# Guard Synthesis for Safety of Hybrid Systems using Sum of Squares Programming

Samuel Coogan and Murat Arcak

***Abstract*— We present a technique for synthesizing switching guards for hybrid systems by using sum of squares (SOS) programming. The guards are defined to be semialgebraic sets calculated from a bilinear SOS program. We present a method for ensuring that synthesized guards satisfy a state-based safety constraint and do not allow Zeno executions. We use an iterative algorithm to solve the bilinear program and demonstrate our approach with an example.**

## I. Introduction

Hybrid systems have emerged as a powerful modeling paradigm for complex systems that incorporate continuous and discrete phenomena. Often, it is desired for these systems to satisfy a *safety* property whereby the system is guaranteed not to enter an unsafe region of the state space. Verifying such properties given a specified hybrid system has received considerable attention, *e.g.* [1], [2], [3].

Synthesizing control strategies ensuring safety of a hybrid system is a challenging task. One approach is to calculate a controlled invariant set via an iterative algorithm. The algorithm is initialized with the safe set and iteratively removes trajectories that may be forced to exit the set due to disturbance inputs or system dynamics. If the algorithm terminates at a fixed point, this final set is the maximal controlled invariant. A controller can be obtained as a byproduct of the iteration procedure, [4], [5], [6]. A similar iterative fixed point algorithm that can accomodate dwell-time requirements is presented in [7].

An alternative approach to controller synthesis is to create a transition system from the hybrid dynamics by partitioning the state space and introducing transitions between partitions which reflect the dynamics and safety properties of the hybrid system model. The relation between the hybrid system and the new transition system is called a *bisimulation*, and a controller for the original system can be synthesized from this bisimulation, [8], [9]. In a separate line of work [10], an optimal switching problem is solved by synthesizing guards, however safety constraints are not explicitly considered.

In this paper, we use sum of squares (SOS) programming to synthesize switching laws that are guaranteed to satisfy a state-based safety constraint. We consider hybrid systems with a finite number of modes in which the state evolution is governed by a differential inclusion, and we synthesize guards that trigger transitions between modes. Guards are assumed to be *semialgebraic* sets, *i.e.* a guard is a subset of the continuous state space which satisfies a collection of polynomial inequalities and equalities. We fix the number of polynomial inequalities and equalities and the polynomial degree to make the problem tractable. Other applications of SOS programming to control theory include region-of-attraction analysis and Lyapunov function calculation, [11], [12], and hybrid system verification, [2]. See [13] for an overview of SOS.

The focus of this paper is on deciding when to switch between discrete modes. We present a synthesis procedure which results in a bilinear feasibility problem as well as an algorithm for iteratively solving this feasibility problem. Our results rely on knowing the reach set from a given set in a particular mode, or at least an overapproximation of this set. Finding such sets can be very difficult and is an active area of research, see *e.g.* [1], [3], [14]. As discussed in the sequel, some techniques are particularly appropriate for sum of squares programming, *e.g.* [15], [11], [2].

This paper is organized as follows: Section II reviews hybrid systems and SOS programming, and Section III states the problem formulation. Our main result is presented in Section IV, which describes our guard synthesis algorithm. In Section V we present an example, and we offer directions for extending our algorithm in Section VI.

## II. Preliminaries

### A. Hybrid Systems

A *hybrid system* is a tuple $H = (Q, X, \text{Init}, f, R)$ where the total state space $Q \times X$ consists of a finite set $Q$ of "modes" and a continuous state space $X = \mathbb{R}^n$. The system is initialized in a set $\text{Init} \subseteq Q \times X$, and we define $\text{Init}(q) \triangleq \{x : (q, x) \in \text{Init}\}$. We consider differential inclusions, [16], [17], and let $f(\cdot, \cdot) : Q \times X \to \mathcal{P}(\mathbb{R}^n)$ where $\mathcal{P}(\cdot)$ denotes the powerset and $\dot{x}(t) \in f(q, x(t))$ constrains the continuous evolution while in mode $q$. Mild assumptions on $f(q, \cdot)$ guarantee the existence of solutions for all time. This formulation is general and can accomodate, for example, parameter uncertainty or disturbance inputs.

We define the reset map as follows: $R(\cdot, \cdot, \cdot) : Q \times Q \times X \to \mathcal{P}(X)$ where $R(q, q', x) \subseteq X$ is the set of continuous states which can be reached when the system undergoes a transition from discrete state $q$ to $q'$ while at $x \in X$. We denote the domain of $R$ for fixed $q, q'$ by $\mathcal{R}_{qq'} \triangleq \text{Dom}(R(q, q', \cdot)) \subseteq X$. We also assume $R(q, q', \cdot)$ can accept set-valued arguments, *i.e.* for a set $W \subseteq \mathcal{R}_{qq'}$, we define $R(q, q', W) \triangleq \bigcup_{x \in W} R(q, q', x)$.

Note that if a transition from $q$ to $q'$ is not possible, then $\mathcal{R}_{qq'} = \emptyset$. Furthermore, we assume $\mathcal{R}_{qq} = \emptyset \ \forall q \in Q$.

Department of Electrical Engineering and Computer Sciences, University of California, Berkeley {scoogan,arcak}@eecs.berkeley.edu

A *set of guards* for a hybrid system is a collection of sets

$$G = \{G_{qq'}\}_{q,q' \in Q} \subset \mathcal{P}(X)$$

such that $G_{qq'} \subseteq \mathcal{R}_{qq'}$. Each $G_{qq'}$ is called a *guard*. Let $G_q \triangleq \bigcup_{q' \in Q} G_{qq'}$. The purpose of the guards is to trigger transitions. We refer to the pair $(H, G)$ as a *controlled hybrid system*.

An *execution* of a controlled hybrid system $(H, G)$ is a sequence of mode transition times $\{\tau_i\}_{i=1}^N$ with $\tau_0 = 0$, $\tau_i \leq \tau_{i+1}$ along with a state trajectory $(q(t), x(t))$ where $q(t)$ is constant for all $t \in [\tau_i, \tau_{i+1})$, $x(t) \in X \backslash G_{q(t)}$, and $\dot{x}(t) \in f(q(t), x(t))$ for all $t \in [\tau_i, \tau_{i+1})$. We denote the continuous state immediately prior to the $i$th transition by $x(\tau_{i-1}')$, i.e. $x(\tau_{i-1}') \triangleq \lim_{t \to \tau_i^-} x(t)$. We further require $x(\tau_i') \in G_{q(\tau_i)q(\tau_{i+1})}$, and $x(\tau_{i+1}) \in R(q(\tau_i), q(\tau_{i+1}), x(\tau_i'))$ for $i = 1, \ldots, N-1$. If $N = \infty$ but $\tau_N < \infty$, the execution is called *Zeno*.

### B. Sum of Squares Programming

For a variable $x \in \mathbb{R}^n$, we denote by $\mathbb{R}[x]$ the set of all polynomials in $x$. Define

$$\Sigma[x] \triangleq \left\{ s(x) \in \mathbb{R}[x] : s(x) = \sum_{i=1}^m f_i(x)^2, f_i(x) \in \mathbb{R}[x] \right\}.$$

A polynomial $s(x) \in \Sigma[x]$ is called a *sum of squares (SOS)* polynomial. Note that if $s(x) \in \Sigma[x]$, then $s(x) \geq 0$ for all $x$. This important property is used in the sequel as a relaxation of the condition $p(x) \in \mathbb{R}[x]$, $p(x) \geq 0$ for all $x$. Given $\{p_i(x)\}_{i=0}^m$ with $p_i \in \mathbb{R}[x]$, the problem of finding $\{q_i(x)\}_{i=1}^m$ with $q_i(x) \in \mathbb{R}[x]$ (or $q_i(x) \in \Sigma[x]$, or a mix of constraints for different $i$'s) such that

$$p_0(x) + \sum_{i=1}^m q_i(x) p_i(x) \in \Sigma[x] \tag{1}$$

is an LMI feasibility problem [13]. An equation of the form (1) is called an *SOS program*, and the MATLAB toolbox SOSTOOLS [18] transforms such SOS programs into semidefinite programs.

## III. Problem Formulation

Consider an *unsafe set* $U \subseteq Q \times X$ which may include undesirable or physically unattainable regions of the state space. Given a controlled hybrid system $(H, G)$, we call an execution of $(H, G)$ *unsafe* if $(q(t), x(t)) \in U$ for some $t \in [0, \tau_N]$. We call $(H, G)$ *safe* if there does not exist an unsafe execution of $(H, G)$.

**Guard Synthesis Problem.** *Given a hybrid system $H$ and an unsafe set $U \subseteq Q \times X$, synthesize a set of guards $G = \{G_{qq'}\}_{q,q' \in Q}$ such that $(H, G)$ is safe.*

### A. Overapproximations of Reach Sets

For $S \subset X$, we call $\Phi \subset X$ an *overapproximation of the reach set* (o.a.r.s) from $(q, S)$ if $\Phi$ contains all trajectories of the continuous dynamics in mode $q$ that originate in $S$ and have either just encountered a guard or remain in $q$ for all time without encountering a guard. Specifically, $\Phi$ is an o.a.r.s. from $(q, S)$ if the following implication holds for every $T > 0$: if for all $t \in [0, T)$

$$x(0) \in S \text{ and } \dot{x}(t) \in f(q, x(t)) \text{ and } x(t) \in (X \backslash G_q)$$

then $x(t) \in \Phi$ for all $t \in [0, T]$. Let

$$\text{OARS}_q(S) \triangleq \{\Phi : \Phi \text{ is an o.a.r.s. from } (q, S)\}.$$

Note that $\emptyset \in \text{OARS}_q(\emptyset)$. Also, if $S \subset X$ is a positively invariant set for the dynamics $\dot{x} \in f(q, x)$, then $S \in \text{OARS}_q(S)$.

A number of techniques exist for obtaining such over-approximations. For example, in [2], the authors consider scalar-valued "barrier functions" $B_q(x)$ and use the fact that if for all $v(x) \in f(q, x)$

$$\nabla B_q(x)^T v(x) \leq 0 \ \forall x \in (X \backslash G_q), \tag{2}$$

then $\{x : B_q(x) \geq 0\} \in \text{OARS}_q(\{x : B_q(x) \geq 0\})$ and propose a technique for constructing such barrier functions from a basis set of functions using an SOS program. This technique can be incorporated into our approach. We discuss this approach and others for calculating o.a.r.s in the example in Section V, but otherwise do not concern ourselves with the computation of $\Phi \in \text{OARS}_q(S)$.

Propositions 1 and 2 are straightforward, but facilitate later proofs:

**Proposition 1.** *If $\Phi \in \text{OARS}_q(S)$, then $S \subset \Phi$.*

**Proposition 2.** *If $W \subseteq S$, then $\Phi \in \text{OARS}_q(S) \implies \Phi \in \text{OARS}_q(W)$.*

We now characterize a sufficient condition for safety using o.a.r.s. which serves as the foundation for our guard synthesis algorithm.

**Lemma 1.** *Given unsafe $U \subseteq Q \times X$ and a controlled hybrid system $(H, G)$, if there exists $\{\Phi_q\} \subset \mathcal{P}(X)$ such that*

$$\Phi_q \in \text{OARS}_q(\text{Init}(q)) \qquad \forall q \in Q \tag{3}$$

$$\Phi_{q'} \in \text{OARS}_{q'}(R(q, q', \Phi_q \cap G_{qq'})) \quad \forall q, q' \in Q \tag{4}$$

$$(q, \Phi_q) \cap U = \emptyset \qquad \forall q \in Q \tag{5}$$

*then $(H, G)$ is safe.*

*Proof:* Suppose not. Then there exists a time $t^*$ and an execution such that $(q(t^*), x(t^*)) \in U$. It must be that $x(t^*) \notin \Phi_{q(t^*)}$ by (5). Let $i^* \triangleq \max\{i : \tau_i \leq t^*\}$. We have $x(\tau_{i^*}) \notin \Phi_{q(\tau_{i^*})}$ since $\Phi_{q(\tau_{i^*})} = \Phi_{q(t^*)}$ is an o.a.r.s for mode $q(\tau_{i^*})$. But $x(\tau_0) \in \Phi_{q(\tau_0)}$ by condition (3), thus $i^\dagger \triangleq \max\{i : x(\tau_i) \in \Phi_{q(\tau_i)}\}$ is well-defined. We have $x(\tau_{i^\dagger}) \in \Phi_{q(\tau_{i^\dagger})} \implies x(\tau_{i^\dagger}') \in \Phi_{q(\tau_{i^\dagger})}$ by the definition of o.a.r.s and $x(\tau_{i^\dagger}') \in G_{q(\tau_{i^\dagger})q(\tau_{i^\dagger+1})}$ by the definition of an execution. Also, $x(\tau_{i^\dagger+1}) \in R(q(\tau_{i^\dagger}), q(\tau_{i^\dagger+1}), x(\tau_{i^\dagger}'))$. Thus

$$x(\tau_{i^\dagger+1}) \in R(q(\tau_{i^\dagger}), q(\tau_{i^\dagger+1}), \Phi_{q(\tau_{i^\dagger})} \cap G_{q(\tau_{i^\dagger})q(\tau_{i^\dagger+1})}) \subset \Phi_{q(\tau_{i^\dagger+1})}$$

by Proposition 1 and (4). But this contradicts the definition of $i^\dagger$. ∎

Note that while Lemma 1 ensures $(H, G)$ is safe, it does not establish the nonexistence of Zeno executions. Below is a sufficient condition for ruling out this phenomenon.

**Proposition 3.** *If* $\mathbf{cl}(\cup_q \Phi_q)$ *is compact and*

$$\mathbf{cl}(R(q, q', \Phi_q \cap G_{qq'})) \cap \mathbf{cl}(G_{q'}) = \emptyset \quad \forall q, q' \in Q \tag{6}$$

*where* $\mathbf{cl}$ *denotes closure then no executions of* $(H, G)$ *are Zeno.*

*Proof:* When $\mathbf{cl}(\cup_q \Phi_q)$ is compact, we have $\mathbf{cl}(R(q, q', \Phi_q \cap G_{qq'})) \subset \Phi_{q'}$ compact, thus the distance between $\mathbf{cl}(R(q, q', \Phi_q \cap G_{qq'}))$ and $\mathbf{cl}(G_{q'})$ is strictly greater than 0. Since this holds for all $q, q' \in Q$, there is a minimum dwell time and thus Zeno executions are prevented. ∎

We will primarily be interested in systems which do not allow trajectories to become unbounded, thus the assumption that $\mathbf{cl}(\cup_q \Phi_q)$ is compact is reasonable. It is possible to extend Proposition 3 to the case when $\mathbf{cl}(\cup_q \Phi_q)$ is not compact by making additional assumptions on the reset map. Condition (6) can also be relaxed by considering all cycles of the hybrid automaton instead of all transitions, see [6]. It is not difficult to adjust the algorithm presented below (specifically, (11)) for this more general case.

## IV. Guard Synthesis Algorithm

Consider the hybrid system $H$ and unsafe set $U \subset Q \times X$. Let $U(q) \triangleq \{x : (q, x) \in U\}$. We assume $\text{Init}(q)$ can be written as $\text{Init}(q) = \{x \in X : \gamma_{\text{Init}(q)}(x) \succeq 0\}$ where $\gamma_{\text{Init}(q)}(\cdot)$ is a vector-valued, polynomial function, 0 is interpreted as a vector of zeros, and $\succeq$ denotes elementwise inequality (similarly for $\succ$, $\preceq$, and $\prec$). We also assume $\mathcal{R} = \{\mathcal{R}_{qq'}\}$ and $\{U(q)\}_{q \in Q}$ can similarly be described with vector-valued polynomial functions $\gamma_{\mathcal{R}_{qq'}}(x)$ and $\gamma_{U(q)}(x)$. The output dimensions of each polynomial need not be the same, and we denote the dimension of a vector $v \in \mathbb{R}^d$ by $\text{Dim}(v) = d$. Finally, we assume the reset map $R(q, q', \cdot)$ is a vector-valued polynomial function for each $q, q' \in Q$.

We now present a theorem which extends [2] and forms the basis of our guard synthesis algorithm by converting conditions (3)–(6) to SOS programs where feasibility is sufficient for each condition. In particular, (7)–(9) below correspond to (3)–(5), and (11) corresponds to (6). Equation (10) ensures that guard transitions are only taken when in the domain of the reset map. See the end of this section for a comparison of our approach to [2].

**Theorem 1.** *Given a hybrid system* $H$ *and a set of bounded o.a.r.s* $\{\Phi_q\}$ *and sets* $\{S_q\}$ *with* $\Phi_q \in OARS_q(S_q)$ *described by* $\Phi_q \triangleq \{x : \phi_q(x) \succeq 0\}$ *and* $S_q(x) \triangleq \{x : \sigma_q(x) \succeq 0\}$ *with* $\phi_q(x), \sigma_q(x)$ *vector-valued, polynomial functions.*

*Consider a set* $\{g_{qq'}(\cdot)\}$ *of vector-valued, polynomial functions defining a set of guards* $G \triangleq \{G_{qq'}\}_{q,q' \in Q}$ *by* $G_{qq'} \triangleq \{x : \phi_q(x) \succeq 0$ *and* $g_{qq'}(x) \succeq 0\}$.

*If there exists a set of SOS polynomial vectors* $\{s_{i,*}(x)\}_{i=1}^{8}$ *with* $*$ *replaced by elements from an appropriate index set such that*

$$\sigma_q^{(i)}(x) - s_{1,q}^T(x)\gamma_{Init(q)}(x) \in \Sigma[x] \quad \forall i = 1, \ldots, Dim(\sigma_q(x)), \forall q \in Q \tag{7}$$

$$\sigma_{q'}^{(i)}(R(q, q', x)) - s_{2,qq'}^T(x)\phi_q(x) - s_{3,qq'}^T(x)g_{qq'}(x) \in \Sigma[x] \quad \forall i = 1, \ldots, Dim(\sigma_{q'}(x)), \forall q, q' \in Q \tag{8}$$

$$-(1 + s_{4,q}^T(x)\gamma_{U(q)}(x) + s_{5,q}^T(x)\phi_q(x)) \in \Sigma[x] \quad \forall q \in Q \tag{9}$$

$$\gamma_{\mathcal{R}_{qq'}}^{(i)}(x) - s_{6,qq'}^T(x)\phi_q(x) - s_{7,qq'}^T(x)g_{qq'}(x) \in \Sigma[x] \quad \forall i = 1, \ldots, Dim(\gamma_{\mathcal{R}_{qq'}}(x)), \forall q, q' \in Q \tag{10}$$

$$-(1 + s_{8,q'q''}^T(x)g_{q'q''}(R(q, q', x)) + s_{8,q}^T\phi_q(x) + s_{8,qq'}^T(x)g_{qq'}(x)) \in \Sigma[x] \quad \forall q, q', q'' \in Q \tag{11}$$

*then* $(H, G)$ *is safe. Furthermore, no execution of* $(H, G)$*is Zeno.*

*Proof:* We will show that (7)–(11) imply (3)–(6) and conclude the results from Lemma 1 and Proposition 3. Observe that (10) implies

$$\begin{bmatrix} \phi_q(x) \\ g_{qq'}(x) \end{bmatrix} \succeq 0 \implies \gamma_{\mathcal{R}_{qq'}}^{(i)}(x) \succeq 0. \tag{12}$$

Indeed, suppose not for a particular $x'$. Then $\gamma_{\mathcal{R}_{qq'}}^{(i)}(x') - s_{6,qq'}^T(x')\phi_q(x') - s_{7,qq'}^T(x')g_{qq'}(x') < 0$ since $s_{6,*}(x') \geq 0$ and $s_{7,*}(x') \geq 0$, contradicting (10). Since this holds for all $i = 1, \ldots, \text{Dim}(\gamma_{\mathcal{R}_{qq'}}(x))$, we have $G_{qq'} = \{x : \begin{bmatrix} \phi_q^T(x) & g_{qq'}^T(x) \end{bmatrix}^T \succeq 0\} \subseteq \mathcal{R}_{qq'}$ and therefore $G$ is a valid guard set.

(7) $\implies$ (3). Applying reasoning similar to (12), we conclude from (7) that $\gamma_{\text{Init}(q)}(x) \succeq 0 \implies \sigma_q^{(i)}(x) \succeq 0$ for all $q \in Q$ and for all $i = 1, \ldots, \text{Dim}(\sigma_q)$. This implies $S_q \supseteq \text{Init}(q)$ for all $q \in Q$, and we invoke Proposition 2.

(8) $\implies$ (4). Similarly, we conclude from (8)

$$\begin{bmatrix} \phi_q(x) \\ g_{qq'}(x) \end{bmatrix} \succeq 0 \implies \sigma_{q'}^{(i)}(R(q, q', x)) \succeq 0$$

for all $i$, thus $R(q, q', \Phi_q \cup G_{qq'}) \subset S_q$. We again invoke Proposition 2.

(9) $\implies$ (5). We have (9) implies

$$\left\{ x : \begin{matrix} \gamma_{U(q)}(x) \succeq 0 \\ \phi_q(x) \succeq 0 \end{matrix} \right\} \text{ is empty.}$$

Indeed, suppose not and let $\gamma_{U(q)}(x') \succeq 0$ and $\phi_q(x') \succeq 0$. Then $-(1 + s_{4,q}^T(x')\gamma_{U(q)}(x') + s_{5,q}^T(x')\phi_q(x')) < 0$, a contradiction.

Applying Lemma 1, we have that $(H, G)$ is safe. Finally, (11) implies

$$\left\{ x : \begin{bmatrix} \phi_q(x) \\ g_{qq'}(x) \\ g_{q'q''}(R(q, q', x)) \end{bmatrix} \succeq 0 \right\} \text{ is empty } \forall q, q', q'' \in Q$$

which gives (6), thus preventing Zeno executions.

■

A number of remarks are in order:

*Remark* 1. It is sometimes more convenient or necessary to represent $U(q)$ as

$$U(q) = \{x : (\gamma_{1,U(q)}(x) \succeq 0) \vee \ldots \vee (\gamma_{J,U(q)}(x) \succeq 0)\}.$$

For such $U(q)$, we can simply verify (9) for each $\gamma_{j,U(q)}(x)$, $j = 1, \ldots, J$.

*Remark* 2. If a convenient vector-valued polynomial inequality description exists for the safe set (i.e., $\text{Safe}(q) = (q, X \backslash U(q)) = \{x : \gamma_{\text{Safe}(q)}(x) \succeq 0\}$), we can replace (9) with an SOS program of the form:

$$\gamma_{\text{Safe}(q)}(x) - s_{4,q}^T(x)\phi_q(x) \in \Sigma[x] \quad \forall q \in Q.$$

*Remark* 3. If $\Phi_q$ is an invariant set for the dynamics in mode $q$, then we can let $\sigma_q(x) = \phi_q(x)$.

We use Theorem 1 as a guide for synthesizing guards. To make the problem numerically tractable, we fix the degrees of the SOS variables and the guards. We also introduce an iterative procedure solving a convex problem at each stage. Such a procedure is widely used when solving SOS programs related to control, see *e.g.* [12], [11], and [2]. Our iterative procedure involves initially loosening the constraints (7)–(11) and iteratively tightening the constraints. The appropriate constraint relaxation depends on the particular problem, but should be such that starting sets $\{\phi_q(x)\}$, $\{g_{qq'}(x)\}$ are easy to find and may include:

- Relaxed safety conditions, i.e. $\tilde{\gamma}_{U(q)}(x) = \gamma_{U(q)}(x) - c$ for $c > 0$,
- Relaxed reset maps or differential inclusions (as in the example in Section V),
- Considering (7)–(11) separately
- Adding positive constants to the left hand side of (7)–(11). For this approach, the norm of the constant vector can be added as a variable to be minimized since the constants appear affinely in the problem.

We have solved the guard synthesis problem if (7)–(11) is feasible. We use a sequence of relaxed problems $\{\mathtt{R}_i\}_{i=1}^T$ where $\mathtt{R}_i$ is a relaxed version of (7)–(11). For example, $\mathtt{R}_i$ might employ the relaxed safety condition $\tilde{\gamma}_{U(q)}^i(x) = \gamma_{U(q)}(x) - c(2^{-i})$.

Our proposed guard synthesis strategy is as follows:

*Guard Synthesis Algorithm*

---

**Initialize** by constructing a sequence $\{\mathtt{R}_i\}_{i=1}^T$ of relaxed problems as describe above where it is easy to randomly generate or construct by hand guards and reach sets such that $\mathtt{R}_1$ is feasible.

**Fix** $\{\phi_q(x)\}$ and $\{g_{qq'}(x)\}$, and **solve** for feasible SOS variables $\{s_{i,*}(x)\}_{i=1}^8$ satisfying $\mathtt{R}_1$.

**Set** $i \leftarrow 2$

**While** $i \leq T$

1) **Fix** SOS variables, and **solve** for feasible $\{\phi_q(x)\}$, $\{g_{qq'}(x)\}$ and $\{s_{4,*}\}$ satisfying $\mathtt{R}_i$.
2) **Fix** $\{\phi_q(x)\}$ and $\{g_{qq'}(x)\}$, and **solve** for feasible SOS variables $\{s_{i,*}(x)\}_{i=1}^8$ satisfying $\mathtt{R}_i$.
3) **Set** $i \leftarrow i + 1$

**End while**

**Fix** SOS variables, and **solve** for feasible $\{\phi_q(x)\}$, $\{g_{qq'}(x)\}$ and $\{s_{4,*}\}$ satisfying (7)–(11).

---

We note that, as for all bilinear feasibility/optimization problems, the proposed iterative procedure is not guaranteed to generate a feasible solution of the original problem.

Our synthesis algorithm is similar in spirit to the non-convex, worst-case safety verification procedure proposed in [2], however there are some key differences. Our proposed algorithm is a method for synthesizing safe control strategies, while [2] seeks to verify that a given strategy is safe. In addition, [2] verifies scalar-valued barrier functions by checking the flow of the vector field along the boundary of the barrier, specifically condition (2). We do not specify how the o.a.r.s. are obtained, and checking the vector field flow along the barrier is a possible method. However, in principle, o.a.r.s. can be obtained using other methods and we allow for vector-valued o.a.r.s as in the example below.

## V. Example: Two Agent Surveillance

Consider two agents (*e.g.* UAVs) patrolling a one-dimensional region with positions $x_i \in \mathbb{R}$, $i = 1, 2$. Let $x \triangleq \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T$ be the state of the system. Let $\Omega_{\text{surv}} = (-\beta, \beta)$ be the region that the agents are patrolling, see Fig. 1.

Each agent has a *left* traveling mode, and a *right* traveling mode, but we assume a finite time $\tau_i \in [\tau_i^{\min}, \tau_i^{\max}] \subset \mathbb{R}_+$ is required to transition between the two modes for each agent where $\mathbb{R}_+$ (resp. $\mathbb{R}_-$) is the set of positive (resp. negative) real numbers. We assume the velocity in the *right* (resp. *left*) mode for each agent at any particular time is $\dot{x}_i \in [v_i^{\min}, v_i^{\max}] \subset \mathbb{R}_+$ (resp. $\dot{x}_i \in [-v_i^{\max}, -v_i^{\min}] \subset \mathbb{R}_-$).

We model each agent as having four modes: a *left* and *right* mode modeled by differential inclusions, and two *transistion* modes for transitioning between *left* and *right*. We assume that each agent has access to its current mode and the position of the other agent. Thus, the synthesized guards will be functions of $x_1$ and $x_2$.

We consider the scenario *unsafe* if there exists a time at which neither agent is in the surveillance region. Additionally, we assume agents are incapable of surveilling the region while in a mode transition. We relax this condition by enforcing that agents do not undergo mode transitions within the surveillance region.

From this formulation, we can set the problem up as a guard synthesis problem where the hybrid system $H = (Q, X, \text{Init}, f, R)$ is as in Fig. 2 with $Q = \{A, B, C, D\}$, $X = \mathbb{R}^2$, $x \triangleq \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T$, $\text{Init} \subset Q \times X$, $f(\cdot, \cdot) : Q \times X \to \mathcal{P}(\mathbb{R}^2)$ such that

$$\begin{aligned}
f(A, \cdot) &= \left\{ \begin{bmatrix} v_1 & v_2 \end{bmatrix}^T : v_i \in [v_i^{\min}, v_i^{\max}] \right\} \\
f(B, \cdot) &= \left\{ \begin{bmatrix} v_1 & -v_2 \end{bmatrix}^T : v_i \in [v_i^{\min}, v_i^{\max}] \right\} \\
f(C, \cdot) &= \left\{ \begin{bmatrix} -v_1 & -v_2 \end{bmatrix}^T : v_i \in [v_i^{\min}, v_i^{\max}] \right\} \\
f(D, \cdot) &= \left\{ \begin{bmatrix} -v_1 & v_2 \end{bmatrix}^T : v_i \in [v_i^{\min}, v_i^{\max}] \right\},
\end{aligned}$$

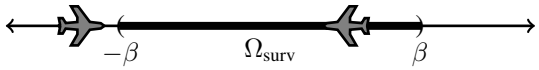


Fig. 1. Two agents are patrolling a one dimensional surveillance region. The scenario is safe if at least one agent is within the surveillance region at all times.

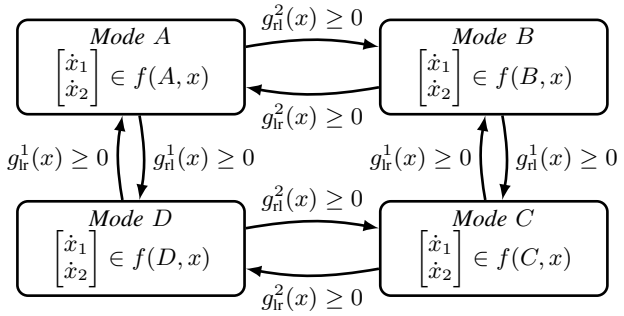


Fig. 2. The system dynamics for the agents in Fig. 1 as a finite state automata. Reset maps are not indicated.

and $R(A,B,x) = R(B,A,x) \triangleq \{\begin{bmatrix}x_1+\tau_1\xi & x_2\end{bmatrix}^T : \xi \in [v_1^{\min}, v_1^{\max}], \tau_1 \in [\tau_1^{\min}, \tau_1^{\max}]\}$ and similarly for $R(B,C,x) = R(C,B,x)$, $R(C,D,x) = R(D,C,x)$, $R(D,A,x) = R(A,D,x)$. Finally, $\mathcal{R}_{qq'} = \{(x_1,x_2) : \gamma_{\mathcal{R}_{qq'}}(x_1,x_2) \geq 0\}$ where $\gamma_{\mathcal{R}_{AB}}(x) = \gamma_{\mathcal{R}_{BA}}(x) = \gamma_{\mathcal{R}_{CD}}(x) = \gamma_{\mathcal{R}_{DC}}(x) \triangleq (x_2^2 - \beta^2)$ and $\gamma_{\mathcal{R}_{DA}}(x) = \gamma_{\mathcal{R}_{AD}}(x) = \gamma_{\mathcal{R}_{BC}}(x) = \gamma_{\mathcal{R}_{CB}}(x) \triangleq (x_1^2 - \beta^2)$.

The unsafe set does not depend on the discrete state and can be formulated in several ways. It proves useful when establishing (10) to use the following formulation:

$$\begin{aligned} U(q) = &\{x : (x_1-\beta) \geq 0, (x_2-\beta) \geq 0\} \quad \cup \\ &\{x : (-x_1-\beta) \geq 0, (x_2-\beta) \geq 0\} \quad \cup \\ &\{x : (-x_1-\beta) \geq 0, (-x_2-\beta) \geq 0\} \cup \\ &\{x : (x_1-\beta) \geq 0, (-x_2-\beta) \geq 0\} \quad \forall q \in Q. \end{aligned}$$

A naïve approach to this problem is to enforce a mode transition as soon as each agent reaches the boundary of the surveillance region. In general, however, this is an unsafe strategy due to the time required for each agent to execute a mode transition. If this strategy is employed, eventually both agents will execute a mode transition just outside the surveillance region, and the scenario is unsafe. Fig. 3 shows an unsafe execution employing this simple scheme.

To synthesize the guards, we seek functions $g^i_{\text{lr}}(x), i = 1,2$ and $g^i_{\text{rl}}(x), i = 1,2$ with

$$\begin{aligned} g_{AB}(x) = g_{DC}(x) &\triangleq g^2_{\text{rl}}(x) \\ g_{BC}(x) = g_{AD}(x) &\triangleq g^1_{\text{rl}}(x) \\ g_{CD}(x) = g_{BA}(x) &\triangleq g^2_{\text{lr}}(x) \\ g_{DA}(x) = g_{CB}(x) &\triangleq g^1_{\text{lr}}(x) \end{aligned}$$

such that $g_{AB}(x) \geq 0 \implies \gamma_{\mathcal{R}_{AB}}(x) \geq 0$, etc.

We can easily describe a positively invariant set $V$ containing $S$ as the intersection of sets satisfying three affine inequalities as shown in Fig. 4. For instance, we

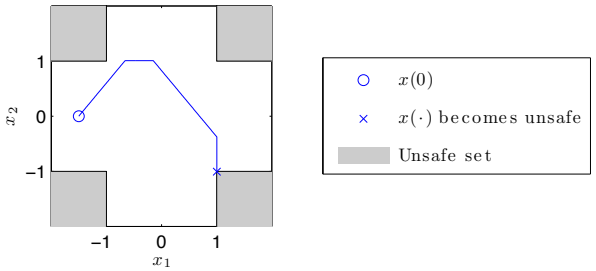


Fig. 3. An execution generated using a straightforward but unsafe control strategy in which each agent initiates a mode transition immediately upon leaving the surveillance region with $\beta = 1$. The trajectory eventually reaches an unsafe state.

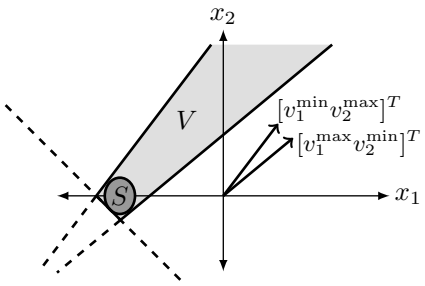


Fig. 4. Overapproximation of the reach set from $S$ in mode $A$.

let $V_A(x) \triangleq \begin{bmatrix}V_A^1(x) & V_A^2(x) & V_A^3(x)\end{bmatrix}^T$ where $V_A^j(x), j = 1,\ldots,3$ are the affine functions describing the positively invariant set. In addition, since trajectories are only possible when guards are not active, we can augment $V_A$ to obtain the final o.a.r.s: $\Phi_A = \{x : \phi_A(x) \succeq 0\}$ where $\phi_A(x) = \begin{bmatrix}V_A(x)^T & -g_{AB}(x) & -g_{AD}(x)\end{bmatrix}^T$. We can easily parameterize the three affine functions for each mode and use SOS to ensure that $\{\phi_A(x) \succeq 0\}$ satisfies the conditions required of an o.a.r.s. Similarly for modes $B$, $C$, and $D$.

For numerical calculations, we let $\text{Init} = \{(A,x) : \gamma_{\text{Init}(A)}(x) \succeq 0\}$ with $\gamma_{\text{Init}(A)} = (x^T - \begin{bmatrix}-1.5 & 0\end{bmatrix})(x - \begin{bmatrix}-1.5 & 0\end{bmatrix}^T) \leq (0.1)^2$, $\beta = 1$, $(v_1^{\min}, v_1^{\max}, v_2^{\min}, v_2^{\max}) = (1.0, 1.1, 1.2, 1.3)$, and $(\tau_1^{\min}, \tau_1^{\max}, \tau_2^{\min}, \tau_2^{\max}) = (0.55, 0.60, 0.45, 0.50)$.

We choose to synthesize guards defined by affine inequalities, thus guards are halfspaces. We initialize the guard synthesis algorithm with o.a.r.s. and guards for the easy case when $\tau_1^{\min} = \tau_1^{\max} = \tau_2^{\min} = \tau_2^{\max} = 0$ and $v_1^{\min} = v_1^{\max} = v_2^{\min} = v_2^{\max} = 1$ and begin the proposed iterative algorithm with relaxed $\{\mathcal{R}_{qq'}\}$. Via the iteration process, we arrive at a solution for the actual problem, shown in Fig 5.

*Remark* 4. Note that since $\Phi_q$ is a positively invariant set for the dynamics in mode $q$, we can in fact conclude that the system is safe for $\text{Init} = (A,\Phi_A) \cup (B,\Phi_B) \cup (C,\Phi_C) \cup (D,\Phi_D)$. Thus, we see that we need a specific $\text{Init} \subset Q \times X$ to begin the guard synthesis algorithm but can conclude safety for a much larger initial set after synthesizing guards and calculating o.a.r.s.

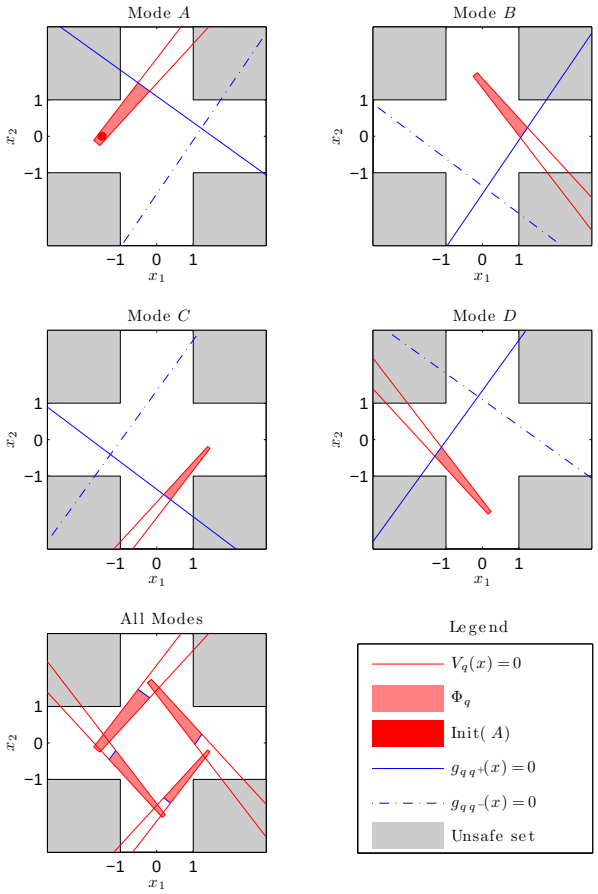


Fig. 5. Overapproximation of reach sets and guards generated by the guard synthesis algorithm. The boundaries of the guards are plotted. The gaps between o.a.r.s. are a result of the time required for each agent to transition from one direction of travel to the other, modeled as reset maps. In the legend, $q^+$ denotes the next mode alphabetically, and $q^-$ denotes the previous mode, with $A^- \triangleq D$ and $D^+ \triangleq A$.

## VI. Conclusions

We present a technique for hybrid system synthesis by designing guards using sum of squares techniques. We employ an iterative algorithm to solve a bilinear SOS program that synthesizes a set of guards satisfying a state-based safety constraint. Our technique relies on describing the guard sets as semialgebraic sets defined by vector-valued polynomial functions found using SOS techniques. While we do not elaborate on the difficult task of computing reach sets in each discrete mode, we do present an example demonstrating a technique for overapproximating these reach sets that is amenable to our SOS approach. Further directions for research include incorporating additional reachability analysis tools into our approach, as well as applying our technique to more complex examples. However, as for all SOS programs, computation time rapidly increases with problem complexity. In addition, it is difficult to incorporate o.a.r.s. that are the union of two or more semialgebraic sets into an SOS program, and these sets often arise in more complex examples. We are also extending our results to include the problem of driving the system to a desired target set while avoiding an unsafe set, and we are investigating specialized bilinear matrix inequality solvers for the bilinear SOS program.

## VII. Acknowledgements

We thank Professor Andy Packard, UC Berkeley, for his insightful comments and suggestions.

This research was supported in part by the Air Force Office of Scientific Research under grant FA9550-11-1-0244. S. Coogan is supported by a National Science Foundation Graduate Research Fellowship.

## References

[1] I. Mitchell and C. Tomlin, "Level set methods for computation in hybrid systems," in *Hybrid Systems: Computation and Control*, vol. 1790 of *Lecture Notes in Computer Science*, pp. 310–323, Springer Berlin / Heidelberg, 2000.

[2] S. Prajna, A. Jadbabaie, and G. Pappas, "A framework for worst-case and stochastic safety verification using barrier certificates," *IEEE Transactions on Automatic Control*, vol. 52, pp. 1415–1428, Aug. 2007.

[3] A. Kurzhanski and P. Varaiya, "Ellipsoidal techniques for reachability analysis: internal approximation," *Systems & Control Letters*, vol. 41, no. 3, pp. 201–211, 2000.

[4] J. Lygeros, C. Tomlin, and S. Sastry, "Controllers for reachability specifications for hybrid systems," *Automatica*, vol. 35, pp. 349–370, 1999.

[5] C. Tomlin, J. Lygeros, and S. Shankar Sastry, "A game theoretic approach to controller design for hybrid systems," *Proceedings of the IEEE*, vol. 88, pp. 949–970, Jul 2000.

[6] E. Asarin, O. Bournez, T. Dang, O. Maler, and A. Pnueli, "Effective synthesis of switching controllers for linear systems," *Proceedings of the IEEE*, vol. 88, pp. 1011–1025, July 2000.

[7] S. Jha, S. Gulwani, S. A. Seshia, and A. Tiwari, "Synthesizing switching logic for safety and dwell-time requirements," in *Proceedings of the 1st ACM/IEEE International Conference on Cyber-Physical Systems*, pp. 22–31, 2010.

[8] E. Haghverdi, P. Tabuada, and G. J. Pappas, "Bisimulation relations for dynamical, control, and hybrid systems," *Theor. Comput. Sci.*, vol. 342, pp. 229–261, Sept. 2005.

[9] A. Girard and G. J. Pappas, "Approximate bisimulation relations for constrained linear systems," *Automatica*, vol. 43, pp. 1307–1317, Aug. 2007.

[10] M. Boccadoro, Y. Wardi, M. Egerstedt, and E. Verriest, "Optimal control of switching surfaces in hybrid dynamical systems," *Discrete Event Dynamic Systems*, vol. 15, pp. 433–448, 2005.

[11] Z. Jarvis-Wloszek, R. Feeley, W. Tan, K. Sun, and A. Packard, "Some controls applications of sum of squares programming," in *Proceedings of the 42nd IEEE Conference on Decision and Control*, vol. 5, pp. 4676–4681, Dec. 2003.

[12] U. Topcu, A. Packard, and P. Seiler, "Local stability analysis using simulations and sum-of-squares programming," *Automatica*, vol. 44, no. 10, pp. 2669–2675, 2008.

[13] P. A. Parrilo, "Semidefinite programming relaxations for semialgebraic problems," *Mathematical Programming Ser. B*, vol. 96, no. 2, pp. 293–320, 2003.

[14] A. Girard, "Reachability of uncertain linear systems using zonotopes," in *Hybrid Systems: Computation and Control*, vol. 3414 of *Lecture Notes in Computer Science*, pp. 291–305, Springer Berlin / Heidelberg, 2005.

[15] F. Blanchini, "Set invariance in control," *Automatica*, vol. 35, no. 11, pp. 1747–1767, 1999.

[16] R. Goebel and A. Teel, "Solutions to hybrid inclusions via set and graphical convergence with stability theory applications," *Automatica*, vol. 42, no. 4, pp. 573–587, 2006.

[17] J. Cortes, "Discontinuous dynamical systems," *IEEE Control Systems Magazine*, vol. 28, pp. 36–73, June 2008.

[18] S. Prajna, A. Papachristodoulou, P. Seiler, and P. A. Parrilo, *SOSTOOLS: Sum of squares optimization toolbox for MATLAB*, 2004.

[19] S. Boyd, L. E. Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*. SIAM, 1994.