuous, Discrete and Impulsive Systems*, (3):395–442, 1997.

[4] M.S. Branicky, V.S. Borkar, and S.K. Mitter. A unified framework for hybrid control: Model and optimal control theory. *IEEE Transactions on Automatic Control*, 43(1):31–45, 1998.

[5] P.E. Caines and Y.J. Wei. Hierarchical hybrid control systems: A lattice theoretic formulation. *IEEE Transactions on Automatic Control*, 43(4):501–508, April 1998.

[6] R.N. Cernikov. The solution to linear programming problems by elimination of unknowns. *Soviet Mathematics Doklady*, 2:1099–1103, 1961.

[7] V. Chandru. Variable elimination in linear constraints. *The Computer Journal*, 36(5):463–472, 1993.

[8] T. Dang and O. Maler. Reachability analysis via face lifting. In *Hybrid Systems: Computation and Control*, vol. 1386 of *LNCS*, pp. 96–109. Springer Verlag, 1998.

[9] A. Dolzmann and T. Sturm. REDLOG: Computer algebra meets computer logic. *ACM SIGSAM Bulletin*, 31(2):2–9, 1997.

[10] L.B.J. Fourier. Analyse des travaux de l'Academie Royale des Sciences, pendant l'annee 1824, Partie matematique. Histoire de l'Academie Royale des Sciences de l'Institut de France 7, 1827.

[11] G. Grammel. Maximum principle for a hybrid system via singular perturbations. *SIAM Journal of Control and Optimization*, 37(4):1162–1175, 1999.

[12] M.R. Greenstreet and I. Mitchell. Integrating projections. In *Hybrid Systems: Computation and Control*, vol. 1386 of *LNCS*, pp. 159–174. Springer Verlag, 1998.

[13] M. Heymann, F. Lin, and G. Meyer. Control synthesis for a class of hybrid systems subject to configuration-based safety constraints. In *Hybrid and Real Time Systems*, vol. 1201 of *LNCS*, pp. 376–391. Springer Verlag, 1997.

[14] C. Lassez and J.-L. Lassez. Quantifier elimination for conjunctions of linear constraints via a convex hull algorithm. In *Symbolic and Numeric Computation for Artificial Intelligence*, pages 103–122. Academic Press, 1992.

[15] J. Lewin. *Differential Games.* Springer-Verlag, 1994.

[16] J. Lygeros, C. Tomlin, and S. Sastry. Controllers for reachability specifications for hybrid systems. *Automatica*, pages 349–370, March 1999.

[17] O. Maler, A. Pnueli, and J. Sifakis. On the synthesis of discrete controllers for timed systems. In *Theoretical Aspects of Computer Science*, vol. 900 of *LNCS*, pp. 229–242. Springer Verlag, 1995.

[18] A. Nerode and W. Kohn. Multiple agent hybrid control architecture. In *Hybrid Systems*, vol. 736 of *LNCS*, pp. 297–316. Springer Verlag, New York, 1993.

[19] G. Pappas, G. Lafferriere, and S. Sastry. Hierarchically consistent control systems. In *IEEE Conference on Decision and Control*, pages 4336–4341, December 1998.

[20] B. Piccoli. Necessary conditions for hybrid optimization. In *IEEE Conference on Decision and Control*, pages 410–415, December 7-10 1999.

[21] P. J. G. Ramadge and W. M. Wonham. The control of discrete event systems. *Proceedings of the IEEE*, Vol.77(1):81–98, 1989.

[22] A. Seidenberg. A new decision method for elementary algebra. *Annals of Mathematics*, 60:387–374, 1954.

[23] H.J. Sussmann. A maximum principle for hybrid optimal control problems. In *IEEE Conference on Decision and Control*, pages 425–430, December 7-10 1999.

[24] A. Tarski. *A decision method for elementary algebra and geometry.* University of California Press, 1951.

[25] W. Thomas. On the synthesis of strategies in infinite games. In Ernst W. Mayr and Claude Puech, editors, *Proceedings of STACS 95, vol. 900 of LNCS*, pp. 1–13. Springer Verlag, Munich, 1995.

[26] C. Tomlin, J. Lygeros, and S. Sastry. Computing controllers for nonlinear hybrid systems. In *Hybrid Systems: Computation and Control*, vol. 1569 of *LNCS*, pp. 238–255. Springer Verlag, 1999.

[27] R. Vidal, S. Schaffert, J. Lygeros, and S. Sastry. Controlled invariance of discrete time systems. Technical Report UCB/ERL M99/65, Electronics Research Laboratory, University of California, Berkeley, 1999.

[28] H. Wong-Toi. The synthesis of controllers for linear hybrid automata. In *IEEE Conference on Decision and Control*, pages 4607–4613, December 10-12 1997.