Synthesizing Controllers for Nonlinear Hybrid Systems*

Claire Tomlin, John Lygeros, and Shankar Sastry

Department of Electrical Engineering and Computer Sciences
University of California,
Berkeley CA 94720
clairet, lygeros, sastry@eecs.berkeley.edu

Abstract. Motivated by an example from aircraft conflict resolution we seek a methodology for synthesizing controllers for nonlinear hybrid automata. We first show how game theoretic methodologies developed for this purpose for finite automata and continuous systems can be cast in a unified framework. We then present a conceptual algorithm for extending them to the hybrid setting. We conclude with a discussion of computational issues.

1 Introduction

In the first part of this paper we show that verification of the safety of continuous nonlinear systems using the Hamilton-Jacobi equation may be considered as the continuous analog of infinite games on finite automata. In the second part we present a conceptual algorithm for calculating maximal controlled invariant sets for nonlinear hybrid systems and we present a motivating example: we describe an iteration process to calculate the maximal set of safe initial conditions for a two-aircraft maneuver. We conclude with a brief discussion of computational issues. The calculation of the control scheme is implicit in the construction of the controlled invariant set.

The idea of posing the controller synthesis problem as a discrete game between the system and its environment is attributed to Church [1], who was studying solutions to digital circuits. The solution to this problem using a version of the von Neumann-Morgenstern discrete game [2] is due to Büchi and Landweber [3] and Rabin [4]. [5] also discusses games on automata. A comprehensive modern survey of infinite discrete games on automata is presented in [6] and [7]. Controller synthesis on timed automata was first developed in [8] and [9]. An algorithm for controller synthesis on linear hybrid automata is presented in [10]. The notion of control invariance for continuous systems is described in [11], and control invariance for hybrid systems is discussed in [12].

The study of differential equations in game theory was first motivated by military problems in the U.S. Air Force (aircraft dog fights, target missiles) and

* Research supported by NASA under grant NAG 2-1039, by the California PATH program under MOU-238 and MOU-288, and by a Zonta Postgraduate Fellowship.

was initially developed by Isaacs in the 1940's and 50's [13]. An excellent modern reference is [14]. Our motivation for this work arose out of attempting to verify the safety of a class of conflict resolution maneuvers for aircraft, in [15]. Similar previous work is that of [16], in which game theoretic methods were used to prove safety of a set of maneuvers for Automated Highway Systems.

Let us first introduce some basic notation. Let PC^0 denote the space of piecewise continuous functions over $\mathbb{R}$, and PC^1 the space of piecewise differentiable functions over $\mathbb{R}$.

Entity	**Discrete**	**Continuous**
State Space	Q	$\mathbb{R}^n$
Input Sets	$\Sigma_0 \times \Sigma_1$	$U \times D$
Input Space	$\Sigma_0^\omega \times \Sigma_1^\omega$	$\mathcal{U} \times \mathcal{D} \subset PC^0 \times PC^0$
Transition Relation	$\delta : Q \times \Sigma_0 \times \Sigma_1 \to 2^Q$	$f : \mathbb{R}^n \times \mathcal{U} \times \mathcal{D} \to \mathbb{R}^n$:
		$\forall \tau, \dot{x}(\tau) = f(x(\tau), u(\tau), d(\tau))$
System Trajectory	$(\gamma, s_0, s_1) \in Q^\omega \times \Sigma_0^\omega \times \Sigma_1^\omega$: $\gamma[i+1] \in \delta(\gamma[i], s_0[i], s_1[i])$	$(x(\cdot), u(\cdot), d(\cdot)) \in PC^1 \times \mathcal{U} \times \mathcal{D}$: $\forall \tau, \dot{x}(\tau) = f(x(\tau), u(\tau), d(\tau))$
Acceptance Conditions	$\Box F$; $\Diamond G$	$\forall \tau, x(\tau) \in F$; $\exists \tau, x(\tau) \in G$

2 Verifying safety in continuous systems: a comparison with discrete $\Box$- and $\Diamond$-games

2.1 Infinite Games on Finite Automata

We summarize a class of two-player games on finite automata, in which the goal of Player 0 is to force the system to remain inside a certain "good" subset of the state space, and the goal of Player 1 is to force the system to leave this same subset. We describe the iteration process for calculating the set of states from which Player 0 can always win, and the set of states from which Player 1 can always win. We then show how this iteration process can be written as a difference equation for a *value* function, similar to the Hamilton-Jacobi equation for differential games on continuous systems.

System Definition and Winning Condition We consider two players P_0 and P_1, playing over a *game automaton* of the form:

$$(Q, \Sigma, \delta, Q_0, \Omega) \tag{1}$$

where Q is a finite set of *states*, Σ is a finite set of *actions*, $\delta : Q \times \Sigma \to 2^Q$ is a *partial transition relation*, $Q_0 \subseteq Q$ is a set of *initial states*, and Ω is a *trajectory acceptance condition*. The set of actions is the product of two sets $\Sigma = \Sigma_0 \times \Sigma_1$ where Σ_i contains the action of P_i, so that each transition between states depends on a joint action (σ_0, σ_1) of P_0 and P_1. In what follows, Σ_0 will be the set of actions of the controller, and Σ_1 will be the set of actions of the environment (or disturbance).

A *system trajectory* is an infinite sequence of states and actions, $(\gamma, s_0, s_1) \in Q^\omega \times \Sigma_0^\omega \times \Sigma_1^\omega$, which satisfies:

$$\gamma[0] \in Q_0 \text{ and } \gamma[i+1] \in \delta(\gamma[i], s_0[i], s_1[i]) \tag{2}$$

We will consider two kinds of trajectory acceptance conditions: $\Omega = (\Box F)$ (meaning that $\forall i, \gamma[i] \in F$), and its dual $\Omega = (\Diamond G)$ (meaning that $\exists i, \gamma[i] \in G$), where F and G are subsets of Q. P_0 wins the game if the trajectory satisfies $\Box F$, otherwise P_1 wins. To illustrate the duality between the two kinds of acceptance conditions we assume that P_1 wins the game $\Omega = (\Diamond G)$ if the trajectory satisfies $\Diamond G$.

State Space Partition Consider the acceptance condition $\Omega = (\Box F)$. The *winning states* for P_0 are those states $W^* \subseteq F$ from which P_0 can force the system to stay in F. The set W^* can be calculated as the fixed point of the following iteration (using a negative index $i \in \mathbb{Z}_-$ to indicate that each step is a predecessor operation):

$$\begin{aligned} W^0 &= F \\ W^{i-1} &= W^i \cap \{q \in Q \mid \exists \sigma_0 \in \Sigma_0 \; \forall \sigma_1 \in \Sigma_1 \; \delta(q, (\sigma_0, \sigma_1)) \subseteq W^i\} \end{aligned} \tag{3}$$

The iteration terminates when $W^i = W^{i-1} \triangleq W^*$. At each step of the iteration, the set W^i contains those states for which P_0 has a sequence of actions which will ensure that the system remains in F for at least i steps, for all possible actions of P_1.

Now consider the acceptance condition $\Omega = (\Diamond G)$. The *winning states* for P_1 are those states $V^* \supseteq G$ from which P_1 can force the system to visit G. It can be calculated iteratively by:

$$\begin{aligned} V^0 &= G \\ V^{i-1} &= V^i \cup \{q \in Q \mid \exists \sigma_1 \in \Sigma_1 \; \forall \sigma_0 \in \Sigma_0 \; \delta(q, (\sigma_0, \sigma_1)) \subseteq V^i\} \end{aligned} \tag{4}$$

terminating when $V^i = V^{i-1} \triangleq V^*$. Here, V^i contains those states for which P_1 has a sequence of actions which will ensure that the system touches G in at most i steps, for all possible actions of P_0.

The Value Function For the acceptance condition $\Omega = (\Box F)$, we inductively define a *value function*:

$$J(q, i) : Q \times \mathbb{Z}_- \to \{0, 1\} \tag{5}$$

by:

$$J(q, i) = \begin{cases} 1 & q \in W^i \\ 0 & q \in (W^i)^c \end{cases} \tag{6}$$

In other words, $W^i = \{q \in Q \mid J(q,i) = 1\}$. Recall that P_0 is trying to keep the system in F while P_1 is trying to force the system to leave F. Therefore,

$$\max_{\sigma_0}\min_{\sigma_1}\min_{q' \in \delta(q,\sigma_0,\sigma_1)} J(q',i) = \begin{cases} 1 \text{ if } \exists\sigma_0 \text{ that } \forall\sigma_1, \delta(q,\sigma_0,\sigma_1) \subseteq W^i \\ 0 \text{ otherwise} \end{cases} \tag{7}$$

The $\min_{q' \in \delta(q,\sigma_0,\sigma_1)}$ in the above compensates for the nondeterminism in δ, and the notation $\max_{\sigma_0}\min_{\sigma_1}$ means that P_0 *plays first*, trying to maximize the minimum value of $J(\cdot)$. P_1 has the advantage in this case, since it has "prior" knowledge of P_0's action when making its own choice. Therefore, in general,

$$\max_{\sigma_0}\min_{\sigma_1}\min_{q' \in \delta(q,\sigma_0,\sigma_1)} J(\cdot) \le \min_{\sigma_1}\max_{\sigma_0}\min_{q' \in \delta(q,\sigma_0,\sigma_1)} J(\cdot) \tag{8}$$

with equality occurring when the action (σ_0, σ_1) is a *saddle solution*, or a *no regret* solution for each player. Here we do not need to assume the existence of a saddle solution, rather we always give advantage to P_1, the player doing its worst to drive the system out of F.

The iteration process (3) may be summarized by the difference equation:

$$J(q,i-1) - J(q,i) = \min\{0, \max_{\sigma_0}\min_{\sigma_1}[\min_{q' \in \delta(q,\sigma_0,\sigma_1)} J(q',i) - J(q,i)]\} \tag{9}$$

which describes the relationship between the change in $J(\cdot)$ due to one step of the iteration and the change in $J(\cdot)$ due to one state transition. The first "min" in equation (9) prevents states outside W^i that can be forced by P_0 to transition into W^i from appearing in W^{i-1}.

To calculate the set of winning states W^* for P_0 we iterate equation (9) until a fixed point is reached, i.e. until for all $q \in Q$, $J(q,i-1) = J(q,i) \triangleq J^*(q)$.

Proposition 1 (Winning States for P_0) *A fixed point $J^*(q)$ of (9) is reached in a finite number of iterations. The set of winning states for P_0 is $W^* = \{q \in Q \mid J^*(q) = 1\}$.*

Definition 1 (Σ_0-controlled invariant set) *A subset $W \subseteq Q$ is called Σ_0-*controlled invariant *if $\forall s_1 \in \Sigma_1^\omega$, $\exists s_0 \in \Sigma_0^\omega$ such that for the system trajectory $(\gamma, s_0, s_1) \in Q^\omega \times \Sigma_0^\omega \times \Sigma_1^\omega$, γ remains in W.*

Proposition 2 (Characterization of W^*) *W^* is the largest Σ_0-controlled invariant subset of F.*

A feedback controller for σ_0 that renders W^* invariant can now be constructed. For all $q \in W^*$ the controller allows only the $\sigma_0 \in \Sigma_0$ for which:

$$\min_{\sigma_1}\min_{q' \in \delta(q,\sigma_0,\sigma_1)} J^*(q') = 1$$

Existence of such σ_0 for all $q \in W^*$ is guaranteed by construction. This control scheme is in fact "least restrictive".

An algorithm for calculating V^* can be constructed similarly. If $G = F^c$, the second game ($\Omega = (\Diamond G)$) is the *dual* of the first game ($\Omega = (\Box F)$) in the sense that if the sequence of actions $(s_0, s_1) \in \Sigma_0^\omega \times \Sigma_1^\omega$ of the first game is a *saddle* or *no regret* solution, then $V^* = (W^*)^c$.

2.2 Dynamic Games on Nonlinear Continuous Systems

Consider now the dynamic counterpart of the above class of discrete games: two-player zero-sum dynamic games on nonlinear continuous-time systems. The acceptance conditions considered here correspond to a class of dynamics games known as *pursuit-evasion* games. Player 0 wins if it can keep the system from entering a "bad" subset of the state space, called the *capture set*. Player 1 wins if it can drive the state into the bad set (if it can capture Player 0). As in the previous section, we describe the calculation of the set of states from which Player 0 can always win.

System Definition and Winning Condition As in the discrete case, we consider two players P_0 and P_1, but now over nonlinear systems of the form

$$\dot{x}(t) = f(x(t), u(t), d(t)) \tag{10}$$

where $x \in \mathbb{R}^n$ is the finite-dimensional *state space*, $u \in U \subseteq \mathbb{R}^u$ is the *control input* which models the actions of P_0, $d \in D \subseteq \mathbb{R}^d$ is the *disturbance input* which models the actions of P_1, and f is a smooth vector field over $\mathbb{R}^n$. The input set $U \times D$ is the analog of the partition $\Sigma_0 \times \Sigma_1$ of the discrete game. The space of acceptable control and disturbance trajectories are denoted by $\mathcal{U} = \{u(\cdot) \in PC^0 \mid u(\tau) \in U \; \forall \tau \in \mathbb{R}\}, \mathcal{D} = \{d(\cdot) \in PC^0 \mid d(\tau) \in D \; \forall \tau \in \mathbb{R}\}$.

A *system trajectory* over an interval $I \subseteq \mathbb{R}$ is a map:

$$(x(\cdot), u(\cdot), d(\cdot)) : I \to \mathbb{R}^n \times U \times D \tag{11}$$

such that $u(\cdot) \in \mathcal{U}$, $d(\cdot) \in \mathcal{D}$, $x(\cdot)$ is continuous and $\forall \tau \in I$ where $u(\cdot)$ and $d(\cdot)$ are continuous, $\dot{x}(\tau) = f(x(\tau), u(\tau), d(\tau))$. We assume that the function f is *globally Lipschitz* in x and continuous in u and d. Then, by the existence and uniqueness theorem of solutions for ordinary differential equations, given an interval I, the value of $x(t)$ for some $t \in I$ and input and disturbance trajectories $u(\tau), d(\tau)$ over I there exists a unique solution $x(\cdot), u(\cdot), d(\cdot)))$ to (10).

We define the capture set as a region G by $G = \{x \in \mathbb{R}^n | l(x) < 0\}$ with boundary $\partial G = \{x \in \mathbb{R}^n | l(x) = 0\}$ where $l : \mathbb{R}^n \to \mathbb{R}$ is a differentiable function of x and $Dl(x) \neq 0$ on ∂G. Defining $F = G^c$, we say P_0 wins the game if for all $\tau \in \mathbb{R}$, $x(\tau) \in F$.

State Space Partition The winning states for P_0 are those states $W^* \subseteq \mathbb{R}^n$ from which P_0 can force the system to stay in $F = G^c$. Define the outward pointing normal to G as:

$$\nu = Dl(x) \tag{12}$$

The states on ∂G which can be forced into G infinitesimally constitute the *usable part* (UP) of ∂G[14]. They are the states for which the disturbance can force the vector field to point inside G:

$$\mathrm{UP} = \{x \in \partial G \mid \forall u \exists d \quad \nu^T f(x, u, d) < 0\} \tag{13}$$

Figure 1 displays a simple example, with the UP of ∂G shown in bold.

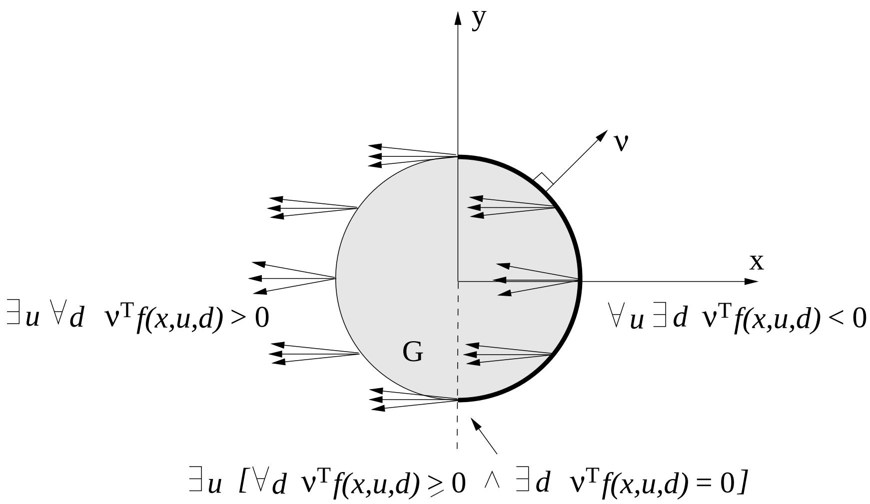


Fig. 1. The capture set G, its outward pointing normal ν, and the cones of vector field directions at points on ∂G.

The Value Function and Hamilton-Jacobi equation Consider the system (10) over the time interval $[t, 0]$, where $t < 0$. The value function of the game is defined by:

$$J(x, u(\cdot), d(\cdot), t) : \mathbb{R}^n \times \mathcal{U} \times \mathcal{D} \times \mathbb{R}_- \to \mathbb{R} \tag{14}$$

such that $J(x, u(\cdot), d(\cdot), t) = l(x(0))$. This value function may be interpreted as the cost of a trajectory $x(\cdot)$ which starts at x at time $t \leq 0$, evolves according to (10) with input $(u(\cdot), d(\cdot))$, and ends at the final state $x(0)$. Note that the value function depends only on the final state: there is no running cost, or *Lagrangian*. This encodes the fact that we are only interested in whether or not the system trajectory ends in G and are not concerned with intermediate states. The game is won by P_1 if the terminal state $x(0)$ belongs inside G (i.e. $J < 0$), and is won by P_0 otherwise.

Let:

$$u^* = \arg \max_{u \in \mathcal{U}} \min_{d \in \mathcal{D}} J(x, u(\cdot), d(\cdot), t) \tag{15}$$

$$J^*(x, t) = \max_{u \in \mathcal{U}} \min_{d \in \mathcal{D}} J(x, u(\cdot), d(\cdot), t) \tag{16}$$

Thus, the set $\{x : J^*(x, t) \geq 0\}$ contains the states for which the system will stay in $F = G^c$ for at least $\mid t \mid$ seconds, regardless of the disturbance d. The continuous-time analog to (3), the iterative method of calculating the winning states for P_0 is therefore:

$$\begin{aligned} W^0 &= G^c \\ W^t &= \{x | J^*(x, t) \geq 0\} \end{aligned} \tag{17}$$

This "iteration" terminates if there exists a $t^* < 0$ such that for all $t < t^*$, $W^t = W^{t^*}$.

We compute $J^*(x,t)$ using standard results in optimal control theory. First, define the *Hamiltonian* of the system as:

$$H(x,p,u,d) = p^T f(x,u,d) \tag{18}$$

where p is a vector in $\mathbb{R}^n$ called the *costate* and is equal to ν at the boundary of G. The optimal Hamiltonian is given by:

$$H^*(x,p) = \max_{u \in U} \min_{d \in D} H(x,p,u,d) \tag{19}$$

If $J^*(x,t)$ is a smooth function of x and t, then it may be calculated for all x and t using the following partial differential equation, known as the *Hamilton-Jacobi equation*:

$$-\frac{\partial J^*(x,t)}{\partial t} = \min\{0, H^*(x, \frac{\partial J^*(x,t)}{\partial x})\} \tag{20}$$

with boundary condition $J^*(x,0) = l(x)$. The derivation of equation (20) may be found in most textbooks on optimal control, for example, see [17]. We added the "min" to the right hand side of (20) for the same reason as in the discrete case: we want to ensure that only the UP of ∂G is propagated backwards, so that states which are once unsafe cannot become safe. Equation (20) is the continuous analog to equation (9) of the preceding discrete game, and describes the relationship between the time and state evolution of $J^*(x,t)$.

Proposition 3 (Winning States for P_0) *If (17) reaches a fixed point at time t^*, then the set of winning states for P_0 is $W^* = \{x | J^*(x,t^*) \geq 0\}$. Otherwise, $W^* = \{x | J^*(x,-\infty) \geq 0\}$. In both cases, $J^*(x,t)$ is the solution of equation (20).*

Definition 2 (U-controlled invariant set) *A subset $W \subseteq \mathbb{R}^n$ is called U-*controlled invariant *if $\exists u(\cdot) \in \mathcal{U}$ such that $\forall d(\cdot) \in \mathcal{D}$, $x(\cdot)$ remains in W for the trajectory $(x(\cdot), u(\cdot), d(\cdot))$.*

Proposition 4 (Characterization of W^*) *W^* is the largest U-controlled invariant set contained in $F = G^c$.*

A feedback controller for u that renders W^* invariant can now be constructed. The controller should be such that on ∂W^* only the u for which:

$$\min_{d \in D} \left(\frac{\partial J^*(x,-\infty)}{\partial x} \right)^T f(x,u,d) \geq 0$$

are applied. In the interior of W^* u is free to take on any value in U. Existence of such u's for $x \in W^*$ is guaranteed by construction. This scheme is in fact least restrictive.

3 Controller synthesis for nonlinear hybrid systems

Nonlinear Hybrid Automata A *hybrid automaton* is a tuple: $H = ((Q \times X), (U \times D), (\Sigma_0 \times \Sigma_1), f, \delta, Inv, (Q_0 \times X_0), \Omega)$ where Q is a finite set of locations, $X = \mathbb{R}^n$, $U \subseteq \mathbb{R}^u$, $D \subseteq \mathbb{R}^d$, $\Sigma = \Sigma_0 \times \Sigma_1$ a finite set of actions, $f : Q \times X \times U \times D \rightarrow \mathbb{R}^n$, $\delta : Q \times X \times \Sigma_0 \times \Sigma_1 \rightarrow 2^{Q \times X}$, $Inv \subseteq Q \times X$, $Q_0 \times X_0 \subseteq Q \times X$ is a subset of initial states, and Ω is an acceptance condition (here $\Omega = (\Box F)$ or $\Omega = (\Diamond G)$ for $F, G \subseteq Q \times X$).

The variables of the hybrid automaton evolve continuously as well as in discrete jumps. A *hybrid time trajectory*, τ, is a finite or infinite sequence of intervals $\tau = \{I_i\}$ satisfying:

- I_i is closed unless τ is finite and I_i is the last interval in the sequence, in which case I_i can be right open.
- Let $I_i = [\tau_i, \tau_i']$. Then $\tau_0 = 0$ and for all i, $\tau_i = \tau_{i-1}'$, $\tau_i \leq \tau_i'$.

We denote by $\mathcal{T}$ the set of all hybrid time trajectories.

A *system trajectory* is a collection $(\tau, (\gamma(\cdot), x(\cdot)), (u(\cdot), d(\cdot)), (s_0, s_1))$ where $\tau \in \mathcal{T}$, $\gamma : \tau \rightarrow Q$, $x(\cdot) : \tau \rightarrow X$, $u(\cdot) : \tau \rightarrow U$, $d(\cdot) : \tau \rightarrow D$, $s_0 \in \Sigma_0^\omega$ and $s_1 \in \Sigma_1^\omega$ and:

- **Initial Condition**: $(\gamma(\tau_0), x(\tau_0)) \in Q_0 \times X_0$
- **Discrete Evolution**: for all i, $(\gamma(\tau_{i+1}), x(\tau_{i+1})) \in \delta((\gamma(\tau_i'), x(\tau_i'), (s_0[i], s_1[i]))$.
- **Continuous Evolution** if $\tau_i' > \tau_i$, then for all $t \in [\tau_i, \tau_i']$, $\gamma(t) = \gamma(\tau_i)$, $\dot{x}(t) = f((\gamma(t), x(t)), (u(t), d(t)))$ and $(\gamma(t), x(t)) \in Inv$.

To ensure that the laws for continuous evolution are meaningful we impose the same assumption on f as in the previous section.

Calculating the Maximal Safe Set Consider the acceptance condition $\Omega = (\Box F)$. We again seek to construct the largest set of states for which the control (in this case both u and σ_0) can guarantee that the acceptance condition Ω is met despite the action of the disturbance (in this case d and σ_1). For any set $K \subseteq Q \times X$ define the controllable and uncontrollable predecessors of K by:

$$\begin{aligned} Pre_0(K) &= \{(q,x) \in Q \times X | \exists \sigma_0 \in \Sigma_0 \; \forall \sigma_1 \in \Sigma_1 \; \delta((q,x),(\sigma_0,\sigma_1)) \subseteq K\} \cap K \\ Pre_1(K) &= \{(q,x) \in Q \times X | \forall \sigma_0 \in \Sigma_0 \; \exists \sigma_1 \in \Sigma_1 \; \delta((q,x),(\sigma_0,\sigma_1)) \cap K^c \neq \emptyset\} \cup K^c \end{aligned} \tag{21}$$

In other words, the controllable predecessor of K, $Pre_0(K)$, contains all states in K for which the controllable actions can force the state to remain in K for at least one step in the discrete evolution. The uncontrollable predecessor, on the other hand, contains all states in K^c as well as all states from which the uncontrollable actions may be able to force the state outside K. Clearly:

Proposition 5 *$Pre_0(K) \cap Pre_1(K) = \emptyset$.*

Consider the algorithm:

Initialization: $W^0 = F$, $W^{-1} = \emptyset$, $i = 0$.

While $W^i \neq W^{i-1}$ **do**

$$W^{i-1} = W^i \setminus \{(q,x) \in Q \times X | \forall u \in \mathcal{U} \exists t \geq 0, d \in \mathcal{D} \text{ such that } (\gamma(t), x(t)) \in Pre_1(W^i) \text{ and } (\gamma(\tau), x(\tau)) \notin Pre_0(W^i)\}$$

$$i = i - 1$$

end

Here $(\gamma(\tau), x(\tau))$ for $\tau \in [0,t]$ represents the continuous trajectory starting at (q,x) under inputs (u,d), i.e. $(\gamma(0), x(0)) = (q,x)$ and for all τ, $\gamma(\tau) = q$, $\dot{x}(\tau) = f((\gamma(\tau), x(\tau)), (u(\tau), d(\tau)))$, and $(\gamma(\tau), x(\tau)) \in Inv$.

The most challenging part of each step of the algorithm is the computation of the set of states that can be driven by d to $Pre_1(W^i)$ without first entering $Pre_0(W^i)$. This computation can be carried out by appropriately modifying the Hamilton-Jacobi construction of Section 2.2.

4 Example

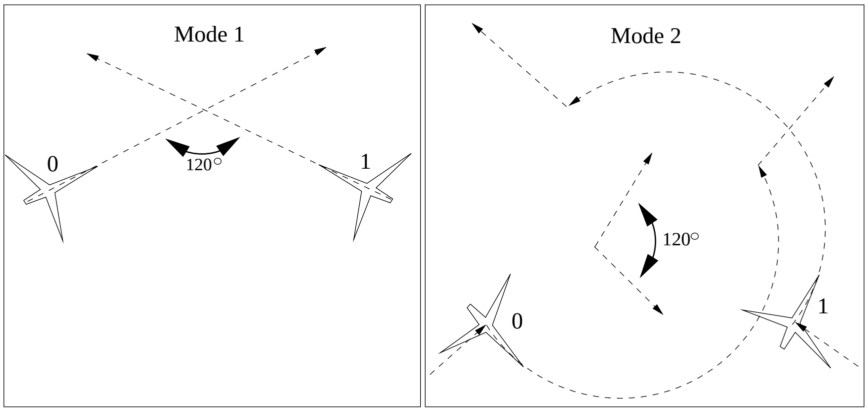


Fig. 2. Two aircraft in two modes of operation: in mode 1 the aircraft follow a straight course and in mode 2 the aircraft follow a half circle. The initial relative heading (120°) is preserved throughout.

Consider a variation of the two aircraft collision avoidance problem of [15], in which there are two modes of operation: a *cruise* mode in which both aircraft follow a straight path, and an *avoid* mode in which both aircraft follow a circular arc path. The protocol of the maneuver is that as soon as the aircraft are within a distance α miles of each other, each aircraft turns 90° to its right and follows a half circle. Once the half circle is complete, each aircraft returns to its original heading and continues on its straight path (Figure 2).

In each mode, the continuous dynamics may be expressed in terms of the *relative motion* of the two aircraft (equivalent to fixing the origin of the relative frame on aircraft 0 and studying the motion of aircraft 1 with respect to aircraft 0):

$$\begin{aligned} \dot{x}_r &= -v_0 + v_1 \cos \psi_r + \omega_0 y_r \\ \dot{y}_r &= v_1 \sin \psi_r - \omega_0 x_r \\ \dot{\psi}_r &= \omega_1 - \omega_0 \end{aligned} \tag{22}$$

in which $(x_r, y_r, \psi_r) \in \mathbb{R}^2 \times [-\pi, \pi]$ is the relative position and orientation of aircraft 1 with respect to aircraft 0, and v_i and ω_i are the linear and angular velocities of each aircraft. In the cruise mode $\omega_i = 0$ for $i = 0, 1$ and in the avoid mode $\omega_i = 1$ for $i = 0, 1$. The control input is defined to be the linear velocity of aircraft 0, $u = v_0 \in U$, and the disturbance input as that of aircraft 1, $d = v_1 \in D$, where U and D denote the range of possible linear velocities of each aircraft. Such a situation could arise, for example, in an airborne collision avoidance algorithm in which the flight management system of aircraft 0 wishes to compute the parameters v_0 and α of its avoidance maneuver and can only predict the velocity of aircraft 1 to within some uncertainty.

The discrete state takes on three possible values ($Q = \{q_1, q_2, q_3\}$). The state q_1 corresponds to cruising before the avoid maneuver, q_2 corresponds to the avoid mode and q_3 corresponds to cruising after the avoid maneuver has been completed. There are two transitions. The first (σ_0) corresponds to the initiation of the avoid maneuver and can be effectively controlled by choosing the range, α, at which the aircraft start turning. The second transition (σ_1)corresponds to the completion of the avoid maneuver. This transition is required to take place after the aircraft have completed a half circle: the continuous state space is augmented with a timer $z \in \mathbb{R}$ to force this transition. Let $x = (x_r, y_r, \psi_r, z)$.

Safety is defined in terms of the relative distance between the two aircraft: throughout the maneuver the aircraft must remain at least 5 nautical miles apart. We define the region at which *loss of separation* occurs as a 5-mile-radius cylinder around the origin in the (x_r, y_r, ψ_r, z) space:

$$G = \{q_1, q_2, q_3\} \times \{x \mid x_r^2 + y_r^2 < 5^2\} \tag{23}$$

This region is referred to in the pursuit-evasion game literature as the *capture set*.

The dynamics of the maneuver can be encoded by the automaton of Figure 3. Let ϵ represent *any action* of either player 0 or player 1.

$$\begin{aligned} Q \times X &= \{q_1, q_2, q_3\} \times (\mathbb{R}^2 \times [-\pi, \pi] \times \mathbb{R}) \\ U \times D &= \mathbb{R}^2 \\ \Sigma_0 \times \Sigma_1 &= \{\sigma_0, \epsilon\} \times \{\sigma_1, \epsilon\} \end{aligned}$$

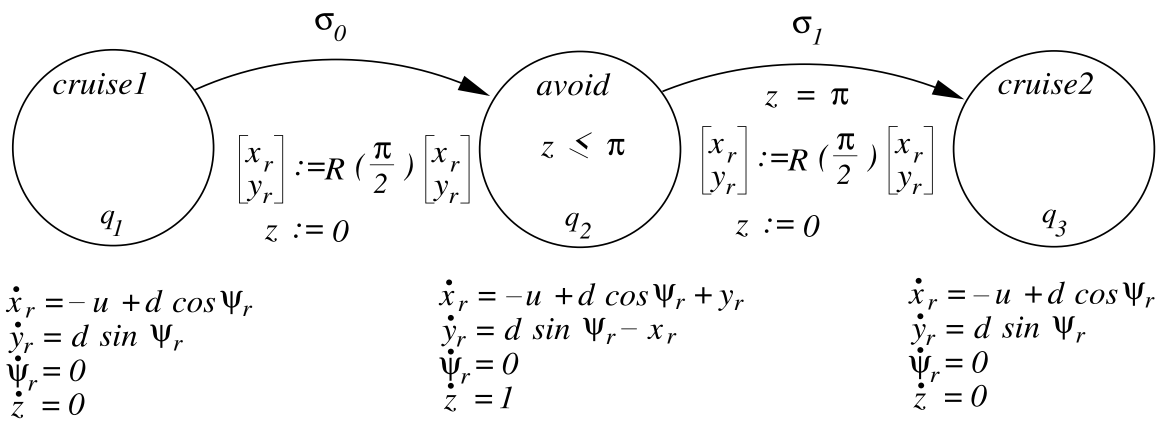


Fig. 3. In q_1 the aircraft follow a straight course, in q_2 the aircraft follow a half circle; in q_3 the aircraft return to a straight course.

$$
\begin{aligned}
f(q, x, u, d) &= \begin{bmatrix} -u + d\cos\psi_r + \omega_0 y_r \\ d\sin\psi_r - \omega_0 x_r \\ \omega_1 - \omega_0 \end{bmatrix} \quad \text{where } \omega_0 = 1 \text{ if } q = q_2, \omega_0 = 0 \text{ otherwise} \\
\delta\,(q_1, x, (\sigma_0, \epsilon)) &= \left(q_2, \begin{bmatrix} R(\pi/2)\begin{bmatrix} x_r \\ y_r \end{bmatrix} \\ \psi_r \\ 0 \end{bmatrix} \right) \\
\delta\left(q_2, \begin{bmatrix} x_r \\ y_r \\ \psi_r \\ \pi \end{bmatrix}, (\epsilon, \sigma_1) \right) &= \left(q_3, \begin{bmatrix} R(\pi/2)\begin{bmatrix} x_r \\ y_r \end{bmatrix} \\ \psi_r \\ 0 \end{bmatrix} \right) \\
\delta(q, x, \sigma) &= \emptyset \text{ otherwise} \\
Inv &= (q_1, X) \cup (q_2, \{x \mid 0 \le z \le \pi\}) \cup (q_3, X) \\
Q_0 \times X_0 &= (q_1, \{x \mid x_r^2 + y_r^2 > 5^2, z = 0\}) \\
\Omega &= \Box G^c
\end{aligned}
$$

It is easy to show that:

Proposition 6 *If* $q \in \{q_1, q_3\}$ *then* $z = 0$.

Following the algorithm of Section 3 we compute W^*, starting from $W^0 = G^c$. Let $T = \{x \mid x_r^2 + y_r^2 < 5^2\}$.

$$
\begin{aligned}
W^0 &= [(q_1, T \cap \{x \mid z = 0\}) \cup (q_2, T) \cup (q_3, T \cap \{x \mid z = 0\})]^c \\
Pre_0(W^0) &= (q_1, T^c \cap \{x \mid z = 0\}) \\
Pre_1(W^0) &= G
\end{aligned}
$$

The set W^{-1} is shown in Figure 5. W^{-2} involves computing $Reach(Pre_1(W^{-1}), Pre_0(W^{-1}))$,

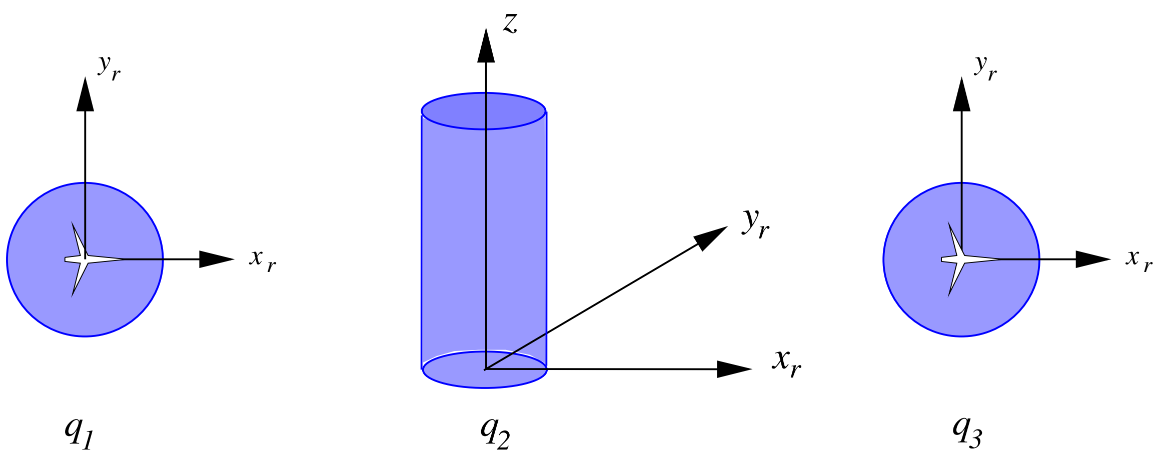


Fig. 4. $(W^0)^c$

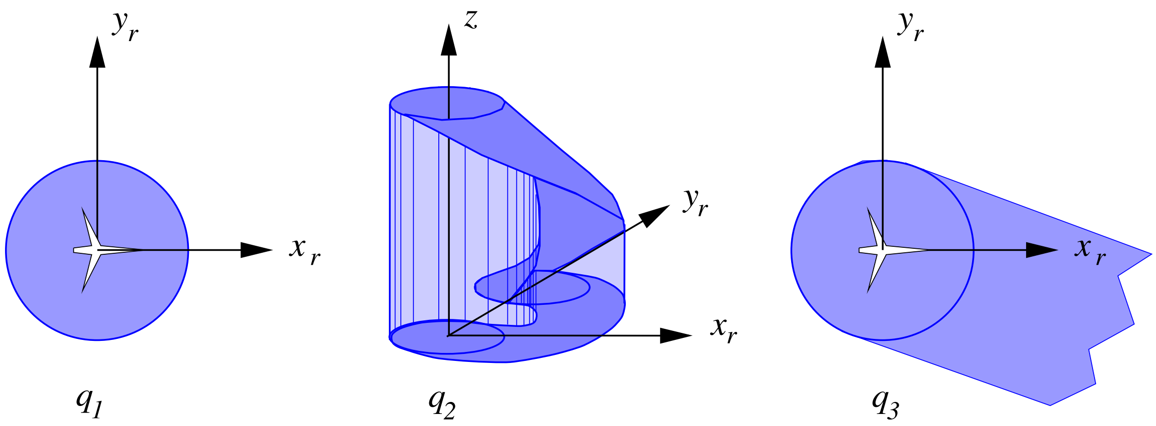


Fig. 5. $(W^{-1})^c$. The jagged edge in q_3 means that the set extends infinitely.

which is shown in Figure 6.

The controller for σ_0 is designed as illustrated in Figure 7(a). The transition σ_0 is not enabled until the system dynamics in q_1 reach the dashed line as shown; at this point σ_0 is enabled, and the transition to state q_2 may occur at any time. The transition is forced to occur before the system reaches the solid boundary of W^*. Note that there are states (x_r, y_r) which are not rendered safe by the maneuver. Indeed, in q_1, if the initial state is in the dark region, then the aircraft are doomed to collide. Figure 7(b) displays the result of increasing the radius of the turn in q_2. Notice that the set W^* increases as the turning radius increases. This implies that the maneuver renders a larger subset of the state space safe. Figure 7(b) shows the critical value of the turning radius, for which the maneuver is guaranteed to be safe, provided the conflict is detected early enough.

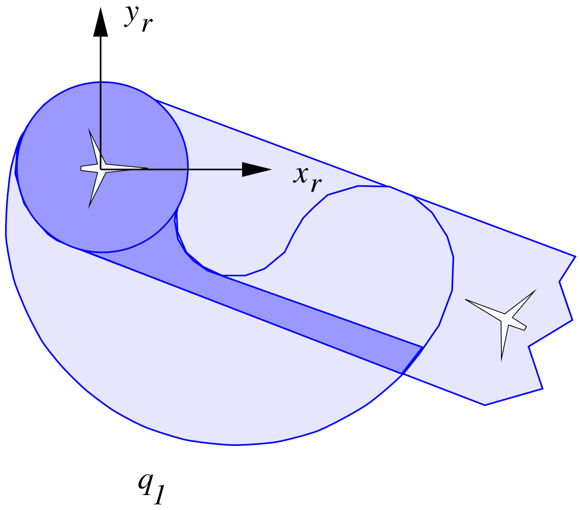


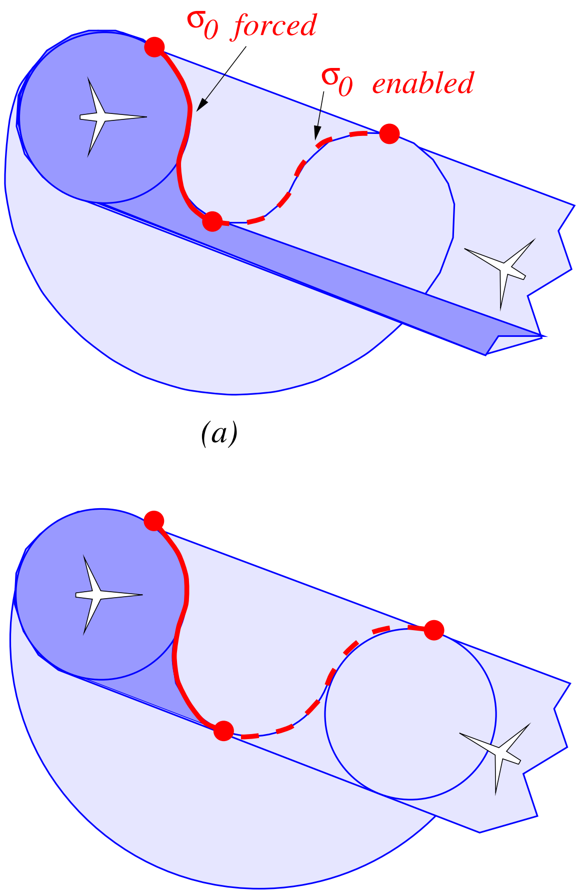
Fig. 6. $Reach(Pre_1(W^{-1}), Pre_0(W^{-1}))$ in q_1

5 Computational Issues

In practice, the usefulness of the proposed synthesis algorithm depends on our ability to efficiently compute solutions the Hamilton-Jacobi equation. We conclude this paper with a brief discussion of some of the computational issues which we are currently investigating.

Numerical methods for computing solutions to the Hamilton-Jacobi PDE have been studied extensively: a survey paper [18] presents a set of computation schemes based on a *level set method* for propagating curves, which uses numerical techniques derived from conservation laws. The approach requires gridding the state space, so while these techniques have been shown to be efficient in two- or three-dimensions, they may become cumbersome in higher dimensions. Also, it is essential that a bound on the error due to approximation be known at each step of the algorithm, in order to guarantee that the computed surface is a conservative approximation to the actual surface.

Numerical solutions are potentially complicated by the fact that the right hand side of equation (20) is non-smooth. This is possibly also the case for the optimal Hamiltonian $H^*(x,p)$. Moreover, as t evolves the solution $J^*(x,t)$ to the Hamilton-Jacobi equation can develop discontinuities (known as *shocks*) as a function of x. Finally, it is unreasonable to assume that the capture set is always described by a level set of a single differentiable function $l(x)$: more generally, we should assume that there exists a collection of differentiable functions $l_i(x)$ where $i = 1 \dots m$ such that the capture set is described by $G = \cap_{i=1}^m \{x \in \mathbb{R}^n \mid l_i(x) \leq 0\}$. Computing solutions with discontinuous Hamiltonian functions is dealt with in [18] using an evolution function which varies across the grid space. Methods to compute solutions in the presence of shocks are presented in



(a)

(b)

Fig. 7. Showing the enabling and forcing boundaries for σ_0 in the frame of state q_1; and the result of increasing the radius of the turn in the avoid maneuver to increase W^*.

[19], and a "viscosity" method to avoid shocks is presented in [20].

References

1. A. Church, "Logic, arithmetic, and automata," in *Proceedings of the International Congress of Mathematicians*, pp. 23–35, 1962.
2. J. von Neumann and O. Morgenstern, *Theory of games and economic behavior*. Princeton university press, 1947.
3. J. R. Büchi and L. H. Landweber, "Solving sequential conditions by finite-state operators," in *Proceedings of the American Mathematical Society*, pp. 295–311, 1969.

4. M. O. Rabin, "Automata on infinite objects and Church's problem," in *Regional Conference Series in Mathematics*, 1972.
5. A. Puri, *Theory of Hybrid Systems and Discrete Event Systems.* PhD thesis, Department of Electrical Engineering, University of California, Berkeley, California, 1995.
6. W. Thomas, "Automata on infinite objects," in *Formal Models and Semantics, volume B of Handbook of Theoretical Computer Science*, Elsevier Science, 1990.
7. W. Thomas, "On the synthesis of strategies in infinite games," in *Proceedings of STACS 95, Volume 900 of LNCS* (E. W. Mayr and C. Puech, eds.), pp. 1–13, Munich: Springer Verlag, 1995.
8. O. Maler, A. Pnueli, and J. Sifakis, "On the synthesis of discrete controllers for timed systems," in *STACS 95: Theoretical Aspects of Computer Science* (E. W. Mayr and C. Puech, eds.), Lecture Notes in Computer Science 900, pp. 229–242, Munich: Springer Verlag, 1995.
9. E. Asarin, O. Maler, and A. Pnueli, "Symbolic controller synthesis for discrete and timed systems," in *Proceedings of Hybrid Systems II, Volume 999 of LNCS* (P. Antsaklis, W. Kohn, A. Nerode, and S. Sastry, eds.), Cambridge: Springer Verlag, 1995.
10. H. Wong-Toi, "The synthesis of controllers for linear hybrid automata," in *Proceedings of the IEEE Conference on Decision and Control*, (San Diego, CA), 1997.
11. W. M. Wonham, *Linear Multivariable Control: a geometric approach.* Springer Verlag, 1979.
12. A. Deshpande and P. Varaiya, "Viable control of hybrid systems," in *Hybrid Systems II* (P. Antsaklis, W. Kohn, A. Nerode, and S. Sastry, eds.), Lecture Notes in Computer Science 999, pp. 128–147, Berlin: Springer Verlag, 1995.
13. R. Isaacs, *Differential Games.* John Wiley, 1967.
14. T. Başar and G. J. Olsder, *Dynamic Non-cooperative Game Theory.* Academic Press, second ed., 1995.
15. C. Tomlin, G. Pappas, and S. Sastry, "Conflict resolution for air traffic management: A case study in multi-agent hybrid systems," tech. rep., UCB/ERL M97/33, Electronics Research Laboratory, University of California, Berkeley, 1997. To appear in the IEEE Transactions on Automatic Control.
16. J. Lygeros, D. N. Godbole, and S. Sastry, "A verified hybrid controller for automated vehicles," Tech. Rep. UCB-ITS-PRR-97-9, Institute of Transportation Studies, University of California, Berkeley, 1997. To appear in the IEEE Transactions on Automatic Control, Special Issue on Hybrid Systems, April 1998.
17. A. E. Bryson and Y.-C. Ho, *Applied Optimal Control.* Waltham: Blaisdell Publishing Company, 1969.
18. J. A. Sethian, "Theory, algorithms, and applications of level set methods for propagating interfaces," tech. rep., Center for Pure and Applied Mathematics (PAM-651), University of California, Berkeley, 1995.
19. J. M. Berg, A. Yezzi, and A. R. Tannenbaum, "Phase transitions, curve evolution, and the control of semiconductor manufacturing processes," in *Proceedings of the IEEE Conference on Decision and Control*, (Kobe), pp. 3376–3381, 1996.
20. P. L. Lions, *Generalized Solutions of Hamilton-Jacobi Equations.* London: Pittman, 1982.