Switching Control of Dynamical Systems from Metric Temporal Logic Specifications

Jun Liu and Pavithra Prabhakar

***Abstract*— Motivated by designing high-level planners for dynamical systems (such as mobile robots) to achieve complex tasks, we consider the synthesis of switching controllers of nonlinear dynamical systems from metric temporal logic (MTL) specifications. MTL is a popular logic that allows to specify timed properties of real-time reactive systems and hence is appropriate for describing the safe and autonomous operations of robotic systems in an uncertain and possibly adversarial environment. We provide constructive means for computing finite-state abstractions that preserve MTL properties for nonlinear systems, under a weak assumption that these nonlinear systems evolve continuously with respect to their initial conditions. We then provide conditions to ensure that the existence of a discrete strategy (obtained by solving a discrete synthesis problem) guarantees the existence of a switching strategy for controlling the continuous-time dynamical systems to satisfy a given MTL specification. We illustrate the results on a motion planning problem.**

I. Introduction

The application of formal methods to robot motion planning has gained prominence in recent years (see, e.g., [3]–[5], [7], [10], [18], [23], [34]). In particular, controller synthesis algorithms from the formal methods area have been utilized to design correct-by-construction planning algorithms [19]. This is particularly useful since it can potentially avoid the extensive testing and simulations involved in the development of embedded control software for robotic systems.

Correct-by-construction synthesis requires as inputs a formal model of the plant (or system), some model of the environment in which the plant operates, and a formal specification of the property the system is required to achieve. The property is formally specified typically in some logic such as linear-time temporal logic (LTL), computation tree logic (CTL) or μ-calculus. Since the region being planned typically has an infinite-state space, state-space reduction algorithms are applied to construct a finite-state abstract model of the system which preserves the controllers of the original system. Such abstractions guarantee that, if a control strategy is synthesized on the finite-state model, a strategy for the original model can be extracted from it. The success of this approach depends heavily on efficient algorithms for computing the finite-state abstractions. For this reason, considerable work has been carried out on abstracting dynamical systems, linear or nonlinear, to finite-state models for controller synthesis (see, e.g., [14]–[16], [20]–[22], [27], [30], [35]).

Jun Liu is with the Department of Automatic Control and Systems Engineering, University of Sheffield, Sheffield S1 3JD, UK (e-mail: j.liu@sheffield.ac.uk). Pavithra Prabhakar is with IMDEA Software Institute, Madrid, Spain (e-mail: pavithra.prabhakar@imdea.org).

Most of the previous work on robot motion planning consider temporal properties which essentially treat time in a qualitative manner (except [15]). However, quantitative timing is essential to achieve hard real-time tasks. For example, a robot might be required to deliver an item within a certain time bound after picking it up, rather than at some arbitrary time in the future. Hence, we consider a quantitative logic called Metric Temporal Logic (MTL) to specify real-time constraints. MTL [17] is an extension of the classical temporal logic which allows time intervals to be specified in the until operator.

We consider the problem of synthesizing switching controllers to achieve MTL specifications. The problem of switching control naturally arises in many applications and in particular robot motion planning. For instance, different modes of a switched system may correspond to the motion of a robot under different mode [32], so-called robot motion primitives [12], or bipedal robot locomotion [28]. 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 controller is to provide a close-loop controller that renders the system satisfy the mission-level specification.

The synthesis problem for MTL is well-studied when the system is specified as a timed automaton [6] (See Section IV-G for more details). We exploit these results to provide algorithms for synthesizing controller for MTL specifications by abstracting nonlinear systems into finite automata interpreted as timed systems. In particular, we provide constructive means for computing finite-state abstractions that preserve MTL properties for nonlinear systems, under a weak assumption that these nonlinear systems evolve continuously with respect to their initial conditions. We then provide a sufficient condition to guarantee the correctness of continuous-time trajectories in terms of the sampling period, the rate of change of the dynamics, and a robustness margin that is needed to account for the possible mismatches between a sampled-data system and a continuous-time system.

II. Preliminaries

Notation: $\mathbb{R}^n$ denotes the n-dimensional Euclidean space with its norm denoted by $|x|$; $\mathbb{R}^+$ is the nonnegative real line; $\mathbb{Z}^n$ is the n-dimensional integer lattice; given $\delta \geq 0$ and $x \in \mathbb{R}^n$, $B_\delta(x) := \{x' \in \mathbb{R}^n : |x' - x| \leq \delta\}$.

A. Metric temporal logic

Metric temporal logic (MTL) is a popular logic for specifying timed properties of real-time reactive systems [17]. It extends the *until* operator of the linear-time temporal logic (LTL) with a time interval. The syntax of MTL over a set of atomic propositions Π is defined inductively as:

$$\varphi := \pi \,|\, \neg\varphi \,|\, \varphi \vee \varphi \,|\, \varphi \mathcal{U}_I \varphi,$$

where $\pi \in \Pi$ and $I \subseteq \mathbb{R}^+$ is an interval whose finite endpoints are integers. Atomic propositions are statements on an observation space Y. A labelling function $L : Y \to 2^\Pi$ maps an observation to a set of propositions that hold true. Metric temporal logic formulas are interpreted over observed signals. A *signal* is a function $\xi : \mathbb{R}^+ \to Y$.

Negation normal form (NNF): All MTL formulas can be transformed into negation normal form, where

- all negations appear only in front of the atomic propositions;
- only other logical operators `true`, `false`, $\wedge$, and $\vee$ can appear; and
- only the temporal operators $\mathcal{U}_I$, and $\mathcal{R}_I$ can appear, where $\mathcal{R}_I$ is the *dual until* operator defined by $\varphi_1 \mathcal{R}_I \varphi_2 \equiv \neg(\neg\varphi_1 \mathcal{U}_I \neg\varphi_2)$.

In this paper, we assume that all MTL formulas are transformed into NNF.

Continuous semantics of MTL: Given a signal ξ, we define $\xi, t \vDash \varphi$ with respect to an MTL formula φ at time t inductively as follows:

- $\xi, t \vDash \pi$ if and only if $\pi \in L(\xi(t))$;
- $\xi, t \vDash \varphi_1 \vee \varphi_2$ if and only if $\xi, t \vDash \varphi_1$ or $\xi, t \vDash \varphi_2$;
- $\xi, t \vDash \varphi_1 \wedge \varphi_2$ if and only if $\xi, t \vDash \varphi_1$ and $\xi, t \vDash \varphi_2$;
- $\xi, t \vDash \varphi_1 \mathcal{U}_I \varphi_2$ if and only if there exists $t' \in I$ such that $\xi, t+t' \vDash \varphi_2$ and for all $t'' \in [0, t')$, $\xi, t+t'' \vDash \varphi_1$;
- $\xi, t \vDash \varphi_1 \mathcal{R}_I \varphi_2$ if and only if for all $t' \in I$ either $\xi, t+t' \vDash \varphi_2$ or there exists $t'' \in [0, t')$ such that $\xi, t+t'' \vDash \varphi_1$.

We write $\xi \vDash \varphi$ if $\xi, 0 \vDash \varphi$. Here, we have chosen the so-called non-strict and non-matching semantics of MTL (for a comparison between different variants of MTL semantics and their expressiveness, see [13], [31]).

B. Transition systems

A *transition system* is a tuple $\mathcal{T} = (Q, Q_0, \mathcal{A}, \to_\mathcal{T}, Y, h, \Pi, L)$, where:

- Q is a (finite or infinite) set of states;
- $\mathcal{A}$ is a (finite or infinite) set of actions;
- $\to_\mathcal{T} \subseteq Q \times \mathcal{A} \times Q$ is a transition relation;
- Y is a (finite or infinite) set of observations;
- $h : Q \to Y$ is an observation map on the states;
- Π is a set of atomic propositions on Y;
- $L : Y \to 2^\Pi$ is a labelling function.

We often write $q \xrightarrow{a} q'$ to indicate $(q, a, q') \in \to$. $\mathcal{T}$ is said to be (1) *finite* if the cardinality of Q and $\mathcal{A}$ are finite, (2) *infinite* otherwise, and (3) *metric* if Y is equipped with a metric. We assume, without loss of generality, that all actions are enabled at every state, that is, for all $q \in Q$ and for all $a \in \mathcal{A}$, there exists at least one $q' \in Q$ such that $q \xrightarrow{a} q'$.

An *execution* of a transition system $\mathcal{T}$ 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 $(q_i, a_i, q_{i+1}) \in \to_\mathcal{T}$ for all $i \geq 0$. A *control strategy* for a transition system $\mathcal{T}$ is a partial function $s : (q_0, a_0, \cdots, q_i) \mapsto a_i$ 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. A *timed* (controlled) execution of a transition system $\mathcal{T}$ is an execution of $\mathcal{T}$ tagged with a strictly increasing sequence of times $\{\tau_i\}_{i=0}^\infty$, which can be written as a sequence of tuples $\rho = (q_0, a_0, \tau_0)(q_1, a_1, \tau_1)(q_2, a_2, \tau_2)\cdots$. This timing sequence can be inherently determined by the sequence of actions (as is the case in this paper).

Given a set of atomic propositions Π on Y, an MTL formula over Π can be interpreted over timed executions of a transition system $\mathcal{T} = (Q, Q_0, \mathcal{A}, \to_\mathcal{T}, Y, h, \Pi, L)$.

Discrete semantics of MTL: Given a timed execution $\rho = (q_0, a_0, \tau_0)(q_1, a_1, \tau_1)(q_2, a_2, \tau_2)\cdots$ of $\mathcal{T}$, we define $\rho, i \vDash \varphi$ with respect to an MTL formula φ inductively as follows:

- $\rho, i \vDash \pi$ if and only if $\pi \in L(h(q_i))$;
- $\rho, i \vDash \varphi_1 \vee \varphi_2$ if and only if $\rho, i \vDash \varphi_1$ or $\rho, i \vDash \varphi_2$;
- $\rho, i \vDash \varphi_1 \wedge \varphi_2$ if and only if $\rho, i \vDash \varphi_1$ and $\rho, i \vDash \varphi_2$;
- $\rho, i \vDash \varphi_1 \mathcal{U}_I \varphi_2$ if and only if there exists $j > i$ such that $\rho, j \vDash \varphi_2$, $\tau_j - \tau_i \in I$, and $\rho, k \vDash \varphi_1$ for all $k \in [i, j)$;
- $\rho, i \vDash \varphi_1 \mathcal{R}_I \varphi_2$ if and only if, for all $j > i$ such that $\tau_j - \tau_i \in I$, either $\rho, j \vDash \varphi_2$ or there exists $k \in [i, j)$ such that $\rho, k \vDash \varphi_1$.

Similarly, we write $\rho \vDash \varphi$ if $\rho, 0 \vDash \varphi$ and say ρ satisfies φ. If there exists a control strategy for $\mathcal{T}$ such that all controlled executions of $\mathcal{T}$ satisfy φ, we say φ is *realizable* on $\mathcal{T}$.

III. Problem Formulation

Consider a family of nonlinear systems,

$$\dot{x}(t) = f_p(x(t)), \quad p \in \mathcal{P}, \quad t \geq 0, \tag{1}$$

where $x(t) \in X \subseteq \mathbb{R}^n$ is the state at time t and $\mathcal{P}$ is a finite index set, and $\{f_p : p \in \mathcal{P}\}$ is a family of nonlinear vector fields satisfying the usual conditions to guarantee the existence and uniqueness of solutions for each of the subsystems in (1). A *switched system* generated by the family (1) can be written as

$$\dot{x}(t) = f_{\sigma(t)}(x(t)), \quad t \geq 0, \tag{2}$$

where $\sigma : [0, \infty) \to \mathcal{P}$ is a *switching signal* taking values in $\mathcal{P}$. Given any initial state $x(0) \in X_0 \subseteq X$, a *solution* to (2) is the trajectory $(x, \sigma) := \{(x(t), \sigma(t))\}_{t \geq 0}$, where the pair $(x(t), \sigma(t))$ satisfies (2) for all $t \geq 0$.

The objective of this paper is to design switching strategies that can generate correct switching signals. At a high level, the switching controller should sense the system state $x(t)$ in real time and decide accordingly on which mode of (1) to activate by choosing $\sigma(t)$. In practice, feedback control laws are mainly implemented in digital platforms. As such, they only sample the system state and update the control signals at discrete times. In this paper, we will adopt this paradigm and consider digital implementations of switching controllers that realize high-level specifications written in metric temporal logic specifications.

We assume that state measurements are taken at *sampling times* $t_k := k\tau_s$, $k = 0, 1, 2, \cdots$, where $\tau_s > 0$ is a fixed sampling period. We define a *switching strategy* to be a partial function $\sigma(x(t_0), \cdots, x(t_k)) = p_k \in \mathcal{P}$, $\forall k = 0, 1, 2, \cdots$ that maps a sequence of state measurements to a sequence of switching modes. Between sampling times, the switching signal is kept constant. In other words, a switching strategy effectively generates a state-feedback switching signal $\sigma(t) = p_k,\ t \in [t_k, t_{k+1})$. This signal, in turn, leads to a solution pair (x, σ) that satisfies (2).

To state the problem, we define the observation space $Y = X$[1].

Problem 1: [**Continuous Switching Synthesis Problem**] Given a family of continuous-time subsystems in (1) and an MTL specification φ, synthesize a switching strategy σ such that the solution (x, σ) satisfies $x \vDash \varphi$.

IV. Main Results

A. Switched systems as transition systems

Switched systems can be formulated as transition systems. Given a family of nonlinear systems (1), we define a transition system $\mathcal{T}_s = (Q, Q_0, \mathcal{A}, \to_{\mathcal{T}_s}, Y, h, \Pi, L)$ by

- $Q = X$ and $Q_0 = X_0$;
- $\mathcal{A} = \mathcal{P} \times \mathbb{R}^+$;
- $(q, (p, \tau), q') \in \to_{\mathcal{T}_s}$ if and only if there exists $\xi : [0, \tau] \to \mathbb{R}^n$ such that $\xi(0) = q$, $\xi(\tau) = q'$, and $\dot{\xi}(s) = f_p(\xi(s))$ for all $s \in [0, \tau]$;
- $Y = X$;
- $h : Q \to Y$ is defined by $\mathbf{id}_Q$, i.e., the identity map on Q;
- Π is a set of atomic propositions on Y;
- $L : Y \to 2^{\Pi}$ is a labelling function.

Each action in $\mathcal{A}$ consists of a mode $p \in \mathcal{P}$ and a duration of maintaining this mode. As discussed earlier, if we are interested in digital implementations of switched systems, the set of actions can be reduced to $\mathcal{A} = \mathcal{P} \times \{k\tau_s : k = 0, 1, 2, \cdots\}$.

B. Finite abstractions of switched systems

The transition system $\mathcal{T}_s$ constructed above is infinite, with infinitely many states and actions. Under the assumption that the set X is compact, we can construct finite abstractions of (1) as follows. For a given parameter $\eta > 0$, define

$$[X]_\eta := \eta\mathbb{Z}^n \cap X, [X_0]_\eta := \eta\mathbb{Z}^n \cap X_0,$$

where $\eta\mathbb{Z}^n := \{\eta z : z \in \mathbb{Z}^n\}$. Clearly, $[X]_\eta$ and $[X_0]_\eta$ are finite sets. Instead of looking at an infinite set of actions, we define $\mathcal{A}_K = \mathcal{P} \times \{k\tau_s : k \in K\}$, where K is a finite subset of nonnegative integers. For example, we can choose $K = \{2^s : s = 0, \cdots N\}$ for some integer $N \geq 0$.

Definition 1: Given a family of nonlinear systems (1), a finite set of nonnegative integers K, and positive constants η and δ, we define a finite transition system $\hat{\mathcal{T}}_s = (\hat{Q}, \hat{Q}_0, \hat{\mathcal{A}}, \to_{\hat{\mathcal{T}}_s}, \hat{Y}, \hat{h}, \hat{\Pi}, \hat{L})$ by

- $\hat{Q} = [X]_\eta$ and $Q_0 = [X_0]_\eta$;
- $\hat{\mathcal{A}} = \mathcal{A}_K = \mathcal{P} \times \{k\tau_s : k \in K\}$;
- $(q, (p, \tau), q') \in \to_{\hat{\mathcal{T}}_s}$ if there exists $\xi : [0, \tau] \to \mathbb{R}^n$ such that $|\xi(0) - q| \leq \eta/2$, $|\xi(\tau) - q'| \leq \eta/2$, and $\dot{\xi}(s) = f_p(\xi(s))$ for all $s \in [0, \tau]$;
- $\hat{Y} = [X]_\eta$ and $\hat{h} = h$ as defined in $\mathcal{T}_s$;
- $\hat{\Pi} = \Pi$;
- $\hat{L} : \hat{Y} \to 2^{\hat{\Pi}}$ is defined by

$$\pi \in \hat{L}(y),\ y \in \hat{Y} \Longleftrightarrow \pi \in L(y'),\ \forall y' \in B_\delta(y), \quad (3)$$

where $B_\delta(y) := \{y' \in Y : |y' - y| \leq \delta\}$ is the closed δ-ball centred at y. We call $\hat{\mathcal{T}}_s$ an (η, δ, K)-abstraction of $\mathcal{T}_s$ and write $\mathcal{T}_s \preceq_{(\eta,\delta,K)} \hat{\mathcal{T}}_s$.

Remark 1: The abstraction relation defined above can be seen as an over-approximation of the system dynamics with discretization granularity η, an under-approximation on the control actions in the sense that these are restricted to a subset of all available actions, and provides a robust margin δ which plays an important role in preserving the correctness of executions. It essentially defines an alternating simulation from $\mathcal{T}_s$ to $\hat{\mathcal{T}}_s$ [2] that takes into account approximation errors (cf. [30]) and provides robustness margins to accommodate these errors. This approach has the potential to deal with imperfect measurements in system state, such as those caused by quantization, time delays, or measurement errors, by incorporating additional robustness margins as recently shown in [20].

C. Computation of abstractions

The only part in $\hat{\mathcal{T}}_s$ that needs to be computed is the transition relations. We do this by simulating a trajectory starting from each of the grid point in $\hat{Q}$ and estimating the state evolution under the dynamics in (1). We rely on the following condition:

$$|x(t; p, \zeta_0) - x(t; p, \zeta_1)| \leq \beta_p(|\zeta_0 - \zeta_1|, t), \quad (4)$$

where $x(t; p, \zeta_0)$ and $x(t; p, \zeta_1)$ denote solutions of $\dot{x} = f_p(x)$ starting from ζ_0 and ζ_1, respectively, and each $\beta_p : \mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}^+$ is a continuous function. Such a condition is called incremental forward completeness in [35] when there is control input. Here, the above condition would naturally follow if each of the subsystem satisfies a continuous dependence condition on its initial conditions. An explicit form of β can usually be obtained using Lyapunov-type techniques.

Proposition 1: Given any tuple (η, δ, K), if $(q, (p, \tau), q') \in \to_{\hat{\mathcal{T}}_s}$ whenever $(q, (p, \tau), q') \in \hat{Q} \times \hat{\mathcal{A}} \times \hat{Q}$ and $|q' - x(\tau; p, q)| \leq \beta_p(\eta/2, \tau) + \eta/2$, then $\mathcal{T}_s \preceq_{(\eta,\delta,K)} \hat{\mathcal{T}}_s$.

D. Discrete synthesis problem

Problem 2: [**Discrete Synthesis Problem**] Given the finite transition system $\hat{\mathcal{T}}$ and an MTL specification $\hat{\varphi}$, synthesize a control strategy s such that all s-controlled executions of $\hat{\mathcal{T}}$ satisfy $\hat{\varphi}$.

[1]It is possible to define the observation space $Y = X \times \mathcal{P}$ to allow specifying system properties related to mode.

E. Transformation of MTL specifications

In general, the existence of a discrete control strategy for Problem 2 with an MTL formula φ *does not* guarantee the existence of a switching strategy σ that solves Problem 1 for the same specification φ. There are two reasons for this. First, we are using discretization-based (or point-based), rather than proposition-preserving partition-based, abstractions. We need extra conditions to ensure correctness of continuous executions from discrete reasoning. This is an issue even for linear temporal logic (LTL) specifications and motivates (3) in the definition of an abstraction in Definition 1, which essentially captures the idea of contracting and expanding atomic propositions as used in [10], [22]. Moreover, the difference between the continuous and discrete semantics of MTL indicates that, when solving Problem 2, we need to synthesize for a different and stronger version of a given MTL formula φ (transformed from φ and denoted by $\hat{\varphi}$) in order to guarantee a solution to the original continuous synthesis Problem 1 under the specification φ. The aim of the next section is to outline an approach to making this transformation (similar ideas were presented in [11]).

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 MTL formula φ in NNF, 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 MTL formulas on Π can be seen as an MTL formula built upon $\hat{\Pi}$ without the negation operator, only that now we need both the logical operators $\vee$ and $\wedge$ and the timed release operator $\mathcal{R}_I$ to derive the full set of MTL formulas in NNF (denoted by MTL).

Given $\Delta \geq 0$, we define an operator $\Re_\Delta$ that transforms MTL formulas by recursively modifying the temporal operators $\mathcal{U}_I$ and $\mathcal{R}_I$ as follows:

$$\Re_\Delta(\varphi_1 \mathcal{U}_I \psi_2) = \Re_\Delta(\varphi_1)\mathcal{U}_{c_\Delta(I)}\Re_\Delta(\varphi_2).$$

and

$$\Re_\Delta(\varphi_1 \mathcal{R}_I \psi_2) = \Re_\Delta(\varphi_1)\mathcal{R}_{e_\Delta(I)}\Re_\Delta(\varphi_2),$$

where

$$e_\Delta(I) = \{t \geq 0 : \exists s \in I \text{ and } s' \in [-\Delta, \Delta] \text{ s.t. } t = s + s'\},$$

and

$$c_\Delta(I) = \{t \geq 0 : t + s \in I, \forall s \in [-\Delta, 0]\}.$$

Intuitively, we expand the interval constraint in $\mathcal{R}_I$ and shrink that in $\mathcal{U}_I$.

F. Proof of correctness

Theorem 1: If $\hat{\mathcal{T}}_s$ is an (η, δ, K)-abstraction of $\mathcal{T}_s$ satisfying $\delta \geq \eta/2 + M\Delta$, where $M = \sup_{x\in X, p\in\mathcal{P}} |f(x)|$ and $\Delta = \sup\{k\tau_s : k \in K\}$, then, given any MTL formula φ, $\hat{\varphi} = \Re_\Delta(\varphi)$ being realizable for $\hat{\mathcal{T}}_s$ implies that φ is realizable for $\mathcal{T}_s$.

Proof: By the definition of an over-approximation, to every control strategy $\hat{f}$ for $\hat{\mathcal{T}}_s$, there corresponds a control strategy f for $\mathcal{T}_s$ such that, to each f-controlled execution of $\mathcal{T}$, there corresponds a $\hat{f}$-controlled execution in $\mathcal{T}$. We denote this correspondence by ρ to $\hat{\rho}$, where $\rho = (q_0, a_0, \tau_0)(q_1, a_1, \tau_1)(q_2, a_2, \tau_2)\cdots$, $\hat{\rho} = (\hat{q}_0, \hat{a}_0, \tau_0)(\hat{q}_1, \hat{a}_1, \tau_1)(\hat{q}_2, \hat{a}_2, \tau_2)\cdots$, and $|q_i - \hat{q}_i| \leq \eta/2$ for all $i \geq 0$. Furthermore, corresponding to ρ, there is the solution (x, σ) with $(x(\tau_i), \sigma(\tau_i)) = (q_i, a_i)$ for all $i \geq 0$.

We have to show that $\hat{\rho} \vDash \hat{\varphi}$ implies $x \vDash \varphi$, where $\hat{\varphi}$ is given by $\hat{\varphi} = \Re_\Delta(\varphi)$. We prove this by proving a stronger statement: $\hat{\rho}, i \vDash \hat{\varphi}$ for $i \geq 0$ implies $x, t \vDash \varphi$ for all $t \in J_i = [\tau_i, \tau_i + \Delta]$.

The proof is by induction on the structure of formulas in MTL.

Case $\varphi = \pi$: To show $x, t \vDash \pi$ for all $t \in J_i$, we have to show that $\pi \in L(x(t))$. This follows from $q_i = x(\tau_i)$, $\pi \in \hat{L}(\hat{q}_i)$, and

$$|x(t) - \hat{q}_i| \leq |x(t) - x(\tau_i)| + |q_i - \hat{q}_i| \leq \eta/2 + M\Delta \leq \delta.$$

Case $\varphi = \varphi_1 \mathcal{U}_I \varphi_2$: To show $x, t \vDash \varphi$ for all $t \in J_i$, we need to show that, for each fixed $t \in J_i$, there exists $t' \in I$ such that $x, t + t' \vDash \varphi_2$ and for all $t'' \in [0, t')$, $x, t + t'' \vDash \varphi_1$. We have $\hat{\rho}, i \vDash \hat{\varphi}$; that is, there exists $j > i$ such that $\hat{\rho}, j \vDash \hat{\varphi}_2$, $\tau_j - \tau_i \in c_\Delta(I)$, and $\hat{\rho}, k \vDash \hat{\varphi}_1$ for all $k \in [i, j)$. Note that we do need the non-strict condition to hold, i.e., $\hat{\rho}, i \vDash \hat{\varphi}_1$. It follows from the inductive assumption that $x, s \vDash \varphi_2$ for all $s \in J_j$ and $x, s \vDash \varphi_1$ for all $s \in J_k$ and all $k \in [i, j)$. Take $t' = \tau_j - t$. Since $\tau_j - \tau_i \in c_\Delta(I)$, we have $t' = (\tau_j - \tau_i) + (\tau_i - t) \in I$. Then $t + t' = \tau_j \in J_j$ and hence $x, t + t' \vDash \varphi_2$. In addition, for all $t'' \in (0, t')$, we have $t + t'' \in J_k$ for some $k \in [i, j)$ and hence $x, t + t'' \vDash \varphi_1$. In fact, $\cup_{i \leq k \leq j-1} J_k = [\tau_i, \tau_{j-1} + \Delta] \supseteq [t, \tau_j) = [t, t + t') \ni t + t''$.

Case $\varphi = \varphi_1 \mathcal{R}_I \varphi_2$: To show $x(t) \vDash \varphi$ for all $t \in J_i$, we need to show that, for each fixed $t \in J_i$, we have, for all $t' \in I$ either $x, t + t' \vDash \varphi_2$ or that there exists $t'' \in [0, t')$ such that $x, t + t'' \vDash \varphi_1$. We have $\hat{\rho}, i \vDash \hat{\varphi}$; that is, for all $j > i$ such that $\tau_j - \tau_i \in e_\Delta(I)$, either $\hat{\rho}, j \vDash \hat{\varphi}_2$ or there exists $k \in [i, j)$ such that $\hat{\rho}, k \vDash \hat{\varphi}_1$. If there exists τ_j such that $\tau_i + t' \leq \tau_j \leq t + t' \leq \tau_i + \Delta + t' \leq \tau_j + \Delta$, then $\tau_j - \tau_i \in (I + [0, \Delta]) \cap \mathbb{R}^+ \subseteq e_\Delta(I)$ and $t + t' \in J_j$. Otherwise, there exists τ_j such that $\tau_j \leq \tau_i + t' \leq t + t' \leq \tau_{j+1} \leq \tau_j + \Delta$. Then $\tau_j - \tau_i \in (I + [-\Delta, 0]) \cap \mathbb{R}^+ \subseteq e_\Delta(I)$ and $t + t' \in J_j$. In both cases, we have either $\hat{\rho}, j \vDash \hat{\varphi}_2$ or that there exists $k \in [i, j)$ such that $\hat{\rho}, k \vDash \hat{\varphi}_1$. It follows that either $x, s \vDash \varphi_2$ for all $s \in J_j$ or there exists $k \in [i, j)$ such that $x, s \vDash \varphi_1$ for all $s \in J_k$. If the former holds, since $t + t' \in J_j$, we get $x, t + t' \vDash \varphi_2$. If the latter holds, since $t + t' \geq \tau_j > \tau_k$ and $\tau_k + \Delta \geq t$, we know $[t, t + t') \cap J_k \neq \emptyset$. Thus, there exists $t'' \in [0, t')$ such that $x, t + t'' \vDash \varphi_1$.

The other cases are straightforward. ■

G. Discrete synthesis

We present two methods to solve the discrete synthesis problem for MTL using the available algorithms in the literature.

a) Formulate as a controller synthesis problem over timed automata with respect to MTL specifications: Transform the finite state system $\hat{\mathcal{T}}_s$ into a timed automaton $\mathcal{T}_A$ such that the state transitions happen at time units of τ_s. Hence, the timed executions of $\hat{\mathcal{T}}_s$ and $\mathcal{T}_A$ are the same. Solve a controller synthesis problem on $\mathcal{T}_A$ with respect to the MTL formula $\hat{\varphi}$.

Controller synthesis problem for timed automaton with respect to MTL has been investigated extensively. In general, the problem is undecidable [6]. However, it is decidable for several fragments of MTL. These depend on translations of the MTL formula into a timed language equivalent timed automaton and solving the controller synthesis problem with respect to a timed automaton specification. When the timed specification is provided as a non-deterministic timed automaton, the controller synthesis problem is undecidable in general; but is decidable when the timed automaton is deterministic or when the available resources are constrained [9]. The sublogic of MTL [1], [25] in which the interval is constrained to be non-singular call MITL (Metric Interval Temporal Logic) can be translated to timed language equivalent non-deterministic timed automata. Though the controller synthesis problem for MITL with unconstrained resources is undecidable [8], the past fragment of MITL can be translated into deterministic timed automata [24], and hence decidable. The full class of MTL can be translated to the deterministic timed automata [26] when the formulas have bounded variability (or non-Zeno).

b) Formulate as a controller synthesis problem over finite state systems with respect to LTL specifications: In this approach, we translate an MTL formula φ into an LTL formula *mtl-ltl*(φ) such that they satisfy the same set of timed executions. The translation is defined inductively as follows: *mtl-ltl*$(\pi) = \pi$, *mtl-ltl*$(\neg\varphi) = \neg$ *mtl-ltl*(φ), *mtl-ltl*$(\varphi_1 \vee \varphi_2) =$ *mtl-ltl*$(\varphi_1) \vee$ *mtl-ltl*(φ_2) and *mtl-ltl*$(\varphi_1 U_I \varphi_2)$ is:

- *mtl-ltl*(φ_2), if $I = [0, 0]$,
- *mtl-ltl*$(\varphi_1) U$*mtl-ltl*(φ_2), if $I = [0, \infty)$,
- *mtl-ltl*$(\varphi_1) \wedge X$*mtl-ltl*$(\varphi_1 U_{\hat{I}} \varphi_2)$ if $0 \notin I$, and
- *mtl-ltl*$(\varphi_2) \vee ($*mtl-ltl*$(\varphi_1) \wedge X$*mtl-ltl*$(\varphi_1 U_{\hat{I}} \varphi_2))$, otherwise

where $\hat{I} = \{i \geq 0 \,|\, i + 1 \in I\}$ and the next operator X is given by $X\varphi = \mathit{True} U_{[1,1]}\varphi$.

While games with complete LTL specifications are known to have prohibitively high computational complexity, for certain fragments of LTL, efficient tools exist for solving the resulting discrete synthesis problem, such as JTLV [29] or more recently developed RaTL toolbox [33]. These fragments can capture most useful motion planning tasks (e.g., safe navigation with collision avoidance, response to certain events, search and rescue, coverage and surveillance).

V. An Example

We illustrate our results on the motion planning of a differential wheeled robot whose motion is described by $\dot{x} = v\cos(\theta)$ and $\dot{y} = v\sin(\theta)$, where (x, y) is the coordinate of the robot, θ is the heading angle, and v is the speed. We can cast this into a switching control problem by restricting the v and θ to a finite set of parameters. Here, we let $v = 2$ and $\theta \in \{0, \pm\pi/2, \pi\}$. Consider a planar grid of size $[0, 11] \times [0, 11]$ as shown in Figure 1. The task of the robot is to pick up an object from upper left corner P and drop it off to the lower left corner D, while avoiding collision with a dynamic obstacle B that can move within the shaded region according to the same dynamics as the robot, but with $v = 1$.

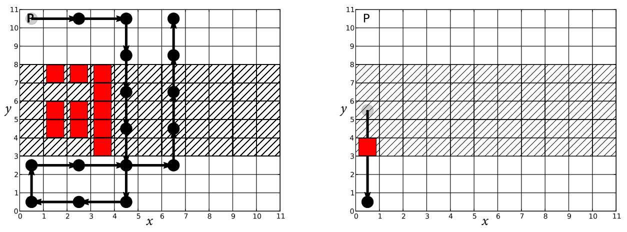

Fig. 1: Illustration of the planar environment for motion planning. Left: a representative trajectory generated by a planner synthesized using an abstraction under a fixed sampling period $\tau_s = 1$. Locations of the robot and dynamical obstacle are marked by black circles and red boxes, respectively. Right: an illustration of the inter-sample behavior of the robot colliding with the obstacle, under a planner synthesized using an abstraction with a fixed sampling period $\tau_s = 2.5$ but the same parameter $\delta = 2.5$. In view of Theorem 1, we need to to choose $\delta \geq 5.5$ to ensure correctness of the continuous trajectory of the robot. This, however, would render the discrete synthesis problem unrealizable.

It is clear that the right-hand side of the system is piecewise constant, yet the system is not incrementally stable, a condition that is often assumed for computing finite abstractions for nonlinear systems [14]. It is, however, easy to check that condition (4) is satisfied with $\beta(r, t) \equiv r$. We can compute an abstraction for this system in view of Proposition 1 with given parameters (η, δ, K). We choose $\eta = 1$ and $K = \{1\}$, i.e., a fixed sampling period of τ_s. The parameter δ should be chosen according to the condition $\delta \geq M\Delta + \eta/2$ in Theorem 1. For example, with $\tau_s = 1$ and hence $\Delta = 1$, Theorem 1 suggests that we need to choose $\delta \geq 2.5$ to ensure correctness of the continuous trajectories (note that $M = 2$).

The specification we would like to enforce is as follows: $\square(P \to \lozenge_{[0,N]} D) \wedge \square(D \to \lozenge_{[0,N]} P) \wedge \square(\neg B)$; that is, whenever the robot pick up an object, it should deliver it to the drop-off location within N unit of time and, vice versa, it should return to the pickup location within N unit of time after each drop-off. At the same time, the robot should avoid the dynamic obstacle. This can in fact be translated to an linear temporal logic (LTL) specification as outlined in Section IV-G. We solve the resulting LTL synthesis problem using the recently developed RaTL toolbox [33]. Figure 1 shows the simulation results of the motion planning (with $N = 22$). It clearly shows that it is necessary to compensate for longer sampling period or faster dynamics by choosing a larger δ in the abstraction.

VI. Conclusions

In this paper, we considered the problem of synthesizing switching controllers for nonlinear hybrid systems constrained by metric temporal logic specifications. These specifications are motivated by common planning tasks such as safe navigation with collision avoidance, response to certain events, search and rescue, coverage and surveillance. Previous work on temporal logic planning for dynamical systems mostly focused on qualitative temporal properties such as LTL properties. In many applications, however, hard real-time tasks are essential. This has motivated us to work on temporal logic control of switched systems for timed specifications.

The synthesized controllers guarantee that the trajectories of the system satisfy certain high-level specifications expressed in metric temporal logic. By means of constructing finite-state transition systems, the switching synthesis problem was lifted to a discrete domain. These abstractions preserve metric temporal logic specifications under an easily verifiable sufficient condition in terms of the rate of the system dynamics, the sampling period, and a parameter (δ) that was intended to account for mismatches between the finite-state abstractions and the continuous-time system.

An important aspect not covered in this work is to determine exactly (or a subset of) of metric temporal logic specifications that can be transformed into specifications expressible in linear temporal logic specifications or even fragments of LTL, for which efficient algorithms for solving the discrete synthesis problem exist. It would also be interesting to quantify the trade-offs between the gained expressiveness by incorporating timed specifications and the resulted increase in complexity for synthesis.

Acknowledgement

The authors would like to thank Necmiye Ozay for helpful discussions and Eric M. Wolff for providing his RaTL (Rabin Temporal Logic) toolbox [33] for solving the example in this paper. This work is supported in part by a Marie Curie Career Integration Grant PCIG13-GA-2013-617377 (to J.L.).

References

[1] R. Alur, T. Feder, and T. A. Henzinger. The benefits of relaxing punctuality. *J. ACM*, 43(1):116–146, 1996.

[2] R. Alur, T. A. Henzinger, O. Kupferman, and M. Y. Vardi. Alternating refinement relations. In *Proc. of CONCUR*, pages 163–178. Springer, 1998.

[3] C. Belta, A. Bicchi, M. Egerstedt, E. Frazzoli, E. Klavins, and G. J. Pappas. Symbolic planning and control of robot motion: Grand challenges of robotics. *IEEE Robotics & Automation Magazine*, 14(1):61–70, 2007.

[4] C. Belta, V. Isler, and G. J. Pappas. Discrete abstractions for robot motion planning and control in polygonal environments. *IEEE Trans. Robotics*, 21:864–874, 2005.

[5] A. Bhatia, L. E. Kavraki, and M. Y. Vardi. Sampling-based motion planning with temporal goals. In *Proc. of ICRA*, pages 2689–2696. IEEE, 2010.

[6] P. Bouyer, L. Bozzelli, and F. Chevalier. Controller synthesis for mtl specifications. In *Proc. of CONCUR*, 2006.

[7] Y. Chen, J. Tumova, A. Ulusoy, and C. Belta. Temporal logic robot control based on automata learning of environmental dynamics. *International Journal of Robotics Research*, 32(5):547–565, 2013.

[8] L. Doyen, G. Geeraerts, J.-F. Raskin, and J. Reichert. Realizability of real-time logics. In *Proc. of FORMATS*, pages 133–148, 2009.

[9] D. D'Souza and P. Madhusudan. Timed control synthesis for external specifications. In *Proc. of STACS*, pages 571–582, 2002.

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

[11] G. E. Fainekos and G. J. Pappas. Robustness of temporal logic specifications for continuous-time signals. *Theoretical Computer Science*, 410(42):4262–4291, 2009.

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

[13] C. A. Furia and M. Rossi. On the expressiveness of mtl variants over dense time. In *Proc. of FORMATS*, pages 163–178. Springer, 2007.

[14] A. Girard, G. Pola, and P. Tabuada. Approximately bisimilar symbolic models for incrementally stable switched systems. *IEEE Trans. Automatic Control*, 55:116–126, 2010.

[15] S. Karaman and E. Frazzoli. Vehicle routing problem with metric temporal logic specifications. In *Proc. of CDC*, pages 3953–3958. IEEE, 2008.

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

[17] R. Koymans. Specifying real-time properties with metric temporal logic. *Real-Time Systems*, 2(4):255–299, 1990.

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

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

[20] J. Liu and N. Ozay. Abstraction, discretization, and robustness in temporal logic control of dynamical systems. In *Proc. of HSCC*, 2014.

[21] J. Liu, N. Ozay, U. Topcu, and R. M. Murray. Synthesis of reactive switching protocols from temporal logic specifications. *IEEE Transactions on Automatic Control*, 58(7):1771–1785, 2013.

[22] J. Liu, U. Topcu, N. Ozay, and R. M. Murray. Reactive controllers for differentially flat systems with temporal logic constraints. In *Proc. of CDC*, 2012.

[23] S. C. Livingston, P. Prabhakar, A. B. Jose, and R. M. Murray. Patching task-level robot controllers based on μ-calculus formula. In *In Proc. ICRA*, 2013.

[24] O. Maler, D. Nickovic, and A. Pnueli. Real time temporal logic: past, present, future. In *Proc. of FORMATS*, pages 2–16. Springer-Verlag, 2005.

[25] O. Maler, D. Nickovic, and A. Pnueli. From MITL to timed automata. In *Proc. of FORMATS*, pages 274–289. Springer, 2006.

[26] D. Nickovic and N. Piterman. From MTL to deterministic timed automata. In *Proc. of FORMATS*, volume 6246, pages 152–167, 2010.

[27] N. Ozay, J. Liu, P. Prabhakar, and R. M. Murray. Computing augmented finite transition systems to synthesize switching protocols for polynomial switched systems. In *Proc. of ACC*, 2013.

[28] H.-W. Park, A. Ramezani, and J. Grizzle. A finite-state machine for accommodating unexpected large ground-height variations in bipedal robot walking. *IEEE Trans. on Robotics*, 29:331–345, 2013.

[29] A. Pnueli, Y. Sa'ar, and L. Zuck. JTLV: A framework for developing verification algorithms. In *Proc. of CAV*, pages 171–174, 2010.

[30] G. Pola and P. Tabuada. Symbolic models for nonlinear control systems: Alternating approximate bisimulations. *SIAM J. Control Optim.*, 48:719–733, 2009.

[31] P. Prabhakar and D. D'Souza. On the expressiveness of MTL with past operators. In *Proc. of FORMATS*, pages 322–336, 2006.

[32] J. M. Toibero, R. Carelli, and B. Kuchen. Switching control of mobile robots for autonomous navigation in unknown environments. In *Proc. of ICRA*, pages 1974–1979. IEEE, 2007.

[33] E. M. Wolff, U. Topcu, and R. M. Murray. Efficient reactive controller synthesis for a fragment of linear temporal logic. In *Proc. of ICRA*, 2013.

[34] T. Wongpiromsarn, U. Topcu, and R. M. Murray. Receding horizon temporal logic planning. *IEEE Trans. Automatic Control*, 57:2817–2830, 2012.

[35] M. Zamani, G. Pola, M. Mazo Jr, and P. Tabuada. Symbolic models for nonlinear control systems without stability assumptions. *IEEE Trans. Automatic Control*, 57:1804–1809, 2012.