Synthesis of Reactive Control Protocols for Differentially Flat Systems

Jun Liu, Ufuk Topcu, Necmiye Ozay, and Richard M. Murray

***Abstract*—We propose a procedure for the synthesis of control protocols for systems governed by nonlinear differential equations and constrained by temporal logic specifications. This procedure relies on a particular finite-state abstraction of the underlying continuous dynamics and a discrete representation of the external environmental signals. A two-player game formulation provides computationally efficient means to construct a discrete strategy based on the finite-state model. We focus on systems with differentially flat outputs, which, in a straightforward manner, allows the construction of continuous control signals from the discrete transitions dictated by the discrete strategy. The resulting continuous-time output trajectories are provably guaranteed to robustly satisfy the original specifications.**

I. Introduction

Synthesis of control protocols from linear temporal logic specifications has recently attracted considerable interest [1]–[7]. Given continuous dynamics of the underlying system and a temporal logic specification, a common approach for synthesizing control protocols is based on the following steps: First construct a finite transition system that se as an "approximation" of the continuous model (which typically has infinitely many states). Then, based on this finite-state model, use tools such as model checking [8] or two-player temporal logic game solvers [9], and construct a discrete protocol that realizes the given specification. Correctness of the continuous implementation of this discrete protocol is then guaranteed from the construction of the finite-state abstraction (which, for example, may simulate the continuous dynamics). Often, the bottleneck in this procedure is in the abstraction step. It either leads to abstract models with number of states exceeding the capabilities of the currently available tools or is not applicable due to the nonlinearity of the underlying continuous dynamics.

Partly motivated by the limitations in the applicability of the synthesis procedure to nonlinear dynamics, we consider the following setup in this paper. The temporal logic specifications are in terms of the observable outputs (not necessarily the same as the states and usually in a lower dimensional space) of the continuous dynamics. The underlying dynamics are allowed to be nonlinear. On the other hand, we focus on a particular class of nonlinear dynamics for which it has been known that solving point-to-point finite-time reachability problems is relatively easy. Namely, we consider nonlinear dynamics with differentially flat outputs [10].

The design procedure outlined in the following sections is based on particular discretizations of the time and continuous input and output sets. Due to the way these discretizations are generated, it is relatively straightforward to establish the transition relations between the discretized points; hence the finite-state abstract model. This finite-state abstract model is then used to construct a discrete protocol that implements the temporal logic specifications. The property that the dynamics are differentially flat is exploited in the continuous implementation of this discrete protocol. That is, once the current output value is observed and the discrete point "closest" to the observation is determined, the discrete protocol dictates the value the output shall reach over the current time interval. Due to the differential flatness, this finite-time point-to-point steering problem can now be solved effectively by using standard trajectory generation techniques [11]. Moreover, by using a notion of robust interpretation of LTL formulas, we formally prove that if the executions of the discrete protocol satisfy a certain specification given by LTL formulas, then the output trajectories generated by continuously implementing the protocol should robustly satisfy the original specifications.

The rest of the paper is organized as follows. In next section, we introduce differentially flat systems and formulate an LTL synthesis problem for such systems. Our procedure for solving the synthesis problem is outlined Section III, including a discretization procedure, temporal logic game formulation and solution technique, real-time implementation, and correctness proof. The procedure is illustrated by a simple example in Section IV. In Section V, we compare our work with existing work on abstraction-based approaches to controller synthesis.

II. Preliminaries and problem formulation

Consider a system of the form

$$\dot{x} = f(x, a), \quad y = h(x), \tag{1}$$

where $x(t) \in \mathbb{R}^n$ is the state, $a(t) \in \mathbb{R}^m$ the control input, and $y(t) \in \mathbb{R}^m$ the output. We assume that functions f and h are sufficiently smooth, i.e., they are kth continuously differentiable for some sufficiently large k. A standing assumption throughout this paper about system (1), called *differential flatness*, is formulated as follows.

Assumption 1. The function h has rank m, and there exist two functions $\Gamma : \mathbb{R}^{m(q+1)} \to \mathbb{R}^n$ and $\Theta : \mathbb{R}^{m(q+2)} \to \mathbb{R}^m$ of rank n and m, respectively, in their domains, such that the integral curve of (1) identically satisfy the equations

$$x = \Gamma(y, \dot{y}, \cdots, y^{(q)}), \quad u = \Theta(y, \dot{y}, \cdots, y^{(q+1)}). \tag{2}$$

This work was supported in part by the NSERC of Canada, the FCRP consortium through the Multiscale Systems Center (MuSyC), and the Boeing Corporation.

The authors are with Control and Dynamical Systems, California Institute of Technology, Pasadena, CA 91125, USA. {liu, necmiye, utopcu, murray}@cds.caltech.edu

Moreover, $m(q+1)=n$, which together with the rank of Γ, implies that Γ has an inverse, denoted by Γ^{-1}.

Differential flatness has proved to be useful in the design of advanced control and supervision schemes for nonlinear systems (see, e.g., [10], [12]–[14]). The statement of Assumption 1 above is taken from [12]. This paper proposes a systematic way of synthesizing correct-by-construction controllers, by exploiting differential flatness of system (1), such that the *output* of the system satisfies certain high-level specifications.

A. Linear temporal logic

We use linear temporal logic (LTL) [15], [16] to formally specify system properties. Standard LTL is built upon a finite set of atomic propositions, logical operators $\neg$ (negation) and $\vee$ (disjunction), and the temporal modal operators $\bigcirc$ (next) and $\mathcal{U}$ (until).

An *atomic proposition* is a statement on the system output variable that has a unique truth value (`True` or `False`) for a given output in $\mathbb{R}^m$. Equivalently, an atomic proposition is uniquely characterized by the subset of $\mathbb{R}^m$ on which the proposition holds true. Formally, given a set of atomic propositions Π, the set of LTL formulas over Π can be defined inductively as follows:

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

Additional logical operators, such as $\wedge$ (conjunction), $\rightarrow$ (material implication), and temporal modal operators $\Diamond$ (eventually), and $\Box$ (always), are defined by:

(a) $\varphi \wedge \psi := \neg(\neg\varphi \vee \neg\psi)$;
(b) $\varphi \rightarrow \psi := \neg\varphi \wedge \psi$;
(c) $\texttt{True} := p \vee \neg p$, where $p \in \Pi$;
(d) $\Diamond\varphi := \texttt{True}\,\mathcal{U}\varphi$;
(e) $\Box\varphi := \neg\Diamond\neg\varphi$.

A *propositional formula* is one that does not include any temporal operators.

In order to reason about the correctness of the output trajectories both at the continuous level and the discrete level, we shall interpret LTL over both the continuous output trajectories of (1) and sequences of output values taking the form $y_0, y_1, y_2, \cdots$. Given an atomic proposition $\pi \in \Pi$, let $[\![\pi]\!]$ denote the subset of $\mathbb{R}^m$ on which π holds true. Formal interpretations of LTL formulas are given as follows.

Continuous Semantics of LTL: Given an LTL formula φ without the next operator $\bigcirc$, we can recursively define the satisfaction of φ over an output trajectory $y(t)$ at time t, written $y(t) \Vdash \varphi$, as follows:

(1) for any atomic proposition $\pi \in \Pi$, $y(t) \Vdash \pi$ if and only if $y(t) \in [\![\pi]\!]$;
(2) $y(t) \Vdash \neg\varphi$ if and only if $y(t) \nVdash \varphi$;
(3) $y(t) \Vdash \varphi \vee \psi$ if and only if $y(t) \Vdash \varphi$ or $y(t) \Vdash \psi$; and
(4) $y(t) \Vdash \varphi\mathcal{U}\psi$ if and only if there exists $t' \geq t$ such that $y(t) \Vdash \psi$ and $y(t) \Vdash \varphi$ for all $s \in [t, t')$.

An output trajectory $y(t)$ starting at 0 is said to satisfy φ if and only if $y(0) \Vdash \varphi$.

Discrete Semantics of LTL: Given an LTL formula φ, we can recursively define the satisfaction of φ over a sequence of outputs of (1), y_0, y_1, y_2, $\cdots$, at position i, written $y_i \vDash \varphi$, as follows:

(1) for any atomic proposition $\pi \in \Pi$, $y_i \vDash \pi$ if and only if $y_i \in [\![\pi]\!]$;
(2) $y_i \vDash \neg\varphi$ if and only if $y_i \nvDash \varphi$;
(3) $y_i \vDash \bigcirc\varphi$ if and only if $y_{i+1} \vDash \varphi$;
(4) $y_i \vDash \varphi \vee \psi$ if and only if $y_i \vDash \varphi$ or $y_i \vDash \psi$;
(5) $y_i \vDash \varphi\mathcal{U}\psi$ if and only if there exists $j \geq i$ such that $y_i \vDash \psi$ and $y_k \vDash \varphi$ for all $k \in [i, j)$.

An output sequence y_0, y_1, y_2, $\cdots$ is said to satisfy φ if and only if $y_0 \vDash \varphi$.

B. Problem formulation

The main problem of the paper is formulated as follows.

Problem 1. Given an LTL formula φ, find a control input $a : \mathbb{R}^+ \rightarrow \mathbb{R}^m$ such that the output trajectory y of (1) satisfies $y(0) \Vdash \varphi$.

C. Robust interpretation of LTL formulas

To reason about the satisfaction of an LTL formula by different output trajectories and/or sequences that are close to each other, we employ a notion of robust interpretation of LTL formulas [2]. Roughly speaking, we need a notion of robust satisfaction that captures the following: if $y_1(t) \Vdash \varphi$ *robustly* for some formula φ and some output trajectory $y_1(t)$ and we know that $y_2(t)$ stays *close* to $y_1(t)$ for all $t \geq 0$, then we should have $y_2(t) \Vdash \varphi$. Similar implications should hold among output sequences and between output sequences and trajectories.

We start with the set of atomic propositions Π and introduce the set of negated atomic propositions $^\neg\Pi := \{\neg\pi : \pi \in \Pi\}$. We form a new set of atomic propositions $\hat{\Pi} := \Pi \cup {}^\neg\Pi$. Given an LTL formula φ, we write it in Negation Normal Form (NNF) [17, Section 9.4]. For all occurrences of negations of atomic propositions in φ in its NNF, we treat them as atomic propositions from $\hat{\Pi}$. In this sense, all LTL formulas on Π can be seen as an LTL formula built upon $\hat{\Pi}$ without the negation operator, only that now we need both the logical operators $\vee$ and $\wedge$ to derive the full set of LTL formulas.

We further define the notions of *ε-contraction* [2] and *ε-inflation* of an atomic proposition π for some given $\varepsilon \geq 0$ (denoted by π_ε and π^ε, respectively) by

$$[\![\pi_\varepsilon]\!] := \{z \in [\![\pi]\!] : \, z + \varepsilon\mathbb{B} \in [\![\pi]\!]\}.$$

and $[\![\pi^\varepsilon]\!] := [\![\pi]\!] + \varepsilon\mathbb{B}$, where $\mathbb{B}$ denote the unit closed ball in $\mathbb{R}^m$. The above definition makes sense because an atomic proposition π is uniquely characterized by the set $[\![\pi]\!]$. Furthermore, ε-contraction and ε-inflation of a set of atomic propositions, in particular, the set $\hat{\Pi}$, are defined by mapping each element of $\hat{\Pi}$ to its ε-contraction and ε-inflation, respectively.

Definition 1. Given an LTL formula φ built upon Π, its ε-contraction (respectively, ε-inflation), denoted by φ^ε (respectively, φ_ε) can be obtained in three steps: i) write φ in NNF; ii) treat it as a formula built upon $\hat{\Pi}$; iii) replace each occurrence of atomic proposition from $\hat{\Pi}$ by its ε-contraction (respectively, ε-inflation).

Proposition 1. For $\delta \geq \varepsilon \geq 0$, we have $\varphi^0 = \varphi_0 = \varphi$, $(\varphi^\varepsilon)^\delta = (\varphi^\delta)^\varepsilon = \varphi^{\varepsilon+\delta}$, $(\varphi^\varepsilon \to \varphi^\delta) = \texttt{True}$, $(\varphi_\delta \to \varphi_\varepsilon) = \texttt{True}$, $((\varphi_\varepsilon)^\delta \to \varphi^{\delta-\varepsilon}) = \texttt{True}$, and $((\varphi_\delta)^\varepsilon \to \varphi_{\delta-\varepsilon}) = \texttt{True}$ for any LTL formula φ.

In addition, the following proposition is useful reason about the satisfaction of an LTL formula by different output trajectories and/or sequences.

Proposition 2. Let $y_0,\, y_1,\, y_2,\, \cdots$ and $y_0',\, y_1',\, y_2',\, \cdots$ be two output sequences and $y(t)$ and $\hat{y}(t)$ two output trajectories. Let $\varepsilon > 0$ be a positive constant. Let φ be an LTL formula without the next operator. Then the following statements hold:

(i) If $y_0 \vDash (\varphi_\varepsilon)^\varepsilon$, then $y_0 \vDash \varphi$. Similarly, $y(0) \Vdash (\varphi_\varepsilon)^\varepsilon$ implies $y(0) \Vdash \varphi$;
(ii) If $y_0 \vDash \varphi$ and $\sup_{k\geq 0} |y_k - y_k'| \leq \varepsilon$, then $y_0' \vDash \varphi^\varepsilon$;
(iii) If $y(0) \Vdash \varphi$ and $\sup_{t\geq 0} |y(t) - \hat{y}(t)| \leq \varepsilon$, then $\hat{y}(0) \Vdash \varphi^\varepsilon$;
(iv) Let $y(t)$ be given by the linear interpolation of the sequence $(t_0, y_0),\, (t_1, y_1),\, (t_2, y_2),\, \cdots$, with $t_k \to \infty$ as $k \to \infty$. If $y_0 \vDash \varphi$ and there exists $\Delta > 0$ such that $\sup_{k\geq 0} |y_{k+1} - y_k| \leq \Delta$, then $y(0) \Vdash \varphi^{\Delta/2}$;
(v) Let $y_k = y(k\tau)$ for some $\tau > 0$ and all $k \geq 0$. If $y(0) \Vdash \varphi$ and $y(t)$ is Lipschitz in t with Lipschitz constant γ, then $y_0 \vDash \varphi^{\gamma\tau/2}$.

Statement (iii) of Proposition 2 above is essentially Theorem 16 of [2] and (ii) is its discrete analogue. The statements given by (iv) and (v) seem to be new and they turn out to be useful in reasoning about the satisfaction of an LTL formula by output trajectories and sequences. In particular, statement (v) above reflects the fact that a continuous trajectory satisfying an LTL formula in general does not imply a sequence taken from this trajectory would satisfy the same LTL formula. Statement (iv) shows the reverse. In both cases, an inflated specification is considered. In (ii) and (iii), it is shown that if two output trajectories or sequences are ε-close, one should satisfy an ε-inflated specification that is satisfied by the other. Finally, (i) shows that we can still enforce a given specification through a composition of inflation and contraction of the specification, an argument that can be useful to reason about implications of correctness among trajectories and sequences.

III. Control protocol synthesis based on flatness and temporal logic games

Based on the continuous-time nonlinear system model (1) and an LTL specification, our approach to control protocol synthesis Problem 1 consists of three steps. We first establish a finite-state abstraction of system (1), which is a finite transition system that approximates the output behavior of (1). By exploiting flatness of the output of (1), such a transition system can be constructed in a somewhat trivial way. Based on this high-level discrete abstraction, we then synthesize a discrete control strategy from a modified LTL specification such that the executions of the transition system satisfy this specification. In particular, we take a specific temporal logic game approach to this synthesis problem, which is amenable to polynomial-time solvers and allows us to further incorporate environmental adversaries in the formulation. Finally, differential flatness of the system allows us to continuously steer the output of the system according to the discrete strategy. Correctness proof is given to ensure that the output trajectories of (1) generated by this approach satisfy the original LTL specification.

A. Construction of finite abstraction

Let $\mathbb{Z}^m$ denote the m-dimensional integer lattice, which is the lattice in $\mathbb{R}^m$ whose lattice points are m-tuples of integers. Consider two compact sets U and Y in $\mathbb{R}^m$. For given parameters $\eta > 0$ and $\mu > 0$, define

$$U_\mu := \mu\mathbb{Z}^m \cap U, \quad Y_\eta := \eta\mathbb{Z}^m \cap Y,$$

where $\mu\mathbb{Z}^m := \{\mu z : z \in \mathbb{Z}^m\}$. We consider the *labelled transition system* $TS = (Y_\eta, U_\mu, \to)$, where $\to \subseteq Y_\eta \times U_\mu \times Y_\eta$ are the labelled transitions defined by $(y, u, y') \subseteq \to$ if and only if $y' = y + \tau u$, where τ is set to be η/μ, as a time discretization parameter. Figure 1 illustrates a typical transition of such a transition system obtained by discretizing a two-dimensional domain. An *execution* of TS is a sequence $(y_0, u_0),\, (y_1, u_1),\, (y_2, u_2),\, \cdots$, where $y_i \in Y_\eta$, $y_i \in U_\mu$, and (q_i, u_i, q_{i+1}) for all $i \geq 0$. Given an LTL formula φ, the above execution is said to satisfy φ if and only if the sequence sequence $y_0,\, y_1,\, y_2,\, \cdots$ satisfies φ.

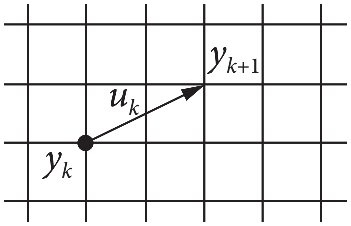

Fig. 1: A typical example of transitions enabled in a finite abstraction obtained by discretization. The transition is given by (y_k, u_k, y_{k+1}), where $y_k,\, y_{k+1} \in Y_\eta$ and $u_k = (y_{k+1} - y_k)/\tau \in U_\mu$. Note that, since $\eta = \mu\tau$, we have $u_k \in U_\mu$ is guaranteed if $u_k \in U$.

B. Solution through temporal logic game

We propose a temporal logic game approach for the synthesis of a discrete strategy to guarantee that the executions of TS will satisfy φ. Our approach leverages recent work on reactive synthesis [18], [19] of controllers for systems interacting with adversarial environments [7], where a control protocol is synthesized to generate a sequence of control signals to ensure that a plant meets its specification for all allowable behaviors of the environment. The synthesis problem is viewed as a two-player game between the environment and the plant: the environment attempts to falsify the specification and the plant tries to satisfy it.

A game is constructed by introducing an additional discrete variable e, which represents the behavior of environmental adverasries at the discrete level. Such adversaries do not impact the continuous dynamics of the system directly, but rather constrain its behavior through GR(1) specifications of the form

$$\varphi = (\varphi_e \to \varphi_s), \tag{3}$$

where φ_e specifies allowable environment behaviors and φ_s is a system level specification that enforces correct system behaviors for all valid environment behaviors. To be more precise, for $\alpha \in \{s, e\}$, each φ_α in (3) has the following structure:

$$\varphi_\alpha := \varphi_{\text{init}}^\alpha \wedge \bigwedge_{i \in I_1^\alpha} \square \varphi_{1,i}^\alpha \wedge \bigwedge_{i \in I_2^\alpha} \square \diamond \varphi_{2,i}^\alpha,$$

where $\varphi_{\text{init}}^\alpha$ is a propositional formula characterizing the initial conditions; $\varphi_{1,i}^\alpha$ are transition relations characterizing safe, allowable moves and propositional formulas characterizing invariants; and $\varphi_{2,i}^\alpha$ are propositional formulas characterizing states that should be attained infinitely often. Many interesting temporal specifications can be transformed into this form. The readers can refer to [18], [19] for more precise treatment on how to use GR(1) game to solve LTL synthesis in many interesting cases (see also [20]–[22] for more examples and applications). A winning strategy for the system, i.e., a strategy such that formula (3) is satisfied, can be solved by a symbolic algorithm within time complexity that is quadratic in the size of the state space [19].

Formally, the two-player game approach can be described as follows.

Two-Player Game: A state of the game $s = (e, y, u)$ is in $E \times Y_\eta \times U_\mu$, where E, Y_η, and U_μ represent finite sets of environment, output, and input values, respectively. A transition of the game is a move of the environment and a move of the output, followed by a move of the control input. A *discrete control strategy* can be defined as a partial function $(s_0 s_1 \cdots s_{t-1}, (e_t, y_t)) \mapsto u_t$, which chooses an input u_t based on the state sequence so far and the current move of the environment. In this sense, a discrete control strategy is a winning strategy for the system such that the specification φ is met for all behaviors of the environment. We say that φ is *realizable* if such a strategy exists. If the specification is realizable, solving the two-player game gives a control strategy in the form of a finite automaton, which stores, for each possible environment move, an appropriate control input that should be taken to ensure that the execution produces correct output sequences.

C. Real-time implementation

We now build from discrete inputs u_k, $k \geq 0$, dictated from the discrete protocol, a continuous-time control input $a : \mathbb{R}^+ \to \mathbb{R}^m$ such that Problem 1 is solved. The procedure exploits the fact that the output of system (1) is flat, for which a point-to-point steering problem can be easily solved without approximation or requiring to integrate the system equations. However, we do require that the output trajectory satisfies the LTL specification for all time, not merely at an output sequence of discrete times. This is in contrast with the approximate simulation-based approaches for continuous-time nonlinear systems, where the approximation simulations are established between a time-discretized system and a finite transition system and hence the closeness of approximation is only enforced for a sequence of states at discrete times.

We start with $y(0) \in Y_\eta$, the observed output of system (1) at time 0. Let y_0 be the point in Y_η that is closest to $y(0)$. If there are multiple of points with same distance, we pick a random one of them. Based on the current environment behavior, we choose u_0 from the discrete control strategy, which by construction of the discrete abstraction, will lead to $y_1 = y_0 + \tau u_0$. We look for a control input on $[0, \tau]$ that steers the output of (1) from $y(0)$ to $y(\tau) = y(0) + \tau u_0$. On each of the intervals $[k\tau, (k+1)\tau]$, $k = 0, 1, \cdots$, we construct $y(t)$ using polynomial interpolation, which is commonly used to construct output trajectory for differentially flat systems [12], [13]. Consider

$$y_{(j)}(t) = \sum_{i=0}^{2q+1} c_{i,j} (t - k\tau)^i / \tau^i, \quad t \in [k\tau, (k+1)\tau], \tag{4}$$

where $y_{(j)}(t)$ denotes the j-th component of $y(t)$ (not to be confused with the sequence of outputs y_0, y_1, $\cdots$) and $c_{i,j}$ are constant coefficients to be determined. For comparison, we also introduce two functions

$$\hat{y}(t) = y(k\tau) + (t - k\tau)u_k, \quad \tilde{y}(t) = y_k + (t - k\tau)u_k, \tag{5}$$

on the interval $[k\tau, (k+1)\tau]$, where $y(k\tau)$ is the output at $t = k\tau$, y_k is the closest point to $y(k\tau)$ we picked from Y_η, and u_k is the control input read from the winning automaton obtained by solving the temporal logic game in the previous section. Note that $|\hat{y}(t) - \tilde{y}(t)| \leq \eta/2$ for all $t \geq 0$. The implementation procedure and the functions $y(t)$, $\hat{y}(t)$, and $\tilde{y}(t)$ are illustrated by Figure 2.

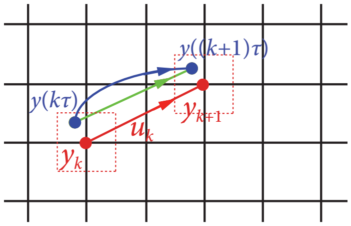

Fig. 2: Illustration of the real-time implementation procedure based on transitions dictated from the discrete strategy: $y(k\tau)$ is the observed output at $k\tau$. The closest point on the grid is y_k. The transition to be enforced by the discrete strategy is $y_{k+1} = y_k + \tau u_k$. The next output value to be reach at the end of the time interval $[k\tau, (k+1)\tau]$ is $y((k+1)\tau) = y(k\tau) + \tau u_k$. The trajectory being generated is denoted by the blue curve, while the functions in comparison, $\hat{y}(t)$ and $\tilde{y}(t)$ are given by the green and red lines, respectively.

By (2), the boundary conditions enforced at $t = 0$ are

$$(y(0), \dot{y}(0), \cdots, y^{(q)}(0)) = \Gamma^{-1}(x(0)). \tag{6}$$

For $k \geq 1$, boundary conditions at $t = k\tau$ should enforce $y(k\tau) = y((k-1)\tau) + \tau u_{k-1}$, while higher-order derivatives of y at $t = k\tau$ can be chosen for design purposes. Here we choose

$$\begin{aligned}(y(k\tau), \dot{y}(k\tau), \cdots, y^{(q)}(k\tau)) \\ = (y((k-1)\tau) + \tau u_{k-1}, u_{k-1}, 0, \cdots, 0),\end{aligned} \quad (7)$$

which is the same as in [12]. Equation (6) can be used to determine the coefficients $c_{0,j}, \cdots, c_{q,j}$ from (4) by

$$c_{i,j} = \frac{\tau^i}{i!} y^{(i)}_{(j)}(k\tau), \quad j = 1, \cdots, m. \quad (8)$$

Equation (7) can be used to determine the rest $m(q+1)$ coefficients. In summary, the equations that determine $c_{q+1,j}$, $\cdots$, $c_{2q+1,j}$ for a given j can be written as

$$\begin{bmatrix} 1 & 1 & \cdots & 1 \\ q+1 & q+2 & \cdots & 2q+1 \\ \vdots & \vdots & \cdots & \vdots \\ (q+1)! & \frac{(q+2)!}{2} & \cdots & \frac{(2q+1)!}{(q+1)!} \end{bmatrix} \begin{bmatrix} c_{q+1,j} \\ \vdots \\ c_{2q+1,j} \end{bmatrix}$$

$$= \begin{bmatrix} y_{(j)}(k\tau) + \tau u_{k,(j)} - \sum_{i=0}^{q} \frac{\tau^i}{i!} y^{(i)}_{(j)}(k\tau) \\ \tau u_{k,(j)} - \sum_{i=1}^{q} \frac{\tau^i}{(i-1)!} y^{(i)}_{(j)}(k\tau) \\ - \sum_{i=2}^{q} \frac{\tau^i}{(i-2)!} y^{(i)}_{(j)}(k\tau) \\ \vdots \\ \tau^q y^{(q)}_{(j)}(k\tau) \end{bmatrix}. \quad (9)$$

By solving the above linear equation at each time $t = k\tau$, we obtain the desired output trajectory $y(t)$ for the next interval $[k\tau, (k+1)\tau]$. The appropriate continuous-time control input $a(t)$ to achieve $y(t)$ can be obtained from the map Θ in (2).

We will formally prove in the next section that the output trajectory generated by this procedure will satisfy the original specification φ and therefore solve our Problem 1.

D. *Proof of correctness*

Note that in both (8) and (9), $y(k\tau)$, $\dot{y}(k\tau)$, $\cdots$, $y^{(q)}(k\tau)$ are given by (6) for $k = 0$ and by (7) for $k \geq 1$. From (4), (8), and (9), we can show that

$$|y(t) - \hat{y}(t)| \leq C\tau, \quad t \in [k\tau, (k+1)\tau], \quad (10)$$

where C is a finite constant independent of τ that can be readily computed from (4), (8), and (9). The finiteness of C follows from the fact that it only depends on $x(0)$, $y(k\tau)$ and u_k and that Y_η and U_μ are compact domains. Note that for the same reason, the continuous-time control signal given by 2 is also bounded.

Recall from (5) that $|\hat{y}(t) - \tilde{y}(t)| \leq \eta/2$ for all $t \geq 0$. Now suppose that there exist $\delta > \varepsilon \geq 0$ such that

$$\varepsilon \leq \delta - u_{\max}\tau/2 - \eta/2 - C\tau, \quad (11)$$

where $u_{\max} > 0$ is such that $|u| \leq u_{\max}$ for all $u \in U_\mu$.

Theorem 1. Let δ and ε be from (11). If the sequence $y_0, y_1, \cdots$ satisfies $y_0 \vDash \varphi_\delta$, then the output trajectory $y(t)$ implemented in the previous section satisfies $y(0) \Vdash \varphi_\varepsilon$, which, in particular, implies $y(0) \Vdash \varphi$.

Proof: Proposition 2(iv) implies that $\tilde{y}(0) \vDash (\varphi_\delta)^{u_{\max}\tau/2}$. Furthermore, since $\tilde{y}$ and $\hat{y}$ are $\eta/2$-close, and $\hat{y}$ and y are $(C\tau)$-close, it follows from Propositions 1 and 2(iii) that $\hat{y}(0) \Vdash (\varphi_\delta)^{u_{\max}\tau/2+\eta/2+C\tau}$, which implies $y(0) \Vdash \varphi_\varepsilon$ and, in particular, $y(0) \Vdash \varphi$. ■

The above theorem says that if there exists a discrete strategy that can realize a given specification φ with certain robustness margin that can compensate for the approximation errors, then we can implement a continuous output trajectory that also realizes φ with some robustness margin.

On the other hand, we can show that, if there exists a continuous output trajectory that satisfies φ with certain robustness margin, then, by refining the discretization for the output space Y, we can make the discrete synthesis problem *nominally* realizable (i.e., when without environmental constraints) with respect to the specification φ, also subject to certain robustness margin.

Suppose that τ and η are chosen sufficiently small such that

$$\delta \leq \varepsilon - \gamma\tau/2 - \eta/2, \quad (12)$$

holds for some $\varepsilon > \delta \geq 0$ and γ, where $u_{\max} > 0$ is such that $|u| \leq u_{\max}$ for all $u \in U_\mu$.

Theorem 2. Let δ, ε, and γ be from (12). Suppose that $y(t)$ be an output trajectory of (1) such that $y(0) \Vdash \varphi_\varepsilon$. Suppose that y is Lipschitz in t with Lipschitz constant γ and U_μ is such that $(\mu+\gamma)B \cap \mu\mathbb{Z}^m \subseteq U_\mu$. Then there exists a sequence $y_0, y_1, \cdots$ in Y_η such that $y_k = y_{k-1} + \tau u_{k-1}$, with $u_{k-1} \in U_\mu$ for all $k \geq 1$, and $y_0 \vDash \varphi_\delta$.

Proof: Proposition 2(v) implies that the sequence of outputs $y(0), y(\tau), y(2\tau), \cdots$ satisfies the specification $(\varphi_\varepsilon)^{\gamma\tau/2}$. By the definition of Y_η, we can pick a sequence $y_0, y_1, \cdots$ in Y_η such that $|y_k - y(k)| \leq \eta/2$. If we define $u_k = (y_{k+1} - y_k)/\tau$ for all $k \geq 0$, then $|u_k| \leq \eta/\tau + \gamma = \mu + \gamma$, which implies $u_k \in U_\mu$. Furthermore, by Proposition 1, $y_0 \vDash (\varphi_\varepsilon)^{\gamma\tau/2+\eta/2}$, which implies $y_0 \vDash \varphi_\delta$. ■

IV. Example

Consider two vehicles driving on two different but intersecting roads. The dynamics of the vehicles is given by

$$\dot{x}_1 = x_2, \quad \dot{x}_2 = u - \rho x_2^2, \quad (13)$$

where the term $-\rho x_2^2$ with $\rho > 0$ models air drag and the static friction can be lumped into the control input. The initial conditions for both vehicles are given by $(0, 0)$. The choice of output $y = x_1$ turns out to be flat, since we have

$$\begin{aligned}(x_1, x_2) &= (y, \dot{y}) = \Gamma(y, \dot{y}), \\ u &= \ddot{y} + \rho\dot{y}^2 = \Theta(y, \dot{y}, \ddot{y}),\end{aligned}$$

which satisfy (2). We will use y_j, $j = 1, 2$, to represent the positions the two vehicles. Suppose that $Y = [0, 10] \times [0, 10]$ and $U = [-1, 1]$. We choose the discretization parameters $\tau = \eta = 0.5$ and $\mu = 1$. Therefore, $Y_\eta = \{0, 1, \cdots, 10\}$ and $U_\mu = \{-1, 0, 1\}$.

The model was taken from [12]. Here we considered more general LTL specifications, including both liveness and safety, and also environmental constraints. The specifications involve: (a) two target areas $T_1 = [0, 2]$ and $T_2 = [8, 10]$ for both vehicles, which they are expected to visit, (b) the intersection of the road $I = \{4.7 < y_1 < 5.3,\ 4.7 < y_2 < 5.3\}$, where the two vehicles cannot appear at the same time, and (c) traffic lights at the intersection, which are enforced only during certain periods of times. Both vehicles are expected to satisfy the following:

(P1) visit the target areas T_1 and T_2 infinitely often;

(P2) if the intersection is not controlled by traffic lights, do not enter I simutanenously;

(P3) if the intersection is controlled by traffic lights, obey the traffic lights.

Note that given a discrete strategy, the trajectories can be generated by (4), where the coefficients $c_{i,j}$, $i = 0,3$ and $j = 1,2$ are given by (8) and (9), which in this case reduce to

$$c_{0,j} = y_j(k\tau), \quad c_{1,j} = \dot{y}_j(k\tau)\tau,$$
$$c_{2,j} = 2\tau(u_{k,(j)} - \dot{y}_j(k\tau)), \quad c_{3,j} = -\tau(u_{k,(j)} - \dot{y}_j(k\tau)),$$

on each of of the intervals $[k\tau, (k+1)\tau]$. To get an estimate on C in (10), we see that

$$|y_j(t) - \hat{y}_j(t)| = |s - 2s^2/\tau + s^3/\tau^2||\dot{y}_j(k\tau) - u_{k,(j)}|,$$

where $0 \le s = t - k\tau \le \tau$ for $t \in [k\tau, (k+1)\tau]$. It is easy to verify that $|s - 2s^2/\tau + s^3/\tau^2| \le 4\tau/27$. Moreover, $\dot{y}_j(k\tau) = u_{k-1,(j)}$ for $k \ge 1$ and $\dot{y}_j(0) = x_{2,(j)}(0) = 0$, which implies $|\dot{y}_j(k\tau) - u_{k,(j)}| \le 2$ for all $k \ge 0$. Therefore C can be chosen as $8/27$ in (10), and (11) is satisfied with $\delta = 0.7$. and $\varepsilon = 0.05$. Theorem 1 says that if we synthesize the discrete strategy such that a δ-contracted version of the specifications (P1)–(P3) are satisfied, which in this particular case means, the target sets now become $[0.7, 1.3]$, while the set to be avoided becomes $I = \{4 < y_1 < 6,\ 4 < y_2 < 6\}$. We can then formulate a game for synthesizing a discrete strategy that enforces this specification. Simulation results are shown in Figure 3, illustrating continuous implementations of this strategy. We can see that the continuous trajectories indeed satisfy the original specifications (P1)–(P2).

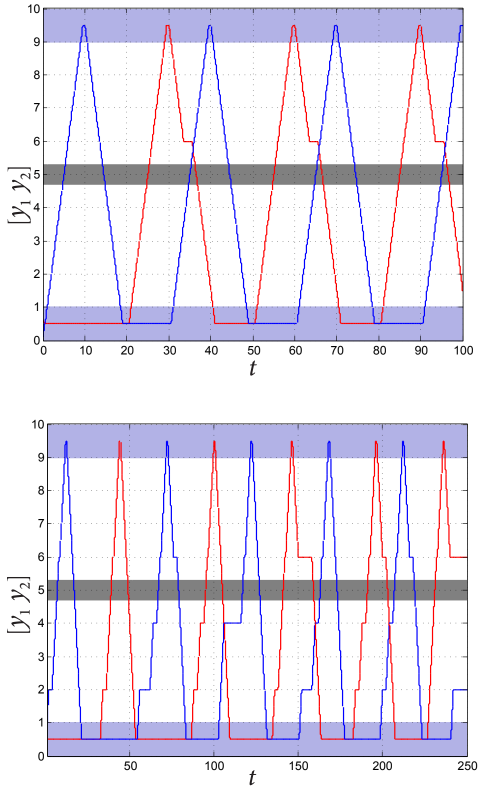

Fig. 3: The upper figure shows case the where there is no traffic lights and two vehicles (positions denoted by read and blue curves) cooperatively try to avoid enter the intersection (grey zone) simultaneously, while visiting each of their respective targets (blue zone) infinitely often. The lower figure shows the case where the intersection is controlled by traffic lights, which can be a random signal (e.g., depending on pedestrians crossing) that enables one of the vehicles to cross the intersection.

V. Comparison with existing work

There has been considerable amount of work on abstraction-based methods for controller synthesis. The success of these methods to synthesize a controller satisfying certain specifications heavily relies on efficient algorithms for constructing a finite abstraction, often in the form of a finite transition system. One way of deriving such abstraction from a continuous-state model is to conduct a partitioning (or discretization) on the continuous-state space. Then transition relations (either deterministic or nondeterministic) among these partitions have to be computed. While finding a finite abstraction is in general challenging, by limiting the underlying dynamics, e.g., to affine systems or affine-in-control systems, efficient methods have been developed and applied for reachability controller synthesis [24]–[26]. These methods can potentially be used to compute the transitions systems independently from the specification, and be used together with model checking tools or game solvers to perform controller synthesis from high-level specifications given by fragments of LTL [23]. There has also been recent work [27], [28] focusing on how to construct such abstractions for discrete-time (sampled) nonlinear systems. In general, such methods are limited by the efficiency of computing finite abstractions for systems with general dynamics. They are also limited by the fact that even though a controller exists, it may not be found by the abstraction-based synthesis procedure (the abstractions essentially only simulate the continuous dynamics). The notion of bisimulation relations [29] between the abstract and concrete systems is used to characterize the case where this does not happen. However, finite bisimulation relations are known to only exist for very limited classes of systems [30]. Recent work then has focused on finding approximate bisimulation relations [31], [32]. In [33]–[35], it has been shown that approximate bismulations can be obtained between a quantized control system and a finite transition system, if the underlying continuous-time nonlinear system is incrementally stable. The work [36] shows that such stability condition can be further relaxed to incremental forward completeness. In both cases, a Lyapunov-like function plays an important role in providing an error bound on the approximation bisimulation relations. In general, these works focus on proving existence of approximate abstractions, and do not explicitly address the problem of controller synthesis for enforcing certain specifications. On the other hand, when temporal logic specifications are considered, the underlying dynamics is usually assumed to be in a relatively simple form, e.g., fully actuated or linear dynamics [1]–[4], [22].

In this paper, we consider a class of nonlinear systems that have differentially flat outputs. This class of systems is general enough to cover a variety of interesting examples, e.g., overhead cranes, cars with trailers, conventional aircraft, induction motors, active magnetic bearings, and chemical reactors (see, e.g., [11] and references therein). Yet the special structure of differential flatness enables us to construct finite abstractions in a straightforward way through discretization. Furthermore, the specifications considered are also quite general, i.e., the GR(1) fragment of LTL. Many interesting examples and applications can be formulated as GR(1) games (see, e.g., [20]–[22]). Such formulation also allows to incorporate environmental adversaries, whereas none of the approaches mentioned in the previous paragraph considers adversarial environment, with [4] as an exception. The checking of realizability of GR(1) game and the synthesis of a discrete strategy have been shown to be of polynomial-time complexity [19].

The work in the current paper has been inspired, in particular, by the work [12] and [2]. In [12], by exploiting differential flatness, the authors construct a supervisor that enforces safety rules. In [2], the authors address the temporal logic motion planning where the underlying dynamics is given by a second-order kinematic model. We borrowed, in particular, from [2], a notion of robust interpretation of LTL formulas that can be used to account for the approximation error. Instead of using this notion between continuous trajectories alone, we use it also to reason about the implications of correctness among continuous trajectories and discrete sequences.

VI. Conclusion

We have proposed a procedure for the synthesis of control protocols based on differential flatness and temporal logic games. The differential flatness makes it possible to construct finite abstraction in a straightforward way through discretization. The GR(1) temporal logic game approach makes it possible to solve for an expressive subclass of LTL specifications in polynomial time. The main limitation of the current work is that it relies on a uniform discretization and a uniform error specification/tolerance. In the future work, nonuniform and adaptive discretization and nonuniform error specifications (cf. [28]) can be considered to reduce the problem size.

References

[1] C. Belta, V. Isler, and G. Pappas, "Discrete abstractions for robot motion planning and control in polygonal environments," *IEEE Transactions on Robotics*, vol. 21, pp. 864–874, 2005.

[2] G. Fainekos, A. Girard, H. Kress-Gazit, and G. Pappas, "Temporal logic motion planning for dynamic robots," *Automatica*, vol. 45, pp. 343–352, 2009.

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

[4] H. Kress-Gazit, G. Fainekos, and G. Pappas, "Temporal-logic-based reactive mission and motion planning," *IEEE Transactions on Robotics*, vol. 25, pp. 1370–1381, 2009.

[5] H. Kress-Gazit, T. Wongpiromsarn, and U. Topcu, "Correct, reactive, high-level robot control," *IEEE Robotics & Automation Magazine*, vol. 18, pp. 65–74, 2011.

[6] P. Tabuada and G. Pappas, "Linear time logic control of discrete-time linear systems," *IEEE Transactions on Automatic Control*, vol. 51, pp. 1862–1877, 2006.

[7] T. Wongpiromsarn, U. Topcu, and R. Murray, "Receding horizon temporal logic planning for dynamical systems," in *Proceedings of the IEEE Conference on Decision and Control*, 2009, pp. 5997–6004.

[8] A. Cimatti, E. Clarke, E. Giunchiglia, F. Giunchiglia, M. Pistore, M. Roveri, R. Sebastiani, and A. Tacchella, "Nusmv 2: An opensource tool for symbolic model checking," in *Computer Aided Verification*. Springer, 2002, pp. 241–268.

[9] A. Pnueli, Y. Saar, and L. Zuck, "JTLV: A framework for developing verification algorithms," in *Computer Aided Verification*. Springer, 2010, pp. 171–174.

[10] M. Fliess, J. Lévine, P. Martin, and P. Rouchon, "Flatness and defect of non-linear systems: introductory theory and examples," *International Journal of Control*, vol. 61, pp. 1327–1361, 1995.

[11] M. Van Nieuwstadt and R. Murray, "Real-time trajectory generation for differentially flat systems," *International Journal of Robust and Nonlinear Control*, vol. 8, pp. 995–1020, 1998.

[12] A. Colombo and D. Del Vecchio, "Supervisory control of differentially flat systems based on abstraction," in *IEEE Conference on Decision and Control*, 2011.

[13] J. Lévine, *Analysis and control of nonlinear systems: A flatness-based approach*. Springer-Verlag, 2009.

[14] M. Van Nieuwstadt, M. Rathinam, and R. Murray, "Differential flatness and absolute equivalence of nonlinear control systems," *SIAM Journal on Control and Optimization*, vol. 36, pp. 1225–1239, 1998.

[15] A. Pnueli, "The temporal logic of programs," in *Proceedings of the Annual Symposium on Foundations of Computer Science*, 1977, pp. 46–57.

[16] Z. Manna and A. Pnueli, *The Temporal Logic of Reactive and Concurrent Systems: Specification*. Springer, 1992.

[17] E. Clarke, O. Grumberg, and D. Peled, *Model Checking*. MIT Press, 2000.

[18] N. Piterman, A. Pnueli, and Y. Saar, "Synthesis of reactive (1) designs," in *Proceedings of the International Conference on Verification, Model Checking, and Abstract Interpretation*, 2006, pp. 364–380.

[19] R. Bloem, B. Jobstmann, N. Piterman, A. Pnueli, and Y. Saar, "Synthesis of reactive (1) designs," *Journal of Computer and System Sciences*, vol. 78, pp. 911–938, 2012.

[20] N. Ozay, U. Topcu, and R. Murray, "Distributed power allocation for vehicle management systems," in *Proceedings of the IEEE Conference on Decision and Control*, 2011.

[21] N. Ozay, U. Topcu, T. Wongpiromsarn, and R. Murray, "Distributed synthesis of control protocols for smart camera networks," in *Proceedings of the ACM/IEEE International Conference on Cyber-Physical Systems*, 2011.

[22] T. Wongpiromsarn, U. Topcu, and R. Murray, "Receding horizon temporal logic planning," *IEEE Transactions on Automatic Control, to appear*, 2012.

[23] J. Liu, N. Ozay, U. Topcu, and R. Murray, "Switching protocol synthesis for temporal logic specifications," in *Proceedings of the American Control Conference*, 2012.

[24] L. Habets, P. Collins, and J. van Schuppen, "Reachability and control synthesis for piecewise-affine hybrid systems on simplices," *IEEE Transactions on Automatic Control*, vol. 51, pp. 938–948, 2006.

[25] B. Roszak and M. Broucke, "Necessary and sufficient conditions for reachability on a simplex," *Automatica*, vol. 42, pp. 1913–1918, 2006.

[26] A. Girard and S. Martin, "Synthesis for constrained nonlinear systems using hybridization and robust controllers on simplices," *IEEE Transactions on Automatic Control, to appear*, 2012.

[27] G. Reiszig, "Computing abstractions of nonlinear systems," *IEEE Transactions on Automatic Control, to appear*, 2012.

[28] Y. Tazaki and J. Imura, "Discrete abstractions of nonlinear systems based on error propagation analysis," *IEEE Transactions on Automatic Control*, vol. 57, pp. 550–564, 2012.

[29] R. Milner, *Communication and Concurrency*. Prentice-Hall, Inc., 1989.

[30] R. Alur, T. Henzinger, G. Lafferriere, and G. Pappas, "Discrete abstractions of hybrid systems," *Proceedings of the IEEE*, vol. 88, pp. 971–984, 2000.

[31] A. Girard and G. Pappas, "Approximation metrics for discrete and continuous systems," *IEEE Transactions on Automatic Control*, vol. 52, pp. 782–798, 2007.

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

[33] A. Girard, G. Pola, and P. Tabuada, "Approximately bisimilar symbolic models for incrementally stable switched systems," *IEEE Transactions on Automatic Control*, vol. 55, pp. 116–126, 2010.

[34] G. Pola, A. Girard, and P. Tabuada, "Approximately bisimilar symbolic models for nonlinear control systems," *Automatica*, vol. 44, pp. 2508–2516, 2008.

[35] G. Pola and P. Tabuada, "Symbolic models for nonlinear control systems: Alternating approximate bisimulations," *SIAM Journal on Control and Optimization*, vol. 48, pp. 719–733, 2009.

[36] M. Zamani, G. Pola, M. Mazo Jr, and P. Tabuada, "Symbolic models for nonlinear control systems without stability assumptions," *IEEE Transactions on Automatic Control, to appear*, 2012.