Computing Augmented Finite Transition Systems to Synthesize Switching Protocols for Polynomial Switched Systems

Necmiye Ozay, Jun Liu, Pavithra Prabhakar, and Richard M. Murray

Abstract— **This work is motivated by the problem of synthesizing mode sequences for continuous-time polynomial switched systems in order to guarantee that the trajectories of the system satisfy certain high-level specifications expressed in linear temporal logic. We use augmented finite transition systems as abstract models of continuous switched systems. Augmented finite transition systems are equipped with liveness properties that can be used to enforce progress in accordance with the underlying dynamics. We then introduce abstraction and refinement relations that induce a preorder on this class of finite transition systems. By construction, the resulting preorder respects the feasibility (i.e., realizability) of the synthesis problem. Hence, existence of a discrete switching strategy for one of these abstract finite transition systems guarantees the existence of a mode sequence for the continuous system such that all of its trajectories satisfy the specification. We also present an algorithm, which can be implemented using sum-of-squares based relaxations, to compute such high fidelity abstract models in a computationally tractable way. Finally, these ideas are illustrated on an example.**

I. Introduction

Synthesizing switching protocols that determine the sequence in which the modes of a switched system are activated to satisfy certain high-level specifications has attracted a considerable attention in the last decade [2], [16], [29], [6], [20]. The problem of switching protocol synthesis arises in many different contexts. For instance, different modes of a switched system may correspond to the evolution of the system under different, pre-designed feedback controllers [13], [19], so-called motion primitives in robot motion planning [9] or bipedal locomotion [22], or different configurations of a system. Each of these modes may meet certain criteria but not necessarily the complete, mission-level specification the system needs to satisfy. The purpose of the switching protocol is to identify a switching sequence such that the resulting switched system satisfies the mission-level specification.

We consider linear temporal logic (LTL) as a specification language. Abstraction-based hierarchical approaches are common in temporal logic planing [28], [8], [12], [15], [27]. The main workflow of these approaches is, (i) lifting the control synthesis problem to discrete level by constructing a finite transition system that abstracts the behavior of the underlying continuous system, (ii) solving the discrete LTL synthesis problem to construct a strategy, (iii) implementing this strategy at the continuous level. Clearly, a prerequisite for the success of such an approach is the ability to construct high fidelity discrete abstractions in the first step. The main objective of this paper is to provide computationally tractable algorithms to compute abstractions for polynomial switched systems. Moreover, we define a refinement relation that can be used to incrementally compute better (in a sense to be made clear later in the paper) abstractions.

As opposed to abstraction-based frameworks that allow a rich (e.g., unconstrained) control input [28], [17], steering the system is fairly challenging in switched systems due to the limited control authority (i.e., only control input is the mode of the switched system). This difficulty usually manifests itself as non-determinism in the abstract finite transition system unless additional assumptions like incremental stability of the switched system are made [6]. In this paper, we do not make any stability assumptions on the dynamics and we compute non-deterministic finite transition systems, for which discrete synthesis problem can be cast as a two-player LTL game [20]. To improve the descriptive power of the abstractions, we propose to augment the finite transition systems with additional liveness conditions that encode transience properties of the underlying dynamics and that enforce progress accordingly. The discrete synthesis problem for these augmented transition systems can also be recast as a two-player game. Ideas similar to augmented transition systems have been considered for verification [14], [4], but they have not been extensively utilized before for hybrid controller synthesis with [32] being an exception where progress properties are implicitly enforced by the synthesis algorithm for discrete-time piecewise affine systems.

Another salient feature of the proposed method is the ability to handle polynomial dynamics. Existing abstraction techniques often limit the dynamics of the systems considered, for instance, to fully-actuated [17], linear [2], [15], piecewise affine [32] or multi-affine [12], for which efficient algorithms for constructing the discrete transition systems have been developed. As for abstracting systems with polynomial dynamics, one can either resort to exact polynomial algebra [30] with prohibitive computational complexity, or use linearization and error-bounds to locally approximate the nonlinear dynamics [10] with some loss of accuracy. In this paper, we propose to use sum-of-squares (SOS) based convex relaxations to perform the abstraction and show with an example that these relaxations achieve a good trade-off between computational complexity and quality of the abstract model.

This work was supported in part by IBM and UTC through iCyPhy Consortium, the FCRP consortium through MuSyC and a Caltech Center for the Mathematics of Information Fellowship.

NO and RMM are with the Computing and Mathematical Sciences Department, California Institute of Technology, Pasadena, CA 91125, USA. JL is with the Department of Automatic Control and Systems Engineering, University of Sheffield, Sheffield S1 3JD, UK. PP is with IMDEA Software Institute, Madrid, Spain. Correspondence: necmiye@caltech.edu

II. Preliminaries

A. Notation

In this section, the notation used in the paper is summarized. We denote by $\mathbb{R}$, the real numbers. $\mathbb{R}^n$ denotes the n dimensional Euclidean space. $\mathbb{R}^+$ ($\mathbb{Z}^+$) is the nonnegative reals (integers). For a given set X, 2^X is its power set, and $|X|$ is its cardinality. For two sets, X_1, X_2, $X_1 \setminus X_2$ stands for the set difference. Let $X \subseteq \mathbb{R}^n$. The dimension, closure and boundary of X are denoted by $dim(X)$, $cl(X)$, and ∂X, respectively. For a point x in a convex polytope $X \subseteq \mathbb{R}^n$, the tangent cone and the normal cone of X at x are defined as in [26] and denoted respectively by $T_X(x)$ and $N_X(x)$. Finally, $\mathcal{C}^1$ stands for continuously differentiable functions.

B. Linear temporal logic

LTL has two kinds of operators: logical connectives and temporal modal operators. The logic connectives are those used in propositional logic: *negation* ($\neg$), *disjunction* ($\vee$), *conjunction* ($\wedge$) and *material implication* ($\rightarrow$). The temporal modal operators include *next* ($\bigcirc$), *always* ($\Box$), *eventually* ($\Diamond$) and *until* ($\mathcal{U}$). An LTL formula over a finite set AP of atomic propositions can be defined inductively as follows:

1) any atomic proposition $\pi \in AP$ is an LTL formula; and
2) given LTL formulas φ and ψ, $\neg\varphi$, $\varphi \vee \psi$, $\bigcirc\varphi$ and $\varphi\, \mathcal{U}\, \psi$ are also LTL formulas.

Other operators can be defined as follows: (i) $\varphi \wedge \psi \triangleq \neg(\neg\varphi \vee \neg\psi)$, (ii) $\varphi \rightarrow \psi \triangleq \neg\varphi \vee \psi$, (iii) $\Diamond\varphi \triangleq$ *True* $\mathcal{U}\, \varphi$, and (iv) $\Box\varphi \triangleq \neg\Diamond\neg\varphi$.

Semantics of LTL: An LTL formula is interpreted over ω-*words*, i.e., infinite sequences in 2^{AP}, where each element of the sequence is called a *letter*. Given such a word $w = w(0)w(1)w(2)\ldots$ and an LTL formula φ, we say that φ *holds at position* $i \geq 0$ of w, written $(w, i) \models \varphi$, if and only if (iff) φ holds for the rest of the word w starting at position i. The semantics of LTL is defined inductively as follows: (i) For an atomic proposition $p \in AP$, $(w, i) \models p$ iff $p \in w(i)$; (ii) $(w, i) \models \neg\varphi$ iff $(w, i) \not\models \varphi$; (iii) $(w, i) \models \varphi \vee \psi$ iff $(w, i) \models \varphi$ or $(w, i) \models \psi$; (iv) $(w, i) \models \bigcirc\varphi$ iff $(w, i+1) \models \varphi$; and (v) $(w, i) \models \varphi\, \mathcal{U}\, \psi$ iff there exists $j \geq i$ such that $(w, j) \models \psi$ and $\forall k \in [i, j), (w, k) \models \varphi$.

Based on this definition, $\bigcirc\varphi$ holds at position i of w iff φ holds at position $i+1$, $\Box\varphi$ holds at position i iff φ holds at every position in w starting at position i, and $\Diamond\varphi$ holds at position i iff φ holds at some position $j \geq i$ in w. The word w is said to satisfy a formula φ, denoted by $w \models \varphi$, iff it satisfies the formula at the initial position, i.e., $(w, 0) \models \varphi$.

LTL without next step: When interpreting a continuous time signal, the notion of next time step does not exist. Hence we consider the fragment of LTL without the next operator to specify properties about continuous time signals, which we denote by LTL$_{\setminus\bigcirc}$. Given a word w, any word w' obtained by replacing any non-empty finite sequence of identical letters by another non-empty finite sequence of the same letter is said to be stutter equivalent to the word w. A word w satisfies an LTL$_{\setminus\bigcirc}$ formula φ if and only if any word that is stutter equivalent to w also satisfies φ. Hence it suffices to consider the words with no repeating letters (maybe except a constantly repeating final letter), which are called *stutter-free* words, to reason about LTL$_{\setminus\bigcirc}$ [3].

III. Problem setup

In this section, we introduce the main components of the problem, namely augmented finite transition systems and continuous-time switched systems. We also give an overview of the solution methodology for switching protocol synthesis problem.

A. Finite transition systems

A *finite transition system* is a tuple $\mathcal{T} = (Q, Q_0, \mathcal{A}, \rightarrow_{\mathcal{T}}, \Pi, L)$ where Q is the finite set of states, $Q_0 \subseteq Q$ is the set of initial states, $\mathcal{A}$ is the finite set of actions (i.e., control inputs), $\rightarrow_{\mathcal{T}} \subseteq Q \times \mathcal{A} \times Q$ is a transition relation, Π is the set of atomic propositions and $L : Q \rightarrow 2^{\Pi}$ is a labeling function respectively. We assume, without loss of generality, that all actions are enabled at every state, that is, for all $q_1 \in Q$ and for all $a \in \mathcal{A}$, there exists at least one $q_2 \in Q$ such that $(q_1, a, q_2) \in \rightarrow_{\mathcal{T}}$.

An *execution* ρ of a finite transition system $\mathcal{T} = (Q, Q_0, \mathcal{A}, \rightarrow_{\mathcal{T}}, \Pi, L)$ is a sequence of pairs $\rho = (q_0, a_0)(q_1, a_1)(q_2, a_2)\cdots$, where $q_0 \in Q_0$ and $(q_i, a_i, q_{i+1}) \in \rightarrow_{\mathcal{T}}$ for all $i \geq 0$. The *word* produced by an execution ρ is $w_\rho = w_\rho(0)w_\rho(1)w_\rho(2)\cdots$, where $w_\rho(i) = (L(q_i), a_i)$ for all $i \geq 0$[1]. An execution ρ is said to satisfy an LTL formula φ over $2^{\Pi \times \mathcal{A}}$, written $\rho \models \varphi$, if and only if the word it produces satisfies φ. If all executions of $\mathcal{T}$ satisfy φ, we say that the finite transition system $\mathcal{T}$ satisfies φ and write $\mathcal{T} \models \varphi$.

An *augmented finite transition system* is a tuple $\mathcal{T}_{aug} = (Q, Q_0, \mathcal{A}, \rightarrow_{\mathcal{T}}, \Pi, L, \mathcal{G})$ (also denoted by $\mathcal{T}_{aug} = (\mathcal{T}, \mathcal{G})$, for short), where $\mathcal{T} = (Q, Q_0, \mathcal{A}, \rightarrow_{\mathcal{T}}, \Pi, L)$ is a finite transition system and $\mathcal{G} : \mathcal{A} \rightarrow 2^{2^Q}$ is a *progress group map*. The progress group map $\mathcal{G}$ maps each action $a \in \mathcal{A}$ to a set of subsets of Q such that the system cannot remain indefinitely within the set of states $G \in \mathcal{G}(a)$ by using only the action a. A set $G \in \mathcal{G}(a)$ is called a *progress group under action* a. Executions and words of augmented finite transition systems are defined similar to those of finite transition systems. Given an execution $\rho = (q_0, a_0)(q_1, a_1)(q_2, a_2)\cdots$ of an augmented transition system, we also define *extended word* $w^e_\rho = w^e_\rho(0)w^e_\rho(1)w^e_\rho(2)\cdots$, where $w^e_\rho(i) = (L(q_i), a_i, q_i)$ for all $i \geq 0$. The progress group map $\mathcal{G}$ imposes that executions of $\mathcal{T}_{aug}$ satisfy the following LTL formula:

$$\varphi_g \doteq \bigwedge_{a \in \mathcal{A}} \bigwedge_{G \in \mathcal{G}(a)} \neg\Diamond\Box((\vee_{q \in G} q) \ \wedge \ a) \qquad (1)$$

interpreted over extended words[2]. Put differently, under the action a, the system states should always eventually progress outside of the set $G \in \mathcal{G}(a)$. This is an additional information

[1] To be precise, each letter $w(i)$ of a word w should be in $2^{\Pi \times \mathcal{A}}$. However, since the actions in $\mathcal{A}$ are mutually exclusive, we define the letters in $2^{\Pi} \times \mathcal{A}$ for simplicity.

[2] With a similar notational abuse as in footnote 1, the letters $w^e(i)$ of extended words are defined in $2^{\Pi} \times \mathcal{A} \times Q$ for simplicity.

about the executions of the finite transition system that can not be encoded simply by the transition relation. Therefore, augmented finite transition systems have more descriptive power than usual finite transition systems.

Finite transition systems considered in this paper are non-deterministic, that is, from a given state with a given action, there are multiple states the system can transition to. Control synthesis for such systems can be seen as finding a winning strategy for a game between the controller and an "adversary" [11], [5]. In the game, the controller chooses the actions, hence limits the possible next states to a certain set encoded by the transition relation, to satisfy a specification φ; and the adversary resolves the non-determinism and determines the next state from this set in order to falsify φ. Formally, a *control strategy* for a (augmented) transition system $\mathcal{T}$ is a partial function $s : (q_0, a_0, \cdots, q_{i-1}, a_{i-1}, q_i) \mapsto a_i$ that maps the execution history to the next action. An *s-controlled execution* of a transition system $\mathcal{T}$ is an execution of $\mathcal{T}$, where for each $i \geq 0$, the action a_i is chosen according to the control strategy s.

Problem 1: **[Discrete Synthesis]** Given a finite transition system $\mathcal{T}$ and an LTL formula φ over $2^{\Pi \times \mathcal{A}}$, synthesize a control strategy s that generates only correct executions in the sense that all s-controlled executions satisfy the specification φ.

The specification φ is said to be *realizable* on $\mathcal{T}$ if and only if there exists a control strategy s so that the specification is satisfied for all controlled executions (i.e., discrete synthesis problem has a solution). Such a strategy s is called a *winning strategy* for the pair $(\mathcal{T}, \varphi)$.

A way to recast Problem 1 as a two-player game is discussed in [20]. For an augmented finite transition system $\mathcal{T}_{aug} = (\mathcal{T}, \mathcal{G})$, the discrete synthesis problem with a specification φ can be reduced to a discrete synthesis problem for $\mathcal{T}$ with the specification $\varphi_{aug} \doteq \varphi_g \to \varphi$, where φ_g is defined as in Eq. (1). Hence it is possible to adopt the method in [20] to solve the discrete synthesis problem for augmented finite transition systems with this reduction. It is worth remarking that if the original specification φ is in the GR(1) fragment of LTL, an expressive class of LTL formulas for which computationally efficient algorithms for solving the discrete synthesis problem are available [5], the augmented specification φ_{aug} is also in the GR(1) fragment. Hence, in this case, the complexity class of solving the discrete synthesis problem for finite transition systems and augmented finite transition systems are the same.

B. Continuous-time polynomial switched systems

A *continuous-time polynomial switched system* is a tuple $\mathcal{S} = (X, X_0, \mathcal{A}, \{f_a\}_{a \in \mathcal{A}}, \Pi, h)$, where $X \subseteq \mathbb{R}^n$ is a compact convex domain (i.e., state space), $X_0 \subseteq X$ is a set of initial states, $\mathcal{A} \doteq \{1, \ldots, s\}$ is a finite index set counting the modes of the system, $\{f_a\}_{a \in \mathcal{A}}$ is a family of vector fields (in particular, for each mode $a \in \mathcal{A}$, $f_a : \mathbb{R}^n \to \mathbb{R}^n$ is a polynomial vector field), $\Pi \doteq \{\pi_{init}, \pi_{out}, \pi_1, \ldots, \pi_{n_p}\}$ is a set of atomic propositions, and $h : \mathbb{R}^n \to 2^\Pi$ is an observation map. The observation map is assumed to satisfy (i) for all $i \in \{1, \ldots, n_p\}$, $\pi_i \in h(\xi)$ if and only if $\xi \in \mathcal{X}_i$, where each $\mathcal{X}_i \subseteq X$ is a convex polytope; (ii) $\pi_{init} \in h(\xi)$ if and only if $\xi \in X_0$; (iii) for all $\xi \in \mathbb{R}^n \setminus X$, $h(\xi) = \{\pi_{out}\}$. In other words, h associates each $\xi \in \mathbb{R}^n$ with the set of atomic propositions that hold at ξ.

The evolution of the system $\mathcal{S}$ is governed by:

$$\dot{x}(t) = f_{\sigma(t)}(x(t)), \tag{2}$$

where $x(t) \in X \subseteq \mathbb{R}^n$ is the state and $\sigma(t) \in \mathcal{A}$ is the mode of the system at time t. We assume switching signal $\sigma : \mathbb{R}^+ \to \mathcal{A}$ is piecewise constant with finite number of discontinuities on every bounded interval. Given an initial condition $x(0) \in X_0$ and a switching signal σ, they together define a unique *state trajectory* $x : I \to X$ that satisfies (2) for all $t \in I$, where $I = \mathbb{R}^+$ if $x(t) \in X$ for all t, or $I = [0, t_f]$ where t_f is the minimum t such that $x(t) \in \partial X$ and $f_{\sigma(t)}(x(t)) \notin T_X(x)$. We need to define the *word* generated by a given solution (x, σ) of the system $\mathcal{S}$, over which satisfiability of a given LTL$_{\setminus \bigcirc}$ formula by (x, σ) is interpreted. Roughly speaking, the word produced by a solution (x, σ) is defined by finding a sequence of maximal intervals $I_k \subseteq \mathbb{R}^+$ on which $(h(x(t)), \sigma(t))$ is constant for all $t \in I_k$ and representing each such interval with a letter $w(k) = (h(x(t)), \sigma(t))$ for some $t \in I_k$ if I_k is bounded, and repeating the letter $w(k)$ infinitely if $(h(x(t)), \sigma(t))$ is constant for all $t > \inf I_k$, or if the trajectory is of finite length (i.e., $I = [0, t_f]$).

The problem of switching protocol synthesis from a temporal logic specification can be formally stated as follows.

Problem 2: **[Continuous Switching Synthesis]** Given a switched system $\mathcal{S} = (X, X_0, \mathcal{A}, \{f_a\}_{a \in \mathcal{A}}, \Pi, h)$, and an LTL$_{\setminus \bigcirc}$ formula φ over $2^{\Pi \times \mathcal{A}}$, find a state-feedback mode signal σ such that the solutions, (x, σ), of $\mathcal{S}$ satisfy φ for all $x(0) \in X_0$.

C. Problem statement

Continuous switching synthesis problem can be solved through a hierarchical approach consisting of four steps: (i) establish finite transition systems which abstract the continuous system $\mathcal{S}$; (ii) formulate a discrete synthesis problem based on this finite abstraction; (iii) synthesize a switching protocol by solving the discrete synthesis problem; and (iv) if a switching protocol can be found in step (iii), implement it on the continuous level.

This approach relies on principled and efficient ways of computing representative discrete transition systems that abstract the behavior of the continuous system $\mathcal{S}$ in the first step, which is the focus of the current paper. In particular, we will use augmented finite transition systems as abstract models and propose a way to compute such models. For details of steps (ii)-(iv), we refer the reader to [20].

IV. Abstraction and Refinement Relations

In this section we define certain relations among augmented transition systems and switched systems. These relations lead to an order that respects the achievable behaviors of the systems. We start by introducing state-space partitions and transience property, two notions relevant in associating a switched system with an augmented finite transition system.

1) Proposition preserving partitions: Given a switched system $\mathcal{S}$ and a partition $P = \{\mathcal{P}_i\}_{i=1}^{N}$ of its domain X, the partition P is said to be *proposition preserving* if all states from a given cell $\mathcal{P}_i$ have the same label, that is, for all $i \in \{1,\ldots,N\}$ and for all $\xi_1, \xi_2 \in X$, if both $\xi_1, \xi_2 \in \mathcal{P}_i$, then $h(\xi_1) = h(\xi_2)$. A proposition preserving partition induces a surjective function α, called a *partitioning function*, from $\mathbb{R}^n$ to an arbitrary set of cardinality $N+1$ as follows. Let $\{1,\ldots,N+1\}$ be the range of α, then:

$$\alpha(\xi) = \left\{ \begin{array}{cl} i & \text{if } \xi \in \mathcal{P}_i \text{ for some } i \\ N+1 & \text{otherwise.} \end{array} \right. \tag{3}$$

By definition α is proposition preserving in the sense that for all $\xi_1, \xi_2 \in \mathbb{R}^n$, $\alpha(\xi_1) = \alpha(\xi_2) \Rightarrow h(\xi_1) = h(\xi_2)$. The partitioning function α will be used in associating the states of switched system with those of the finite transition system.

2) Transience: An important concept related to progress in discrete transitions is transience in the continuous dynamics. Given a switched system $\mathcal{S} = (X, X_0, \mathcal{A}, \{f_a\}_{a\in\mathcal{A}}, \Pi, h)$, we say that a set $Y \subseteq X$ is *transient* on mode $a \in \mathcal{A}$ if and only if for any trajectory starting from some state $\xi_0 \in Y$, there exists a finite time τ such that the solution of $\dot{x}(t) = f_a(x(t))$ leaves Y at time τ (i.e., there exists $\tau < \infty$ such that $x(\tau) \notin Y$). An equivalent characterization of transience is given in terms of positively invariant sets.

Proposition 1: Given $\mathcal{S}$, a set $Y \subseteq X$ is transient on mode $a \in \mathcal{A}$ if and only if Y does not contain a positively invariant set for subsystem a.

Proof: ($\Longrightarrow$) This direction is trivial. For ($\Longleftarrow$), assume Y is not transient on mode a and it does not contain a positively invariant set for subsystem a. Since Y is not transient, there exists $\xi_0 \in Y$ such that the complete state trajectory x of subsystem a of $\mathcal{S}$ on $\mathbb{R}^+$ starting from ξ_0 is contained in Y. However, this means the trajectory (or, more precisely, the set $\{\xi | x(t) = \xi \text{ for some } t\} \subseteq Y$) is a positively invariant set that is contained in Y which is a contradiction. ■

We next define the type of finite-state approximations (abstractions) of switched systems considered in this paper.

Definition 1: An augmented finite transition system $\mathcal{T} = (Q, Q_0, \mathcal{A}, \to_{\mathcal{T}}, \Pi, L, \mathcal{G})$ is said to be an *over-approximation* for the switched system $\mathcal{S} = (X, X_0, \mathcal{A}, \{f_a\}_{a\in\mathcal{A}}, \Pi, h)$, denoted by $\mathcal{T} \underset{\text{O.A.}}{\succeq} \mathcal{S}$, if there exists a proposition preserving partitioning function $\alpha : \mathbb{R}^n \to Q$ such that the following statements hold.

(i) Given $q \in Q$, $L(q) = h(\xi)$ for some $\xi \in \alpha^{-1}(q)$.

(ii) For all $q \in Q_0$, $\pi_{init} \in L(q)$. There exists a unique $q_{out} \in Q$ such that $\pi_{out} \in L(q_{out})$ and for some $\xi \in \mathbb{R}^n \setminus X$, we have $\alpha(\xi) = q_{out}$.

(iii) Given states $q, q' \in Q$, $q \neq q_{out}$, there is a transition $(q, a, q') \in\to_{\mathcal{T}}$, if there exist $\xi_0 \in \alpha^{-1}(q)$ and $\tau > 0$ such that the corresponding trajectory x of the subsystem f_a starting from ξ_0, i.e., $x : [0,\tau] \to \mathbb{R}^n$ with $x(0) = \xi_0$ and $\dot{x}(t) = f_a(x(t))$, for all $t \in (0,\tau)$ satisfies

$$x(\tau) \in \alpha^{-1}(q') \quad x(t) \in \alpha^{-1}(q) \cup \alpha^{-1}(q'), \quad t \in [0,\tau].$$

For all $a \in \mathcal{A}$, $(q_{out}, a, q_{out}) \in\to_{\mathcal{T}}$.

(iv) The progress group map $\mathcal{G}$ is such that given an action $a \in \mathcal{A}$, for all $G \in \mathcal{G}(a)$, the set $\bigcup_{q\in G} \alpha^{-1}(q)$ is transient on mode a of $\mathcal{S}$.

In the above definition, statements (i) and (ii) say that if a state q of the finite transition system $\mathcal{T}$ is associated with a state ξ of the switched system $\mathcal{S}$ via the abstraction map, the propositions corresponding to their labels and observations, respectively, should be the same. Statement (iii) intuitively means there is a transition $(q, a, q') \in\to_{\mathcal{T}}$ in the finite transition system if there is a corresponding trajectory of subsystem a of the switched system implementing that transition without visiting extra cells. Finally, statement (iv) encodes the transience properties of the continuous switched system, and it means if the map $\mathcal{G}$ imposes a progress property on a set of states of $\mathcal{T}$ on a given action a, the corresponding states of $\mathcal{S}$ should form a transient set on mode a. Encoding such properties is important when designing controllers for liveness specifications since this can help eliminating words generated by repeating spurious cycles (or self-transitions) indefinitely in the discrete transition system. Given an over-approximation $\mathcal{T}$ of a switched system $\mathcal{S}$, we also call $\mathcal{T}$ an abstract model for $\mathcal{S}$.

The next definition gives a relation between two augmented finite transition systems.

Definition 2: Given two augmented finite transition systems $\hat{\mathcal{T}} = (\hat{Q}, \hat{Q}_0, \mathcal{A}, \to_{\hat{\mathcal{T}}}, \Pi, \hat{L}, \hat{\mathcal{G}})$ and $\mathcal{T} = (Q, Q_0, \mathcal{A}, \to_{\mathcal{T}}, \Pi, L, \mathcal{G})$, $\mathcal{T}$ is said to be a *refinement* of $\hat{\mathcal{T}}$ (or, $\hat{\mathcal{T}}$ is an *abstract model* of $\mathcal{T}$), denoted by $\hat{\mathcal{T}} \underset{\text{A.S.}}{\succeq} \mathcal{T}$, if there exists a function $\beta : Q \to \hat{Q}$ such that the following conditions hold.

(i) For all $q \in Q$, $L(q) = \hat{L}(\beta(q))$.

(ii) For all $q \in Q_0$, $\beta(q) \in \hat{Q}_0$.

(iii) For all $(q_1, a, q_2) \in\to_{\mathcal{T}}$, $(\beta(q_1), a, \beta(q_2)) \in\to_{\hat{\mathcal{T}}}$.

(iv) For all $a \in \mathcal{A}$, for all $\hat{G} \in \hat{\mathcal{G}}(a)$, there exists $G \in \mathcal{G}(a)$ such that for all $\hat{q} \in \hat{G}$, we have $\beta^{-1}(\hat{q}) \subseteq G$.

If we ignore the progress group maps and consider usual finite transition systems, statements (i)-(ii)-(iii) in Def. 2, say that the function β is an abstraction function, that is the states of $\hat{\mathcal{T}}$ can be seen as aggregations of disjoint sets of equally labeled states in $\mathcal{T}$ (see [7], [3]). For every transition in the refinement $\mathcal{T}$, there is a corresponding transition in the abstract model $\hat{\mathcal{T}}$, but we allow $\hat{\mathcal{T}}$ to have additional transitions (i.e., more non-determinism). On the other hand, statement (iv) means that for each progress group $\hat{G}$ of the abstract model $\hat{\mathcal{T}}$, there is a corresponding progress group G of the refinement $\mathcal{T}$, that contains all the states in Q related (via β^{-1}) to those in $\hat{G}$.

Definition 3: Given a switched system $\mathcal{S} = (X, X_0, \mathcal{A}, \{f_a\}_{a\in\mathcal{A}}, \Pi, h)$ and an augmented transition system $\mathcal{T} = (Q, Q_0, \mathcal{A}, \to_{\mathcal{T}}, \Pi, L, \mathcal{G})$ such that $\mathcal{T} \underset{\text{O.A.}}{\succeq} \mathcal{S}$, a transition system $\mathcal{T}' = (Q', Q'_0, \mathcal{A}, \to_{\mathcal{T}'}, \Pi, L', \mathcal{G}')$ is called an *abstract model of $\mathcal{S}$ refined with respect to $\mathcal{T}$* if $\mathcal{T} \underset{\text{A.S.}}{\succeq} \mathcal{T}' \underset{\text{O.A}}{\succeq} \mathcal{S}$.

We call $\mathcal{T}'$ a *refinement of $\mathcal{T}$*, for short, whenever $\mathcal{S}$ is clear from context.

We summarize some important properties of the relations introduced in this section with two proposition.

Proposition 2: Given $\hat{\mathcal{T}} \underset{\text{A.S.}}{\succeq} \mathcal{T}$, the set of all words that can be generated by $\mathcal{T}$ is a subset of those that can be generated by $\hat{\mathcal{T}}$. Moreover, given an LTL formula φ over $2^{\Pi \times \mathcal{A}}$, if it is realizable on $\hat{\mathcal{T}}$, then it is realizable on $\mathcal{T}$.

Proof: First we show that for any execution $\rho = (q_0, a_0)(q_1, a_1)(q_2, a_2) \cdots$ of the refinement $\mathcal{T}$, there exists an execution of $\hat{\mathcal{T}}$ of the form $\hat{\rho} = (\beta(q_0), a_0)(\beta(q_1), a_1)(\beta(q_2), a_2) \cdots$. This follows from conditions (ii)-(iii) in Def. 2 and condition (iv), which says $\mathcal{G}$ restricts the infinite executions of $\mathcal{T}$ at least as much as $\hat{\mathcal{G}}$ restricts the infinite executions of $\hat{\mathcal{T}}$. Finally noting that ρ and $\hat{\rho}$ generate the same words according to condition (i) of Def. 2 proves the first part. For the second part, let $\hat{s} : (\hat{q}_0, a_0, \cdots, \hat{q}_{i-1}, a_{i-1}, \hat{q}_i) \mapsto a_i$ be winning strategy for the pair $(\hat{\mathcal{T}}, \varphi)$, then the strategy s for $\mathcal{T}$ defined as $s(q_0, a_0, \cdots, q_{i-1}, a_{i-1}, q_i) = \hat{s}(\beta(q_0), a_0, \cdots, \beta(q_{i-1}), a_{i-1}, \beta(q_i))$ is a winning strategy for $(\mathcal{T}, \varphi)$. This follows by noting that the words generated by the s-controlled executions of $\mathcal{T}$ is a subset of the words generated by $\hat{s}$-controlled executions of $\hat{\mathcal{T}}$ by conditions (i)-(ii)-(iii) of Def. 2. ■

Proposition 3: Given $\mathcal{T} \underset{\text{O.A.}}{\succeq} \mathcal{S}$, the set of all words that can be generated by $\mathcal{S}$ is a subset of stutter-free versions of words that can be generated by $\mathcal{T}$. Moreover, given an LTL$_{\setminus \bigcirc}$ specification φ over $2^{\Pi \times \mathcal{A}}$, if there exists a discrete strategy that realizes φ on $\mathcal{T}$, then[3] there exists a switching protocol that realizes φ on $\mathcal{S}$.

Proof: Let (x, σ) be a solution to $\mathcal{S}$ and the word generated by (x, σ) be $w = w(1)w(2) \cdots$. By definition, w is a stutter-free word. Now let α be the partitioning function in Def. 1 and let $\rho = \rho(1)\rho(2) \cdots$, where $\rho(i) = (q_i, a_i)$, and ρ is the exact sequence of values taken by the pairs $(\alpha(x(t)), \sigma(t))$ on a sequence of countably many non-overlapping intervals $\{I_i\}$ covering the domain of the solution, over which the value of $(\alpha(x(t)), \sigma(t))$ is constant (if the solution is defined on a finite interval, $[0, t_f]$, there are finitely many such intervals but ρ can be completed to an infinite sequence by repeating $\rho(i) = (q_{out}, \sigma(t_f))$). By Def. 1, ρ is an execution of $\mathcal{T}$. Let $w' = w'(1)w'(2) \cdots$ be the word generated by ρ, i.e., $w'(i) = (L(q_i), a_i)$ for all i. On each of the interval I_i, we have $(h(x(t)), \sigma(t)) = (L \circ \alpha(x(t)), \sigma(t)) = w'(i)$ for all $t \in I_i$. Therefore, w is a stutter-free version of w', which proves the first part. The second part proceeds along the lines of proof of Prop. 2 and follows from Theorem 1 in [21]. ■

Definitions 2 and 3 together establish a preorder between the over-approximations of a switched system $\mathcal{S}$. It follows from Prop. 2 that this preorder respects the realizability of the switching synthesis problem. Therefore, given two over-approximation $\hat{\mathcal{T}}$ and $\mathcal{T}$ of $\mathcal{S}$, if $\hat{\mathcal{T}}$ is a refined abstract model of $\mathcal{S}$ with respect to $\mathcal{T}$, then we can deduce that $\hat{\mathcal{T}}$ is a better abstraction in the sense that given any LTL (LTL$_{\setminus \bigcirc}$) formula φ, whenever we can find a strategy for realizing φ on $\mathcal{T}$, we can also find a strategy realizing φ on $\hat{\mathcal{T}}$.

V. Computation of Abstractions

In this section, we present an explicit algorithm for computing over-approximations, argue the correctness of this algorithm and discuss how similar ideas can be used to compute refinements.

Given a switched system $\mathcal{S}$ and a proposition preserving partition P, an over-approximation of $\mathcal{S}$ in the sense of Def. 1 can be computed using Alg. 1. The two key subroutines in this algorithm are $isBlocked$ and $isTransient$. For the correctness of the algorithm we require the following.

R.1) $isBlocked(\mathcal{P}_1, \mathcal{P}_2, f)$: Given two convex polytopes $\mathcal{P}_1, \mathcal{P}_2 \in \mathbb{R}^n$ and a vector field $f : \mathbb{R}^n \to \mathbb{R}^n$, return $True$ if a certificate for the non-existence of a continuous trajectory segment (that does not go through a third set) from $\mathcal{P}_1$ to $\mathcal{P}_2$ under the flow of f is found, and return $False$ otherwise.

R.2) $isTransient(\mathcal{P}, f)$: Given a set $\mathcal{P} \in \mathbb{R}^n$ and a vector field $f : \mathbb{R}^n \to \mathbb{R}^n$, return $True$ if a certificate for transience of $\mathcal{P}$ under the flow of f is found, and return $False$ otherwise.

Algorithm 1 Abstraction Procedure

```
Input: switched system $\mathcal{S} = (X, X_0, \mathcal{A}, \{f_a\}_{a \in \mathcal{A}}, \Pi, h)$, proposition preserving partition $P = \{\mathcal{P}_i\}_{i=1}^{N}$
Output: augmented finite transition system $\mathcal{T} = (Q, Q_0, \mathcal{A}, \to_{\mathcal{T}}, \Pi, L, \mathcal{G})$ such that $\mathcal{T} \underset{\text{O.A}}{\succeq} \mathcal{S}$
 1: Let $\alpha$ be as defined in Eq. (3)
 2: Set $Q = \{1, \ldots, N+1\}$, $Q_0 = \{i : \mathcal{P}_i \subseteq X_0\}$, $L = h \circ \alpha^{-1}$
 3: Initialize $\to_{\mathcal{T}} = (Q \setminus \{N+1\}) \times \mathcal{A} \times Q$
 4: for  $a \in \mathcal{A}$ do
 5:     $\mathcal{G}(a) = \emptyset$
 6:     $\to_{\mathcal{T}} = \to_{\mathcal{T}} \cup \{(N+1, a, N+1)\}$
 7:     for $i \in \{1, \ldots, N\}$ do
 8:         for $j = \{1, \ldots, N+1\} \setminus \{i\}$ do
 9:             if isBlocked$(\alpha^{-1}(i), \alpha^{-1}(j), f_a)$  then
10:                 $\to_{\mathcal{T}} = \to_{\mathcal{T}} \setminus \{(i, a, j)\}$
11:         if  isTransient$(\alpha^{-1}(i), f_a)$  then
12:             $\mathcal{G}(a) = \mathcal{G}(a) \cup \{\{i\}\}$
13: return $\mathcal{T} = (Q, Q_0, \mathcal{A}, \to_{\mathcal{T}}, \Pi, L, \mathcal{G})$
```

Correctness of the abstraction procedure is established next.

Proposition 4: The abstraction procedure is sound. That is, if the requirements R.1 and R.2 are satisfied, given a switched system $\mathcal{S}$, and a proposition preserving partition P of its domain, the abstraction procedure (Alg. 1) returns an augmented finite transition system $\mathcal{T}$ such that $\mathcal{T} \underset{\text{O.A.}}{\succeq} \mathcal{S}$.

Proof: We can analyze the algorithm line by line to show that the transition system $\mathcal{T}$ returned by Alg. 1 satisfies all conditions of Def. 1. On line 1, an abstraction function

[3] Under mild conditions, related to continuous implementations being non-Zeno (see [21] for details).

as in Eq. (3) is computed. On line 2, states and labeling function are defined so as to match the conditions (i) and (ii) in Def. 1. The transition relation $\rightarrow_{\mathcal{T}}$ is initialized to $((Q \setminus \{N+1\}) \times \mathcal{A} \times Q) \cup (\{N+1\} \times \mathcal{A} \times \{N+1\})$, which trivially satisfies condition (iii) in Def. 1. Similarly, the progress group map $\mathcal{G}$ is initialized to sets of empty sets, which trivially satisfies condition (iv) in Def. 1. Then, by requirements R.1 and R.2, $isBlocked$ conservatively removes transitions from $\rightarrow_{\mathcal{T}}$ only if their non-existence can be verified; and $isTransient$ conservatively adds progress groups only if their existence can be verified; hence, adhering to conditions (iii) and (iv), respectively. ■

In what follows, we show that certificates mentioned in requirements R.1 and R.2 can be computed by solving appropriate polynomial programming problems. Hence, computation of these certificates is amenable to SOS-based convex relaxations [23].

Let F be the common boundary of the sets $\mathcal{P}_1$ and $\mathcal{P}_2$ (i.e., $F = cl(\mathcal{P}_1) \cap cl(\mathcal{P}_2)$). If the sets $\mathcal{P}_1$ and $\mathcal{P}_2$ are not neighbors (i.e., $F = \emptyset$), $isBlocked$ returns $True$ since there cannot be a continuous trajectory in between these two sets that remains in their union. If the sets are neighbors, then $isBlocked$ searches for a certificate that guarantees none of the continuous trajectories of $\mathcal{S}$ starting in $\mathcal{P}_1$ leaves $\mathcal{P}_1$ through the face F under the flow of f. The following condition gives such a certificate:

$$n(\xi)^T f(\xi) < 0; \quad \forall \xi \in F \ \text{ and } \ \forall n(\xi) \in N_{\mathcal{P}_1}(\xi), \tag{4}$$

which essentially checks the angle $f(\xi)$ makes with the normal cone $N_{\mathcal{P}_1}(\xi)$ of $\mathcal{P}_1$ at all $\xi \in F$[4].

As for finding a certificate for transience (i.e., implementing $isTransient$), we use an idea similar to barrier certificates [24].

Proposition 5: A set Y is transient on a mode a of a switched system $\mathcal{S}$, if there exists a $\mathcal{C}^1$ function $B : \mathbb{R}^n \rightarrow \mathbb{R}$ such that

$$\dot{B}(\xi) = \frac{\partial B(\xi)}{\partial \xi} f_a(\xi) \leq -\varepsilon, \quad \forall \xi \in Y \tag{5}$$

for some $\varepsilon > 0$.

Proof: Assume there exists a $\mathcal{C}^1$ function $B : \mathbb{R}^n \rightarrow \mathbb{R}$ satisfying (5), and, by contradiction, assume there exists an $\xi_0 \in Y$ such that the state trajectory x of subsystem a of (2) on $\mathbb{R}^+$ starting from ξ_0 is contained in Y. Since x is contained in Y, we have $\inf_{t \in \mathbb{R}^+} B(x(t)) \geq \min_{\xi \in cl(Y)} B(\xi) > -\infty$ (existence of a finite lower bound follows from compactness of $cl(Y) \subseteq X$ together with continuity of B). However, since B is strictly decreasing over the trajectories in Y, we have $\lim_{t \rightarrow \infty} B(x(t)) = -\infty$, which, together with the fact that $\lim_{t \rightarrow \infty} B(x(t)) \geq \inf_{t \in \mathbb{R}^+} B(x(t))$, leads to a contradiction. ■

Remark 1: If a switching protocol synthesis problem is unrealizable for an over-approximation $\mathcal{T}$ of a system $\mathcal{S}$, by Props. 2-3 and Def. 3, it might be possible to find a protocol for $\mathcal{S}$ using a refined finite transition system. A refinement procedure based on refining the proposition preserving partition P can be devised, for instance by splitting certain cells in the partition, applying the $isBlocked$ and $isTransient$ subroutines to the cells that are affected by splitting and by inheriting the transience property from parent cells to children cells. Two key points for the refinement procedure to be effective are the choice of the cell to split and how to split that cell, which are beyond the scope of the current paper, and subjects of future research.

Remark 2: The definitions of abstractions and transience can be readily extended to the case where each subsystem in (2) is subject to disturbances. Moreover, when disturbance affects the dynamics polynomially, it is possible to use SOS-based convex optimization to compute certificates. We skip the details due to page limitation but illustrate this extension with an example in Sec. VI-B.2.

VI. Implementation and Examples

In this section, we first provide implementation details and then illustrate the approach with a numerical example.

A. Implementation details

We used Multi-Parametric Toolbox (MPT) [18] for polytope manipulations and SOSTOOLS [25] for solving relaxed versions of polynomial problems (4) and (5). MPT is used to partition the state-space into convex polytopes that are proposition preserving.

In principle, given two convex polytopes $\mathcal{P}_1, \mathcal{P}_2 \in \mathbb{R}^n$, the condition (4) can be converted to a finite dimensional optimization form by using the fact that normal cones of convex polytopes are finitely parametrized, however computation can be involved when different dimensional faces meet on F. We made a simplifying assumption when implementing $isBlocked$ and considered the two sets $\mathcal{P}_1, \mathcal{P}_2$ to be neighbors only if $dim(F) = n-1$. Under this simplification[5], a certificate of being blocked can be obtained if the solution γ^* of the following optimization problem is negative:

$$\gamma^* = \max_{\xi \in F} \quad n^T f(\xi) \tag{6}$$

where n is the normal vector of supporting hyperplane of $\mathcal{P}_1$ that contains F. We computed an upper-bound $\bar{\gamma}$ of γ^* using sum-of-squares relaxations [23]. If $\bar{\gamma} < 0$, this gives a certificate for being blocked.

In order to find transience certificates, we restricted B in Proposition 5 to be a polynomial of bounded degree and resorted to standard SOS methods [25] to search for B. For completeness, we present below the associated SOS program. $\mathbb{P}_{n,d}$ be the set of real polynomials and $\Sigma_{n,d}$ be the set of real SOS polynomials in n variables and of degree less than or equal to d. As B is a polynomial, so is $\tilde{B}(\xi) \doteq \frac{\partial B(\xi)}{\partial \xi}$. Given a convex polytope $\mathcal{P}$ that has a representation $H\xi \leq k$

[4] Another (less conservative) way to obtain a certificate is to check if $f(\xi) \notin T_{\mathcal{P}_2}(\xi)$ for all $\xi \in F$ (i.e., whether any trajectory enters $\mathcal{P}_2$ through F). However for simplicity we use (4).

[5] Note that almost all trajectories pass through $n-1$ dimensional faces and the effect of this simplification on resulting trajectories is negligible for all practical purposes.

for some $H \in \mathbb{R}^{m\times n}$ and $k \in \mathbb{R}^m$, consider the following SOS program:

$$\begin{array}{ll} \text{find} & \tilde{B} \in \mathbb{P}_{n,d-d_f},\ s_i \in \Sigma_{n,d-1}\ \forall i \in \{1,\dots,m\} \\ \text{s.t.} & [s_1(\xi),\dots,s_m(\xi)](H\xi - k) - \tilde{B}(\xi)f(\xi) - \epsilon \in \Sigma_{n,d} \end{array} \tag{7}$$

where d_f is the degree of f, $d \geq d_f$ is a fixed relaxation order, and ϵ is a fixed arbitrary positive number. Feasibility of Problem (7) implies feasibility of Eq. (5), therefore is a certificate of transience of the set $\mathcal{P}$ under the flow of f. Problem (7) can be reduced to a semidefinite program, and its feasibility can be checked e.g., using SOSTOOLS [25].

B. Numerical example

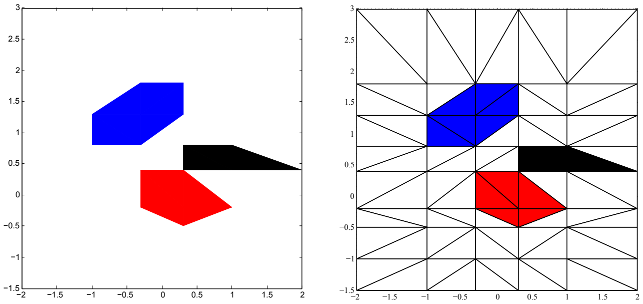
We consider the polynomial dynamical system

$$\begin{array}{rcl} \dot{x}_1 & = & -x_2 - 1.5x_1 - 0.5x_1^3 + u_1 \\ \dot{x}_2 & = & x_1 + u_2 \end{array} \tag{8}$$

where $x_1 \times x_2 \in X = [-2,2] \times [-1.5,3]$. Four different state feedback controllers, K_i, $i \in \{1,2,3,4\}$ and $[u_1,u_2] = K_i(x_1,x_2)$, are designed for this system, two stabilizing it around desired equilibrium points, and two other to provide some fast dynamics within the region of interest X shown in Fig. 1. In particular, following controllers are used: $K_1(x_1,x_2) = [0, -x_2^2 + 2]$, $K_2(x_1,x_2) = [0, -x_2]$, $K_3(x_1,x_2) = [2, 10]$, and $K_4(x_1,x_2) = [-1.5, -10]$. Therefore, the switched system has four modes: $f_1 = \begin{bmatrix} -x_2 - 1.5x_1 - 0.5x_1^3 \\ x_1 - x_2^2 + 2 \end{bmatrix}$, $f_2 = \begin{bmatrix} -x_2 - 1.5x_1 - 0.5x_1^3 \\ x_1 - x_2 \end{bmatrix}$, $f_3 = \begin{bmatrix} -x_2 - 1.5x_1 - 0.5x_1^3 + 2 \\ x_1 + 10 \end{bmatrix}$, $f_4 = \begin{bmatrix} -x_2 - 1.5x_1 - 0.5x_1^3 - 1.5 \\ x_1 - 10 \end{bmatrix}$.

1) Comparison of abstractions: In this section we compare the proposed SOS-based abstraction procedure (Alg. 1) with the linearization-based hybridization procedure inspired by the one in [10].

Given the family of vector fields $\{f_a\}_{a=1}^4$, the domain X, propositions $\Pi = \{\pi_1, \pi_2, \pi_3\}$ together with an observation map h associating them to regions A_1, A_2, A_3 shown in Fig. 1(a), and a proposition preserving partition as in Fig. 1(b), we used the proposed SOS-based abstraction algorithm to compute an over-approximation $\mathcal{T} = (Q, Q_0, \mathcal{A}, \rightarrow_{\mathcal{T}}, \Pi, L, \mathcal{G})$ of $\mathcal{S} = (X, X_0, \mathcal{A}, \{f_a\}_{a\in\mathcal{A}}, \Pi, h)$. As a comparison, we adapted the abstraction procedure from [10] to linearize the system within each cell of the proposition preserving partition for each mode and computed the uncertainty bounds and transitions accordingly. We denote the finite transition system obtained by linearization-based abstraction with $\mathcal{T}^{lin} = (Q, Q_0, \mathcal{A}, \rightarrow_{\mathcal{T}^{lin}}, \Pi, L, \mathcal{G})$. Note that $\mathcal{T}^{lin}$ is also an over-approximation of $\mathcal{S}$. The resulting transition relation $\rightarrow_{\mathcal{T}}$ was a proper subset of $\rightarrow_{\mathcal{T}^{lin}}$ (i.e., $\rightarrow_{\mathcal{T}} \subsetneq \rightarrow_{\mathcal{T}^{lin}}$) with $\mid \rightarrow_{\mathcal{T}} \mid = 802$ and $\mid \rightarrow_{\mathcal{T}^{lin}} \mid = 905$. Hence, $\mathcal{T}$ is a better approximation of $\mathcal{S}$ in the sense that $\mathcal{T}^{lin} \underset{\text{A.S.}}{\succeq} \mathcal{T} \underset{\text{O.A.}}{\succeq} \mathcal{S}$ (see Defs. 2-3, with the function β chosen to be identity). It is worth emphasizing that the less transitions there are, the easier it is to control the system due to decreased non-determinism within the modes. Though, it should be noted that this improvement comes with some added computational complexity (solving a set of semidefinite programs for the proposed SOS-based abstraction vs. some matrix/vector arithmetic for the linearization-based approach). However since abstractions are computed offline, it might be desirable to use SOS-based abstractions to obtain higher fidelity approximations with the same number of discrete states.



(a) (b)

Fig. 1. (a) Domain X and regions A_1 (red), A_2 (blue), A_3 (black) associated with the propositions. (b) A proposition preserving partition.

2) Control protocol synthesis: In this section, we use the proposed method to solve a continuous switching protocol synthesis problem. We consider the system $\mathcal{S} = (X, X_0, \mathcal{A}, \{f_a\}_{a\in\mathcal{A}}, \Pi, h)$ from Sec. VI-B.1 subject to additive disturbance $\delta = (\delta_1, \delta_2)$ with $||\delta||_\infty \leq 0.005$. That is the family $\{\tilde{f}_a\}_{a=1}^4$ of vector fields is given by $\tilde{f}_a = f_a + \delta$ for $a \in \mathcal{A}$. Given the family $\{\tilde{f}_a\}_{a=1}^4$, the domain X, propositions $\Pi = \{\pi_1, \pi_2, \pi_3\}$ together with an observation map h associating them to regions A_1, A_2, A_3 shown in Fig. 1(a), the goal was to synthesize a switching protocol to guarantee that the trajectories of the system satisfy:

$$\varphi = \square(\neg\pi_{out}) \wedge \square(\neg\pi_3) \wedge \square\diamond(\pi_1) \wedge \square\diamond(\pi_2). \tag{9}$$

We used the proposition preserving partition in Fig. 1(b) and computed an augmented finite transition system $\mathcal{T}_\delta = (Q, Q_0, \mathcal{A}, \rightarrow_{\mathcal{T}_\delta}, \Pi, L, \mathcal{G})$ using the proposed method. Compared to $\mathcal{T}$ in Sec. VI-B.1, $\mathcal{T}_\delta$ had 6 additional transitions. TuLiP toolbox [31] was employed to solve the discrete synthesis problem. The problem was realizable and we obtained a control strategy that was represented by an automaton with 24 states. The discrete strategy was then implemented on the continuous system. Fig. 2 shows a sample continuous trajectory which repeatedly visits A_1 and A_2 and avoids A_3 as expected.

VII. Conclusions

In this paper we considered the problem of synthesizing switching protocols for polynomial switched systems so that the closed-loop trajectories of the system satisfy a high level specification expressed in linear temporal logic. The main contributions of this paper lie in (i) introducing augmented finite transition systems as abstract models for polynomial switched systems, and (ii) proposing a computationally

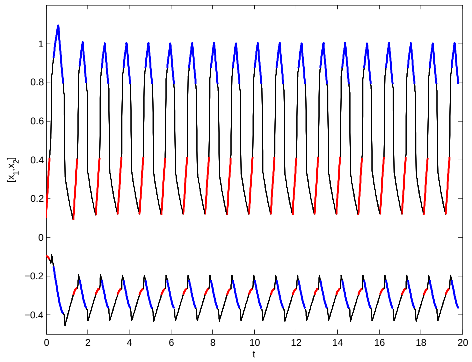


Fig. 2. A simulation of the continuous trajectory with the synthesized switching protocol for the specification in (9). Lower curve shows x_1 and upper curve shows x_2, where the portions of the trajectory in regions A_1 and A_2 are highlighted with red and blue segments respectively.

tractable method for computing such abstract models. Sum-of-squares based relaxations were used for computing the transition relations and transience properties. These relaxations were shown, with an example, to achieve a good trade-off between computational complexity and quality of the abstract model obtained. Proposed abstraction method can easily be integrated with methods as in [21] to accommodate specifications that require reacting to possibly adversarial external events in runtime.

We also defined a refinement relation between augmented finite transition systems that can be used for incrementally computing finite transition systems abstracting a switched system, with potentially higher fidelity. Future work will focus on abstraction refinement based incremental synthesis of switching protocols by leveraging counter-example guided automatic refinement ideas in hybrid system verification [1], [7].

References

[1] R. Alur, T. Dang, and F. Ivancic. Counterexample-guided predicate abstraction of hybrid systems. *Theor. Comput. Sci.*, 354(2):250–271, March 2006.

[2] E. Asarin, O. Bournez, T. Dang, O. Maler, and A. Pnueli. Effective synthesis of switching controllers for linear systems. *Proceedings of the IEEE*, 88(7):1011–1025, 2000.

[3] C. Baier and J.P. Katoen. *Principles of Model Checking*. MIT Press, 2008.

[4] G. Batt, C. Belta, and R. Weiss. Temporal logic analysis of gene networks under parameter uncertainty. *IEEE Trans. on Automatic Control*, 53(Special Issue):215 –229, jan. 2008.

[5] R. Bloem, B. Jobstmann, N. Piterman, A. Pnueli, and Y. Saar. Synthesis of reactive (1) designs. *J. Comput. System Sci.*, 78:911–938, 2012.

[6] J. Cámara, A. Girard, and G. Gössler. Synthesis of switching controllers using approximately bisimilar multiscale abstractions. In *Hybrid Systems: Computation and Control*, pages 191–200, 2011.

[7] E. Clarke, A. Fehnker, Z. Han, B. Krogh, O. Stursberg, and M. Theobald. Verification of hybrid systems based on counterexample-guided abstraction refinement. *Tools and Algorithms for the Construction and Analysis of Systems*, pages 192–207, 2003.

[8] G.E. Fainekos, A. Girard, H. Kress-Gazit, and G.J. Pappas. Temporal logic motion planning for dynamic robots. *Automatica*, 45(2):343–352, 2009.

[9] E. Frazzoli, M.A. Dahleh, and E. Feron. Maneuver-based motion planning for nonlinear systems with symmetries. *IEEE Trans. on Robotics*, 21(6):1077–1091, 2005.

[10] A. Girard and S. Martin. Synthesis for constrained nonlinear systems using hybridization and robust controllers on simplices. *IEEE Trans. on Automatic Control*, 57(4):1046 –1051, april 2012.

[11] E. Grädel, W. Thomas, and T. Wilke, editors. *Automata, Logics, and Infinite Games: A Guide to Current Research*, volume 2500 of *Lecture Notes in Computer Science*. Springer, 2002.

[12] L. Habets, M. Kloetzer, and C. Belta. Control of rectangular multi-affine hybrid systems. In *Decision and Control, 2006 45th IEEE Conference on*, pages 2619–2624. IEEE, 2006.

[13] J.P. Hespanha and A.S. Morse. Switching between stabilizing controllers. *Automatica*, 38(11):1905 – 1917, 2002.

[14] Y. Kesten and A. Pnueli. Verification by augmented finitary abstraction. *Information and Computation*, 163(1):203 – 243, 2000.

[15] M. Kloetzer and C. Belta. A fully automated framework for control of linear systems from temporal logic specifications. *IEEE Trans. on Automatic Control*, 53:287–297, 2008.

[16] T. Koo, G. Pappas, and S. Sastry. Mode switching synthesis for reachability specifications. In *Hybrid Systems: Computation and Control*, pages 333–346. Springer, 2001.

[17] H. Kress-Gazit, G.E. Fainekos, and G.J. Pappas. Temporal-logic-based reactive mission and motion planning. *IEEE Trans. on Robotics*, 25:1370–1381, 2009.

[18] M. Kvasnica, P. Grieder, and M. Baotić. Multi-Parametric Toolbox (MPT), 2004.

[19] D. Liberzon and A.S. Morse. Basic problems in stability and design of switched systems. *IEEE Control Systems Magazine*, 19(5):59–70, 1999.

[20] J. Liu, N. Ozay, U. Topcu, and R.M. Murray. Switching protocol synthesis for temporal logic specifications. *American Control Conference*, 2012.

[21] J. Liu, N. Ozay, U. Topcu, and R.M. Murray. Synthesis of reactive switching protocols from temporal logic specifications. *IEEE Trans. on Automatic Control*, 2013.

[22] H-W. Park, A. Ramezani, and J.W. Grizzle. A finite-state machine for accommodating unexpected large ground height variations in bipedal robot walking. *IEEE Trans. on Robotics*, 2013.

[23] P.A. Parrilo. Semidefinite programming relaxations for semialgebraic problems. *Mathematical Programming*, 96(2):293–320, 2003.

[24] S. Prajna. *Optimization-based methods for nonlinear and hybrid systems verification*. PhD thesis, California Institute of Technology, 2005.

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

[26] R.T. Rockafellar. *Convex Analysis*. Princeton Mathematical Series. Princeton University Press, 1970.

[27] P. Tabuada. *Verification and control of hybrid systems: a symbolic approach*. Springer-Verlag New York Inc, 2009.

[28] P. Tabuada and G. Pappas. Model checking LTL over controllable linear systems is decidable. *Hybrid systems: computation and control*, pages 498–513, 2003.

[29] A. Taly and A. Tiwari. Switching logic synthesis for reachability. In *Proceedings of the ACM International Conference on Embedded Software*, pages 19–28, 2010.

[30] A. Tiwari. Abstractions for hybrid systems. *Formal Methods in Systems Design*, 32:57–83, 2008.

[31] T. Wongpiromsarn, U. Topcu, N. Ozay, H. Xu, and R.M. Murray. TuLiP: a software toolbox for receding horizon temporal logic planning. In *Hybrid Systems: Computation and Control*, pages 313–314, 2011.

[32] B. Yordanov, J. Tumova, I. Cerna, J. Barnat, and C. Belta. Temporal logic control of discrete-time piecewise affine systems. *IEEE Trans. on Automatic Control*, 57(6):1491 –1504, june 2012.