

# Trajectory-based Controller Design for Hybrid Systems with Affine Continuous Dynamics

A. Agung Julius

Department of Electrical, Computer and Systems Engineering

Rensselaer Polytechnic Institute

Troy, NY 12180, USA.

Email: `agung@ecse.rpi.edu`

## Abstract

In this paper we propose a method for feedback controller synthesis using the concept of control auto-bisimulation function. Control autobisimulation function (CAF) is the analog of control Lyapunov function for approximate bisimulation. Approximate bisimulation has been used to establish robustness (in  $\ell_\infty$  sense) of execution trajectories of dynamical systems and hybrid systems, resulting in trajectory-based safety verification procedures.

CAF is used to characterize the family of all feedback control laws that result in a close loop system with an autobisimulation function. Further, we use a heuristic potential function based idea to construct a safe feedback control law for a nominal initial state, and use the trajectory-robustness property to guarantee that the control law is also safe for other initial states in a neighborhood of the nominal initial state.

**Keywords:** hybrid system, trajectory based, controller synthesis.

## I. INTRODUCTION

The issue of safety/reachability is very important in the theory of hybrid systems. The *analysis* part of this issue, i.e. the investigation whether a given hybrid system model with given initial conditions can reach a certain state, or set of states has received a lot of attention from the hybrid systems community. It has also resulted in a lot of practical applications, for example in the safety analysis of air traffic systems [1], design verification for electronic circuits [2], design verification for synthetic biology (e.g. [3]), and model analysis for biochemical processes [4].

There is a variety of methods that have been developed to compute the reachable set of hybrid systems. The class of safety/reachability analysis methods that is closely related to this paper is the *trajectory-based analysis*. These are methods that aim to assess the safety/reachability based on the execution trajectories of the system, or the simulations thereof. Within the family of trajectory based reachability analysis techniques itself there are different approaches. Some methods, for example, conduct state space exploration through randomized testing [5] or by using Rapidly exploring Random Trees (RRT) or its adaptations [6], [7]. Methods based on linearization of the system's nonlinear dynamics along the execution trajectory have also been proposed, for example in [8], [9]. The methods presented in [10], [11] use sensitivity analysis in computing the robustness of the trajectories with respect to parameter variations. Other related methods incorporate local gain/contraction analysis [12], [13] to measure the difference between neighboring trajectories. The main conceptual tool that we use in this paper, the *approximate bisimulation*, was developed by Girard and Pappas [14], and has been used for trajectory based analysis of hybrid systems in [15], [16], [17].

The *synthesis* part of the safety/reachability issue deals with the construction of control laws/algorithms for systems with input that result in safe executions. Some of the methods for safety/reachability analysis can be

extended for controller synthesis. For example, the optimal control method in [18] and the simulation based method in [19] directly characterize the influence of the control input in the reachability formulation. The predicate abstraction technique for systems with piecewise affine dynamics in polytope sets leads to a control procedure based on the transversality of the vector field on the facets of the polytopes [20], [21]. The technique for discrete-time system presented in [22] utilizes partitioning of the state space by polygonal approximation of the reachable set. The notion of approximate bisimulation has previously been used for controller synthesis for nonlinear dynamical systems [23], [24]. In this case, the notion is used to establish a quantization of the continuous state space, which can result in a countable transition system approximation of the original dynamics.

In this paper we present a controller synthesis method that is based on trajectory-based analysis. We introduce the notion of *control autobisimulation function (CAF)*, to characterize a class of feedback laws, called the *admissible feedback laws*, that result in closed loop systems that admit an (auto)bisimulation function. Therefore, the control autobisimulation function can be thought of as an analog of control Lyapunov function [25], [26] for autobisimulation. For any given initial condition, we use a heuristic method based on the idea of potential function to construct an admissible feedback law that results in a "valid" execution trajectory<sup>1</sup>. The use of CAF enables us to use trajectory robustness (a la approximate bisimulation) to guarantee formally the validity of the control law for a neighborhood around that initial condition. By repeating this procedure for a finite set of initial conditions, we can cover a compact set of initial conditions.

The controller design method presented in this paper therefore consists of two steps. The first step is to characterize the class of admissible feedback laws. The second step is to construct an admissible feedback law for each initial condition, that can be verified to result in a valid trajectory (for example, through simulation). We present an example demonstrating that although the presented design method has a heuristic step in it, the resulting controller is formally guaranteed to be correct, and that it is possible to achieve that with only a few simulation runs. This approach can thus be regarded as a highly parallelizable and lightweight (no quantization of state space is required) complement to the more formal approaches, such as [23], [24].

## II. AUTOBISIMULATION FUNCTION

We recall the use of (auto)bisimulation function for establishing trajectory robustness for autonomous systems (systems without input). Given an autonomous dynamical system

$$\Sigma_{\text{aut}} : \frac{dx}{dt} = f(x), x \in \mathbb{R}^n, \quad (1)$$

with  $f(x)$  locally Lipschitz.

*Notation 1:* The trajectory of the dynamical system (1) with initial condition  $x(0) = x_0$  is denoted as  $\xi(t, x_0)$ .

We define an autobisimulation function as follows.

*Definition 1:* [16] A continuously differentiable function  $\phi : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}_+$  is an **autobisimulation function** of (1) if for any  $x, x' \in \mathbb{R}^n$ ,

$$\phi(x, x') \geq \|x - x'\|, \quad (2)$$

$$\nabla_x \phi(x, x') f(x) + \nabla_{x'} \phi(x, x') f(x') \leq 0. \quad (3)$$

Autobisimulation function is used to provide a formal guarantee that the distance between two execution trajectories of the dynamical system (1) is bounded in the  $\ell_\infty$  sense. This result is stated in the following proposition.

<sup>1</sup>What "valid" means will be discussed later.

*Proposition 1:* [16] Given a dynamical system (1) and an autobisimulation function  $\phi$ , for any  $x_0, x'_0 \in \mathbb{R}^n$ ,

$$\|\xi(t, x_0) - \xi(t, x'_0)\| \leq \phi(x_0, x'_0), \quad \forall t \geq 0. \quad (4)$$

Further, if  $\phi(x, x')$  is designed to be a (pseudo)metric in  $\mathbb{R}^n$ ,  $\phi(x, x') = \|x - x'\|_\phi$ , or if it is a class  $\mathcal{K}$  function of a metric in  $\mathbb{R}^n$ ,  $\phi(x, x') = \alpha(\|x - x'\|)$ , where  $\alpha$  is a class  $\mathcal{K}$  function<sup>2</sup>, then Proposition 1 can be used as the foundation of trajectory-based safety analysis. This is illustrated in Figure 1.

*Remark 1:* The notion of autobisimulation is related to but different from the notion of incremental global asymptotic stability ( $\delta$ GAS) introduced by Angeli (cf. [28]) in the following sense. With autobisimulation, we do not require asymptotic convergence of the trajectories as in  $\delta$ GAS. Therefore, any Lyapunov function that characterizes  $\delta$ GAS can be used as autobisimulation function, but not vice versa.

Suppose that there is a given compact set of initial states  $\text{Init} \subset \mathbb{R}^n$ , where the state is initiated at  $t = 0$ , i.e.  $x(0) \in \text{Init}$ . Also, we assume that there is a set of goal states,  $\text{Goal} \subset \mathbb{R}^n$ . We require that any execution trajectory starting in  $\text{Init}$  enters the goal set before time  $t = T > 0$ . For a given initial condition  $x_0 \in \mathbb{R}^n$ , suppose that

$$\inf_{0 \leq t \leq T} d_\phi(\xi(t, x_0), \text{Unsafe}) = \delta > 0, \quad (5)$$

$$\xi(T, x_0) \in \text{Goal}, \quad (6)$$

$$d_\phi(\xi(T, x_0), \text{Goal}^C) < \delta, \quad (7)$$

where

$$d_\phi(\xi(t, x_0), \text{Unsafe}) := \inf_{x' \in \text{Unsafe}} \phi(\xi(t, x_0), x'),$$

$$d_\phi(\xi(T, x_0), \text{Goal}^C) := \inf_{x' \notin \text{Goal}} \phi(\xi(T, x_0), x').$$

Notice that this implies that for the time interval  $0 \leq t \leq T$  the trajectory  $\xi(t, x_0)$  is safe (i.e. it does not enter the Unsafe set).

*Notation 2:* For any  $x \in \mathbb{R}^n$  and  $\delta \geq 0$ , we denote the set  $\{x' \in \mathbb{R}^n \mid \phi(x, x') \leq \delta\}$  as  $B_\phi(x, \delta)$ .

We can show (c.f. [16]) that in this case, any initial condition  $x'_0 \in B_\phi(x_0, \delta)$  will also result in a safe trajectory, i.e.  $\xi(t, x'_0) \notin \text{Unsafe}$ , for  $0 \leq t \leq T$ . Moreover, we can formally guarantee that the trajectory  $\xi(t, x'_0)$  will enter the goal set before time  $t = T$ . Therefore, by computing the execution trajectory  $\xi(t, x_0)$ , we can generalize its safety property to a nonzero measure neighborhood of initial states around  $x_0$ . This is the foundation for formal safety verification of a compact set of initial states using a finite number of execution trajectories.

### III. CONTROL AUTOBISIMULATION FUNCTION AND TRAJECTORY-BASED FEEDBACK CONTROL

Consider a dynamical system with input

$$\Sigma_{\text{inp}} : \frac{dx}{dt} = f(x, u), \quad x \in \mathbb{R}^n, \quad u \in \mathcal{U} \subset \mathbb{R}^m. \quad (8)$$

where the function  $f(x, u)$  is locally Lipschitz in  $x$  and continuous in  $u$ . As discussed in the previous section, assume that we also have the set of initial states,  $\text{Init}$ , and the goal set,  $\text{Goal}$ . Suppose that we are given the following control problem:

*Problem 1:* Design a feedback control law  $u = g(x)$  such that for any initial state  $x_0 \in \text{Init}$ , the trajectory of the closed loop system enters  $\text{Goal}$  before time  $t = T > 0$ , and in the time interval  $[0, T]$  the trajectory is safe.

<sup>2</sup>A function  $\alpha : \mathbb{R}_+ \rightarrow \mathbb{R}_+$  is a class  $\mathcal{K}$  function if it is continuous, monotonically increasing, and  $\alpha(0) = 0$  (see e.g. [27]).

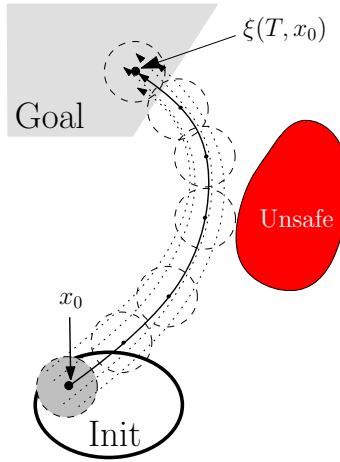


Fig. 1. An illustration for the use of Proposition 1 in trajectory-based safety analysis. The dashed circles represent several level sets of the autobisimulation function. In particular, the set  $B_\phi(x_0, \delta)$  is represented by the shaded circle. The solid curve represents the nominal execution trajectory  $\xi(t, x_0)$ , while the dotted curves represent execution trajectories originating in  $B_\phi(x_0, \delta)$ . Notice that at time  $t = T$ , the set  $B_\phi(\xi(T, x_0), \delta)$  is entirely included in Goal.

We will discuss how the notion of trajectory-robustness discussed in the previous section can also be used in trajectory-based controller synthesis. The key concept in this approach is the *control autobisimulation function (CAF)*.

*Definition 2:* A continuously differentiable function  $\psi : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}_+$  is a **control autobisimulation function** of (8) if for any  $x, x' \in \mathbb{R}^n$ ,

$$\psi(x, x') \geq \|x - x'\|, \quad (9)$$

and there exists a function  $k : \mathbb{R}^n \rightarrow \mathcal{U}$  such that

$$\nabla_x \psi(x, x') f(x, k(x)) + \nabla_{x'} \psi(x, x') f(x', k(x')) \leq 0. \quad (10)$$

The control autobisimulation function is an analog of the control Lyapunov function (CLF) [25], for approximate bisimulation [14], [16]. While control Lyapunov function has been used to construct control laws that guarantee stability (e.g. [26]), we shall use the control autobisimulation function to construct control laws that guarantee trajectory robustness.

*Remark 2:* One can compare the control autobisimulation function with control Lyapunov function of the product of the system (8) with itself

$$\frac{d}{dt} \begin{bmatrix} x \\ x' \end{bmatrix} = \begin{bmatrix} f(x, u) \\ f(x', u') \end{bmatrix}.$$

In this case, notice that unlike for CLF, for CAF we cannot set  $u$  and  $u'$  to be any functions of  $x$  and  $x'$ . Rather, the inputs  $u$  and  $u'$  must be the same function of their respective states ( $x$  and  $x'$ ). Therefore, in this aspect, the requirement for CAF is more stringent than that for CLF.

A consequence of the existence of a CAF as in Definition 2 is the existence of feedback control laws

$$u = k(x), \quad (11)$$

such that the closed loop system obtained from (8) and (11),

$$\frac{dx}{dt} = f(x, k(x)), x \in \mathbb{R}^n, \quad (12)$$

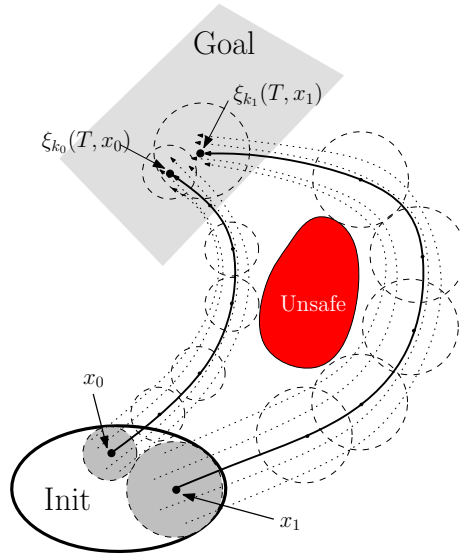


Fig. 2. An illustration for trajectory-based controller synthesis.

has  $\psi(\cdot, \cdot)$  as an autobisimulation function. This fact can be deduced immediately from Definition 2.

*Definition 3:* For a given dynamical system with input  $\Sigma_{\text{inp}}$  and a control autobisimulation function  $\psi$ , the class of all feedback control laws  $k(\cdot)$  that satisfy (10) is called the **class of admissible feedback laws**,  $\eta(\Sigma_{\text{inp}}, \psi)$ .

*Notation 3:* For a given dynamical system with input  $\Sigma_{\text{inp}}$  and a feedback control law  $u = k(x)$ , the closed loop trajectory with initial condition  $x(0) = x_0$  is denoted by  $\xi_k(t, x_0)$ .

The controller synthesis paradigm in this paper can be stated as follows. We construct feedback controllers from the class of feasible feedback laws. By definition, the closed loop system will then admit a predefined autobisimulation function. This means that the trajectory-robustness property discussed in Section II is guaranteed to hold. Please refer to Figure 2. Suppose that for a given initial state  $x_0 \in \text{Init}$ , we can design a feedback law  $u = k_0(x)$  that results in a closed loop execution trajectory  $\xi_{k_0}(t, x_0)$  satisfying

$$\inf_{0 \leq t \leq T} d_\phi(\xi_{k_0}(t, x_0), \text{Unsafe}) = \delta_0 > 0, \quad (13)$$

$$\xi_{k_0}(T, x_0) \in \text{Goal}, \quad (14)$$

$$d_\phi(\xi_{k_0}(T, x_0), \text{Goal}^C) < \delta_0. \quad (15)$$

Then, as previously shown, we can obtain a neighborhood around  $x_0$ ,  $B_\phi(x_0, \delta_0)$  consisting of other initial states for which the feedback law  $u = k_0(x)$  is guaranteed to result in execution trajectories that are safe and meet the goal state.

We can repeat the procedure for a different initial state, say  $x_1 \in \text{Init}$ . Suppose that we can then design a feedback law  $u = k_1(x)$  that results in a closed loop execution trajectory  $\xi_{k_1}(t, x_1)$  that is safe and meets the goal set as shown in Figure 2. As before, we also obtain a neighborhood around  $x_1$  for which the feedback law  $u = k_1(x)$  is guaranteed to yield execution trajectories that are safe and meet the goal state. As the result of this process, we now obtain two feedback laws which are valid for two different subsets of Init (not necessarily disjoint). The goal of the controller synthesis procedure is then to cover the entire initial set Init with different control laws as such.

#### IV. CONTROLLER SYNTHESIS FOR SYSTEMS WITH AFFINE DYNAMICS

Based on the exposition in the previous section, it is clear that to implement the idea of trajectory-based controller synthesis, we need to have a control autobisimulation function (CAF)  $\psi$ , and the feedback control laws for each initial condition that we evaluate. Moreover, each the feedback control laws must belong to the class of admissible controller  $\eta(\Sigma_{\text{inp}}, \psi)$ . The synthesis of the CAF and the controllers for systems with linear affine dynamics is discussed in this section.

##### A. Two-stage Controller Design

A specific class of these systems are systems with linear affine dynamics. These are systems of the form

$$\Sigma_{\text{lin}} : \frac{dx}{dt} = Ax + f + Bu, \quad x \in \mathbb{R}^n, u \in \mathbb{R}^m, \quad (16)$$

where  $A \in \mathbb{R}^{n \times n}$ ,  $f \in \mathbb{R}^n$ , and  $B \in \mathbb{R}^{n \times m}$ . For such systems, we propose to construct CAF as quadratic functions [14], [16], [29]. That is, we assume that

$$\psi(x, x') = \frac{1}{2}(x - x')^T P(x - x'), \quad (17)$$

where  $P \in \mathbb{R}^{n \times n}$  is a positive definite matrix. From Definition 2, it follows that inequality (10) becomes

$$(x - x')^T P (A(x - x') + B(k(x) - k(x'))) \leq 0. \quad (18)$$

We propose to construct a feedback law of the form

$$u(t) = k(x) = Kx + v(t), \quad (19)$$

where  $K \in \mathbb{R}^{m \times n}$  and  $v(t) \in \mathbb{R}^m$  is a time-varying function, both to be determined later. By substituting (19) into (18), we obtain

$$(x - x')^T P (A + BK) (x - x') \leq 0. \quad (20)$$

Finding  $K$  that satisfies inequality (20) is equivalent to finding  $K$  such that  $(A + BK)$  is Hurwitz. A well known result in control theory (c.f. [30], [31]) states that there exist  $P$  and  $K$  such that (20) holds if and only if  $(A, B)$  is stabilizable. In this case, there are well known methods to synthesize the suitable  $P$  and  $K$ . For example, by solving the following linear matrix inequality<sup>3</sup> (LMI) [32]

$$A\tilde{P} + B\tilde{D} + \tilde{P}A^T + D^T B \leq 0, \quad \tilde{P} > 0, \quad (21)$$

for  $\tilde{P} \in \mathbb{R}^{n \times n}$  and  $D \in \mathbb{R}^{m \times n}$ . The feedback gain  $K$  can be computed from

$$K^T = D\tilde{P}^{-1}. \quad (22)$$

From here,  $P$  can be obtained by solving the Lyapunov equation

$$(A + BK)^T P + P(A + BK) \leq 0, \quad P > 0. \quad (23)$$

By applying the feedback control law (19) to  $\Sigma_{\text{lin}}$ , we obtain a closed loop system

$$\Sigma_{\text{cl}} : \frac{dx}{dt} = (A + BK)x + f + Bv, \quad x \in \mathbb{R}^n, v \in \mathbb{R}^m. \quad (24)$$

<sup>3</sup>We use the fact that  $A$  is Hurwitz if and only if  $A^T$  is Hurwitz.

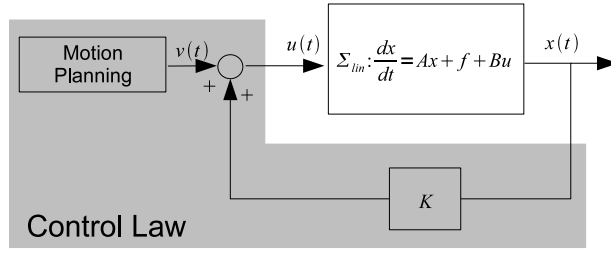


Fig. 3. The control law to be designed is shown in the shaded area. First, we compute the feedback gain  $K$  such that the inner loop system admits an autobisimulation function. Then, the signal  $v(t)$  is designed such that the control goal is achieved for a given initial state.

Notice that following to the discussion above, given that  $(A + BK)$  is Hurwitz, we are still free to design  $v(t)$ . In other words, whatever  $v(t)$  is, the control law is admissible (see Definition 3). The remaining task in the controller design is therefore to use  $v(t)$  to steer the trajectories of the closed loop system. The goal is to steer any given initial state in  $\text{Init}$  to the goal set, without entering the unsafe set.

We propose to use heuristic ideas related to motion planning problem with obstacle avoidance. There are many techniques that have been developed to solve this problem [33]. In Subsection IV-C, we will discuss an example where this problem is solved. Please refer to Figure 3 for the block diagram of the controller synthesis.

### B. Bounded Input Set

Consider the case where the input set is bounded, i.e. for all  $t \in \mathbb{R}_+$

$$\|u(t)\| \leq M, \quad (25)$$

for some positive bound  $M$ . We need to ensure that the feedback law given in (19) satisfies this condition. The fact that

$$\|u(t)\| \leq \|K\| \|x\| + \|v(t)\|, \quad (26)$$

indicates that minimizing  $\|K\|$  can alleviate the difficulty of designing  $v(t)$  that satisfies the control input bound, as demonstrated by the example in Subsection IV-C. We can approach this problem by modifying the LMI (21) into the following semidefinite programming problem [34]

$$\begin{aligned} \min \|D\| \quad \text{subject to} \\ A\tilde{P} + BD + \tilde{P}A^T + D^T B^T \leq 0, \\ \tilde{P} - I > 0. \end{aligned} \quad (27)$$

It is clear that any  $(\tilde{P}, D)$  that is feasible for (21) can be scaled so that it is feasible for (27). However, we also have

$$\|K\| \leq \|D\| \left\| \tilde{P}^{-1} \right\| \leq \|D\|, \quad (28)$$

which shows that solving (27) effectively leads to the minimization of an upper bound for  $\|K\|$ .

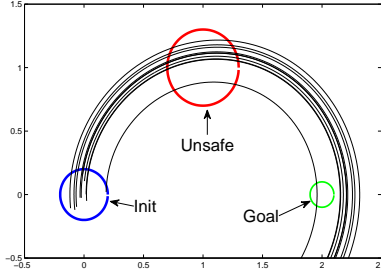


Fig. 4. A diagram showing the initial set, the goal set, and the unsafe set. The curves are execution trajectories starting from Init with zero input.

### C. Numerical Example

The following example illustrates the controller synthesis procedure. Consider the following problem. Given an affine linear system

$$\Sigma : \frac{dx}{dt} = Ax + f + Bu, \quad (29)$$

$$A = \begin{bmatrix} 0 & 1 \\ -1 & 0.1 \end{bmatrix}; B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}; f = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

We want to design feedback control laws that will bring any state in the initial set

$$\text{Init} = \{x \in \mathbb{R}^2 \mid \|x\| \leq 0.2\}, \quad (30)$$

to the goal set

$$\text{Goal} = \{x \in \mathbb{R}^2 \mid \|x - [2, 0]^T\| \leq 0.1\}, \quad (31)$$

without entering the unsafe set

$$\text{Unsafe} = \{x \in \mathbb{R}^2 \mid \|x - [1, 1]^T\| \leq 0.3\}, \quad (32)$$

with the input constraint  $\|u(t)\| \leq 2$ . These sets and some execution trajectories originating in Init with zero input are shown in Figure 4.

We implement a feedback control

$$u(t) = Kx + v(t) = \begin{bmatrix} k_1 & k_2 \end{bmatrix} x + v(t). \quad (33)$$

Since  $A$  and  $B$  are in the controllable canonical form, we know that the system is controllable (stronger than stabilizable) and we can apply the Routh-Hurwitz criteria to characterize all feedback gain  $K$  that stabilize the system [31]. This is given by

$$k_1 - 1 < 0, \quad k_2 + 0.1 < 0. \quad (34)$$

The problem of minimizing  $\|K\|$  subject to the constraint (34) can be solved analytically, yielding the optimal gain

$$K_{\text{opt}} = \begin{bmatrix} 0 & -0.1 \end{bmatrix}, \quad \text{and } \|K_{\text{opt}}\| = 0.1. \quad (35)$$

As a comparison, we can approximate the optimal  $K$  by solving (27). In this case, using the convex optimization

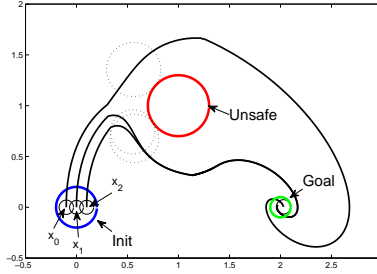


Fig. 5. The close loop execution trajectories resulting from the controller synthesis procedure. For each initial state, we can observe that the trajectories are robustly safe and meet the goal set. We also show the robust neighborhood around each initial state.

solver `cvx` [35], we calculate the optimal gain as (up to 4 decimals)

$$K_{\text{sdp}} = D_{\text{opt}} \tilde{P}_{\text{opt}}^{-1} = \begin{bmatrix} 0.0000 & -0.1000 \end{bmatrix}, \quad (36)$$

which is the same as the result of the analytic calculation (up to numerical precision). The control autobisimulation function

$$\psi(x, x') = \frac{1}{2}(x - x')^T P(x - x') \quad (37)$$

needs to satisfy (20). By implementing the optimal feedback gain  $K_{\text{opt}}$ , we obtain

$$A + BK_{\text{opt}} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}, \quad (38)$$

which implies that  $P$  can be taken as the identity matrix.

For any given initial condition  $[x_{1,0}, x_{2,0}]^T \in \text{Init}$ , the next step in this controller design is to calculate the steering input  $v(t)$  of the closed loop system

$$\Sigma_{\text{cl}} : \frac{dx}{dt} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} v, \quad (39)$$

that will bring the initial state to the goal state without entering the unsafe set<sup>4</sup>. We also need to satisfy the bound on the input magnitude

$$\|K_{\text{opt}}x(t) + v(t)\| \leq 2. \quad (40)$$

The design that we pick is rather *ad hoc*, and inspired by the potential function/navigation function technique in motion planning with obstacle avoidance (see e.g. [36], [37]). We notice that there are two tasks at hand: (i) steer the trajectory to the goal set, and (ii) avoid the unsafe set. We design a controller for each task, and combine them. The controller for the first task can be designed as a feedback law as follows

$$v_{\text{goal}}(t) = 1 - x_2(t). \quad (41)$$

The rationale behind this design is that it makes the center of the goal set an attractive equilibrium. Thus, if we ignore the unsafe set, this steering input will ensure that the goal set will be reached.

The design of the second controller borrows an idea from potential function technique. We first define a scalar

<sup>4</sup>i.e.  $v(t)$  can vary depending on the initial condition.

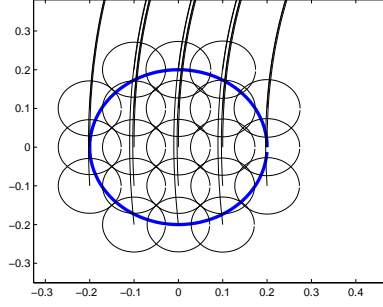


Fig. 6. The set of initial states,  $\text{Init}$ , can be covered with 21 robust neighborhoods.

weight function  $W : \mathbb{R}^2 \rightarrow \mathbb{R}_+$ ,

$$W(x) = \begin{cases} \frac{0.1}{(0.3 - \|[1,1]^T - x\|)^2}, & \|[1,1]^T - x\| \leq 0.7 \\ 0, & \text{otherwise.} \end{cases} \quad (42)$$

Observe that this function is concentric around the unsafe set, zero at distance more than 0.4 from the unsafe set, and asymptotically raises to infinity as  $x$  approaches the boundary of the unsafe set. We then define the second controller also as feedback law, as follows

$$v_{\text{safe}}(t) = \begin{cases} W(x), & \text{if } B^T([1,1]^T - x) \geq 0 \\ -W(x), & \text{if } B^T([1,1]^T - x) < 0. \end{cases} \quad (43)$$

The rationale behind this controller design is as follows. Notice that the sign of  $v_{\text{safe}}(t)$  is chose such that  $Bv_{\text{safe}}(t)$  always pushes the state away from the unsafe set. The magnitude of this "pushing force" is defined by the weight function  $W(x)$ . Thus, it is zero if the state is sufficiently far away from the unsafe set, and blows up to infinity as it approaches the unsafe set.

Finally, we need to ensure that (40) is met. But first, we need the following observation. Based on the size of the goal set, any robust neighborhood that we obtain for any particular initial condition cannot have radius more than 0.1 (see Figure 2). Therefore, we can bound the variation of the control input  $u(t)$  within any robust neighborhood as follows

$$\|\Delta u\| \leq \|K\| \|\Delta x\| = 0.1 \cdot 0.1 = 0.01. \quad (44)$$

We then define the overall steering input such that

$$u(t) = Z(K_{\text{opt}}x(t) + v_{\text{goal}}(t) + v_{\text{safe}}(t), 1.99), \quad (45)$$

where  $Z : \mathbb{R} \times \mathbb{R}_+ \rightarrow \mathbb{R}$  is simply the saturation function

$$Z(x, y) := \begin{cases} -1.99, & x < -y. \\ x, & |x| \leq y \\ 1.99, & x > y. \end{cases} \quad (46)$$

Therefore, for any initial condition, we guarantee that the designed feedback law meets the constraint on the input magnitude, even including the possible input variation within the robust neighborhood.

Notice that while the controller that we design above is sensible, there is no proof that it will indeed meet the objectives. For example, there are some parameters that might need tuning. However, the benefit of the

trajectory-based controller synthesis **allows us to simply simulate and tune the controller for a finite set of initial conditions**. The robustness property of the trajectories then allows us to generalize the results to cover the whole initial set. Figure 5 shows the application of this design procedure for three different initial conditions  $x_{0,1,2} \in \text{Init}$ . In each case, we can see that indeed the designed steering inputs  $v(t)$  result in safe trajectories that meet the goal set. We also compute the robust neighborhood around each initial state. We only show three trajectories, so as not to clutter the figure. However we can show that, for example, with uniform sampling of the initial set, we can cover it with 21 robust neighborhood, as shown in Figure 6.

## V. CONTROLLER SYNTHESIS FOR HYBRID SYSTEMS

The design paradigm presented in the previous section can be also applied to hybrid systems. Consider a standard model of hybrid systems,  $\mathcal{H} = (\mathcal{X}, \mathcal{L}, E, \text{Inv}, \Sigma)$ , where  $\mathcal{X}$  is the continuous state space of the system,  $\mathcal{L}$  is the finite set of discrete states (locations),  $E$  is the set of transitions,  $\text{Inv} : \mathcal{L} \rightarrow 2^{\mathcal{X}}$  is the invariant set of a location, and  $\Sigma$  is a family of dynamical systems with input that defines the continuous dynamics in each location. That is, for each location  $l \in \mathcal{L}$ , we define the continuous dynamics as

$$\Sigma(l) : \frac{dx}{dt} = f_l(x, u), x \in \mathcal{X}, u \in \mathcal{U}. \quad (47)$$

A transition  $e \in E$  is a 4-tuple  $(l, l', g, r)$ , where  $l \in \mathcal{L}$  is the origin of the transition,  $l' \in \mathcal{L}$  is the target of the transition and that each location,  $g \subset \partial \text{Inv}(l)$  is the guard of the transition, which is a subset of the boundary of the invariant set of location  $l$ , and  $r : g \rightarrow \text{Inv}(l')$  is the reset map that resets the continuous state at the new location. We assume that the reset map  $r$  is continuous, the continuous state space is  $\mathbb{R}^n$ , the invariant sets are closed,  $f_l(x, u)$  is locally Lipschitz in  $x$  and continuous in  $u$  for all  $l \in \mathcal{L}$ , the transitions are deterministic in the sense that the guards of all outgoing transitions from a location are disjoint, and that the system does not deadlock or possess Zeno behavior. In analyzing the safety of the system, we assume that there is a subset  $\text{Unsafe} \subset \mathcal{X} \times \mathcal{L}$  of unsafe states. A trajectory of the hybrid system corresponds to an unsafe execution if it intersects with the unsafe set.

*Notation 4:* We denote the set of all outgoing transitions from a location  $l \in \mathcal{L}$  as  $\text{Out}(l)$ .

### A. Control Problem Formulation and Hierarchical Control Synthesis

To define the control problem, we define a set of initial state  $\text{Init} \subset \mathcal{X} \times \mathcal{L}$ , in which we assume the hybrid state begins at  $t = 0$ . We also define a goal set,  $\text{Goal} \subset \mathcal{X} \times \mathcal{L}$  in which all executions must terminate. As before, the control problem is defined as finding the feedback control strategy that is guaranteed to bring any initial state in  $\text{Init}$  to the goal set without entering the unsafe set. Without any loss of generality, we can assume that the set  $\text{Init}$  is contained in (the invariant set of) one location, called  $l_{\text{init}} \in \mathcal{L}$ . If this is not the case, we can divide the problem into several subproblems, each with an  $\text{Init}$  set contained in a specific location. Similarly, we can assume the  $\text{Goal}$  is also entirely contained in one location, called  $l_{\text{goal}} \in \mathcal{L}$ .

We approach this problem with a *hierarchical control* design, which can be described in the following steps: **Step 1: Discrete Synthesis.** We compute a discrete trajectory that starts in  $l_{\text{init}}$  and ends in  $l_{\text{goal}}$ . By discrete trajectory, we mean an alternating sequence of locations and transitions

$$l_{\text{init}} = l_0 \xrightarrow{e_1} l_1 \xrightarrow{e_2} l_2 \xrightarrow{e_3} \dots \xrightarrow{e_N} l_N = l_{\text{goal}}. \quad (48)$$

Each transition  $e_{i,i \in \{1, \dots, N\}}$  is an element of  $E$ , originating in  $l_{i-1}$ , and targeting  $l_i$ . Such a discrete trajectory is not necessarily unique, but at this step we only need one. The computation of a discrete trajectory like this, albeit formally undecidable, is a standard procedure in formal verification of discrete event systems [38].

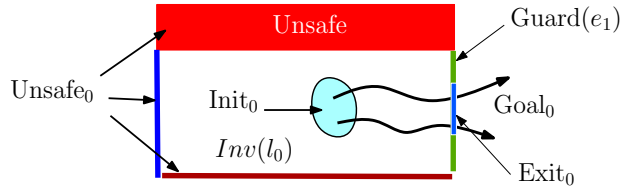


Fig. 7. An illustration for the definitions related to controller synthesis for  $l_0$ . The feedback controller is expected to guide any initial condition in  $\text{Init}_0$  to the goal set, beyond the guard of  $e_1$ , while avoiding the unsafe set and other guards.

**Step 2: Continuous Synthesis.** In this step, we synthesize the continuous controller for each of the visited locations  $(l_{0,1,\dots,N})$  in order to implement the computed discrete trajectory. Basically, in each location  $l_i$ , we define an initial set based on how  $l_i$  is reached from  $l_{i-1}$ . We then formulate the control problem of bringing the continuous state from this initial set to the goal set, which is defined as a set beyond the guard transition  $e_i$  that will bring the state to location  $l_{i+1}$  without entering the forbidden set. The forbidden set is defined as the union of Unsafe, and the guards of other outgoing transitions from  $l_i$ . If we are able to construct a continuous controller that implements the discrete trajectory, then the hybrid control problem is solved. Otherwise, we go back to Step 1, and compute another discrete trajectory.

*Remark 3:* Similar two-step approach to solve the control problem with application in motion control synthesis for fully actuated robots has been discussed in the literature (see [39] and the references therein). The discrete part of the control goal in [39] is expressed as a temporal logic formula, which is richer than the one presented in this paper. However, we would like to point out that the continuous synthesis presented in this paper can also be applied to implement the continuous part of the controller in [39].

### B. Continuous Synthesis of the Hybrid Controller

Given the discrete trajectory (48), we can synthesize the continuous controller in each location as follows.

**First Location** ( $l_{\text{init}} = l_0$ ). Define the Init as the initial set (recall that we assume the Init is entirely contained in the invariant set of  $l_{\text{init}}$ ), i.e.  $\text{Init}_0 := \text{Init}$ . Define the initial unsafe set as

$$\text{Unsafe}_0 := \text{Unsafe}|_{l_{\text{init}}} \bigcup_{e \in \text{Out}(l_{\text{init}}) \setminus e_1} \text{Guard}(e), \quad (49)$$

where for  $l \in \mathcal{L}$ ,

$$\text{Unsafe}|_l := \{x \in \text{Inv}(l) \mid (x, l) \in \text{Unsafe}\}, \quad (50)$$

and  $\text{Guard}(e)$  refers to the guard set of transition  $e \in E$ . Define as the goal set

$$\text{Goal}_0 = \text{Guard}(e_1) \cup \text{Inv}(l_{\text{init}})^C. \quad (51)$$

Essentially, this means that we define the guard set of  $e_i$  and beyond the invariant set of  $l_{\text{init}}$  as our goal. These definitions are illustrated in Figure 7.

We implement the controller synthesis discussed in Section III to develop a feedback control for the dynamical system  $\Sigma(l_{\text{init}})$  such that any initial condition in  $\text{Init}_0$  is guaranteed to reach  $\text{Goal}_0$  without entering  $\text{Unsafe}_0$ . If  $\Sigma(l_{\text{init}})$  is an affine linear system, we can use the design procedure discusses in Section IV.

After we can successfully design and implement such a controller, we define the subset of  $\text{Guard}(e_1)$  that can be reached by initial state in  $\text{Init}_0$  as  $\text{Exit}_0$ . In practice, we can use an overapproximation to compute  $\text{Exit}_0$ . For example, if  $x_i, i \in \{0,1,\dots,M-1\}$  are  $M$  initial conditions that we tested in order to cover  $\text{Init}_0$  (see Figure 2),

and if we define the continuous trajectories of the closed loop system starting from those initial conditions as  $\xi_i, i \in \{0, 1, \dots, M-1\}(t)$ , then  $\text{Exit}_0$  can be overapproximated by the intersection of  $\text{Guard}(e_1)$  with the union of all trajectory tubes around  $\xi_i, i \in \{0, 1, \dots, M-1\}(t)$ .

**Intermediate Locations** ( $l_i, i \in \{1, 2, \dots, N-1\}$ ). Define  $\text{Init}_i = r_i(\text{Exit}_{i-1})$ , where  $r_i$  is the reset map of  $e_i$ . Define the unsafe set as

$$\text{Unsafe}_i := \text{Unsafe}|_{l_i} \bigcup_{e \in \text{Out}(l_i) \setminus e_{i+1}} \text{Guard}(e), \quad (52)$$

and the goal set as

$$\text{Goal}_i = \text{Guard}(e_{i+1}) \cup \text{Inv}(l_i)^C. \quad (53)$$

As before, the objective of the controller synthesis is to guide any initial condition in  $\text{Init}_i$  to reach  $\text{Goal}_i$  without entering  $\text{Unsafe}_i$ . Once we obtain and implement such a controller, we compute (or overapproximate)  $\text{Exit}_i$  in the same way as  $\text{Exit}_0$ .

**Final Location** ( $l_{\text{goal}} = l_N$ ). Define  $\text{Init}_N = r_N(\text{Exit}_{N-1})$ , where  $r_N$  is the reset map of  $e_N$ . Define the unsafe set as

$$\text{Unsafe}_N := \text{Unsafe}|_{l_{\text{goal}}} \bigcup_{e \in \text{Out}(l_{\text{goal}})} \text{Guard}(e), \quad (54)$$

and the goal set as

$$\text{Goal}_N = \text{Goal}. \quad (55)$$

Design a controller as before. For this location, we do not need to compute the exit set.

### C. Numerical Example

We present a numerical example of hybrid controller synthesis for a hybrid system with three locations,  $\mathcal{L} = \{l_{\text{init}}, l_{\text{goal}}, l_{\text{other}}\}$ . The continuous state space is  $\mathbb{R}^2$ . The first location,  $l_{\text{init}}$ , has the invariance

$$\text{Inv}(l_{\text{init}}) = \{(x, y) \mid -0.5 \leq x \leq 3, -0.5 \leq y \leq 2\}, \quad (56)$$

and has the the same continuous dynamics as the one given in the example in Subsection IV-C, i.e.

$$\begin{aligned} \Sigma(l_{\text{init}}) : \frac{dx}{dt} &= A_1 x + f_1 + B_1 u, \quad \|u\| \leq 4. \\ A_1 &= \begin{bmatrix} 0 & 1 \\ -1 & 0.1 \end{bmatrix}; B_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}; f_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \end{aligned} \quad (57)$$

The second location,  $l_{\text{goal}}$ , has the invariance

$$\text{Inv}(l_{\text{goal}}) = \{(x, y) \mid 0 \leq x \leq 3, 0 \leq y \leq 3\}, \quad (58)$$

and has the following continuous dynamics

$$\begin{aligned} \Sigma(l_{\text{goal}}) : \frac{dx}{dt} &= A_2 x + f_2 + B_2 u, \quad \|u\| \leq 4 \\ A &= \begin{bmatrix} 0 & 1 \\ -1 & 2 \end{bmatrix}; B_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}; f = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \end{aligned} \quad (59)$$

The invariance and continuous dynamics of the other location,  $l_{\text{other}}$ , are not relevant in this example, and will not be specified.

There are two outgoing transitions from  $l_{\text{init}}$ , which are:  $e_1 = (l_{\text{init}}, l_{\text{goal}}, g_1, r_1)$  and  $e_2 = (l_{\text{init}}, l_{\text{other}}, g_2, r_2)$ . There is one outgoing transition from  $l_{\text{goal}}$ , which is  $e_3 = (l_{\text{goal}}, l_{\text{other}}, g_3, r_3)$ . The guards are defined as follows  $g_1 = \{(x, y) \mid x = 3, 0 \leq y \leq 3\}$ ,  $g_2 = \partial \text{Inv}(l_{\text{init}}) \setminus g_1$ ,  $g_3 = \partial \text{Inv}(l_{\text{goal}})$ . The reset map  $r_1$  is defined as

$$r_1 : g_1 \rightarrow \text{Inv}(l_{\text{goal}}), r_1(x, y) = (0.5, 0.75 + y). \quad (60)$$

This basically means that the reset  $r_1$  maps  $g_1$ , which is the line segment between (3,-0.5) and (3,2) to the line segment between (0,0.25) and (0,2.75) in  $\text{Inv}(l_{\text{goal}})$ . The other reset maps are irrelevant to this example and will not be specified.

Suppose that we specify the initial set

$$\text{Init} = \{l_{\text{init}}\} \times \{x \in \mathbb{R}^2 \mid \|x\| \leq 0.2\}, \quad (61)$$

and the goal set  $\text{Goal} = \{l_{\text{init}}\} \times \text{Goal}_1$ , where

$$\text{Goal}_1 = \{x \in \mathbb{R}^2 \mid \|x - [2, 1.5]^T\| \leq 0.1\}. \quad (62)$$

Moreover, we also specify the unsafe set

$$\begin{aligned} \text{Unsafe} = & \{l_{\text{init}}\} \times \{x \in \mathbb{R}^2 \mid \|x - [1, 1]^T\| \leq 0.3\} \cup \\ & \{l_{\text{goal}}\} \times \{x \in \mathbb{R}^2 \mid \|x - [1, 1.5]^T\| \leq 0.3\}. \end{aligned}$$

Notice that this example is constructed such that Figure 5 without the goal set depicts the invariant set of  $l_{\text{init}}$ .

The discrete synthesis for this example is trivial, and we can obtain one (and the only) valid discrete trajectory, which is  $l_{\text{init}} \xrightarrow{e_1} l_{\text{goal}}$ . For the continuous synthesis, following the procedure outlined in Subsection V-B, we start with a controller for  $l_{\text{init}}$ . This continuous dynamics has been analyzed before, we are going to design a controller of the form (see (33)-(35))

$$u(t) = \begin{bmatrix} 0 & -0.1 \end{bmatrix} x(t) + v(t). \quad (63)$$

the design for  $v(t)$  has two goals: (i) bring the continuous state to cross  $g_1$  and (ii) avoid the unsafe set and other guards. As before, we plan to compose  $v(t)$  out of two components. The first component,  $v_{\text{goal}}(t)$  is designed to keep the state in the first quadrant. This way, the horizontal velocity of the continuous state remains positive, and will eventually cross  $g_1$ . We can therefore propose to define  $v_{\text{goal}}(t)$  through a weight function that will effectively makes the boundary between the first quadrant and the fourth quadrant repelling, e.g.

$$W_1(x, y) = \begin{cases} (2 - 10y)^2, & 0 \leq y \leq 0.2 \\ 4, & y < 0, \\ 0, & \text{otherwise.} \end{cases}, \quad (64)$$

$$v_{\text{goal}}(t) = W_1(x, y). \quad (65)$$

The second component of  $v(t)$ ,  $v_{\text{safe}}(t)$ , is designed to keep the state away from the unsafe set and the guard

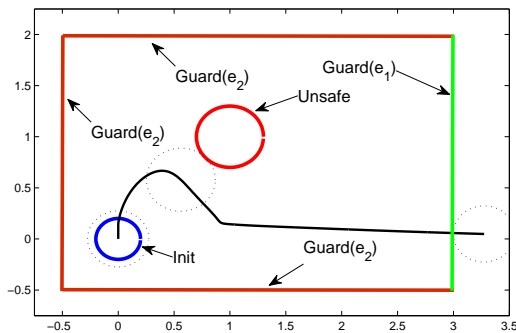


Fig. 8. The close loop system for location  $l_{init}$ .

$g_2$ . As before, we propose to define  $v_{safe}(t)$  through some weight functions

$$W_2(x, y) = \begin{cases} \frac{1}{(y-2)^2}, & y \geq 1.9 \\ 0, & \text{otherwise.} \end{cases}, \quad (66)$$

$$W_3(x, y) = \begin{cases} \frac{0.1}{(0.3 - \sqrt{(1-x)^2 + (1-y)^2})^2}, & \left\| \begin{bmatrix} 1 \\ 1 \end{bmatrix} - \begin{bmatrix} x \\ y \end{bmatrix} \right\| \leq 0.7 \\ 0, & \text{otherwise.} \end{cases} \quad (67)$$

$$v_{safe}(t) = \begin{cases} W_3(x) - W_2(x), & \text{if } B_1^T [1 - x, 1 - y]^T \geq 0 \\ -W_3(x) - W_2(x), & \text{if } B_1^T [1 - x, 1 - y]^T < 0. \end{cases} \quad (68)$$

Notice that  $W_2$  is designed to prevent the state from hitting the "top wall" of  $Inv(l_{init})$ , while  $W_3$  is essentially the same as the one given in (42). We do not define a weight function to prevent the state from hitting the "bottom wall" of  $Inv(l_{init})$  since this has been taken care of by  $v_{safe}(t)$ .

Considering the size of  $Init$ , we assume that the radius of the robust neighborhood is capped at 0.2. Therefore, by performing a calculation similar to (44) we can bound the control input variation within any robust neighborhood as

$$\|\Delta u\| \leq \left\| \begin{bmatrix} 0 & -0.1 \end{bmatrix} \right\| 0.2 = 0.02. \quad (69)$$

In order to make sure the the bound on the magnitude of the input signal is satisfied, we therefore propose to define

$$u(t) = Z\left(\begin{bmatrix} 0 & -0.1 \end{bmatrix} x(t) + v_{goal}(t) + v_{safe}(t), 3.98\right). \quad (70)$$

Figure 8 depicts a simulation run of the implementation of the above controller with initial condition (0,0) (the center of  $Init$ ). We can see that we actually cover the whole initial set with the robust neighborhood of this initial condition. In this case,  $Exit_0$  is overapproximated as the line segment between (3,-0.19) and (3,0.31).

Proceeding to the next location,  $Init_1$  is given as the image of  $Exit_0$  under the reset map  $r_1$ . Following similar ideas as above, we design and implement a feedback controller. Because of space limitation, the design is not detailed here, but the result is shown in Figure 9. We can conclude that we have successfully designed a hybrid controller that achieves the control objective.

## VI. DISCUSSION

The result presented in this paper can be generalized by replacing the motion planning box in Figure 3 with other means of obtaining valid trajectories for given initial conditions. These include other heuristics based

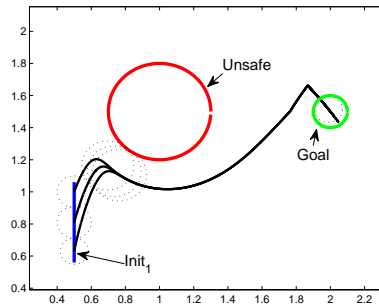


Fig. 9. After verifying only three initial conditions in  $\text{Init}_1$ , we find a controller design that meets the control objective.

methods, such as fuzzy control [40], or expert system based methods (cf. [41]) that allow for integration of human operators' experience into the control strategy. The advantage offered by the theory of trajectory-based analysis is that we can formally guarantee the safety and correctness of the resulting controllers.

Finally, we would like remark that although in this paper we restrict our attention to affine continuous dynamics, the theory of trajectory-based analysis is applicable to nonlinear dynamics as well. In further investigation, we will explore the use of local trajectory-based analysis techniques for nonlinear dynamics, extending the results presented in [29].

**Acknowledgement:** This work is partially supported by an NSF CAREER grant (#0953976). The author would like to thank Jeff Ban for the stimulating discussions that lead to this paper.

## REFERENCES

- [1] M. Prandini, J. Hu, J. Lygeros, and S. Sastry, "A probabilistic approach to aircraft conflict detection," *IEEE Trans. on Intelligent Transportation Systems*, vol. 1(4), pp. 199–220, 2000.
- [2] T. Dang, A. Donze, and O. Maler, "Verification of analog and mixed-signal circuits using hybrid system techniques," in *Formal Methods in Computer-Aided Design*, vol. 3312 of *LNCS*, pp. 21–36, Springer, 2004.
- [3] G. Batt, B. Yordanov, R. Weiss, and C. Belta, "Robustness analysis and tuning of synthetic gene networks," *Bioinformatics*, vol. 23, no. 18, pp. 2415–2422, 2007.
- [4] D. Riley, X. Koutsoukos, and K. Riley, "Modeling and analysis of the sugar cataract development process using stochastic hybrid systems," *IET Systems Biology*, vol. 3(3), pp. 137–154, 2009.
- [5] J. M. Esposito, "Randomized test case generation for hybrid systems verification," in *Proc. 36th Southeastern Symposium of Systems Theory*, 2004.
- [6] M. S. Branicky, M. M. Curtiss, J. Levine, and S. Morgan, "RRTs for nonlinear, discrete, and hybrid planning and control," in *Proc. IEEE Conf. Decision and Control*, (Hawaii, USA), 2003.
- [7] A. Bhatia and E. Frazzoli, "Incremental search methods for reachability analysis of continuous and hybrid systems," in *Hybrid Systems: Computation and Control*, vol. 2993 of *LNCS*, pp. 142–256, Springer, 2004.
- [8] A. Asarin, T. Dang, and A. Girard, "Reachability analysis of nonlinear systems using conservative approximation," in *Hybrid Systems: Computation and Control*, vol. 2623 of *LNCS*, pp. 20–35, Springer, 2003.
- [9] Z. Han and B. H. Krogh, "Reachability analysis of nonlinear systems using trajectory piecewise linearized models," in *Proc. American Control Conference*, (Minneapolis), 2006.
- [10] I. A. Hiskens and M. A. Pai, "Trajectory sensitivity analysis of hybrid systems," *IEEE Trans. Circuits and Systems - Part I*, vol. 47(2), pp. 204–220, 2000.
- [11] A. Donze and O. Maler, "Systematic simulations using sensitivity analysis," in *Hybrid Systems: Computation and Control*, vol. 4416 of *LNCS*, Springer, 2007.
- [12] W. Lohmiller and J. J. E. Slotine, "On contraction analysis for nonlinear systems," *Automatica*, vol. 34, no. 6, pp. 683–696, 1998.
- [13] W. Tan, A. Packard, and T. Wheeler, "Local gain analysis on nonlinear systems," in *Proc. American Control Conference*, (Minneapolis), 2006.

- [14] A. Girard and G. J. Pappas, "Approximation metrics for discrete and continuous systems," *IEEE Trans. Automatic Control*, vol. 52, no. 5, pp. 782–798, 2007.
- [15] A. Girard and G. J. Pappas, "Verification using simulation," in *Hybrid Systems: Computation and Control*, vol. 3927 of *LNCS*, pp. 272–286, Springer Verlag, 2006.
- [16] A. A. Julius, G. Fainekos, M. Anand, I. Lee, and G. J. Pappas, "Robust test generation and coverage for hybrid systems," in *Hybrid Systems: Computation and Control*, vol. 4416 of *LNCS*, pp. 329–342, Springer Verlag, 2007.
- [17] F. Lerda, J. Kapinski, E. M. Clarke, and B. H. Krogh, "Verification of supervisory control software using state proximity and merging," in *Hybrid Systems: Computation and Control*, vol. 4981 of *LNCS*, pp. 344–357, 2008.
- [18] I. Mitchell and C. J. Tomlin, "Level set methods in for computation in hybrid systems," in *Hybrid Systems: Computation and Control*, vol. 1790 of *LNCS*, pp. 310–323, Springer Verlag, 2000.
- [19] J. Kapinski, B. H. Krogh, O. Maler, and O. Stursberg, "On systematic simulation of open continuous systems," in *Hybrid Systems: Computation and Control*, vol. 2623 of *LNCS*, pp. 283–297, Springer, 2003.
- [20] C. Belta and L. Habets, "Controlling a class of nonlinear systems on rectangles," *IEEE Trans. Automatic Control*, vol. 51, no. 11, pp. 1749–1759, 2006.
- [21] L. Habets, P. J. Collins, and J. H. van Schuppen, "Reachability and control synthesis for piecewise-affine hybrid systems on simplices," *IEEE Trans. Automatic Control*, vol. 51, no. 6, pp. 938–948, 2006.
- [22] G. Reissig, "Computation of discrete abstractions of arbitrary memory span for nonlinear sampled systems," in *Hybrid Systems: Computation and Control*, vol. 5469 of *LNCS*, pp. 306–320, Springer, 2009.
- [23] P. Tabuada, "An approximate simulation approach to symbolic control," *IEEE Trans. Automatic Control*, vol. 53, no. 6, pp. 1406–1418, 2008.
- [24] G. Pola, A. Girard, and P. Tabuada, "Approximately bisimilar symbolic models for nonlinear control systems," *Automatica*, vol. 44, no. 10, pp. 2508–2516, 2008.
- [25] Z. Artstein, "Stabilization with relaxed controls," *Nonlinear Analysis*, vol. 15, no. 11, pp. 1163–1170, 1983.
- [26] E. D. Sontag, "A 'universal' construction of Artstein's theorem on nonlinear stabilization," *Systems and Control Letters*, vol. 13, no. 2, pp. 117–123, 1989.
- [27] H. K. Khalil, *Nonlinear Systems*. Prentice Hall, 3rd ed., 2002.
- [28] D. Angeli, "A Lyapunov approach to incremental stability properties," *IEEE Trans. Automatic Control*, vol. 47, no. 3, pp. 410–421, 2002.
- [29] A. A. Julius and G. J. Pappas, "Trajectory based verification using local finite-time invariance," in *Hybrid Systems: Computation and Control*, vol. 5469 of *LNCS*, pp. 223–236, Springer, 2009.
- [30] W. L. Brogan, *Modern control theory*. New Jersey: Prentice Hall International, 1991.
- [31] B. Friedland, *Control System Design: An Introduction to State-Space Methods*. Dover, 2005.
- [32] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in Systems and Control Theory*. Philadelphia: SIAM, 1994.
- [33] S. M. LaValle, *Planning algorithms*. Cambridge University Press, 2006.
- [34] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004. Available online at [www.stanford.edu/~boyd/cvxbook/](http://www.stanford.edu/~boyd/cvxbook/).
- [35] S. Boyd and M. C. Grant, "cvx – MATLAB software for disciplined convex programming," 2005. <http://www.stanford.edu/~boyd/cvx/>.
- [36] A. Rimon and D. E. Koditschek, "Exact robot navigation using artificial potential fields," *IEEE Trans. on Robotics and Automation*, vol. 8, no. 5, pp. 501–518, 1992.
- [37] D. C. Conner, A. A. Rizzi, and H. Choset, "Composition of local potential functions for global robot control and navigation," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3546–3551, 2003.
- [38] E. M. Clarke, O. Grumberg, and D. A. Peled, *Model Checking*. MIT Press, 1999.
- [39] G. E. Fainekos, A. Girard, H. Kress-Gazit, and G. J. Pappas, "Temporal logic motion planning for dynamic mobile robots," *Automatica*, vol. 45, no. 2, pp. 343–352, 2009.
- [40] K. M. Passino and S. Yurkovich, *Fuzzy Control*. Addison-Wesley, 1998.
- [41] B. K. Bose, "Expert system, fuzzy logic, and neural network applications in power electronics and motion control," *Proceedings of the IEEE*, vol. 82, no. 8, pp. 1303 – 1323, 1994.