Formal Methods for Adaptive Control of Dynamical Systems

Sadra Sadraddini and Calin Belta

Abstract— **We develop a method to control discrete-time systems with constant but initially unknown parameters from linear temporal logic (LTL) specifications. We introduce the notions of (non-deterministic) parametric and adaptive transition systems and show how to use tools from formal methods to compute adaptive control strategies for finite systems. For infinite systems, we first compute abstractions in the form of parametric finite quotient transition systems and then apply the techniques for finite systems. Unlike traditional adaptive control techniques, our method is correct-by-design, does not require a reference model, and can handle a much wider range of systems and specifications. Illustrative examples are included.**

I. Introduction

One particular limitation of current adaptive (self-learning) control methods is handling systems that involve discontinuities. Most adaptive control techniques rely on the continuity of the model and its parameterization. In many realistic models, state, control or parameters take values from both continuous and discrete domains. Within methods that do not entirely depend on the continuity of the model, a promising direction is using multiple models/controllers [1]–[4], where the objective is to achieve stability via designing a switching law to coordinate the controllers. Model reference adaptive control (MRAC) of specific forms of scalar input piecewise affine systems were studied in [5], [6]. However, it is still not clear how to deal with general discrete or hybrid systems.

Another remaining open problem in adaptive control is dealing with specifications richer than stability. In many engineering applications, we are interested in complex requirements composed of safety (something bad never happens), liveness (something good eventually happens), sequentiality of tasks, and reactiveness. Temporal logics [7] provide a natural framework for specifying such requirements. The main challenge in designing adaptive control techniques from formal specifications is handling hard constraints on the evolution of the system. Even for the simpler problem of constraints defined as a safe set in the state-space, designing adaptive control strategies is challenging. Existing works on this problem [8]–[12] apply robust control techniques to ensure infinite-time constraint satisfaction for all admissible parameters. This approach may be severely conservative since if a robust control strategy does not exist for all admissible parameters, it does not necessarily indicate that constraints can not be satisfied after some measurements are taken from the system and a more accurate model is available. Even though [9], [11] update the model and synthesize controls in a receding horizon manner, they decouple constraint satisfaction and learning. However, there exists a deep coupling: when synthesizing controls, not only constraints must be taken into account, but also the evolution of the system should also lead to subsequent measurements that are more informative about the uncertainties in the model. In other words, control decisions influence how the way the model is updated.

We use tools from formal methods [7] to develop a framework for correct-by-design adaptive control that can deal with complex systems and specifications. Formal methods have been increasingly used in control theory in recent years [13], [14]. We consider discrete-time systems with constant but initially unknown parameters. We describe system specifications using linear temporal logic (LTL) [7]. As in any other adaptive control technique, we require an online parameter estimator. Our parameter estimator maps the history of the evolution of the system to the set of "all possible" parameters, which contains the actual parameters. We embed the parameterized system in a (non-deterministic) parametric transition system (PTS), from which we construct a (non-deterministic) adaptive transition system (ATS) that contains all the possible combinations of transitions with the unfoldings of the parameter estimator. The main results and contributions of this paper are as follows:

- For finite systems, the LTL adaptive control problem reduces to a Rabin game [15] on the product of the finite ATS and the Rabin automaton corresponding to the LTL specification. The method is correct by design and it is complete, i.e. it finds a solution if one exists;
- For infinite systems, we construct finite quotient ATSs by partitioning the state and the parameter space and quantizing the control space. An adaptive control strategy found for the quotient is guaranteed to ensure the LTL formula for the original infinite system. The method may be conservative.

This paper is related to recent works on formal methods approach to learning and control. Statistical safety certificates were investigated in [16], [17]. The idea was based on implementing MRAC from a set of different initial conditions and parameters and observing if the trajectories were safe. However, the design of MRAC itself did not take into account the constraints. MRAC may not be suitable for the purpose of this paper as it is not clear how a reference model should be chosen given LTL formula. If a reference model is able to satisfy the specification, the matching condition may not hold, i.e., there may not exist a controller for the original system to make it behave as

The authors are with the Department of Mechanical Engineering, Boston University, Boston, MA 02215 {sadra,cbelta}@bu.edu. This work was partially supported by the NSF under grants CPS- 1446151 and CMMI-1400167.

the reference model. Reinforcement learning (RL) methods are conceptually similar to adaptive control, but are used in a probabilistic framework and require a reward mechanism to generate control policies. The authors in [18] studied RL from LTL specifications, where large rewards were dedicated to the pairs in the Rabin automaton to incentivize the system to visit them regularly or avoid them. In [19], Q-learning was applied to control MDPs from signal temporal logic (STL) specifications, where the reward was the STL robustness score - a measure of distance to satisfaction. In [20], [21], the LTL controller inferred the "grammar" of actions taken by the environment. However, this approach also decoupled adaptation (learning) and control. The LTL formula may be violated during the grammar learning. While RL and grammar learning have the advantage that they require less or no prior knowledge about the system, they are not suitable for performance-critical systems with constraints that should never be violated, even during the learning process.

This paper is organized as follows. First, we provide the necessary background on LTL, transition systems and LTL control in Sec. II. The problem is formulated in Sec. III. We define PTSs in Sec. IV. Technical details for the solutions for finite and infinite systems are explained in Sec. V and VI, respectively. Two case studies are presented in Sec. VII.

II. Background

A. Notation

The set of real and Boolean values are denoted by $\mathbb{R}$ and $\mathbb{B}$ respectively. The empty set is denoted by $\emptyset$. Given a set S, we use $|S|$, 2^S, $2^S_{-\emptyset}$ to denote its cardinality, power set, and power set excluding the empty set, respectively. An alphabet $\mathcal{A}$ is a finite set of symbols $\mathcal{A} = \{a_1, a_2, \cdots, a_A\}$. A finite (infinite) word is a finite-length (infinite-length) string of symbols in $\mathcal{A}$. For example, $w_1 = a_1a_2a_1$ is a finite word, and $w_2 = a_1a_2(a_1)^\omega$ and $w_3 = a_1(a_2a_1)^\omega$ are infinite words over $\mathcal{A} = \{a_1, a_2\}$, where ω stands for infinitely many repetitions. We use $\mathcal{A}^*$ and $\mathcal{A}^\omega$ to denote the set of all finite and infinite words that can be generated from $\mathcal{A}$, respectively.

B. Linear Temporal Logic

The formal definition of LTL syntax and semantics is not provided here as it can be found in the literature [7]. Here we provide an informal introduction and the necessary notation. LTL consists of a finite set of atomic propositions Π, temporal operators $\mathbf{G}$ (globally/always), $\mathbf{F}$ (future/eventually), $\mathbf{U}$ (Until), and Boolean connectives $\wedge$ (conjucntion), $\vee$ (disjunction), and $\neg$ (negation). LTL semantics are interpreted over infinite words over 2^Π. The set of all infinite words that satisfy an LTL formula φ is denoted by $L(\varphi)$, $L(\varphi) \subset (2^\Pi)^\omega$, and is referred to as the *language* of φ.

Definition 1: A Deterministic Rabin Automaton (DRA) is defined as the tuple $\mathcal{R} = (S, s^0, \mathcal{A}, \alpha, \Omega)$, where:

- S is a set of states;
- s^0 is the initial state;
- $\mathcal{A}$ is a finite set of inputs (alphabet);
- α is a transition function $\alpha : S \times \mathcal{A} \to S$;
- $\Omega = \{(F_1, I_1), \cdots, (F_r, I_r)\}$ is a finite set of pairs of sets of states, where $F_i, I_i \subset S, i = 1, \cdots, r$.

An infinite word $w \in \mathcal{A}^\omega$ determines a sequence of inputs for $\mathcal{R}$ that results in the *run* $\zeta(w) = s_0 s_1 \cdots$, where $s_{k+1} = \alpha(s_k, a_k)$, $s_0 = s^0$, and a_k is the k'th input appearing in w. We define $Inf(\zeta) = \{s | s \text{ appears infinitely often in } \zeta\}$. A run ζ is *accepted* by $\mathcal{R}$ if there exists $i \in \{1, \cdots, m\}$ such that $Inf(\zeta) \cap F_i = \emptyset$ and $Inf(\zeta) \cap I_i \neq \emptyset$. In other words, F_i is visited finitely many times and I_i is visited infinitely often for some i. The language of $\mathcal{R}$, denoted by $L(\mathcal{R})$, $L(\mathcal{R}) \subset \mathcal{A}^\omega$, is defined as the set of all elements in $\mathcal{A}^\omega$ that produce accepting runs. It is known that given an LTL formula φ over Π, one can construct a DRA $\mathcal{R}_\varphi$ with input set $\mathcal{A} = 2^\Pi$ such that $L(\mathcal{R}_\varphi) = L(\varphi)$ [15]. Thus, verifying whether an infinite word satisfies an LTL formula is equivalent to checking the Rabin acceptance condition, for which tere exists well-established algorithms [22].

C. Transition Systems

Definition 2: A transition system is defined as the tuple $\mathcal{T} = (X, U, \beta, \Pi, O)$, where:

- X is a (possibly infinite) set of states;
- U is a (possibly infinite) set of control inputs;
- β is a transition function $\beta : X \times U \to 2^X$;
- $\Pi = \{\pi_1, \pi_2, \cdots, \pi_m\}$ is a finite set of atomic propositions;
- $O : X \to 2^\Pi$ is an observation map.

We assume that $\mathcal{T}$ is *non-blocking* in the sense that $|\beta(x, u)| \neq 0$ for all $x \in X, u \in U$. A transition system $\mathcal{T}$ is *deterministic* if $|\delta(x, u)| = 1, \forall x \in X, \forall u \in U$, and is *finite* if X and U are finite sets. A trajectory of $\mathcal{T}$ is an infinite sequence of visited states $x_0x_1x_2\cdots$. The infinite word produced by such a trajectory is $O(x_0)O(x_1)O(x_2)\cdots$. The alphabet here is 2^Π. The set of all infinite words that can be generated by $\mathcal{T}$ is a subset of $(2^\Pi)^\omega$.

Definition 3: A control strategy Λ is a function $\Lambda : X^* \times U^* \to U$ that maps the history of visited states and applied controls to an admissible control input, where $u_k = \Lambda(x_0 \cdots, x_k, u_0 \cdots, u_{k-1}), \forall k \in \mathbb{N}$.

Given a transition system $\mathcal{T} = (X, U, \beta, \Pi, O)$, a control strategy Λ, $u_k = \Lambda(x_0 \cdots, x_k, u_0 \cdots, u_{k-1})$, and a set of initial states $X_0 \in X$, we define following:

$$L(\mathcal{T}, \Lambda, X_0) := \Big\{ \quad O(x_0)O(x_1)\cdots \in (2^\Pi)^\omega \Big| \\ x_0 \in X_0, x_{k+1} \in \beta(x_k, u_k), k \in \mathbb{N} \Big\}.$$

D. Quotient Transition System

Consider a transition system $\mathcal{T} = (X, U, \beta, \Pi, O)$. A (finite) set $Q \subset 2^X$ is a (finite) partition for X if 1) $\emptyset \notin Q$, 2) $\bigcup_{q \in Q} q = X$, and 3) $q \cap q' = \emptyset, \forall q, q' \in Q, q \neq q'$. A partition Q is *observation preserving* if for all $q \in Q$, we have $O(x) = O(x'), \forall x, x' \in q$.

Definition 4: Given a transition system $\mathcal{T} = (X, U, \beta, \Pi, O)$ and an observation preserving partition Q for X, the *quotient transition system* is defined as the tuple $\mathcal{T}_Q = (Q, U, \beta_Q, \Pi, O_Q)$ such that:

- for all $q \in Q$, we have $q' \in \beta_Q(q,u)$ if and only if $\exists x \in q$, $\exists x' \in q'$ such that $x' \in \beta(x,u)$;
- for all $q \in Q$, we have $O_Q(q) = O(x)$ for any $x \in q$.

Given a control strategy for the quotient $\Lambda_Q : Q^* \times U^* \to U$, and a set of initial conditions Q_0, we construct $\Lambda^{(Q)} : X^* \to U$ such that $\Lambda^{(Q)}(x_0 \cdots x_k) = \Lambda_Q(q_0 \cdots q_k)$, $x_i \in q_i$, $0 \leq i \leq k$, $k \in \mathbb{N}$, and $X_0^{(Q)} = \{x_0 | x_0 \in q_0, q_0 \in Q_0\}$. It is easy to show that $L(\mathcal{T}, \Lambda^{(Q)}, X_0^{(Q)}) \subseteq L(\mathcal{T}_Q, \Lambda_Q, Q_0)$, which stems from the fact that $\mathcal{T}_Q$ simulates $\mathcal{T}$.

E. LTL Control

Given a finite transition system $\mathcal{T} = (X, U, \beta, \Pi, O)$ and an LTL formula φ over Π, we are interested in finding a control strategy Λ and the largest set of initial conditions $X_0^{\max}$ such that $L(\mathcal{T}, \Lambda, X_0^{\max}) \subseteq L(\varphi)$. In other words, we require φ to be satisfied for all trajectories that are allowed by the non-determinism in $\mathcal{T}$.

Definition 5: Given a transition system $\mathcal{T} = (X, U, \beta, \Pi, O)$ and a DRA $\mathcal{R}_\varphi = (S, s^0, \mathcal{A}, \alpha, \Omega)$ corresponding to LTL formula φ, the product automaton $\mathcal{T}_\varphi^P = \mathcal{T} \otimes \mathcal{R}_\varphi$ is defined as the tuple $\left(X^P, X^{P,0}, U, \beta^P, \Omega^P\right)$, where:

- $X^P = X \times S$ is the set of product states;
- $X^{P,0} = \{(x, s^0) | x \in X\}$ is the set of initial product states;
- U is the set of control inputs;
- $\beta^P : X^P \times U \to 2^{X^P}$ is the product transition function, where $x^{P'} \in \delta(x^P, u)$, $x^P = (x,s), x^{P'} = (x', s')$, if and only if $x' \in \beta(x,u)$ and $s' = \alpha(s, O(x))$.
- $\Omega^P = \left\{(F_1^P, I_1^P), \cdots, (F_r^P, I_r^P)\right\}$ is a finite set of pairs of sets of states, where $F_i^P = \{(x,s) | x \in X, s \in F_i\}, I_i^P = \{(x,s) | x \in X, s \in I_i\}, i = 1, \cdots, r$.

The solution to the problem of finding a control strategy to satisfy φ is accomplished by solving the Rabin game on the product automaton. The details are not presented here but can be found in [23]. It can be shown that the control strategy is memoryless on the product automaton in the form $\Lambda : X \times S \to U$. In other words, the history of the system is incorporated into the state of the Rabin automaton. The largest set of admissible initial conditions $X_0^{\max}$ corresponds to the winning region of the Rabin game.

If the transition system $\mathcal{T}$ is infinite, a finite quotient is constructed. If U is infinite, it can be quantized to obtain a finite set. It is known that if a control strategy satisfying φ exists for the finite quotient, it also satisfies φ if implemented on the original system. However, unless the quotient and the original transition system are bisimilar, the non-existence of a control strategy for the quotient does not indicate that one does not exist for the original system. Thus, completeness may be lost using using finite quotients [13], [14].

III. Problem Formulation and approach

We consider discrete-time systems of the following form:

$$\begin{aligned} x^+ &= F(x, u, \theta, d), \\ y_i &= \mu_i(x), i = 1, \cdots, m, \end{aligned} \tag{1}$$

where $x \in X$ is the state, $u \in U$ is the control input, $\theta \in \Theta$ represents the parameters of the system, $d \in D$ is the disturbance (adversarial input), $F : X \times U \times \Theta \times D \to X$ is the system evolution function, and $y_i, i = 1, \cdots, m$, are Boolean system outputs, where $\mu_i : X \to \mathbb{B}$. We define the set of atomic propositions $\Pi = \{\pi_1, \cdots, \pi_m\}$ such that $x \models \pi_i \Leftrightarrow \mu_i(x) = \text{True}, i = 1, \cdots, m$. The sets X, U, Θ, D are the admissible sets for states, controls, parameters and disturbances respectively. All sets may be finite or infinite. System (1) is finite if X, U, Θ, D are all finite.

As mentioned in the introduction, we distinguish between the uncertainty in parameters and disturbances. Disturbances usually have unknown (fast) variations in time. In this paper, we assume that θ is a constant but its value θ^* is initially unknown. If we treat the uncertainties in parameters and disturbances in the same way, we are required to design control strategies that are robust versus all values in both Θ and D. This approach is severely conservative and often fails to find a solution. The key idea of adaptive control is to take advantage of the fact that θ^* can be (approximately) inferred from the history of the evolution of the system. Therefore, adaptive control is often significantly more powerful than pure robust control and it is also more difficult to design and analyze. In engineering applications, parameters are related to the physical attributes of the plant whereas disturbances are related to effects of stochastic nature such as imperfect actuators/sensors and perturbations in the environment.

Problem 1: Given system (1) and an LTL formula φ over Π, find a control strategy $\Lambda : X^* \times U^* \to U$ and a set of initial states $X_0 \subseteq X$ such that all the trajectories of the closed loop system starting from X_0 satisfy φ.

Our aim is to convert Problem 1 to an LTL control problem described in Sec.II-E and use the standard tools for Rabin games. To this end, we need to incorporate adaptation into control synthesis. The central tool to any adaptive control technique is parameter estimation. Note that an adaptive control strategy has the form $\Lambda : X^* \times U^* \to U$, since parameters are estimated using the history of the evolution of the system. We take the following approach to convert Problem 1 into an LTL control problem. We embed system (1) in a parametric transition system (PTS), which is defined in Sec. IV. We construct a finite adaptive transition system (ATS) from a finite PTS. An ATS is an ordinary transition system as in Sec. II-C, but parameters are also incorporated into its states and transitions in appropriate way, which is explained in Sec. V. We deal with an infinite PTS by constructing a finite quotient PTS in Sec. VI.

IV. Parametric Transition System

Definition 6: A parametric transition system (PTS) is defined as the tuple $\mathcal{T}^\Theta = (X, U, \Theta, \gamma, \Pi, O)$, where:

- X is a (possibly infinite) set of states;
- U is a (possibly infinite) set of control inputs;
- Θ is a (possibly infinite) set of parameters;
- γ is a transition function $\gamma : X \times U \times \Theta \to 2^X$.

- $\Pi = \{\pi_1, \pi_2, \cdots, \pi_m\}$ is a finite set of atomic propositions;
- $O : X \to 2^{\Pi}$ is an observation map.

The only difference between a PTS and a transition system is that its transitions depend on parameters. Note that if $|\Theta| = 1$, a PTS becomes a transition system.

System (1) can represented in the form of a PTS. The sets X, U, Θ are inherited from (1) (we have used the same notation). The transition function γ is constructed such that

$$\gamma(x, u, \theta) = \left\{ F(x, u, \theta, d) \middle| d \in D \right\}. \tag{2}$$

The observation map $O : X \to 2^{\Pi}$ is given by:

$$O(x) = \left\{ \pi_i \middle| \mu_i(x) = \text{True}, i = 1 \cdots, m \right\}. \tag{3}$$

Therefore, $\mathcal{T}^{\Theta} = (X, U, \Theta, \gamma, \Pi, O)$ captures everything in system (1). One can interpret a PTS as a (possibly infinite) family of transition systems. The actual transitions are governed by a single parameter θ^*, which is initially unknown to the controller, which has to find out which transition system is the ground truth.

V. Control Synthesis for Finite Systems

Here we assume the PTS embedding system (1) is finite.

A. Parameter Estimation

Definition 7: A *parameter estimator* Γ is a function

$$\Gamma : X^* \times U^* \to 2^{\Theta}_{-\emptyset} \tag{4}$$

that maps the history of visited states and applied controls to a subset of parameters. We have $\vartheta_k = \Gamma(x_0 \cdots x_k; u_0 \cdots u_{k-1})$, where:

$$\vartheta_k = \left\{ \theta \in \Theta \middle| x_{i+1} \in \gamma(x_i, u_i, \theta), 0 \le i \le k-1 \right\}. \tag{5}$$

?he parameter estimator (5) is "sound" in the sense that $\theta^* \in \vartheta_k, \forall k \in \mathbb{N}$. We have $\vartheta_0 = \Gamma(x_0) = \Theta$, by definition. Note that our parameter estimator is different from the traditional ones, which are often in the form $X^* \times U^* \to \Theta$, as they return only an estimate $\hat{\theta}$ rather than the set of all possible parameters. For our formal setup, it is important that the controller take into account all possible parameters.

Proposition 1: The following recursive relation holds:

$$\vartheta_{k+1} = \left\{ \theta \in \vartheta_k \middle| x_{k+1} \in \gamma(x_k, u_k, \theta) \right\}. \tag{6}$$

Thus, the set of estimated parameters never grows: $\vartheta_{k+1} \subseteq \vartheta_k, \forall k \in \mathbb{N}$. We obtain a recursive parameter estimator $\Gamma_{rec} : 2^{\Theta}_{-\emptyset} \times X \times U \times X \to 2^{\Theta}_{-\emptyset}$ as $\vartheta_{k+1} = \Gamma_{rec}(\vartheta_k, x_k, u_k, x_{k+1})$.

B. Adaptive Transition System

As mentioned in the introduction, a primary challenge of provably correct adaptive control is coupling parameter estimation and control synthesis. In order to combine these two, we provide the following definition.

Definition 8: Given a PTS $\mathcal{T}^{\Theta} = (X, U, \Theta, \gamma, \Pi, O)$, we define the adaptive transition system (ATS) as the tuple $\mathcal{T}^{adp} = \left(X^{adp}, U, \gamma^{adp}, \Pi, O^{adp}\right)$, where U, Π are inherited from $\mathcal{T}^{\Theta}$ with the same meaning and

- $X^{adp} \subseteq X \times 2^{\Theta}_{-\emptyset}$ is the set of states;
- $\gamma^{adp} : X^{adp} \times U \to 2^{X^{adp}}$ is the transition function, where we have $(x', \vartheta') \in \gamma^{adp}((x, \vartheta), u)$ if and only if $x' \in \gamma(x, u)$ and $\vartheta' = \Gamma_{rec}(\vartheta, x, u, x')$;
- $O^{adp} : X^{adp} \to 2^{\Pi}$ is the observation function where $O^{adp}(x, \vartheta) = O(x), \forall x \in X, \vartheta \in 2^{\Theta}_{-\emptyset}$.

The number of states in the ATS is upper-bounded by $|X|(2^{|\Theta|}-1)$, which shows an exponential explosion with the number of parameters. Fortunately, not all states in $X \times 2^{\Theta}_{-\emptyset}$ are reachable from the set $\{(x, \theta) | x \in X, \theta \in \Theta\}$, which is the set of possible initial states in the ATS. Algorithm 1 constructs the ATS consisting of only these reachable states.

Algorithm 1 Constructing ATS from PTS

Require: $\mathcal{T}^{\Theta} = (X, U, \Theta, \gamma, \Pi, O)$
$X^{adp,new} = \{(x, \Theta) | x \in X\}$
$X^{adp} = X^{adp,new}$
while $X^{adp,new} \neq \emptyset$ **do**
  $X^{adp,new} \leftarrow \emptyset$
  for $(x, \vartheta) \in X^{adp}$ **do**
    for $u \in U$ **do**
      $\gamma^{adp}((x, \vartheta), u) = \emptyset$
      $\vartheta' = \emptyset$
      for $\theta \in \vartheta$ **do**
        for $x' \in \gamma(x, u, \vartheta)$ **do**
          for $\theta' \in \vartheta$ **do**
            if $x' \in \gamma(x, u, \theta')$ **then**
              $\vartheta' \leftarrow \vartheta' \cup \theta'$
          $\gamma^{adp}((x, \vartheta), u) \leftarrow \gamma^{adp}((x, \vartheta), u) \cup (x', \vartheta')$
          if $(x', \vartheta') \notin X^{adp}$ **then**
            $X^{adp,new} \leftarrow X^{adp,new} \cup (x', \vartheta')$
            $X^{adp} \leftarrow X^{adp} \cup (x', \vartheta')$
            $O^{adp}(x', \vartheta') = O(x')$
return $\mathcal{T}^{adp} = \left(X^{adp}, U, \gamma^{adp}, \Pi, O^{adp}\right)$

C. Control Synthesis

Finally, given an ATS $\mathcal{T}^{adp}$ and an LTL formula φ, we construct the product automaton $\mathcal{T}^{adp} \otimes \mathcal{R}_{\varphi}$ as explained in Sec. II-E, and find the memoryless control strategy on $\mathcal{T}^{adp} \otimes \mathcal{R}_{\varphi}$ by solving the Rabin game. We also find the largest set of admissible initial conditions $X_0^{adp,\max}$ as the winning region of the Rabin game. In order to find $X_0^{\max}$, we perform the following projection:

$$X_0^{\max} = \left\{ x_0 \middle| (x_0, \Theta) \in X_0^{adp,\max} \right\}. \tag{7}$$

The adaptive control strategy takes the memoryless form $\Lambda : X \times 2^{\Theta}_{-\emptyset} \times S \to U$, which maps the current state in the PTS, the set of current possible ground truth parameters and the state in the Rabin automaton to an admissible control action.

Theorem 1: Given a finite system (1), an initial condition $x_0 \in X$, an LTL formula over Π, there exists a control strategy $\Lambda^* : X^* \times U^* \to U$ such that $O(x_0)O(x_1) \cdots \models \varphi$, $\forall \theta \in \Theta, \forall d_k \in D$, $x_{k+1} = F(x_k, u_k, \theta, d_k), \forall k \in \mathbb{N}$, if and only if $x_0 \in X_0^{\max}$. .

VI. Control Synthesis for Infinite Systems

Here we assume that PTS embedding (1) is not finite. We provide the solution for the case where X, U, Θ are infinite.

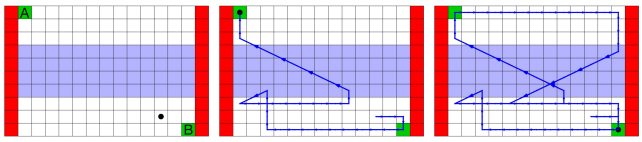

Fig. 1. Case Study 1: [Left]: The Robot (shown in black) and its environment. [Middle]: Snapshots of the executed Motion at time $k = 33$, and [Right] $k = 62$. The robot satisfies the specification.

We consider a finite observation preserving (see Sec. II-D) partition $Q_X = \{q_X^1, \cdots, q_X^{p_X}\}$ for X and a finite partition $Q_\Theta = \{q_\Theta^1, \cdots, q_\Theta^{p_\Theta}\}$ for Θ. We also quantize U to obtain a finite $U_{\text{qtz}} = \{u_{qtz}^1, \cdots, u_{qtz}^{p_u}\}$. In this paper, we do not consider any particular guideline for how to partition and leave this problem to our future work. In general, the finer the partitions, the less conservative the method is with a price of higher computational effort. "Smart" partition refinement procedures were studied in [24], [25].

Once partitions and quantizations are available, we compute the transitions. We denote the successor (post) of set q_X, under parameter set q_Θ and control u by

$$\text{Post}(q_X, q_\Theta, u) := \Big\{ x \in X \Big| \exists x \in q_X, \exists \theta \in q_\Theta, x \in \gamma(x, \theta, u) \Big\}. \tag{8}$$

A computational bottleneck is performing the post computation in (8). For additive parameters, the post computation is exact for piecewise affine systems using polyhedral operations [14]. For multiplicative parameters, an over-approximations of post can be computed [26], which introduces further conservativeness but retains correctness. Finally, we construct the quotient PTS from the infinite PTS. The procedure is outlined in Algorithm 2.

Algorithm 2 Constructing quotient PTS from infinite PTS

Require: $\mathcal{T}^\Theta = (X, U, \Theta, \gamma, \Pi, O)$
Require: $Q_X, Q_\Theta, U_{\text{quantized}}$
for $q_X \in Q_X$ **do**
  $O^Q(q_X) = O(x)$ for some $x \in q_X$
  for $q_\Theta \in Q_\Theta$ **do**
    for $u_{qtz} \in U_{qtz}$ **do**
      $X_{\text{post}} = \text{Post}(q_X, q_\Theta, u)$
      $\gamma^Q(q_X, u_{qtz}, q_\Theta) = \emptyset$
      for $q'_X \in Q_X$ **do**
        if $X_{\text{post}} \cap q'_X \neq \emptyset$ **then**
          $\gamma^Q(q_X, u_{qtz}, q_\Theta) \leftarrow \gamma^Q(q_X, u_{qtz}, q_\Theta) \cup q'_X$
return $\mathcal{T}^{Q,\Theta} = (Q_X, U_{\text{quantized}}, Q_\Theta, \gamma^Q, \Pi, O^Q)$

VII. Case Studies

We present two case studies. The first one is a simple finite deterministic system. The second case study involves a linear parameterized system that is infinite and non-deterministic due to the presence of additive disturbances.

A. Persistent Surveillance

We consider a robot motion planning problem. The environment is modeled as a finite number of cells illustrated in Fig. 1. Each cell corresponds to a state in X. We have $|X| = 150$. The set of control inputs is given by $U = \{$ **left, right, up, down**$\}$, where the transition enabled by each input corresponds to its unambiguous meaning. There exists an constant drift in the horizontal direction in the purple region, but its direction to left or right and its intensity are unknown. The set of possible drifts is $\Theta = \{+2, +1, 0, -1, -2\}$, where positive sign corresponds to the left direction. At each time, if the robot is in a purple cell, the drift is added to its subsequent position. For example, if the robot applies $u =$**right**, and $\theta^* = 2$, the robot actually ends up in a cell to the left. Similarly, if $u =$**up** and $\theta^* = -2$, the robot moves a cell up and two cells to the right. The red cells are "unsafe" regions that must be avoided, and the green cells A, B are "interesting" regions, which have to be persistently visited. The LTL formula describing this specification is:

$$\varphi = \mathbf{GF}A \ \wedge \ \mathbf{GF}B \ \wedge \ \mathbf{G}(\neg \text{unsafe}).$$

We implemented the procedure outlined in Sec. V. It is worth to note that there does not exist a pure robust control solution to this problem. In other words, if the robot ignores estimating the drift, it can not find a control strategy. For example, if the robot enters the purple region around the middle and persistently applies **up**, a maximum drift in either direction can drive the robot into the unsafe cells before it exits the purple region. Therefore, the only way the robot can fulfill the specification is to learn the drift. The robot first enters the drifty region to find out its value and then moves back and re-plans its motion. Notice that this procedure is fully automated using the solution of the Rabin game on the product $\mathcal{T}^{adp} \otimes \mathcal{R}_\varphi$. Two snapshots of the executed motion for the case $\theta^* = +2$ are shown in Fig. 1.

B. Safety Control

Consider the following one-dimensional linear system:

$$x^+ = (1 + \theta_1)x + \theta_2 u + \theta_3 + d, \tag{9}$$

where $\theta_1 \in [-0.5, 0.5]$, $\theta_2 \in [1, 2]$, and $\theta_3 \in [-0.2, 0.2]$ are fixed parameters, and $d \in D$, is the additive disturbance, $D = [-0.1, 0.1]$. The set of admissible control inputs is $U = [-1, 1]$. We desire to restrict x to the $[-1, 1]$ interval for all times, which is described by the following LTL formula:

$$\varphi = \mathbf{G}(x \leq 1) \wedge \mathbf{G}(x \geq -1).$$

We have $\Theta = [-0.5, 0.5] \times [1, 2] \times [-0.2, 0.2]$. We partitioned the intervals of θ_1, θ_2, θ_3, and X into 2,2,4, and 10 evenly spaced intervals, respectively. Thus, we have partitioned Θ into 16 cubes ($|Q_\Theta| = 16$) and X into 10 intervals ($|Q_X| = 10$). U is quantized to obtain $U_{qtz} = \{-1, -0.8, \cdots, 0.8, 1\}$. We implemented Algorithm 2 to obtain the quotient PTS and Algorithm 1 to find the corresponding ATS. The computation times were 0.1 (Algorithm 2) and 152 (Algorithm 1) seconds on a 3.0 GHz MacBook Pro. Even though $|X \times 2^{Q_\Theta}_{-\emptyset}| = 655350$, the number of reachable states obtained from Algorithm 1 was 14146. We solved the safety game on the ATS, which took less than a second and found a winning region

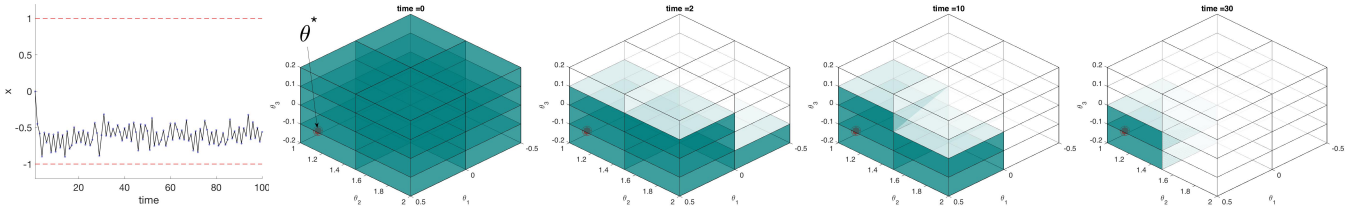

Fig. 2. Case Study 2: [Left]: trajectory of the system versus time, which is always between -1 and 1. [right]: Snapshots of ϑ_k at various times, which are illustrated by the shaded regions. They always contain the ground truth parameter $\theta_1^* = 0.45$, $\theta_2^* = 1.11$, $\theta_3^* = -0.18$.

containing 14008 states. The winning region in the state-space is $X_0 = [-0.6, 0.6]$. Since the solution is conservative, $X_0^{\max}$ may be larger if a finer partitioning is used. We also found that the winning region is empty if we had sought a pure robust control strategy. We simulated the system for 100 time steps starting from $x_0 = 0$. The values of disturbances at each time were chosen randomly with a uniform distribution over D. The specification is satisfied, and the sets given by the parameter estimator shrink over time and always contain the ground truth parameter, which in this case is $\theta_1^* = 0.45$, $\theta_2^* = 1.11$, $\theta_3^* = -0.18$. The results are shown in Fig. 2.

VIII. Conclusion and Future Work

We developed a framework to combine the recent advances in applications of formal methods in control theory with classical adaptive control. We used the concepts from transition systems, finite quotients, and product automata to introduce adaptive transition systems and correct-by-design adaptive control. Like most of other formal methods applications, our results suffer from high computational complexity. As discussed in the paper, the number of states in the ATS can be very large. Also, constructing finite quotients for infinite systems is computationally difficult. We plan to develop efficient methods to construct finite adaptive transition systems for special classes of hybrid systems such as mixed-monotone systems and piecewise affine systems.

References

[1] A. S. Morse, "Supervisory control of families of linear set-point controllers-part i. exact matching," *IEEE Transactions on Automatic Control*, vol. 41, no. 10, pp. 1413–1431, 1996.

[2] K. S. Narendra and C. Xiang, "Adaptive control of discrete-time systems using multiple models," *IEEE Transactions on Automatic Control*, vol. 45, no. 9, pp. 1669–1686, 2000.

[3] B. Anderson, T. Brinsmead, D. Liberzon, and A. Stephen Morse, "Multiple model adaptive control with safe switching," *International journal of adaptive control and signal processing*, vol. 15, no. 5, pp. 445–470, 2001.

[4] J. P. Hespanha, D. Liberzon, and A. S. Morse, "Overcoming the limitations of adaptive control by means of logic-based switching," *Systems & control letters*, vol. 49, no. 1, pp. 49–65, 2003.

[5] M. di Bernardo, U. Montanaro, and S. Santini, "Hybrid model reference adaptive control of piecewise affine systems," *IEEE Transactions on Automatic Control*, vol. 58, no. 2, pp. 304–316, 2013.

[6] M. di Bernardo, U. Montanaro, R. Ortega, and S. Santini, "Extended hybrid model reference adaptive control of piecewise affine systems," *Nonlinear Analysis: Hybrid Systems*, vol. 21, pp. 11–21, 2016.

[7] C. Baier, J.-P. Katoen, *Principles of model checking*. MIT press, 2008.

[8] M. Guay and M. Bürger, "Adaptive control of state constrained nonlinear systems in strict feedback form," in *American Control Conference (ACC), 2012*. IEEE, 2012, pp. 1143–1148.

[9] A. Aswani, H. Gonzalez, S. S. Sastry, and C. Tomlin, "Provably safe and robust learning-based model predictive control," *Automatica*, vol. 49, no. 5, pp. 1216–1226, 2013.

[10] S. Di Cairano, "Indirect adaptive model predictive control for linear systems with polytopic uncertainty," in *American Control Conference (ACC), 2016*. IEEE, 2016, pp. 3570–3575.

[11] M. Tanaskovic, L. Fagiano, R. Smith, and M. Morari, "Adaptive receding horizon control for constrained mimo systems," *Automatica*, vol. 50, no. 12, pp. 3019–3029, 2014.

[12] W. He, Y. Chen, and Z. Yin, "Adaptive neural network control of an uncertain robot with full-state constraints," *IEEE Transactions on Cybernetics*, vol. 46, no. 3, pp. 620–629, 2016.

[13] P. Tabuada, *Verification and Control of Hybrid Systems* . Springer Science & Business Media, 2008.

[14] C. Belta, B. Yordanov, and E. Aydin Gol, *Formal Methods for Discrete-Time Dynamical Systems*. Springer, 2017.

[15] W. Thomas, T. Wilke, *et al.*, *Automata, logics, and infinite games: a guide to current research*. Springer Science & Business Media, 2002, vol. 2500.

[16] J. F. Quindlen, U. Topcu, G. Chowdhary, and J. P. How, "Region-of-convergence estimation for learning-based adaptive controllers," in *American Control Conference (ACC), 2016*. IEEE, 2016, pp. 2500–2505.

[17] A. Kozarev, J. Quindlen, J. How, and U. Topcu, "Case studies in data-driven verification of dynamical systems," in *Proceedings of the 19th International Conference on Hybrid Systems: Computation and Control*. ACM, 2016, pp. 81–86.

[18] D. Sadigh, E. S. Kim, S. Coogan, S. S. Sastry, and S. A. Seshia, "A learning based approach to control synthesis of markov decision processes for linear temporal logic specifications," in *53rd IEEE Conference on Decision and Control*. IEEE, 2014, pp. 1091–1096.

[19] D. Aksaray, A. Jones, Z. Kong, M. Schwager, and C. Belta, "Q-learning for robust satisfaction of signal temporal logic specifications," in *Decision and Control (CDC), 2016 IEEE 55th Conference on*. IEEE, 2016, pp. 6565–6570.

[20] J. Fu, H. G. Tanner, J. Heinz, and J. Chandlee, "Adaptive symbolic control for finite-state transition systems with grammatical inference," *IEEE Transactions on Automatic Control*, vol. 59, no. 2, pp. 505–511, 2014.

[21] K. J. Leahy, P. Kannappan, A. Jardine, H. Tanner, J. Heinz, and C. Belta, "Integration of deterministic inference with formal synthesis for control under uncertainty," in *2016 American Control Conference (ACC)*, July 2016, pp. 4829–4834.

[22] J. Klein and C. Baier, "Experiments with deterministic ω-automata for formulas of linear temporal logic," *Theoretical Computer Science*, vol. 363, no. 2, pp. 182–195, 2006.

[23] K. Chatterjee and T. A. Henzinger, "A survey of stochastic ω-regular games," *Journal of Computer and System Sciences*, vol. 78, no. 2, pp. 394–413, 2012.

[24] B. Yordanov, J. Tümová, I. Černá, J. Barnat, and C. Belta, "Formal analysis of piecewise affine systems through formula-guided refinement," *Automatica*, vol. 49, no. 1, pp. 261–266, 2013.

[25] P. Nilsson and N. Ozay, "Incremental synthesis of switching protocols via abstraction refinement," in *53rd IEEE Conference on Decision and Control*. IEEE, 2014, pp. 6246–6253.

[26] B. Yordanov and C. Belta, "Formal analysis of piecewise affine systems under parameter uncertainty with application to gene networks," in *2008 American Control Conference*. IEEE, 2008, pp. 2767–2772.