Observer-based control of block-triangular discrete time hybrid automata on a partial order[‡]

Domitilla Del Vecchio[*, †]

Systems Laboratory, University of Michigan, Ann Arbor, Michigan, U.S.A.

SUMMARY

The safety control problem for the class of discrete time block-triangular order preserving hybrid automata with imperfect continuous state information is addressed. A dynamic feedback law is constructed in order to guarantee that the continuous state is always outside a bad set. The order preserving properties of the dynamics are exploited to construct state estimation and control algorithms that have linear complexity in the number of variables. Such algorithms adopt an interval abstraction approach, in which sets of interest are represented and propagated only through suitable upper and lower bounds. The proposed algorithms are applied to a collision avoidance problem arising in the context of intelligent transportation. Copyright © 2008 John Wiley & Sons, Ltd.

1. INTRODUCTION AND BACKGROUND

The problem addressed in this paper is the dynamic control (state estimator plus control) of the parallel composition of a class of discrete time hybrid automata (triangular order preserving hybrid automata) under safety specifications. Motivating applications both for the model and for the problem considered include multi-agent hybrid systems, in which each agent evolves along a path or route in a network of routes and conflicts arise at routes crossings. Examples include intelligent transportation systems and railway network control systems. In these systems, each agent (a vehicle) can be modeled as a hybrid automaton, in which the continuous state dynamics has triangular structure and models the physical motion of the agent along its path. The discrete

---

[*]Correspondence to: Domitilla Del Vecchio, Electrical Engineering and Computer Science, University of Michigan, 1301 Beal Avenue, Ann Arbor, MI 48109, U.S.A.
[†]E-mail: ddv@umich.edu
[‡]Parts of this work appeared in the Proceedings of the Conference on Decision and Control, 2007.

state can model the control mode, in which the agent can be (braking, accelerating, run-out, etc.) or it can model input and state constraints. The entire system is given as the parallel composition of the component systems modeling the agents. In particular, one problem for which automated solutions are sought is the collision prediction and avoidance at traffic intersections and at railway mergings [1, 2]. In these systems, the state (speed and position, for example) is known to the controller only within some uncertainty bounds. This uncertainty is due to measurement errors or to missing measurements as it occurs, for example, with the position measurement obtained by the global positioning system or by road-side sensor systems.

The control problem under safety specifications assuming that the perfect state information has been addressed by several researchers (see [3–5], for example). General control design problems under language specification (safety, for example) [3, 6–9] have been extensively studied for discrete systems in the computer science literature (see [10] for an overview). A control perspective in the context of discrete event systems was given by Ramadge and Wonham [11]. The approach has been extended to specific classes of hybrid systems such as timed automata [12, 13] and rectangular automata [14]. For these classes of hybrid systems, implementation results using tools such as [15] showed that in practice the synthesis procedure is limited to control problems with a small number of control modes. Most of the work on safety controller design for general classes of hybrid systems has been concerned with the computation of reachable sets (see, for example, [3, 8, 9] and the references therein). In these works, the safety control problem has been addressed by computing the set of states that lead to an unsafe configuration, a bad set of states, independently of the input choice. This set has been called the *capture set* in the differential games literature as independently of any input sequence, the state of the system will be captured by the bad set at some time. Then, a feedback is computed that guarantees that any state that is outside such a set is kept outside it [3–5, 16]. We mathematically make precise the notion of the capture set by representing a system through the transition systems formalism [17]. In such a formalism, a system is a tuple $\Sigma = (S, \mathcal{I}, \tau)$, in which S is a set of states, $\mathcal{I}$ is a set of inputs, and $\tau : S \times \mathcal{I} \to S$ is a transition map. We denote a state by $s \in S$ and an input by $u \in \mathcal{I}$. An execution of Σ is an infinite sequence $\{s^k\}_{k \in \mathbb{N}}$ such that $s^{k+1} = \tau(s^k, u^k)$ for $u^k \in \mathcal{I}$. Let $B \subseteq S$ be the set of bad states. For all $n \geq 0$, we define $\tau^n(s, \{u^k\}_{0 \leq k \leq n})$ by the following relations: $\tau^0(s, u^0) := s$ for all $u^0 \in \mathcal{I}$, $\tau^n(s, \{u^k\}_{0 \leq k \leq n}) := \tau(\tau^{n-1}(s, \{u^k\}_{0 \leq k \leq n-1}), u^n)$. The escape set is mathematically characterized as

$$C = \{s \in S | \forall \{u^k\}_{k \in \mathbb{N}} \exists N \text{ such that } \tau^N(s, \{u^k\}_{0 \leq k \leq N}) \in B\}$$

The safety control problem is thus the problem of designing a feedback law $u = g(s)$ such that for all executions $\{s^k\}_{k \in \mathbb{N}}$ starting with $s \notin C$, we have that $s^k \notin C$ for all k. A bottleneck in solving this problem is complexity. For classes of hybrid automata, for which the continuous dynamics reachable set can be computed, computational constraints usually limit the system to four or five continuous variables and to two or three discrete states. Furthermore, the proposed algorithms are not guaranteed to terminate [3, 4]. To reduce the computational load, approximate algorithms have been proposed to compute an over-approximation of the capture set [8, 15, 18].

The safety control problem with imperfect or partial state information has been scarcely addressed in the literature. Some results in this direction can be found in [19, 20], for example. In [19], a controller that relies on a state estimator is proposed for finite state systems. The results are then extended to control a class of rectangular hybrid automata with imperfect state information. The approach employed by De Wulf *et al.* [19] is similar to the one adopted for the full state information as it also relies on the computation of the capture set. To make this more precise, we

illustrate the basic idea employing again the transition system formalism. Consider now a transition system with output $\Sigma = (S, \mathscr{I}, \mathscr{Y}, \tau, \gamma)$, in which S is a set of states, $\mathscr{I}$ is a set of inputs, $\mathscr{Y}$ is a set of outputs, $\tau : S \times \mathscr{I} \to S$ is a transition map, and $\gamma : \mathscr{Y} \to 2^S$ is the output map, in which 2^S denotes the set of all subsets of S. If a state $s \in \gamma(y)$, we say that s is compatible with measurement y. An output sequence is denoted by $\{y^k\}_{k \in \mathbb{N}}$, in which y^k is such that $s^k \in \gamma(y^k)$. Since the state is not measured, one determines the set of all current system states. We thus define the operator $\hat{\tau} : 2^S \times \mathscr{I} \times \mathscr{Y} \to 2^S$ as follows. Let $\hat{s} \subseteq S$, then $\hat{\tau}(\hat{s}, u, y) := \tau(\hat{s}, u) \cap \gamma(y)$. Given an output sequence of Σ, $\{y^k\}_{k \in \mathbb{N}}$, corresponding to the execution $\{s^k\}_{k \in \mathbb{N}}$, the set of all states at step k that are compatible with such an output sequence up to step k and with the system transition map is given as

$$\hat{s}^{k+1} = \hat{\tau}(\hat{s}^k, u^k, y^{k+1})$$
$$\hat{s}^0 = \gamma(y^0) \tag{1}$$

One can verify that $s^k \in \hat{s}^k$ for all k. We refer to Equations (1) as a *state estimator* for the transition system Σ. In the differential game literature, the set $\hat{s}$ is also referred to as information state [21]. A *state estimator-based control strategy* is one in which the control law depends on the state estimate $\hat{s}$, that is, $u = g(\hat{s})$. We define the notation $\hat{\tau}^{n+1}(\hat{s}, \{u^k\}_{0 \leqslant k < n+1}, \{y^k\}_{0 \leqslant k \leqslant n+1}) := \hat{\tau}(\hat{\tau}^n(\hat{s}, \{u^k\}_{0 \leqslant k < n}, \{y^k\}_{0 \leqslant k \leqslant n}), u^n, y^{n+1})$, $\hat{\tau}^0(\hat{s}, \{u^k\}_{0 \leqslant k < 0}, \{y^k\}_{0 \leqslant k \leqslant 0}) := \gamma(y^0)$, to denote the set of states, to which an initial set of states $\hat{s}$ is mapped after $n+1$ steps with input sequence $\{u^k\}_{0 \leqslant k < n+1}$ and output sequence $\{y^k\}_{0 \leqslant k \leqslant n+1}$. The capture set $C \subseteq 2^S$ is the set of all subsets of S such that if the state estimator is initialized with one of such subsets of S, then there will be an output sequence for which the state estimate at a later time will intersect the bad set no matter what input sequence is applied to the system. Mathematically, this is denoted as

$$C = \{X \in 2^S | \forall \{u^k\}_{k \in \mathbb{N}}, \exists N \text{ and } \{y^k\}_{0 \leqslant k \leqslant N}$$

$$\text{such that } \hat{\tau}^N(X, \{u^k\}_{0 \leqslant k < N}, \{y^k\}_{0 \leqslant k \leqslant N}) \cap B \neq \emptyset\} \tag{2}$$

To maintain safety, a controller must thus guarantee that the state estimator (1) will never have one of the sets in C as a state. For a general system with only discrete states, the set C can be computed in a finite number of steps, but the complexity of the algorithm that computes it is exponential [19]. For a system with also continuous states, the computation of C is impractical because the sets are infinite.

In this paper, we exploit order preserving properties and a triangular structure of the system dynamics to show

(a) that a tight over-approximation of the capture set for the perfect state information case can be symbolically computed by an algorithm with linear complexity in the number of state variables;

(b) that with imperfect information one does not need to re-compute C, but the same symbolic computation obtained for the perfect information case can be employed by evaluating suitable lower and upper bounds on the lower and upper bounds of the state estimate.

The basic idea is to propagate and represent sets by means of their upper and lower bounds in the chosen partial order. If the dynamics of the system preserve such an order, in a way that is made mathematically precise in the paper, one can compute the uncontrollable predecessor of an interval just by computing its lower and upper bounds. In this way, one can compute the capture set by

simply back propagating upper and lower bounds as opposed to back propagating entire sets. This approach is inspired by interval abstraction techniques [22], which have been extensively employed in the static analysis of programs. Within this approach, representing the set of current system states as an interval is crucial for determining the control map. In fact, such a control map relies on the comparison between suitable upper and lower bounds for exploiting the order preserving properties of the dynamics. Thus, we employ state estimators on partial orders as developed in [23, 24], as opposed to employing other set-valued approaches to state estimation such as the ones proposed by Alamoa *et al.* [25] and by the references therein. The net result is a dynamic control algorithm for safety specifications, which has linear complexity in the number of state variables for systems with order preserving properties.

The contents of this paper are as follows. In Section 2, we introduce the class of systems considered, that is, the class of triangular order preserving hybrid automata. In Section 3, we solve the perfect information case control problem, whereas in Section 4, we solve the imperfect information case control problem. Finally, Section 5 presents the application of the developed techniques to collision avoidance problems in intelligent transportation systems.

2. CLASS OF SYSTEMS CONSIDERED

In this section, we introduce basic notions on partial orders and the class of systems that we consider. A partial order [26] is a set P with a partial order relation '$\leqslant$', and we denote it by the pair $(P, \leqslant)$. For all $x, w \in P$, the sup$\{x, w\}$, denoted $x \curlyvee w$, is the smallest element that is larger than both x and w. The inf$\{x, w\}$, denoted $x \curlywedge w$, is the largest element that is smaller than both x and w. If $S \subseteq P$, $\vee S := \sup S$ and $\wedge S := \inf S$. If $x \curlywedge w \in X$ and $x \curlyvee w \in X$ for all $x, w \in X$, then $(X, \leqslant)$ is a *lattice*. Any interval sublattice of $(P, \leqslant)$ is given by $[L, U] = \{w \in P | L \leqslant w \leqslant U\}$ for $L, U \in P$. That is, this special sublattice can be represented by only two elements. A special type of partial ordering can be considered on $\mathbb{R}^n$, and it is given by the component-wise partial ordering, defined as follows. For all $x, y \in \mathbb{R}^n$ with $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$, we have that $x \leqslant y$ if and only if $x_i \leqslant y_i$ for all $i \in \{1, \ldots, n\}$. Let $(P, \leqslant)$ and $(Q, \leqslant)$ be partially ordered sets. A map $f : P \rightarrow Q$ is (i) an *order preserving map* if $x \leqslant w \Longrightarrow f(x) \leqslant f(w)$ and (ii) an *order isomorphism* if $x \leqslant w \Longleftrightarrow f(x) \leqslant f(w)$ and it maps P *onto* Q.

We next specify the general class of transition systems to a class of systems that has both continuous and discrete states. That is, we define a hybrid automaton H as a transition system $\Sigma_H = (S, \mathscr{I}, \mathscr{Y}, \tau, \gamma)$, in which $S = Q \times X$, $\mathscr{I} = \mathscr{I}_D \times \mathscr{I}_C$, $\tau = (f, R)$, with $f : Q \times X \times \mathscr{I}_C \rightarrow X$, $R : Q \times X \times \mathscr{I}_D \rightarrow Q$, and $\gamma : \mathscr{Y} \rightarrow 2^X \times 2^Q$. In particular, we give the following definition, which is analogous to the continuous time counterpart [3].

Definition 1

A *discrete time hybrid automaton* is a tuple $H = (Q, X, \mathscr{I}, \iota, \mathscr{Y}, f, \text{Dom}, R, \gamma)$, in which $Q = \{q_1, \ldots, q_m\}$ is a set of discrete states (or modes); $X = \mathbb{R}^n$ is the set of continuous states; $\mathscr{I} = \mathscr{I}_D \times \mathscr{I}_C$ is the set of discrete and continuous inputs, respectively; $\iota : Q \rightarrow 2^{\mathscr{I}}$ is a function that attaches to each discrete state the set of enabled inputs; $\mathscr{Y}$ is a set of outputs; $f : Q \times X \times \mathscr{I}_C \rightarrow X$ is the continuous state update function; $\text{Dom} : Q \rightarrow 2^X$ is a map that for each mode establishes the domain in X in which such a mode is enabled; $R : Q \times X \times \mathscr{I}_D \rightarrow Q$ is the discrete state update map, which for any current discrete state, continuous state, and input determines the new discrete state; and $\gamma : \mathscr{Y} \rightarrow 2^X$ is the output map.

We denote the mode by $q \in Q$, the continuous state by $x \in X$, the continuous input by $u \in \mathscr{I}_C$, and the discrete input by $\sigma \in \mathscr{I}_D$. We assume that the reset function is static, that is, it does not contain memory of previous discrete states. Thus, we have $q = R(x, \sigma)$. We make an explicit distinction between two types of modes: the modes q such that $\mathrm{Dom}(q) = X$ and the modes q such that $\mathrm{Dom}(q) \neq X$. In particular, we assume that a transition to a mode with $\mathrm{Dom}(q) = X$ can occur *only* by a suitable choice of discrete inputs, whereas a transition to a mode with $\mathrm{Dom}(q) \neq X$ can occur only autonomously and thus cannot be controlled. This is formalized by the following structure of R:

$$R(x, \sigma) := \begin{cases} R_1(\sigma) & \text{if } \sigma \neq \emptyset \\ R_2(x) & \text{if } \sigma = \emptyset \end{cases} \tag{3}$$

in which we define $R_2(x) := q$ if $x \in \mathrm{Dom}(q)$. One can verify that this update is *deterministic* if $\mathrm{Dom}(q_1) \cap \mathrm{Dom}(q_2) = \emptyset$ whenever $\mathrm{Dom}(q_1) \neq \mathbb{R}^p$ and $\mathrm{Dom}(q_2) \neq \mathbb{R}^p$. In addition, we assume that for any mode with $\mathrm{Dom}(q) = \mathbb{R}^p$, there exists a discrete input $\sigma \in \mathscr{I}_D$ such that $q = R(\sigma)$. The *non-blocking* condition can be guaranteed if $\bigcup_{\{q | \mathrm{Dom}(q) \neq \mathbb{R}^p\}} \mathrm{Dom}(q) = \mathbb{R}^p$. In the sequel, we use the notation $\overline{Q} := \{q \in Q | \mathrm{Dom}(q) \neq \mathbb{R}^p\}$. The general class of discrete time hybrid automata is next restricted to the class of hybrid automata, in which the continuous state update map is order preserving.

Definition 2
Let $(\mathbb{R}^n, \leqslant)$ be the partial order established according to component-wise ordering. A *triangular order preserving hybrid automaton* is a hybrid automaton $H = (Q, X, \mathscr{I}, \iota, \mathscr{Y}, f, \mathrm{Dom}, R, \gamma)$, in which

  (i) The update map $f(q, x, u)$ for every $q \in Q$ and $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$ has the following triangular structure $f(q, x, u) = (f_1(x_1, \ldots, x_n), \ldots, f_i(x_i, \ldots, x_n), \ldots, f_n(x_n, q, u))$, in which $f_i : \mathbb{R}^{n-(i-1)} \to \mathbb{R}$ for $i \in \{1, \ldots, n-1\}$, $f_n : \mathbb{R} \times Q \times \mathscr{I}_C \to \mathbb{R}$ with $\mathscr{I}_C = \mathbb{R}$, and $\mathrm{Dom}(q) \subseteq \mathbb{R}^n$.

  (ii) We assume that the set of discrete states with $\mathrm{Dom}(q) = \mathbb{R}^n$ is a lattice with minimum α and with maximum β, that is, $\{q \in Q | \mathrm{Dom}(q) = \mathbb{R}^n\} = [\alpha, \beta]$. For all $q \in Q$, we assume that $\iota(q)$ is an interval in $\mathbb{R}$, that is, $\iota(q) = [u_L(q), u_U(q)]$. In addition, the functions $u_L(\cdot)$ and $u_U(\cdot)$ are order preserving in q with $\mathrm{Dom}(q) = \mathbb{R}^n$.

  (iii) We assume that f_i is order preserving in all of its arguments, that is, if $(x_i^a, \ldots, x_n^a) \leqslant (x_i^b, \ldots, x_n^b)$, then $f_i(x_i^a, \ldots, x_n^a) \leqslant f_i(x_i^b, \ldots, x_n^b)$ for $i < n$, and $f_n(x_n^a, q, u) \leqslant f_n(x_n^b, q, u)$. In addition, $f_n : Q|_{\{q \in Q | \mathrm{Dom}(q) = \mathbb{R}^n\}} \times \mathbb{R} \times \mathscr{I}_C \to \mathbb{R}$ is order preserving in all of its arguments. Additionally, f_i is one–one and onto in x_i, that is, fixed $x_{i+1}, \ldots, x_n, q, u$, for any x_i' there is one and only one x_i such that $f_i(x_i, \ldots, x_n) = x_i'$ if $i < n$ or $f_i(x_i, q, u) = x_i'$ if $i = n$. We denote the first one by $f_i^{-1}(x_i', x_{i+1}, \ldots, x_n)$ and the second one by $f_i^{-1}(x_i', q, u)$.

  (iv) The maps f_i are non-decreasing: $f_i(x_i, \ldots, x_n) \geqslant x_i$, for $i < n$ and $f_n(x_n, q, u_U(q)) > x_n$ for all q.

  (v) For all $y \in \mathscr{Y}$, the set $\gamma(y) \subseteq \mathbb{R}^n$ is an interval in $(\mathbb{R}^n, \leqslant)$, that is, $\gamma(y) = [\wedge \gamma(y), \vee \gamma(y)]$.

Example 1
A practical motivation for the structure of the dynamics in Definition 2 is given by the longitudinal dynamics of vehicles along their lanes or paths. Let $x \in \mathbb{R}$ represent the coordinate of a vehicle

along its lane, then

$$\ddot{x} = R^2/(J_\text{w} + mR^2)\left(f_\text{w} - f_\text{brake} - \frac{\rho_\text{air}}{2}C_\text{D}A_\text{f}U^2 - C_\text{rr}mg - mg\sin(\theta_\text{road})\right) \tag{4}$$

in which J_w is the wheel inertia, m is the mass of the vehicle, R is the tire radius, f_brake is the brake force, $f_\text{w} = \tau_\text{w}/R$ where τ_w is the drive-shaft output torque, U is the longitudinal vehicle velocity, ρ_air is the air density, C_D is the drag coefficient, A_f is the projected front area of the vehicle, C_rr is the rolling resistance coefficient, R is the tire radius, and θ_road is the road gradient. For more details on this model, the reader is referred to [27] and to the references therein. For automatic driving, f_w and f_brake are control inputs to the longitudinal dynamics of the vehicle. Let the total force $F = f_\text{w} - f_\text{brake}$, $\delta = R^2/(J_\text{w} + mR^2)(-(\rho_\text{air}/2)C_\text{D}A_\text{f}U^2 - C_\text{rr}mg)$, $b = R^2/(J_\text{w} + mR^2)$, and $\theta_\text{road} = 0$. Assuming that all the parameters are exactly known, then δ is also known as the vehicle measures on-board its own longitudinal velocity U. Thus, one can set $F = (u - \delta)/b$ so that the resulting discrete time model can be expressed as

$$x_1' = x_1 + x_2\Delta T, \quad x_2' = x_2 + u\Delta T \tag{5}$$

in which $x_1 = x$, $x_2 = \dot{x}_1$, and $\Delta T > 0$ is the discretization time. Such a model not only has a triangular structure but also the update map is order preserving and in particular f_i are invertible with respect to the x_i. In addition, condition (iv) of Definition 2 is always satisfied because the vehicles cannot move in reverse along their lane and if u is large (F is a sufficiently large positive force), then the vehicle will accelerate.

The parallel composition of a number of triangular order preserving hybrid automata generates a block-triangular order preserving hybrid automaton. This is made more precise by defining the parallel composition of hybrid automata in a way similar to that in [28].

Definition 3
Let $H_1 = (Q_1, X_1, \mathscr{I}_1, \iota_1, \mathscr{Y}_1, f_1, \text{Dom}_1, R_1, \gamma_1)$ and $H_2 = (Q_2, X_2, \mathscr{I}_2, \iota_2, \mathscr{Y}_2, f_2, \text{Dom}_2, R_2, \gamma_2)$ be two hybrid automata. The parallel composition, denoted $H = H_1 \| H_2$, is given by $H = (Q, X, \mathscr{I}, \iota, \mathscr{Y}, f, \text{Dom}, R, \gamma)$, in which $Q = Q_1 \times Q_2$, $X = X_1 \times X_2$, $\mathscr{I} = \mathscr{I}_C \times \mathscr{I}_\text{D}$ with $\mathscr{I}_C = \mathscr{I}_{C,1} \times \mathscr{I}_{C,2}$ and $\mathscr{I}_\text{D} = \mathscr{I}_{D,1} \times \mathscr{I}_{D,2}$; $\iota: Q \to \mathscr{I}_C$ is given by $\iota = (\iota_1, \iota_2)$; $\mathscr{Y} = \mathscr{Y}_1 \times \mathscr{Y}_2$; $f: Q \times X \times \mathscr{I}_C \to X$ is given by $f = (f_1, f_2)$; $\text{Dom}(q) = \text{Dom}_1(q_1) \times \text{Dom}_2(q_2)$; $R(x, \sigma) = (R_1(x_1, \sigma_1), R_2(x_2, \sigma_2))$; $\gamma = (\gamma_1, \gamma_2)$.

Definition 4
A *block-triangular order preserving hybrid automaton* is the parallel composition of N triangular order preserving hybrid automata $H_1, \ldots, H_N$.

Let $x_i = (x_{1,i}, \ldots, x_{n,i}) \in \mathbb{R}^n$, $q_i \in Q_i$, $u_i \in \iota(q_i)$, and $\sigma_i \in \mathscr{I}_{D,i}$ represent the continuous state, the discrete state, the continuous input, and the discrete input of the triangular hybrid automaton H_i, respectively. Then, in each mode $q = (q_1, \ldots, q_N)$ of the hybrid automaton $H = H_1 \| \cdots \| H_N$, the continuous state update map has the following form:

$$\begin{aligned} x_{j,i}' &= f_{j,i}(x_{j,i}, \ldots, x_{n,i}), \quad j < n, \ i \in \{1, \ldots, N\} \\ x_{n,i}' &= f_{n,i}(x_{n,i}, q_i, u_i), \quad i \in \{1, \ldots, N\} \end{aligned} \tag{6}$$

in which primed variables denote updated variables. In the sequel, we will use the notation

$$f_i(q_i, x_i, u_i) = (f_{1,i}(x_{1,i}, \ldots, x_{n,i}), \ldots, f_{n,i}(q_i, x_{n,i}, u_i))$$

For system $H = H_1 \| \cdots \| H_N$, we model the safety requirement by requesting that the state x never enters the bad set:

$$
\begin{aligned}
B &= \{(x_{1,1}, \ldots, x_{n,1}, \ldots, x_{1,N}, \ldots, x_{n,N}) | (x_{1,1}, \ldots, x_{1,N}) \in \overline{B}\} \\
\overline{B} &= [L_1, U_1] \times \cdots \times [L_N, U_N] \quad \text{with } L_i, U_i \in \mathbb{R}
\end{aligned}
\tag{7}
$$

In the sequel, we denote $L = (L_1, \ldots, L_N)$ and $U = (U_1, \ldots, U_N)$. This choice of the safety requirement to involve only the variables $(x_{1,1}, \ldots, x_{1,N})$ is motivated by applications, such as collision avoidance, in which a collision occurs whenever the positions of the agents are too close to each other independently of the values of their speeds, accelerations, jerks, etc.

In the following two sections, we first address the case in which the whole state is measured (perfect information case). Then we consider the case in which the continuous state x is subject to uncertainty (imperfect information case). In such a case, the control map is the same as the perfect information case, but it is evaluated on the state estimates as opposed to being evaluated on the state.

3. THE CASE OF PERFECT INFORMATION

In this section, we construct the control map by computing an approximation $\bar{C}$ of the capture set C. To explain the idea of the algorithm that computes the over-approximation $\bar{C}$ of the capture set, we first introduce a very simple example.

Example 2
Consider the system

$$
\begin{aligned}
x'_1 &= x_1 + u_1, \quad u_1 \in [u_m, u_M] \\
x'_2 &= x_2 + u_2, \quad u_2 \in [u_m, u_M]
\end{aligned}
\tag{8}
$$

with bad set $B = [L, U] \times [L, U]$, with $L, U \in \mathbb{R}$ and $L < U$. In this case, the exact capture set C can be computed by employing the following very simple reasoning. The set of all (x_1, x_2) that are mapped by (8) inside B for all inputs $(u_1, u_2) \in [u_m, u_M] \times [u_m, u_M]$ is given by $\{(x_1, x_2) | x_1 \in [L - u_1, U - u_1]$ and $x_2 \in [L - u_2, U - u_2]$ for all $(u_1, u_2) \in [u_m, u_M] \times [u_m, u_M]\}$, which is simply given by $B^1 := \{(x_1, x_2) |; x_1 \in [L - u_m, U - u_M]$ and $x_2 \in [L - u_m, U - u_M]\}$. One can then compute the set of all (x_1, x_2) that are mapped by (8) inside B^1 for all inputs $(u_1, u_2) \in [u_m, u_M] \times [u_m, u_M]$, to obtain $[L - 2u_m, U - 2u_M] \times [L - 2u_m, U - 2u_M] := B^2$. As a consequence, one obtains

$$C = \bigcup_{k \geqslant 0} B^k, \quad B^k = [L - ku_m, U - ku_M] \times [L - ku_m, U - ku_M]$$

which is depicted in Figure 1. Let us denote the ends of B^k by $L^k := L - ku_m$ and $U^k := U - ku_M$. Once the capture set C is computed this way, the control map $(u_1, u_2) = g(x_1, x_2)$ is simply determined by leaving the input arbitrary if this input does not map through (8) the state (x_1, x_2)
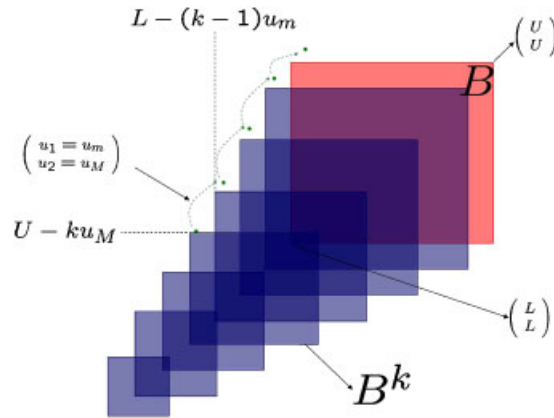
Figure 1. Example of capture set C (given by the union of the intervals) for the simple system in Equations (8). If a state (the dot) is 'above' the capture set C, the control $(u_1, u_2) = (u_m, u_M)$ guarantees that it is mapped still 'above' the capture set.

in C. Otherwise, set $u_1 = u_m$ and $u_2 = u_M$ if $x_1 \leqslant L^k$ and $x_2 \geqslant U^{k-1}$ for some k (that is, (x_1, x_2) is 'above' C), or set $u_1 = u_M$ and $u_2 = u_m$ if $x_1 \geqslant U^k$ and $x_2 \leqslant U^{k+1}$ for some k (that is, (x_1, x_2) is 'below' C). One can check that such an input map will keep any state (x_1, x_2) outside C still outside C. This is pictorially shown in Figure 1.

In this example, the computation of C has been obtained by back propagating only the lower and upper bounds of B^k, which is possible because the dynamics preserves the ordering on the state. In addition, the exploitation of the order preserving property of the dynamics with respect to the input allows to easily compute the control map by comparing (x_1, x_2) with the lower and upper bounds of B^k.

We next generalize the reasoning of this simple example to the class of systems given in Definition 4 in the case of perfect state information, that is, in the case in which $\gamma(y) = \{x\}$. Let thus B be as given in Equations (7). Denote

$$F_i(x_{2,i}, \ldots, x_{n,i}, q_i, u_i) := (f_{2,i}(x_{2,i}, \ldots, x_{n,i}), \ldots, f_{n,i}(x_{n_i}, q_i, u_i))$$

$\bar{x}_i = (x_{2,i}, \ldots, x_{n,i})$, and

$$F_i^k(\bar{x}_i, q_i, u_i) := F(F^{k-1}(\bar{x}_i, q_i, u_i), q_i, u_i)$$

Let $\bar{x} = (\bar{x}_1, \ldots, \bar{x}_N)$. Then, we have that $\bar{C} = \{(x_{1,1}, \ldots, x_{n,1}, \ldots, x_{1,N}, \ldots, x_{n,N}) | (x_{1,1}, \ldots, x_{1,N}) \in \bar{C}^*(\bar{x})\}$, in which $\bar{C}^*(\bar{x})$ is given by the following algorithm.

Algorithm 1
$\bar{C}^*(\bar{x}) = \bigcup_{k=0}^{k=k^*-1} [\bar{L}^k, \bar{U}^k]$, $\bar{L}^0 = L$, $\bar{U}^0 = U$, $\bar{L}^k = (\bar{L}_1^k, \ldots, \bar{L}_N^k)$, $\bar{U}^k = (\bar{U}_1^k, \ldots, \bar{U}_N^k)$ with

$$\bar{L}_i^1(\bar{x}_i) = f_{1,i}^{-1}(L_i^0, \bar{x}_i)$$

$$\bar{U}_i^1(\bar{x}_i) = f_{1,i}^{-1}(U_i^0, \bar{x}_i)$$

while for $k>1$, we have

$$L_i^k(\bar{x}_i) = L_i^{k,a}(\bar{x}_i) \curlyvee L_i^{k,b}(\bar{x}_i) \tag{9}$$

$$L_i^{k,a}(\bar{x}_i) = \bigwedge_{q_i \in \bar{Q}_i} f_{1,i}^{-1}(\bar{L}_i^{k-1}(F_i(\bar{x}_i, q_i, u_L(q_i))), \bar{x}_i) \tag{10}$$

$$L_i^{k,b}(\bar{x}_i) = f_{1,i}^{-1}(\bar{L}_i^{k-1}(F_i(\bar{x}_i, \alpha_i, u_L(\alpha_i))), \bar{x}_i) \tag{11}$$

$$U_i^k(\bar{x}_i) = U_i^{k,a}(\bar{x}_i) \curlywedge U_i^{k,b}(\bar{x}_i) \tag{12}$$

$$U_i^{k,a}(\bar{x}_i) = \bigvee_{q_i \in \bar{Q}_i} f_{1,i}^{-1}(\bar{U}_i^{k-1}(F_i(\bar{x}_i, q_i, u_U(q_i))), \bar{x}_i) \tag{13}$$

$$U_i^{k,b}(\bar{x}_i) = f_{1,i}^{-1}(\bar{U}_i^{k-1}(F_i(\bar{x}_i, \beta_i, u_U(\beta_i))), \bar{x}_i) \tag{14}$$

with (removing the dependence on $\bar{x}_i$ for shortness of notation)

$$\bar{L}_i^k = \inf(L_i^k, \bar{L}_i^{k-1}) \tag{15}$$

$$\bar{U}_i^k = \begin{cases} \sup(U_i^k, \bar{L}_i^{k-1}) & \text{if } \exists j \text{ such that } U_j^k > \bar{L}_j^{k-1} \\ U_i^k & \text{if } U_j^k \leqslant \bar{L}_j^{k-1} \ \forall j \end{cases} \tag{16}$$

with k^* the smallest k such that

$$U_i^k \leqslant \bar{L}_i^{k-1} \quad \forall i \text{ and } \exists j \text{ such that } \bar{U}_j^k < \bar{L}_j^k$$

For a fixed $\bar{x}$, the set $\bar{C}^*(\bar{x})$ is the union of k^* intervals in $\mathbb{R}^N$. Expressions (9) and (12) of the extremes of such intervals depend on the values of the variables $(x_{2,i}, \ldots, x_{n,i})$ for all i. For computation, one can off-line symbolically compute the iterative expressions (9) and (12) and evaluate them only when the value of $(x_{2,i}, \ldots, x_{n,i})$ becomes available on-line. The set $\bar{C}$ is obtained by computing at each iteration $(n-1)N$ computations for evaluating $f_{j,i}$ for $j>1$ and for $i \in [2, N]$. This procedure has thus linear complexity with the number of continuous state variables.

We next show that Algorithm 1 computes an over-approximation of C, that is, $\bar{C} \supseteq C$, by showing that for all $x \notin \bar{C}$, there is always an input such that x is mapped outside $\bar{C}$. We show this in two parts. First, we demonstrate that whenever $x \notin \bar{C}$ (and thus $(x_{1,1}, \ldots, x_{1,N}) \notin \bar{C}^*(\bar{x}) \subseteq \mathbb{R}^N$) there is a two-dimensional projection of $\bar{C}^*(\bar{x})$ and of $(x_{1,1}, \ldots, x_{1,N})$ along coordinate axis (i, j) in $\mathbb{R}^N$, such that $(x_{1,i}, x_{1,j})$ is not contained in $\bigcup_{k=0}^{k^*-1} [\bar{L}_i^k(\bar{x}_i), \bar{U}_i^k(\bar{x}_i)] \times [\bar{L}_j^k(\bar{x}_j), \bar{U}_j^k(\bar{x}_j)]$ (Proposition 1). From a geometric point of view, the situation, in such a plane, becomes then similar to the one in Figure 1 obtained for the simple example. Second, we consider such two-dimensional projection of $\bar{C}^*(\bar{x})$ and of $(x_{1,1}, \ldots, x_{1,N})$ to compute an input that maps the two-dimensional projection of $(x_{1,1}, \ldots, x_{1,N})$ outside the two-dimensional projection of $\bar{C}^*(\bar{x})$ (Proposition 2). Let us denote the interior of a set for a set S in a metric space by $\text{Int}(S)$.

Proposition 1
Given Algorithm 1, the following are equivalent:

(i) $(x_{1,1}, \ldots, x_{1,N}) \notin \text{Int}(\bar{C}^*(\bar{x}))$;
(ii) there is a pair of coordinates (i, j) with $i \neq j$ and a $\bar{k} < k^*$ such that either $x_{1,i} \leqslant \bar{L}_i^k(\bar{x}_i)$ for $k \leqslant \bar{k}$ and $x_{1,j} \geqslant \bar{U}_j^k(\bar{x}_j)$ for $k > \bar{k}$ or $x_{1,i} \leqslant \bar{L}_i^{k^*-1}(\bar{x}_i)$.

Proof

We omit here the dependence of $\bar{C}^*$, $\bar{L}^k$, and $\bar{U}^k$ on $\bar{x}$. (The proof that (ii) implies (i) follows directly by the fact that (ii) implies that $(x_{1,1}, \ldots, x_{1,N})$ is not contained in any of the intervals composing $\bar{C}^*$.) We thus prove that (i) implies (ii). If $(x_{1,1}, \ldots, x_{1,N}) \notin \mathrm{Int}(\bar{C}^*)$, then $(x_{1,1}, \ldots, x_{1,N})$ is not in any of the component rectangles of $\bar{C}^*$, that is, $(x_{1,1}, \ldots, x_{1,N}) \notin \mathrm{Int}([\bar{L}_1^k, \bar{U}_1^k] \times \cdots \times [\bar{L}_N^k, \bar{U}_N^k])$ for all k. This, in turn, implies that for all k there is at least one i_k such that either (a) $x_{1,i_k} \leqslant \bar{L}_{i_k}^k$ or (b) $x_{1,i_k} \geqslant \bar{U}_{i_k}^k$. Let k be the smallest integer less than k^* such that there is a i_k with $x_{1,i_k} \geqslant \bar{U}_{i_k}^k$. If it does not exist, it means that there is i_{k^*-1} such that $x_{1,i_{k^*-1}} \leqslant \bar{L}_{i_{k^*-1}}^{k^*-1}$. If it exists, it implies that $x_{1,i_k} \geqslant \bar{U}_{i_k}^k$ and that there is a i_{k-1} such that $x_{1,i_{k-1}} \leqslant \bar{L}_{i_{k-1}}^{k-1}$. Here, we can have two cases: (1) $i_{k-1} \neq i_k$ and (2) $i_{k-1} = i_k$. In case (1), thus we have $x_{1,i_k} \geqslant \bar{U}_{i_k}^k \geqslant \bar{U}_{i_k}^{k+1} \geqslant \cdots \geqslant \bar{U}_{i_k}^{k^*}$ and $x_{1,i_{k-1}} \leqslant \bar{L}_{i_{k-1}}^{k-1} \leqslant \bar{L}_{i_{k-1}}^{k-2} \leqslant \cdots \leqslant \bar{L}_{i_{k-1}}^0$ because the sequences $\{\bar{L}^k\}$ and $\{\bar{U}^k\}$ are non-increasing. In case (2), we have that $\bar{U}_{i_k}^k \leqslant x_{1,i_k} \leqslant \bar{L}_{i_{k-1}}^{k-1}$. This in turn is possible if $\bar{U}_{i_k}^k < \bar{L}_{i_k}^{k-1}$, which by Equations (16) is possible if $\bar{U}_j^k \leqslant \bar{L}_j^{k-1}$ for all j. Then, either there is a $j \neq i_k$ such that $x_{1,j} > \bar{U}_j^k$ or for all $j \neq i_k$, we have that $x_{1,j} \leqslant \bar{U}_j^k$ that implies $x_{1,j} \leqslant \bar{L}_j^{k-1}$. The relation $x_{1,j} \geqslant \bar{U}_j^k$ with $j \neq i_k$ and with $x_{1,i_k} \leqslant \bar{L}_{i_{k-1}}^{k-1}$ falls back into case (1). Similarly, having that $x_{1,j} \leqslant \bar{L}_j^{k-1}$ for all $j \neq i_k$ and $\bar{U}_{i_k}^k \leqslant x_{1,i_k}$ also falls back into case (1). $\qquad\square$

Proposition 2

Let $\bar{L}_i^k(\bar{x}_i)$ and $\bar{U}_i^k(\bar{x}_i)$ be as in Algorithm 1. If $x_{1,i} \leqslant \bar{L}_i^k(\bar{x}_i)$, $(x_{1,i} \geqslant \bar{U}_i^k(\bar{x}_i))$, then there exists a continuous/discrete control law such that $x_{1,i}' \leqslant \bar{L}_i^{k-1}(\bar{x}_i')$ $(x_{1,i}' \geqslant \bar{U}_i^{k-1}(\bar{x}_i'))$. In particular, such a control law is as follows:

$$\text{if } x_{1,i} \leqslant \bar{L}_i^k(\bar{x}_i) \quad \text{then}$$

$$R_{1,i}(\sigma_i) = \alpha_i, \quad u_i = u_L(\alpha_i) \quad \text{if } L_i^{k,a}(\bar{x}_i) < L_i^{k,b}(\bar{x}_i) \tag{17}$$

$$R_{2,i}(x_i) = q_i, \quad u_i = u_L(q_i) \quad \text{if } L_i^{k,a}(\bar{x}_i) \geqslant L_i^{k,b}(\bar{x}_i)$$

$$\text{if } x_{1,i} \geqslant \bar{U}_i^k(\bar{x}_i) \quad \text{then}$$

$$R_{1,i}(\sigma_i) = \beta_i, \quad u_i = u_U(\beta_i) \quad \text{if } U_i^{k,a}(\bar{x}_i) > U_i^{k,b}(\bar{x}_i) \tag{18}$$

$$R_{2,i}(x_i) = q_i, \quad u_i = u_U(q_i) \quad \text{if } U_i^{k,a}(\bar{x}_i) \leqslant U_i^{k,b}(\bar{x}_i)$$

Proof

In the case in which $x_{1,i} \leqslant \bar{L}_i^k(\bar{x}_i)$, we also have by Expressions (16) that $x_{1,i} \leqslant L_i^k(\bar{x}_i)$. If also $L_i^{k,a}(\bar{x}_i) > L_i^{k,b}(\bar{x}_i)$, we will have that $x_{1,i} \leqslant L_i^{k,a}(\bar{x}_i)$. Applying $f_{1,i}$ on both sides and taking into account that $f_{1,i}$ preserves the ordering, we obtain that $f_{1,i}(x_{1,i}, \bar{x}_i) \leqslant f_{1,i}(L_i^{k,a}(\bar{x}_i), \bar{x}_i)$. By Equation (10) and by the order isomorphism property of $f_{1,i}$ in its first argument, we have that $f_{1,i}(L_i^{k,a}(\bar{x}_i), \bar{x}_i) = \bigwedge_{q_i \in \bar{Q}_i} \bar{L}_i^{k-1}(F_i(\bar{x}_i, q_i, u_L(q_i)))$. In addition, we have that $\bigwedge_{q_i \in \bar{Q}_i} \bar{L}_i^{k-1}(F_i(\bar{x}_i, q_i, u_L(q_i))) \leqslant \bar{L}_i^{k-1}(F_i(\bar{x}_i, q_i, u_L(q_i)))$. As a consequence, if we choose the control action such that $q_i = R_{2,i}(x_i)$ and $u_i = u_L(q_i)$, we obtain that $F_i(\bar{x}_i, q_i, u_L(q_i)) = (x_{2,i}', \ldots, x_{n,i}') = \bar{x}_i'$ and therefore that $x_{1,i}' = f_{1,i}(x_{1,i}, \bar{x}_i) \leqslant \bar{L}_i^{k-1}(\bar{x}_i')$. If $L_i^{k,a}(\bar{x}_i) \leqslant L_i^{k,b}(\bar{x}_i)$. We

can proceed similarly to obtain that $x_{1,i} \leqslant L_i^{k,b}(\bar{x}_i)$ implies by the order preserving property of $f_{1,i}$ that $f_{1,i}(x_{1,i}, \bar{x}_i) \leqslant f_{1,i}(L_i^{k,b}(\bar{x}_i), \bar{x}_i)$. By Equation (11), we also have that $f_{1,i}(L_i^{k,b}(\bar{x}_i), \bar{x}_i) = \bar{L}_i^{k-1}(F_i(\bar{x}_i, \alpha_i, u_L(\alpha_i)))$, which by choosing $R_{1,i}(\sigma_i) = \alpha_i$ and $u_i = u_L(\alpha_i)$ is equal to $\bar{L}_i^{k-1}(x'_{2,i}, \ldots, x'_{n,i}) = \bar{L}_i^{k-1}(\bar{x}'_i)$. As a consequence, we have again that $x'_{1,i} = f_{1,i}(x_{1,i}, \bar{x}_i) \leqslant \bar{L}_i^{k-1}(x'_{2,i}, \ldots, x'_{n,i}) = \bar{L}_i^{k-1}(\bar{x}'_i)$. If $x_{1,i} \geqslant \bar{U}_i^k(\bar{x}_i)$, then it follows from expressions (16) that $x_{1,i} \geqslant U_i^k(\bar{x}_i)$ and the proof proceeds in a way similar as performed above. $\qquad \square$

When the measurement x becomes available, the extremes $\bar{L}_i^k(\bar{x}_i)$ and $\bar{U}_i^k(\bar{x}_i)$ can be evaluated. Then, one checks whether maintaining the current input will cause that $(x'_{1,1}, \ldots, x'_{1,N})$ will enter any of the intervals $[\bar{L}_1^k(\bar{x}'_1), \bar{U}_1^k(\bar{x}'_1)] \times \cdots \times [\bar{L}_N^k(\bar{x}'_N), \bar{U}_N^k(\bar{x}'_N)]$ for all k. If not, the input is maintained constant. Otherwise, the input is changed according to the following algorithm.

Algorithm 2

 (i) If there is a $k \in [0, k^* - 1]$ and a pair of coordinates (i, j) such that $x_{1,i} \geqslant \bar{U}_i^{k+1}$ and $x_{1,j} \leqslant \bar{L}_j^k$, then set $(R_i(x_i, \sigma_i), u_i)$ as in Equation (18) with $k+1$ in place of k, and set $(R_j(x_j, \sigma_j), u_j)$ as in Equation (17);

 (ii) If instead $(x_{1,1}, \ldots, x_{N,1}) \leqslant (\bar{L}_1^{k^*-1}(\bar{x}_1), \ldots, \bar{L}_N^{k^*-1}(\bar{x}_N))$, select (i, j) such that $x_{1,i} \leqslant \bar{L}_i^{k^*-1}(\bar{x}_i)$ and $x_{1,j} \leqslant \bar{L}_j^{k^*-1}(\bar{x}_j)$ with $\bar{U}_j^{k^*}(\bar{x}_j) < \bar{L}_j^{k^*}(\bar{x}_j)$. If $x_{1,j} \geqslant \bar{L}_j^{k^*}(\bar{x}_j)$ then $(x_{1,j} \geqslant U_j^{k^*}(\bar{x}_j))$, set $(R_j(x_j, \sigma_j), u_j)$ as in Equation (18) and set $(R_i(x_i, \sigma_i), u_i)$ as in Equation (17). If $x_{1,j} \leqslant \bar{L}_{1,j}^{k^*}$, set $(R(x_j, \sigma_j), u_j)$ as in Equation (17) and set $(R_i(x_i, \sigma_i), u_i)$ arbitrarily (if $R_i(x_i, \sigma_i) = R_{2,i}(x_i)$, then set $u_i \in \iota_i(q_i)$).

Algorithm 2 thus provides switching control laws $u = g_C(x, q)$ and $\sigma = g_D(x)$ with $q = R(x, \sigma)$ such that if $x \notin \bar{C}$ with $\bar{C} = \{(x_{1,1}, \ldots, x_{n,1}, \ldots, x_{1,N}, \ldots, x_{n,N}) | (x_{1,1}, \ldots, x_{1,N}) \in \bar{C}^*(\bar{x})\}$ and $\bar{C}^*(\bar{x})$ as computed by Algorithm 1, then $x' \notin \bar{C}$.

4. THE CASE OF IMPERFECT INFORMATION

Consider the class of systems given in Definition 4, in which now $\gamma(y)$ returns a set of possible continuous states compatible with the output measurement y. In order to proceed, we construct a state estimator and a controller is then determined on the basis of the state estimates.

4.1. State estimator

Consider hybrid automaton H and let $\hat{x} \subseteq X$. A set valued state estimator for H of the type of the one in Equation (1) can take, for example, the form

$$\hat{x}' = f(\hat{q}, \hat{x}, u) \cap \gamma(y') \quad \text{with } \hat{q} = R(\hat{x}, \sigma) \tag{19}$$

and

$$R(\hat{x}, \sigma) = \begin{cases} R_1(\sigma) & \text{if } \sigma \neq \emptyset \\ R_2(\hat{x}) & \text{if } \sigma = \emptyset \end{cases}$$

in which $R_2(\hat{x}) = \{q \in \overline{Q} | \exists x \in \hat{x},$ with, $x \in \text{Dom}(q)\}$ and $\hat{x}^0 = \gamma(y^0)$. One can verify that $x^k \in \hat{x}^k$ for all k. This type of estimator is impractical for implementation because the sets $\hat{x}$ are in general infinite sets. However, since H is the parallel composition of order preserving triangular hybrid automata H_i, the update maps f_i are order preserving and $\gamma(y_i) = [\wedge\gamma(y_i), \vee\gamma(y_i)]$. As a consequence, one can keep track of the lower and upper bounds of $\hat{x}$ as follows. Let $\vee\hat{x} = (\vee\hat{x}_1, \ldots, \vee\hat{x}_N)$ and $\wedge\hat{x} = (\wedge\hat{x}_1, \ldots, \wedge\hat{x}_N)$ denote the upper and lower bounds of $\hat{x}$, respectively. Then, we have that $\vee\hat{x}_i = (\vee\hat{x}_{1,i}, \ldots, \vee\hat{x}_{n,i}) \in \mathbb{R}^n$ and $\wedge\hat{x}_i = (\wedge\hat{x}_{1,i}, \ldots, \wedge\hat{x}_{n,i}) \in \mathbb{R}^n$ are the lower and upper bounds of $\hat{x}_i$, respectively, in which $\hat{x}_i \subseteq \mathbb{R}^n$ is the state estimate of the component automaton H_i. Then, the bounds of $\hat{x}_i$ for all i are updated according to the following equations:

$$\text{if } \sigma_i \neq \emptyset, \quad \begin{aligned} \wedge\hat{x}'_i &= f_i(\wedge\hat{x}_i, R_{1,i}(\sigma_i), u_i) \curlyvee \wedge\gamma(y'_i) \\ \vee\hat{x}'_i &= f_i(\vee\hat{x}_i, R_{1,i}(\sigma_i), u_i) \curlywedge \vee\gamma(y'_i) \end{aligned} \tag{20}$$

$$\text{if } \sigma_i = \emptyset, \quad \begin{aligned} \wedge\hat{x}'_i &= \bigwedge_{q_i \in \hat{\overline{Q}}_i} f_i(\wedge\hat{x}_i, q_i, u_i) \curlyvee \wedge\gamma(y'_i) \\ \vee\hat{x}'_i &= \bigvee_{q_i \in \hat{\overline{Q}}_i} f_i(\vee\hat{x}_i, q_i, u_i) \curlywedge \vee\gamma(y'_i) \end{aligned} \tag{21}$$

in which y'_i is the output observation of H_i, $\hat{\overline{Q}}_i$ is the set of possible modes that are compatible with the interval of states $[\wedge\hat{x}_i, \vee\hat{x}_i]$, that is,

$$\hat{\overline{Q}}_i = \{q_i \in \overline{Q}_i | \exists x_i \in [\wedge\hat{x}_i, \vee\hat{x}_i] \text{ such that } x_i \in \text{Dom}_i(q_i)\} \tag{22}$$

$$\wedge\hat{x}_i^0 = \wedge\gamma(y_i^0), \quad \vee\hat{x}_i^0 = \vee\gamma(y_i^0) \tag{23}$$

Proposition 3
Let $\{u^k\}_{k\in\mathbb{N}}$ be an input sequence for the block-triangular order preserving hybrid automaton $H = H_1 \| \cdots \| H_N$, and let $\{x^k\}_{k\in\mathbb{N}}$ and $\{y^k\}_{k\in\mathbb{N}}$ be the corresponding execution and output sequence. Let $\{\wedge\hat{x}^k\}_{k\in\mathbb{N}}$ and $\{\vee\hat{x}^k\}_{k\in\mathbb{N}}$ with $\vee\hat{x} = (\vee\hat{x}_1, \ldots, \vee\hat{x}_N)$ and $\wedge\hat{x} = (\wedge\hat{x}_1, \ldots, \wedge\hat{x}_N)$ be generated by Equations (20)–(23). Then, $x^k \in [\wedge\hat{x}^k, \vee\hat{x}^k]$ for all k.

The proof of this proposition is a consequence of the order preserving property of f_i and of the interval structure of $\gamma(y_i)$. For more details on these types of estimators and for convergence conditions, the reader is referred to [23, 24].

4.2. Dynamic feedback

Once the set of all possible current states is known, we can design a control input that maps such a set as a whole forward in such a way that it will never intersect the bad set B. Consider again Example 2. We illustrate how in such a case it is not necessary to compute the capture set for the imperfect information case as defined in Equation (2). Instead, the same capture set as computed in the perfect information case is employed.

Example 3
Consider again the system in Equations (8) and consider the capture set C computed earlier and shown in Figure 2. Now consider also a state uncertainty as given by the state estimator. This
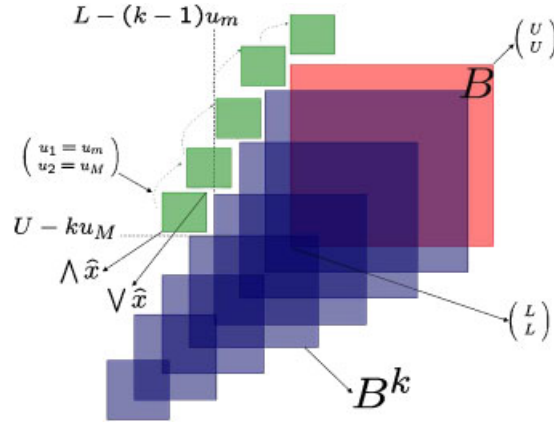
Figure 2. Capture set C for the system in Example 3. The state estimate is given by the interval $[\wedge\hat{x}, \vee\hat{x}]$. A control input $(u_1, u_2) = (u_m, u_M)$ guarantees that the state estimate is kept 'above' the capture set.

uncertainty is an interval $\hat{x} = [\wedge\hat{x}, \vee\hat{x}]$ as shown in Figure 2. Since $\wedge\hat{x}_2 > U^k$ and $\vee\hat{x}_1 < L^{k-1}$, one can set $u_1 = u_m$ and $u_2 = u_M$ (as in the perfect information case) so that the state estimate $\hat{x}$ is mapped still outside the capture set C as shown in Figure 2. A similar reasoning would have lead to $u_1 = u_M$ and $u_2 = u_m$ if the state estimate $\hat{x}$ were 'below' the capture set C.

This example exploits the computation of C obtained in the perfect information case and exploits the order preserving property of the dynamics in order to determine an input that maps a set $\hat{x}$ not intersecting C to a set that still does not intersect C. This problem can be generally formulated as follows.

Problem 1 (*State estimator-based safety control problem*)
Given transition system $\Sigma = (S, \mathcal{I}, \mathcal{Y}, \tau, \gamma)$ with bad set $B \subseteq S$, determine the smallest set $C \subseteq S$ with $B \subseteq C$, if it exists, and a dynamic feedback law $u^k = g(\hat{s}^k)$, with $\hat{s}^k = \hat{\tau}(\hat{s}^{k-1}, u^{k-1}, y^k)$, $\hat{s}^0 = \gamma(y^0)$, and $\{y^k\}_{k\in\mathbb{N}}$ output sequence of Σ such that if $\hat{s}^0 \cap C = \emptyset$, then $\hat{s}^k \cap C = \emptyset$ for all k.

Instead of computing the capture set $C \subset 2^S$ as defined in Equation (2), we seek to compute a set $C \subset S$ such that if the information state $\hat{s}$ does not intersect it, then there is a control input that will map such information state still outside C. This, if possible, simplifies the computation of C. In particular, we show that a good over-approximation of such a set C for the general class of block-triangular order preserving hybrid automata is provided by the following algorithm, which provides a slightly augmented set with respect to the one computed by Algorithm 1. Then, we have $\bar{C} = \{(x_{1,1}, \ldots, x_{n,1}, \ldots, x_{1,N}, \ldots, x_{n,N}) | (x_{1,1}, \ldots, x_{1,N}) \in \bar{C}^*(\bar{x})\}$, in which $\bar{C}^*(\bar{x})$ is given by the following algorithm.

Algorithm 3
$\bar{C}^*(\bar{x}) = \bigcup_{k\geqslant 0}[\bar{L}^k, \bar{U}^k]$, $\bar{L}^0 = L$, $\bar{U}^0 = U$, $\bar{L}^k = (\bar{L}_1^k, \ldots, \bar{L}_N^k)$, $\bar{U}^k = (\bar{U}_1^k, \ldots, \bar{U}_N^k)$ with

$$\bar{L}_i^1(\bar{x}_i) = f_{1,i}^{-1}(L_i^0, \bar{x}_i)$$
$$\bar{U}_i^1(\bar{x}_i) = f_{1,i}^{-1}(U_i^0, \bar{x}_i)$$

while for $k > 1$, we have

$$L_i^k(\bar{x}_i) = L_i^{k,a}(\bar{x}_i) \curlyvee L_i^{k,b}(\bar{x}_i) \tag{24}$$

$$L_i^{k,a}(\bar{x}_i) = \bigwedge_{q_i \in \bar{Q}_i} f_{1,i}^{-1}(\bar{L}_i^{k-1}(F_i(\bar{x}_i, q_i, u_L(q_i))), \bar{x}_i) \tag{25}$$

$$L_i^{k,b}(\bar{x}_i) = f_{1,i}^{-1}(\bar{L}_i^{k-1}(F_i(\bar{x}_i, \alpha_i, u_L(\alpha_i))), \bar{x}_i) \tag{26}$$

$$U_i^k(\bar{x}_i) = U_i^{k,a}(\bar{x}_i) \curlywedge U_i^{k,b}(\bar{x}_i) \tag{27}$$

$$U_i^{k,a}(\bar{x}_i) = \bigvee_{q_i \in \bar{Q}_i} f_{1,i}^{-1}(\bar{U}_i^{k-1}(F_i(\bar{x}_i, q_i, u_U(q_i))), \bar{x}_i) \tag{28}$$

$$U_i^{k,b}(\bar{x}_i) = f_{1,i}^{-1}(\bar{U}_i^{k-1}(F_i(\bar{x}_i, \beta_i, u_U(\beta_i))), \bar{x}_i) \tag{29}$$

with (removing the dependence on $\bar{x}_i$ for shortness of notation)

$$\bar{L}_i^k = \inf(L_i^k, \bar{L}_i^{k-1}, U_i^k) \tag{30}$$

$$\bar{U}_i^k = \sup(U_i^k, \bar{L}_i^{k-1}, \bar{L}_i^k) \tag{31}$$

The following proposition provides an easy check for establishing whether the state estimate set $[\wedge\hat{x}, \vee\hat{x}]$ does not intersect $\bar{C}$.

Proposition 4
We have that $[\wedge\hat{x}, \vee\hat{x}] \cap \overline{C} = \emptyset$ if

$$[\wedge\hat{x}_{1,1}, \vee\hat{x}_{1,1}] \times \cdots \times [\wedge\hat{x}_{1,N}, \vee\hat{x}_{1,N}] \cap \bigcup_{k \geqslant 0} [\overline{L}^k(\vee\hat{\bar{x}}), \overline{U}^k(\wedge\hat{\bar{x}})] = \emptyset$$

Proof
We show that if $[\wedge\hat{x}_{1,1}, \vee\hat{x}_{1,1}] \times \cdots \times [\wedge\hat{x}_{1,N}, \vee\hat{x}_{1,N}] \cap \bigcup_{k \geqslant 0} [\overline{L}^k(\vee\hat{\bar{x}}), \overline{U}^k(\wedge\hat{\bar{x}})] = \emptyset$, then $[\wedge\hat{x}, \vee\hat{x}] \cap \overline{C} = \emptyset$. This derives directly from the fact that $\overline{C} = \{x | (x_{1,1}, \ldots, x_{1,N}) \in \overline{C}^*(\overline{x})\}$, in which $\overline{C}^*(\overline{x})$ is given by Algorithm 1, and by the fact that the functions $\overline{L}_i^k(\overline{x}_i)$ and $\overline{U}_i^k(\overline{x}_i)$ are order reversing functions of their arguments. □

By virtue of this proposition, if we can guarantee that $[\wedge\hat{x}_{1,1}, \vee\hat{x}_{1,1}] \times \cdots \times [\wedge\hat{x}_{1,N}, \vee\hat{x}_{1,N}] \cap \bigcup_{k \geqslant 0} [\overline{L}^k(\vee\hat{\bar{x}}), \overline{U}^k(\wedge\hat{\bar{x}})] = \emptyset$ at all time trough a suitable choice of a control map, then we also guarantee that $[\wedge\hat{x}, \vee\hat{x}] \cap \overline{C} = \emptyset$. The following proposition provides an intermediate result (analogous to Proposition 1 for the perfect information case) that plays a central role for establishing such a control map.

Proposition 5
The following are equivalent:

  (i) $[\wedge\hat{x}_{1,1}, \vee\hat{x}_{1,1}] \times \cdots \times [\wedge\hat{x}_{1,N}, \vee\hat{x}_{1,N}] \cap \bigcup_{k \geqslant 0} [\overline{L}^k(\vee\hat{\bar{x}}), \overline{U}^k(\wedge\hat{\bar{x}})] = \emptyset$;
 (ii) There is a pair of coordinates (i, j) with $i \neq j$ and an index $\bar{k}$ such that $\vee\hat{x}_{1,j} \leqslant \bar{L}_j^k(\vee\hat{\bar{x}}_j)$ for all $k \leqslant \bar{k}$ and $\wedge\hat{x}_{1,i} \geqslant \bar{U}_i^k(\wedge\hat{\bar{x}}_i)$ for all $k > \bar{k}$.

Proof
The fact that (ii) implies (i) follows because (ii) implies that the two-dimensional projection of the interval $[\wedge\hat{x}_{1,1}, \vee\hat{x}_{1,1}] \times \cdots \times [\wedge\hat{x}_{1,N}, \vee\hat{x}_{1,N}]$ on the (i,j) plane does not intersect any of the projections of the intervals that compose $\bigcup_{k\geqslant 0}[\overline{L}^k(\vee\hat{x}), \overline{U}^k(\wedge\hat{x})]$. We thus focus on proving that (i) implies (ii). This proof exploits the following properties of Algorithm 3. The sequences $\{\bar{U}^k(\bar{x})\}$ and $\{\bar{L}^k(\bar{x})\}$ for a fixed $\bar{x}$ are non-increasing and they tend to $-\infty$. Furthermore, the set $\bigcup_{k\geqslant 0}[\overline{L}^k(\vee\hat{x}), \overline{U}^k(\wedge\hat{x})]$ is connected for any fixed $\bar{x}$. If $[\wedge\hat{x}_{1,1}, \vee\hat{x}_{1,1}] \times \cdots \times [\wedge\hat{x}_{1,N}, \vee\hat{x}_{1,N}] \cap \bigcup_{k\geqslant 0}[\overline{L}^k(\vee\hat{x}), \overline{U}^k(\wedge\hat{x})] = \emptyset$, then, neglecting the dependence of $\bar{L}^k$ and of $\bar{U}^k$ on their arguments, for all k there is an i_k such that $\wedge\hat{x}_{i_k} > \bar{U}^k_{i_k}$ or $\bar{L}^k_{i_k} > \vee\hat{x}_{i_k}$. Let $\bar{k}$ be the smallest integer k such that $\wedge\hat{x}_{i_k} > \bar{U}^k_{i_k}$. This index is finite because the sequence $\{\bar{U}^k(\bar{x})\}$ for a fixed $\bar{x}_i$ tends to $-\infty$. As a consequence, because the sequence $\{\bar{U}^k(\bar{x})\}$ for a fixed $\bar{x}_i$ is also non-increasing, we have that $\wedge\hat{x}_{i_{\bar{k}}} > \bar{U}^k_{i_{\bar{k}}}$ for all $k\geqslant\bar{k}$. For $k=\bar{k}-1$ we thus have that $\bar{L}^{\bar{k}-1}_{i_{\bar{k}-1}} > \vee\hat{x}_{i_{\bar{k}-1}}$, and since the sequence $\{\bar{L}^k(\bar{x})\}$ for a fixed $\bar{x}$ is non-increasing, we also have that $\bar{L}^k_{i_{\bar{k}-1}} > \vee\hat{x}_{i_{\bar{k}-1}}$ for all $k\leqslant\bar{k}-1$. This proves (ii) because the set $\bigcup_{k\geqslant 0}[\overline{L}^k(\vee\hat{x}), \overline{U}^k(\wedge\hat{x})]$ is a connected set. In fact, it must be that $i_{\bar{k}} \neq i_{\bar{k}-1}$. If instead $i_{\bar{k}} = i_{\bar{k}-1}$, we would have that $\bar{L}^{\bar{k}-1}_{i_{\bar{k}}} \geqslant \vee\hat{x}_{i_{\bar{k}}} \geqslant \wedge\hat{x}_{i_{\bar{k}}} > \bar{U}^{\bar{k}}_{i_{\bar{k}}}$. This, in turn, would imply that $\bar{L}^{\bar{k}-1}_{i_{\bar{k}}} > \bar{U}^{\bar{k}}_{i_{\bar{k}}}$, which contradicts the connectedness of the set $\bigcup_{k\geqslant 0}[\overline{L}^k(\vee\hat{x}), \overline{U}^k(\wedge\hat{x})]$. $\qquad\square$

Assumption 1
We assume that $\bigcap_{q_i \in \overline{Q}_i}[u_L(q_i), u_U(q_i)] \neq \emptyset$.

This assumption guarantees that if there are a number of possible current discrete states, then there is at least an input that is enabled by all of the possible current discrete states.

Proposition 6
Let $\overline{L}^k_i(\overline{x}_i)$ and $\overline{U}^k_i(\overline{x}_i)$ be as in Algorithm 3 and let Assumption 1 hold. Let $\vee\hat{x}_i, \wedge\hat{x}_i \in \mathbb{R}^n$ and let $\vee\hat{x}'_i, \wedge\hat{x}'_i \in \mathbb{R}^n$ be the updated values according to Equations (20) and (21). If $\vee\hat{x}_{1,i} < \overline{L}^k_i(\vee\hat{\bar{x}}_i)$, $(\wedge\hat{x}_{1,i} > \overline{U}^k_i(\wedge\hat{\bar{x}}_i))$ then there exists a continuous/discrete control law such that $\vee\hat{x}'_{1,i} < \overline{L}^k_i(\vee\hat{\bar{x}}'_i)$, $(\wedge\hat{x}'_{1,i} > \overline{U}^k_i(\wedge\hat{\bar{x}}'_i))$. In particular, such a control law is as follows:

$$
\begin{aligned}
&\text{if } \vee\hat{x}_{1,i} < \overline{L}^k_i(\vee\hat{\bar{x}}_i) \quad \text{then} \\
&R_{1,i}(\sigma_i) = \alpha_i, \quad u_i = u_L(\alpha_i) \quad \text{if } L^{k,a}_i(\vee\hat{\bar{x}}_i) < L^{k,b}_i(\vee\hat{\bar{x}}_i) \\
&\qquad\qquad u_i = \bigvee_{q_i \in \hat{\bar{Q}}_i} u_L(q_i) \quad \text{if } L^{k,a}_i(\vee\hat{\bar{x}}_i) \geqslant L^{k,b}_i(\vee\hat{\bar{x}}_i)
\end{aligned}
\tag{32}
$$

$$
\begin{aligned}
&\text{if } \wedge\hat{x}_{1,i} > \overline{U}^k_i(\wedge\hat{\bar{x}}_i) \quad \text{then} \\
&R_{1,i}(\sigma_i) = \beta_i, \quad u_i = u_U(\beta_i) \quad \text{if } U^{k,a}_i(\wedge\hat{\bar{x}}_i) > U^{k,b}_i(\wedge\hat{\bar{x}}_i) \\
&\qquad\qquad u_i = \bigwedge_{q_i \in \hat{\bar{Q}}_i} u_U(q_i) \quad \text{if } U^{k,a}_i(\wedge\hat{\bar{x}}_i) \leqslant U^{k,b}_i(\wedge\hat{\bar{x}}_i)
\end{aligned}
\tag{33}
$$

Proof

We show that if $\wedge\hat{x}_{1,i} > \overline{U}_i^k(\wedge\hat{\overline{x}}_i)$ then there exists a continuous/discrete control (σ_i, u_i) such that $\wedge\hat{x}'_{1,i} > \overline{U}_i^k(\wedge\hat{\overline{x}}'_i)$ (the other case can be shown in a similar way).

If $\wedge\hat{x}_{1,i} > \overline{U}_i^k(\wedge\hat{\overline{x}}_i)$, then $\wedge\hat{x}_{1,i} > U_i^k(\wedge\hat{\overline{x}}_i)$ and $U_i^{k,a} < U_i^{k,b}$, we will have that $\wedge\hat{x}_{1,i} > U_i^{k,a}(\wedge\hat{\overline{x}}_i)$. Applying $f_{1,i}$ on both sides and taking into account that $f_{1,i}$ preserves the ordering, we obtain that $f_{1,i}(\wedge\hat{x}_i) > f_{1,i}(U_i^{k,a}(\wedge\hat{\overline{x}}_i), \wedge\hat{\overline{x}}_i)$. By expression (13), we have that $f_{1,i}(U_i^{k,a}(\wedge\hat{\overline{x}}_i), \wedge\hat{\overline{x}}_i) = \bigvee_{q_i \in \overline{Q}_i} \overline{U}_i^{k-1}(F_i(\wedge\hat{\overline{x}}_i, q_i, u_U(q_i)))$. By the fact that $U_i(\cdot)$ are order reversing functions of their arguments, we have that $\bigvee_{q_i \in \overline{Q}_i} \overline{U}_i^{k-1}(F_i(\wedge\hat{\overline{x}}_i, q_i, u_U(q_i))) = \overline{U}_i^{k-1}(\bigwedge_{q_i \in \overline{Q}_i} F_i(\wedge\hat{\overline{x}}_i, q_i, u_U(q_i)))$. As a consequence, if we choose $u_i = \bigwedge_{q_i \in \hat{\overline{Q}}_i} u_U(q_i)$, we obtain that $\wedge\hat{x}'_{1,i} > \overline{U}_i^{k-1}(\wedge\hat{\overline{x}}'_i)$ by virtue of Assumption 1. If instead $U_i^{k,a} \geqslant \overline{U}_i^{k,b}$, we will have that $\hat{x}_{1,i} > U_i^{k,b}(\wedge\hat{\overline{x}}_i)$. Applying $f_{1,i}$ on both sides and taking into account that $f_{1,i}$ preserves the ordering, we obtain that $f_{1,i}(\wedge\hat{x}_i) > f_{1,i}(U_i^{k,b}(\wedge\hat{\overline{x}}_i), \wedge\hat{\overline{x}}_i)$. By expression (13), we have that $f_{1,i}(U_i^{k,b}(\wedge\hat{\overline{x}}_i), \wedge\hat{\overline{x}}_i) = \overline{U}_i^{k-1}(F_i(\wedge\hat{\overline{x}}_i, \beta_i, u_U(\beta_i)))$. Therefore, choosing $R_{1,i}(\sigma_i) = \beta_i$ and $u_i = u_U(\beta_i)$, we obtain that $\wedge\hat{x}'_{1,i} > \overline{U}_i^{k-1}(\wedge\hat{\overline{x}}'_i)$. $\square$

The idea of this proposition is that all points that have the jth coordinate larger than $\overline{U}_j^{k+1}$ can be mapped to points with jth coordinate larger than $\overline{U}_j^k$ by suitable choice of input u_j (or σ_j). Similarly, all points with the ith coordinate smaller than $\overline{L}_i^k$ can be mapped to points with ith coordinate smaller than $\overline{L}_i^{k-1}$ by a suitable choice of input u_i (or σ_i).

Algorithm 4

  (i) Evaluate all $\overline{L}_i^k(\vee\hat{\overline{x}}_i)$ and $\overline{U}_i^k(\wedge\hat{\overline{x}}_i)$ for all i and all k until $\wedge\hat{\overline{x}}_{1,i} > \overline{U}_i^k(\wedge\hat{\overline{x}}_i)$ for all i.

  (ii) If there is a k and a pair of coordinates (i, j) such that $\wedge\hat{x}_{1,i} > \overline{U}_i^{k+1}(\wedge\hat{\overline{x}}_i)$ and $\vee\hat{x}_{1,j} < \overline{L}_j^k(\vee\hat{\overline{x}}_j)$, then set (σ_i, u_i) as in Equation (33) with $k+1$ in place of k, and set (σ_j, u_j) as in Equation (32);

Theorem 1

The control law of Algorithm 4 guarantees that if initially $[\wedge\hat{x}_{1,1}, \vee\hat{x}_{1,1}] \times \cdots \times [\wedge\hat{x}_{1,N}, \vee\hat{x}_{1,N}] \cap \bigcup_{k \geqslant 0}[\overline{L}^k(\vee\hat{\overline{x}}), \overline{U}^k(\wedge\hat{\overline{x}})] = \emptyset$, then $[\wedge\hat{x}, \vee\hat{x}] \cap \bar{C} = \emptyset$ at all time. Furthermore, Algorithm 4 terminates.

The fact that Algorithm 4 keeps $[\wedge\hat{x}_{1,1}, \vee\hat{x}_{1,1}] \times \cdots \times [\wedge\hat{x}_{1,N}, \vee\hat{x}_{1,N}] \cap \bigcup_{k \geqslant 0}[\overline{L}^k(\vee\hat{\overline{x}}), \overline{U}^k(\wedge\hat{\overline{x}})] = \emptyset$ at all time is a consequence of Propositions 5 and 6. Therefore, by virtue of Proposition 4, also $[\wedge\hat{x}, \vee\hat{x}] \cap \bar{C} = \emptyset$ at all time. Algorithm 4 terminates if and only if (i) of Algorithm 4 terminates. Termination of step (i) is guaranteed by the fact that the sequence $\{\overline{U}_i^k(\wedge\hat{\overline{x}}_i)\}_{k \in \mathbb{N}}$ tends to $-\infty$ for a fixed $\wedge\hat{\overline{x}}_i$.

In summary, the overall dynamic control strategy is established as follows. Based on the current values of $\wedge\hat{x}$ and of $\vee\hat{x}$, the predicted values of $\wedge\hat{x}$ and $\vee\hat{x}$ at the next step, denoted $\wedge\hat{x}_{\text{pred}}$ and

$\vee \hat{x}_{\text{pred}}$, are determined by

$$\text{if } \sigma_i \neq \emptyset, \quad \begin{aligned} \wedge \hat{x}_{\text{pred}} &= f_i(\wedge \hat{x}_i, R_{1,i}(\sigma_i), u_i) \\[4pt] \vee \hat{x}_{\text{pred}} &= f_i(\vee \hat{x}_i, R_{1,i}(\sigma_i), u_i) \end{aligned}$$

$$\text{if } \sigma_i = \emptyset, \quad \begin{aligned} \wedge \hat{x}_{\text{pred}} &= \bigwedge_{q_i \in \hat{\bar{Q}}_i} f_i(\wedge \hat{x}_i, q_i, u_i) \\[8pt] \vee \hat{x}_{\text{pred}} &= \bigvee_{q_i \in \hat{\bar{Q}}_i} f_i(\vee \hat{x}_i, q_i, u_i) \end{aligned}$$

with $\hat{\bar{Q}}_i$ as in Equation (22), with u_i and σ_i the inputs applied at the previous step. Thus, the intervals $[\overline{L}^k(\vee \hat{\bar{x}}_{\text{pred}}), \overline{U}^k(\wedge \hat{\bar{x}}_{\text{pred}})]$ are computed to check whether $[\wedge \hat{x}_{\text{pred}}, \vee \hat{x}_{\text{pred}}] \cap \overline{C} = \emptyset$ using Proposition 4. If the intersection is empty, then the current input is set to its previous value. If the intersection is not empty, we use Algorithm 4 to compute the new current input.

5. SIMULATION RESULTS

We consider two examples: one example involves the collision prediction and avoidance of vehicles at a traffic intersection and it involves only autonomous discrete state transitions. The second example instead considers a similar problem but with two trains at a railway merging and it involves only controlled mode transitions.

Vehicles at a traffic intersection. Let us consider two vehicles converging to a traffic intersection (represented in Figure 3). Each vehicle longitudinal dynamics can be modeled by the second-order
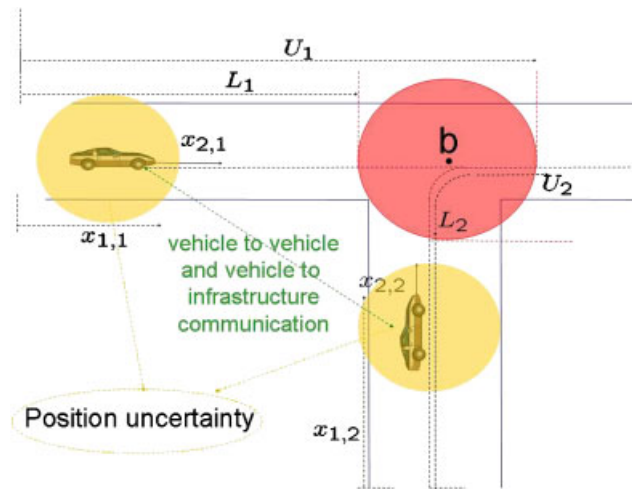

Figure 3. Vehicles converging at a traffic intersection. The bad set is defined to be the set of all vehicle 1/vehicle 2 configurations in which the vehicles are both in the ball centered at b.

system in Equations (5), that is,

$$x'_{1,i} = x_{1,i} + x_{2,i}\Delta T$$
$$x'_{2,i} = x_{2,i} + u_i\Delta T, \quad i \in \{1, 2\} \tag{34}$$

in which $x_{1,i}$ represents the position of vehicle i with respect to a coordinate axis along its path, while $x_{2,i}$ represents the longitudinal velocity of vehicle i along the same path (Figure 3). When a vehicle is inside the intersection, it cannot stop as it has to free the intersection as soon as possible, while it can stop before entering the intersection. In addition, a vehicle cannot move backwards in its lane. These constraints can be modeled by requiring that (for a suitable $x^A_{1,i}$) for $x_{1,i} \leqslant x^A_{1,i}$ then $x_{2,i} \geqslant 0$, while for $x_{1,i} > x^A_{1,i}$ we must have $x_{2,i} \geqslant v_m$ with $v_m > 0$. Let $u_m < 0 < u_M$ be lower and upper bounds for each u_i. To enforce this requirement, we assume that u_i satisfy

$$\text{when } x_{1,i} \leqslant x^A_{1,i}, \quad u_i \in \begin{cases} [0, u_M] & \text{if } x_{2,i} \leqslant 0 \\ [u_m, u_M] & \text{if } x_{2,i} > 0 \end{cases}$$

$$\text{when } x_{1,i} > x^A_{1,i}, \quad u_i \in \begin{cases} [0, u_M] & \text{if } x_{2,i} \leqslant v_m \\ [u_m, u_M] & \text{if } x_{2,i} > v_m \end{cases}$$

in which $v_m > 0$ is a lower bound on the speed. Thus, each vehicle can be described by a hybrid automaton with two modes: $q_i = q_{1,i}$ if $(x_{1,i} \leqslant x^A_{1,i}$ and $x_{2,i} \leqslant 0)$ or $(x_{1,i} > x^A_{1,i}$ and $x_{2,i} \leqslant v_m)$; $q_i = q_{2,i}$ if $(x_{1,i} \leqslant x^A_{1,i}$ and $x_{2,i} > 0)$ or $(x_{1,i} > x^A_{1,i}$ and $x_{2,i} > v_m)$. In each one of these modes, the update map f is given by Equations (34), in which $\iota(q_{1,i}) = [0, u_M]$, $\iota(q_{2,i}) = [u_m, u_M]$. Since $\mathscr{I}_D = \emptyset$, the hybrid automaton admits only autonomous mode transitions. We assume that all continuous state variables are subject to bounded uncertainty, that is, $\gamma : \mathbb{R}^2 \to 2^{\mathbb{R}^2}$ and $\gamma(y_i) = [y_i - \Delta, y_i + \Delta]$, for some $\Delta \geqslant 0$. The safety requirement is modeled by requesting that the two vehicles are never in the ball centered at b of Figure 3 at the same time. This is encoded by a bad set $B = \{x | (x_{1,1}, x_{1,2}) \in \bar{B}\}$, in which $\bar{B} = [L_1, U_1] \times [L_2, U_2]$ for suitable $L_1, U_1, L_2, U_2 \in \mathbb{R}$. The results obtained by applying Algorithms 3 and 4 are shown in Figure 4. In all simulations $\Delta T = 1$.

Trains at a railway merging. Consider two trains in the proximity of a railway merging. Assuming a second-order dynamics along their rail, each train can be modeled again as in Equations (34). However, now the input sets will be different from the previous example. In digital control mode [29], the input u_i can take four values corresponding to a 'hard-brake' mode, a 'run-out' mode, a 'constant-speed' mode, and an 'acceleration' mode. Let these four values be denoted by $\alpha, \gamma, \delta, \beta$, respectively, so that $\alpha < \gamma < \delta < \beta$. Each vehicle dynamics can thus be modeled by a hybrid automaton with four modes such that $q_i = q_{1,i}$ iff $u_i = \alpha$, $q_i = q_{2,i}$ iff $u_i = \gamma$, $q_i = q_{3,i}$ iff $u_i = \delta$, and $q_i = q_{4,i}$ iff $u_i = \beta$. There are no autonomous switches in this system, so that for each train $R_i(x_{1,i}, x_{2,i}, \sigma_i) = R_{i,2}(\sigma_i)$, where σ_i is the discrete input and can take four values, each corresponding to one of the modes. Algorithms 1 and 2 were implemented for the perfect information case and results are in Figure 5. Algorithms 3 and 4 were implemented for the imperfect information case and results are in Figure 6. For the perfect information case, Algorithms 1 and 2 provide a tight over-approximation of the capture set as demonstrated by the fact that the trajectories of the system pass very close to the bad set (Figure 5). For the imperfect information case, the resulting dynamic control law still guarantees safety but is more conservative (Figure 6). The reason is not in Algorithm 3, which is basically the same as Algorithm 1, but in Algorithm 4, in which the lower and upper bounds
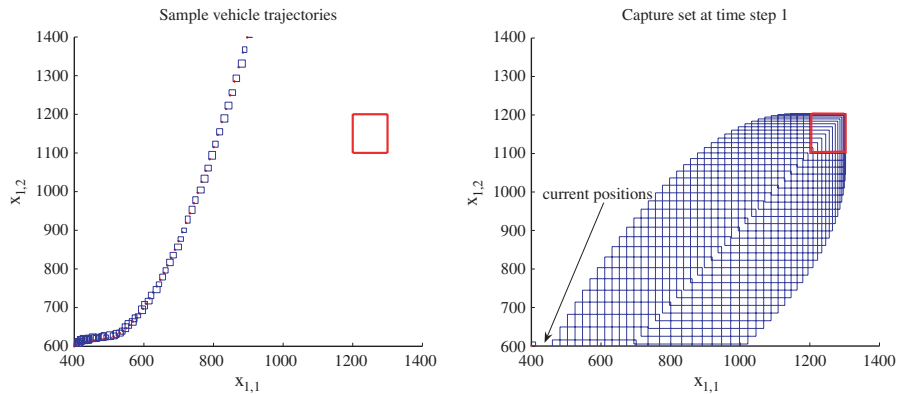
Figure 4. Vehicles at a traffic intersection (imperfect state information). Left: show a sample trajectory in the $x_{1,1}, x_{1,2}$ plane with initial conditions $x_{2,1} = x_{2,2} = 1$, and $u_1 = u_2 = 0$ is shown. The values of u_m and u_M are chosen to be $u_m = -0.2$ and $u_M = 1$. The dots represent the position of the vehicles and the rectangle surrounding them is given by the state estimator (Equations (21)). The measurement uncertainty is $\Delta = 10$. Right: show a slice of the set $\overline{C}$ at the initial time for the initial values of velocities $x_{2,1}, x_{2,2}$ is shown. The box with bold sides represents the set $\bar{B}$.
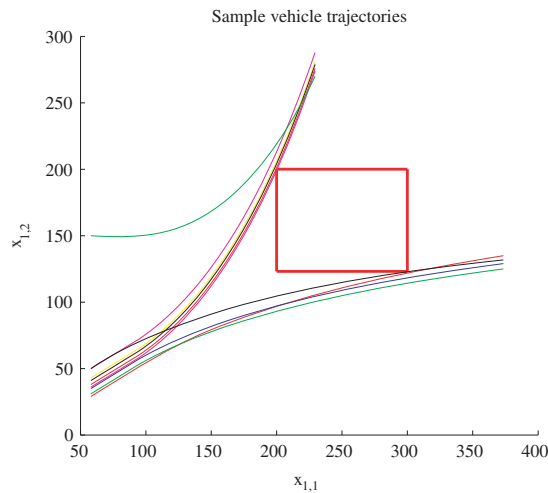

Figure 5. Trains at a railway merging (perfect state information). The box represents the set $\bar{B}$. The plot shows the $x_{1,1}, x_{1,2}$ trajectories of trains at a railway merging. Each trajectory corresponds to a different choice of initial values for $x_{1,1}, x_{1,2}, x_{2,1}, x_{2,2}, u_1, u_2$. The trajectories pass very close to the bad set. This shows that Algorithms 1 and 2 are tight.

expressions of Algorithm 3 are evaluated on the lower and upper bounds of the state estimate. In fact, Proposition 4 gives only a sufficient condition for the non-intersection of the estimate set with the capture set, but not a necessary condition.
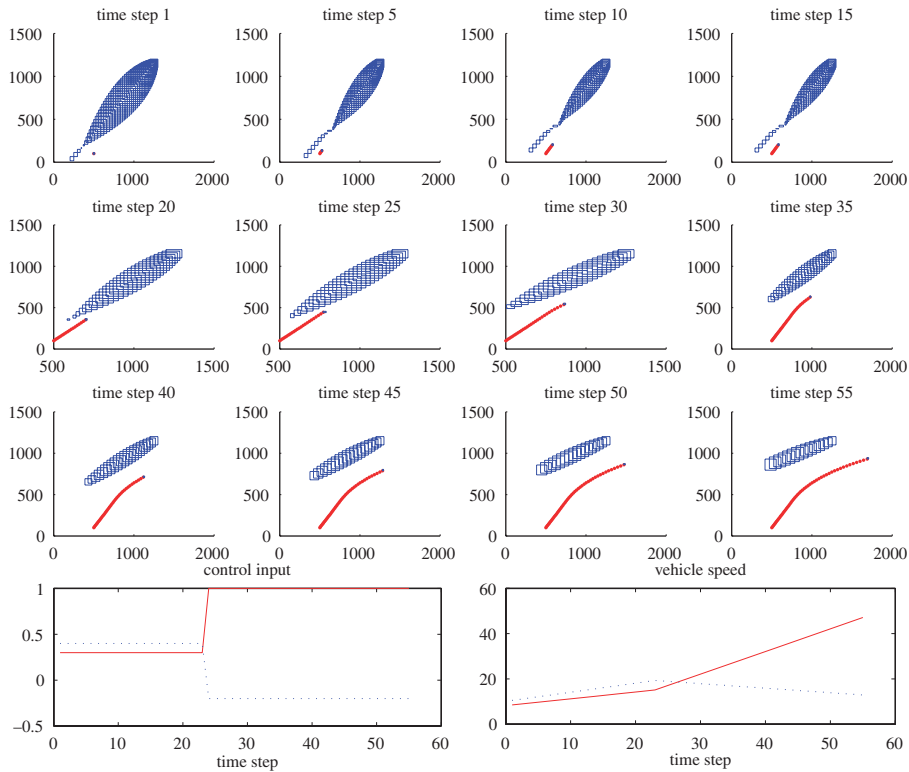
Figure 6. Trains at a railway merging (imperfect state information). This figure shows a run of Algorithm 4 at different time instants in the upper plots. The trajectory of the trains in the $x_{1,1}, x_{1,2}$ plane and the set $\bigcup_{k\geqslant 0}[\overline{L}^k(\vee\hat{x}), \overline{U}^k(\wedge\hat{x})]$ are both shown. The set $\overline{B}$ in each plot is the last up-right one in the set $\bigcup_{k\geqslant 0}[\overline{L}^k(\vee\hat{x}), \overline{U}^k(\wedge\hat{x})]$. The lower left side plot shows the control commands applied to each train. These commands are left to their initial values until at time step $t=25$ it is predicted that leaving these commands constant will cause the trajectory to enter the set $\bigcup_{k\geqslant 0}[\overline{L}^k(\vee\hat{x}), \overline{U}^k(\wedge\hat{x})]$. As a consequence, one vehicle is slowed down by braking ($u_2=-0.2$) and the other is accelerated ($u_1=+1$). The corresponding train speeds are shown in the lower right side plot.

6. CONCLUSION AND FUTURE WORK

We have proposed a dynamic feedback law for safety control in a class of triangular order preserving hybrid automata with imperfect state information. The structure of the system allowed to compute a tight over-approximation of the capture set and the dynamic control map through algorithms that have linear complexity in the number of variables.

In our future work, we plan to extend these results to continuous time hybrid automata, relax the assumptions to only require monotone flows, and state a separation principle between state estimation and control. We plan to modify Algorithm 4 so as to obtain a less conservative control map for the imperfect state information case. In addition, we will consider the more general case

in which the discrete state needs also to be estimated and the discrete state update map is not static. Unknown, bounded disturbances will also be considered to handle modeling uncertainty.

ACKNOWLEDGEMENTS

REFERENCES

1. Office of Safety Analysis. Accidents/Incidents Counts. Federal Railroad Administration, 2005. Available from: http://safetydata.fra.dot.gov/officeofsafety.
2. Laberteaux K, Caminiti L, Caveney D, Hada H. Pervasive vehicular networks for safety. *IEEE Pervasive Computing*, *Spotlight* 2006; **4**:60–62.
3. Tomlin CJ, Lygeros J, Sastry S. A game theoretic approach to controller design for hybrid systems. *Proceedings of the IEEE* 2000; **88**(7):949–970.
4. Shakernia O, Pappas GJ, Sastry S. Semi-decidable synthesis for triangular hybrid systems. In *Hybrid Systems*: *Computation and Control*, Di Benedetto MD, Sangiovanni-Vincentelli A (eds). Lecture Notes in Computer Science, vol. 2034. Springer: Berlin, 2001.
5. Maler O, Asarin E, Pnueli A. Symbolic controller synthesis for discrete and timed systems. In *Hybrid Systems II*, Antsaklis P, Kohn W, Nerode A, Sastry S (eds). Lecture Notes in Computer Science, vol. 999. Springer: Berlin, 1995; 1–20.
6. Prajna S, Jadbabaie A. Safety verification of hybrid systems using barrier certificates. In *Hybrid Systems*: *Computation and Control*, Alur R, Pappas G (eds). Lecture Notes in Computer Science, vol. 2993. Springer: Berlin, 2004; 477–492.
7. Kurzhanski AB, Varaiya P. Ellipsoidal techniques for hybrid dynamics: the reachability problem. In *New Directions and Applications in Control Theory*, Dayawansa WP, Lindquist A, Zhou Y (eds). Lecture Notes in Control and Information Sciences, vol. 321. Springer: Berlin, 2005; 193–205.
8. Tomlin CJ, Mitchell I, Bayen AM, Oishi M. Computational techniques for the verification of hybrid systems. *Proceedings of the IEEE* 2003; **91**(7):986–1001.
9. Lygeros J, Tomlin CJ, Sastry S. Controllers for reachability specifications for hybrid systems. *Automatica* 1999; **35**(3):349–370.
10. Thomas W. On the synthesis of strategies in infinite games. In *Proceedings of the STACS 95*, Mayr EW, Puech C (eds). Springer: Berlin, 1995; 1–13.
11. Ramadge PJ, Wonham WM. The control of discrete event systems. *Proceedings of the IEEE* 1989; **77**(1):81–98.
12. Heymann M, Lin F, Meyer G. Synthesis and viability of minimal interventive legal controllers for hybrid systems. *Discrete Event Dynamic Systems*: *Theory and Applications* 1998; **8**(2):105–135.
13. Asarin E, Maler O, Pnueli A. Symbolic controller design for discrete and timed systems. In *Hybrid Systems*: *Computation and Control*, Antsakilis P, Kohn W, Nerode A, Sastry S (eds). Lecture Notes in Computer Science, vol. 999. Springer: Berlin, 1995; 1–20.
14. Wong-Toi H. The synthesis of controllers for linear hybrid automata. *Conference on Decision and Control*, San Diego, 1997; 4607–4613.
15. Henzinger TA, Ho PH, Wong-Toi H. A user guide to HyTech. In *TACAS 95*: *Tools and Algorithms for the Construction and Analysis of Systems*, Brinksma E, Cleaveland W, Larsen K, Margaria T, Steffen B (eds). Lecture Notes in Computer Science, vol. 1019. Springer: Berlin, 1995; 41–71.
16. Henzinger TA, Kopke PW. Discrete-time control for rectangular hybrid automata. *Theoretical Computer Science* 1999; **221**:369–392.
17. Manna Z, Pnueli A. *The Temporal Logic of Reactive and Concurrent Systems*: *Specification*. Springer: Berlin, 1992.
18. Henzinger TA, Horowitz B, Majumdar R, Wong-Toi H. Beyond HyTech: hybrid systems analysis using interval numerical methods. In *Hybrid Systems*: *Computation and Control*, Krogh B, Lynch N (eds). Lecture Notes in Computer Science, vol. 1790. Springer: Berlin, 2000; 130–144.
19. De Wulf M, Doyen L, Raskin J-F. A lattice theory for solving games of imperfect information. In *Hybrid Systems*: *Computation and Control*, Hespanha J, Tiwari A (eds). Lecture Notes in Computer Science, vol. 3927. Springer: Berlin, 2006; 153–168.

20. Del Vecchio D. A partial order approach to discrete dynamic feedback in a class of hybrid systems. In *Hybrid Systems*: *Computation and Control*, Bemporad A, Bicchi A, Buttazzo G (eds). Lecture Notes in Computer Science, vol. 4416. Springer: Pisa, Italy, 2007; 159–173.
21. LaValle SM. *Planning Algorithms*. Cambridge University Press: Cambridge, 2006.
22. Cousot P, Cousot R. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. *Conference Record of the 4th ACM Symposium on Principles of Programming Languages*, 1977; 238–252.
23. Del Vecchio D, Murray RM, Klavins E. Discrete state estimators for systems on a lattice. *Automatica* 2006; **42**(2):271–285.
24. Del Vecchio D, Murray RM. Existence of cascade discrete-continuous state estimators on a partial order. In *Hybrid Systems*: *Computation and Control*, Morari M, Thiele L, Rossi F (eds). Lecture Notes in Computer Science, vol. 3414. Springer: Zurich, 2005; 226–241.
25. Alamoa T, Bravo JM, Redondo MJ, Camachoa EF. A set-membership state estimation algorithm based on DC programming. *Automatica* 2008; **44**:216–224.
26. Davey BA, Priesteley HA. *Introduction to Lattices and Order*. Cambridge University Press: Cambridge, 2002.
27. Verma R, Del Vecchio D, Fathy H. Development of a scaled vehicle with longitudinal dynamics of a HMMWV for an its testbed. *IEEE/ASME Transactions on Mechatronics* 2008; **13**(1):46–57.
28. Henzinger TA. The theory of hybrid automata. *Proceedings of the 11th Annual Symposium on Logic in Computer Science*. IEEE Press: New York, 1996; 278–292.
29. Pachl J. *Railway Operation and Control*. VTD Rail Publishing: Mountlake Terrace, 2002.