2020 American Control Conference
Denver, CO, USA, July 1-3, 2020

Continuous-time Signal Temporal Logic Planning with Control Barrier Functions

Guang Yang, Calin Belta and Roberto Tron

Abstract— **Temporal Logic (TL) guided control problems have gained enormous interests in recent years. A wide range of properties, such as liveness and safety, can be specified through TL. On the other hand, Control Barrier Functions (CBF) have shown success in the context of safety critical applications that require constraints on the system states. In this paper, we consider linear cyber-physical systems with continuous dynamics, where controls are generated by digital computers in discrete time. We propose an offline trajectory planner for such systems subject to linear constraints given as Signal Temporal Logic (STL) formulas. The proposed planner is based on a Mixed Integer Quadratic Programming (MIQP) formulation that utilizes CBFs to produce system trajectories that are valid in continuous time; moreover we allow STL predicates with arbitrary time constraints, in which asynchronous control updates are allowed. We validate our theoretical results through numerical simulations.**

I. Introduction

Temporal Logic (TL)-based control has been widely used in the context of persistent surveillance [1], traffic control [2] and distributed sensing [3]. While originating from the field of formal methods [4], TLs are now used to describe specifications for a variety of system behaviors, as attested by the proliferation of many different specialized languages (such as Linear Temporal Logic [5], Computation Tree Logic [6] and Time Window Temporal Logic [7]). For applications that require the definition of real values with bounded time constraints, Signal Temporal Logic (STL) [8] and Metric Temporal Logic (MTL) [9] have been introduced.

The notion of STL robustness over real-valued signals [10], also known as space robustness, provides a quantitative semantics of how well a signal satisfies a given STL formula.[1] In discrete time, it is possible to encode the robustness function of a formula into the constraints of a Mixed Integer Quadratic Program (MIQP), thus allowing for relatively efficient control synthesis [12]–[14]. The major drawback of this type of approach is that it is limited to the discrete-time setting for verifying the satisfaction of each predicate (since all time instants need to be represented with variables in the MILP); if the same paradigm is applied to the discretization of continuous-time systems, it does not guarantee the satisfaction of the formula in between two sampled time steps (see Figure 3 for an example).

Moreover, on the one hand, practical systems evolve in continuous time, and time intervals in the specification can also involve arbitrary (application-driven) continuous-time intervals. On the other hand, there is usually a limitation on the control and actuation rates that can be achieved, and the control updates cannot be generally assumed to coincide with the time intervals in the specification.

Control Barrier Functions (CBFs, first introduced in [15]), are related to Control Lyapunov Functions (CLF), but instead of stability they guarantee that the trajectories of a system remain in a pre-defined *forward invariant* set. CBFs have been extended to Exponential CBFs [16] and High Order CBF (HOCBF) [17] for systems with relative degree higher than one. CBFs have been applied to adaptive cruise control [18], swarm manipulation [19], heterogeneous multi-robot manipulation [20], and bipedal robotic walking [21]. A typical CBF formulation involves a continuous-time system and results in a Quadratic Program (QP) that need to be solved at every control update. For real-world systems with discrete time updates, the computed controls are applied in a Zero Order Hold (ZOH) manner, but special care needs to be taken in order to ensure that the CBF constraints hold true in between the two control updates [22], [23].

In this paper, we consider trajectory planning for continuous linear systems with discrete control updates, constrained by linear CBF safety sets, and STL specifications with linear predicates. We are interested in synthesizing a discrete sequence of controls to satisfy an STL formula in continuous time while remaining in the safety set specified by the CBFs. There exist some work [24] that combines TLs with CBFs using continuous dynamics; however in that formulation predicates are guaranteed to be satisfied only at discrete times and does not guarantee continuous-time satisfaction. To the best of our knowledge, there have been no attempts for trajectory planning under continuous-time Signal Temporal Logic specification with discrete control updates using CBFs.

In our proposed method, we formulate a MIQP with constraints obtained from the STL specification and CBF functions in such a way that continuous-time satisfaction is guaranteed with discrete ZOH updates. The CBFs are used to derive constraints based on linear predicates that guarantee continuous satisfaction between time instants. Our main contributions are as follows. First, we propose a novel integer encoding method that provides lower bounds for linear CBF constraints over fixed time intervals. Second, we overcome the drawbacks of control synthesis based on discrete-time STL robustness by encoding the STL robustness as time-varying CBF constraints. Third, we provide an answer to the issue of asynchronicity of update times between a given STL specification and the actual system sampling and control update rates.

[1]There also exists a notion of time robustness [11]. In this paper, when we use the term robustness, we refer to space robustness.

II. Preliminaries

A. Notation

We use $\mathbb{Z}$ and $\mathbb{R}^n$ to denote the set of integers and the n-dimensional real space, respectively. A function $f : \mathbb{R}^n \mapsto \mathbb{R}^m$ is called *Lipschitz continuous* on $\mathbb{R}^n$ if there exists a positive real constant $L \in \mathbb{R}^+$, such that $\|f(y) - f(x)\| \leq L\|y - x\|, \forall x, y \in \mathbb{R}^n$. Given a continuously differentiable function $h : \mathbb{R} \mapsto \mathbb{R}$, we use $h^{(r)}$ to denote its r-th order derivative with respect to time t. A continuous function $\alpha : [-b, a) \mapsto [-\infty, \infty)$, for some $a > 0, b > 0$, is said to be of class $\mathcal{K}$ if α is strictly increasing, and $\alpha(0) = 0$.

B. System Dynamics

Consider a continuous-time linear control system:

$$\dot{x}(t) = Ax(t) + Bu, \tag{1}$$

where $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, while $x \in \mathbb{R}^n$ and $u \in \mathbb{R}^m$ represent the state and control inputs.

We assume that we are only able to update the control inputs only at regular discrete sampling instants. We denote t_k as the k-th sampling time instant, and the time interval between control updates as $\tau = t_{k+1} - t_k, k = \{1, 2, \ldots\}$. For $t \in [t_k, t_{k+1})$, we implement the Zeroth-Order Hold control mechanism which holds a control signal at t_k constantly until t_{k+1}. For each update interval, the dynamics (1) can be exactly integrated as

$$x(t) = e^{A(t-t_k)}x[t_k] + \int_{t_k}^{t_{k+1}} e^{A(t-s)}\mathrm{d}sBu[t_k], \tag{2}$$

for $t_k \leq t < t_{k+1}$. Let $V^{-1}QV$ be the Jordan decomposition of A, where V an invertible matrix, and Q a block-diagonal matrix containing κ Jordan blocks. We denote $s(i)$ and λ_i the size and eigenvalue associated with i-th Jordan block, respectively, $i \in \{1, \ldots, \kappa\}$. With this decomposition, we can rewrite (2) as:

$$x(t) = e^{A(t-t_k)}x[t_k] + e^{A(t-t_k)}V\int_{t_k}^{t_{k+1}} e^{-Qs}\mathrm{d}sV^{-1}Bu[t_k]. \tag{3}$$

C. Safety Set

We define N_c safety sets $\mathcal{C}_l, l = 1, ..., N_c$ as

$$\mathcal{C}_l = \{x \in \mathbb{R}^n | h_l(x) \geq 0\}, \tag{4}$$

and use $\partial\mathcal{C}_l$ and $Int(\mathcal{C}_l)$ to denote the boundary and the interior of $\mathcal{C}_l$. Given an initial time t_0, we call the set $\mathcal{C}_l$ *forward invariant* for system (1) if $x(t_0) \in \mathcal{C}_l$ implies $x(t) \in \mathcal{C}_l, \forall t \geq t_0$; note that invariance is maintained under complement, intersection and union of sets.

In this paper, we consider affine safety constraints as smooth functions in the form

$$h_l(x) = \nu_l^{\mathrm{T}} x + \gamma_l, \quad l = 1, \ldots, N_c, \tag{5}$$

where $\nu_l \in \mathbb{R}^n$ and $\gamma_l \in \mathbb{R}$. We define the Lie derivative of a smooth function $h_l(x(t))$ along the dynamics (1) as $\pounds_{Ax}h_l(x) := \frac{\partial h_l(x(t))}{\partial x(t)}Ax(t)$, $\pounds_B h_l(x) := \frac{\partial h_l(x(t))}{\partial x(t)}B$. The relative degree $r_b \geq 1$ is defined as the smallest natural number such that $\pounds_B \pounds_{Ax}^{r_b-1} h_l(x)u \neq 0$. The time derivatives of h_l can then be expressed as

$$h_l^{(r_b)}(x) = \pounds_{Ax}^{r_b} h_l(x) + \pounds_B \pounds_{Ax}^{r_b-1} h_l(x)u. \tag{6}$$

By combining (5) and (6), we have

$$h_l^{(r_b)}(x) = \nu_l^T (A)^{r_b} x + \nu_l^T (A)^{r_b-1} Bu, \tag{7}$$

where $(A)^{r_b}$ is the r_b-th power of A.

D. Exponential Control Barrier Function

We use Exponential Control Barrier Functions (ECBF, [16]) to ensure forward invariance of the set $\mathcal{C}_l$ with relative degree r_b. Before the formal definition, we first introduce a virtual input-output linearized system

$$\begin{aligned}\dot{\xi}_b(x) &= A_b\xi_b(x) + B_b\mu,\\ h_l(x) &= C_b\xi_b(x),\end{aligned}$$

where $\xi_b(x) = \left[h_l(x), \dot{h}_l(x), ..., h_l^{r_b}(x)\right]^T$,with

$$A_b = \begin{bmatrix} 0 & 1 & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & \cdot & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, B_b = \begin{bmatrix} 0 \\ \cdot \\ 0 \\ 1 \end{bmatrix},$$
$$C_b = [1 \cdots 0].$$

Now we can formally define the ECBF as follows:

*Definition 1 (*Exponential Control Barrier Function*):* Consider the dynamical system (1), a safety set $\mathcal{C}_l$ defined in (4) with $h_l(x)$ having relative degree $r_b \geq 1$. Then, $h_l(x)$ is an exponential control barrier function (ECBF) if there exists $K_b \in \mathbb{R}^{1\times r_b}$ such that, $\forall x \in Int(\mathcal{C}_l)$,

$$\inf_{u\in U} [\pounds_{Ax}^{r_b} h_l(x) + \pounds_B \pounds_{Ax}^{r_b-1} h_l(x)u + K_b\xi_b(x)] \geq 0, \tag{8}$$

and K_b is a feedback gain that stabilizes a system in standard controllable form (see [16] for details).

E. Signal Temporal Logic

The syntax of STL is recursively defined as:

$$\varphi := \top|\mu|\neg\varphi|\varphi_1 \wedge \varphi_2|\varphi_1 \vee \varphi_2|\mathbf{F}_{[a,b]}\varphi|\mathbf{G}_{[a,b]}\varphi|\varphi_1\mathcal{U}_{[a,b]}\varphi_2, \tag{9}$$

where $\top$ is the Boolean constant *true*, and μ is a predicate. We consider predicates μ_i of the form

$$\mu := y(t) \geq 0, \tag{10}$$

where y is a linear function over the states of (1). The *Eventually* temporal operator $\mathbf{F}_{[a,b]}\varphi$ specifies that φ holds true at some time step between $[a, b]$. The *Always* operator $\mathbf{G}_{[a,b]}\varphi$ states that φ must holds true $\forall t \in [a, b]$. To state that a signal y satisfies a specification (formula) φ at time t we use the notation $x(t) \models \varphi$. The STL semantics is the defined as follows:

$$\begin{aligned}
&(y,t) \models \mu \Leftrightarrow y(t) \geq 0\\
&(y,t) \models \neg\mu \Leftrightarrow \neg((y,t) \models \mu)\\
&(y,t) \models \mu_1 \wedge \mu_2 \Leftrightarrow (y,t) \models \mu_1 \wedge (y,t) \models \mu_2\\
&(y,t) \models \mathbf{F}_{[a,b]}\mu \Leftrightarrow \exists t' \in [t+a, t+b] s.t. (y,t') \models \mu \qquad (11)\\
&(y,t) \models \mathbf{G}_{[a,b]}\mu \Leftrightarrow \neg\mathbf{F}_{[a,b]}(\neg\mu)\\
&(y,t) \models \varphi_1 \mathcal{U}_{[a,b]} \varphi_2 \Leftrightarrow \exists t' \in [t+a, t+b]\\
&s.t. (y,t') \models \varphi_2 \wedge \forall t'' \in [t,t'], (y,t'') \models \varphi_1.
\end{aligned}$$

All STL temporal operators have bounded time intervals in continuous time. The *horizon* of an STL formula is the minumum time needed to decide its satisfaction. For an STL formula that has no nested operators, its horizon is determined by the largest upper bound in all operators.

F. Mixed Integer Formulation for STL

In this section, we review the binary encoding of STL robustness using mixed integer constraints proposed in [14]. This encoding is based on the big-M method, where a sufficiently large number M is introduced to enforce logical constraints. For the i-th predicate μ_i and the corresponding binary variable $z_{\mu_i}[t_k] \in \{0,1\}$, we use the constraints

$$y_i[t_k] \leq M z_\mu[t_k], \qquad -y_i[t_k] \leq M(1 - z_\mu[t_k]), \qquad (12)$$

to establish the relation

$$y_i[t_k] \geq 0 \iff z_\mu[t_k] = 1. \qquad (13)$$

For an STL formula φ with horizon N, we denote $z_\varphi[t_k] \in \{0,1\}$, with $(x,t) \models \varphi \iff z_\varphi[t] = 1$. We also denote $z_{\varphi_i}[t]^k \in \{0,1\}$ for the i-th subformula which is recursively defined based on the STL semantics (9).

Given a STL formula φ, we can recursively encode the rest of the logical operators by using the binary variables of subformula and predicates as shown in Table I (we dropped the k for simplicity in the table).

As mentioned in the introduction, this encoding guarantees satisfaction of the entire STL formula over a horizon N only at the sampling time instants $t_k, k = 1, ..., N$, and for temporal operators having the boundary of each time interval (e.g., a and b in $\mathbf{G}_{[a,b]}$) coincide with update instants t_k.

	Definition	**Encoding Rule**
$\wedge$	$z_\varphi[t] = \wedge_{i=1}^p z_{\psi_i}[t]$	$z_\varphi[t] \leq z_{\psi_i}[t], z_\varphi[t] \geq 1 - p + \sum_{i=1}^p z_{\psi_i}[t]$
$\vee$	$z_\varphi^t = \vee_{i=1}^p z_{\psi_i}[t]$	$z_\varphi[t] \geq z_{\psi_i}[t], z_\varphi[t] \leq \sum_{i=1}^p z_{\psi_i}[t]$
$\neg$	$z_\varphi[t] = \neg z_\psi[t]$	$z_\varphi[t] = 1 - z_\psi[t]$
F	$\varphi = \mathbf{F}_{[a,b]}\psi$	$z_\varphi[t] = \bigvee_{\tau=t+a}^{t+b} z_{\psi_i}^\tau$
G	$\varphi = \mathbf{G}_{[a,b]}\psi$	$z_\varphi[t] = \bigwedge_{\tau=t+a}^{t+b} z_{\psi_i}^\tau$
$\mathcal{U}$	$\varphi = \psi_1 \mathcal{U}_{[a,b]} \psi_2$	$\mathbf{G}_{[0,a]}\psi_1 \wedge \mathbf{F}_{[a,b]}\psi_2 \wedge \mathbf{F}_{[a,a]}\psi_1 \mathcal{U} \psi_2$

TABLE I: STL Encoding with Mixed-integer

III. Problem Statement

Problem 1: Given the linear system (1) with initial state $x_0 \in X \subseteq \mathbb{R}^n$, where X is an initial feasible set that satisfies all safety constraints (4), and given a STL formula φ with horizon t_f, synthesize a sequence of control inputs $u[t_k], k = 1, ..., N$, that minimizes a quadratic cost function $J(u(t))$ over the horizon, while the trajectory satisfies a formula φ.

IV. STL based control with Control Barrier Function

This section contains the main theoretical and algorithmic contributions of this paper. We present the CBF formulation in section IV-A. Next, in section IV-B, we demonstrate how to obtain the lower bound of a given linear CBF constraint using mixed-integer encoding. In section IV-C, we show how certain STL formulas can be encoded as CBF constraints and achieve continuous-time satisfaction. Finally, the MIQP based planner is formally defined in section IV-E.

A. CBFs for Linear Constraints

From the closed form solution (3) for the dynamical system and safety constraint (5), we can write the CBF constraint (8) at the k-th update instant as

$$\zeta_k(t) = \sigma + \sum_{i=1}^{\kappa} \sum_{j=0}^{s(i)-1} (c_{k,i,j}^{(x)T} x[t_k] e^{\lambda_i t} t^j + c_{k,i,j}^{(u)T} u[t_k] e^{\lambda_i t} t^j),$$
$$\zeta_k(t) \geq 0, \forall t \in [t_k, t_{k+1}]. \qquad (14)$$

where $c_{k,i,j}^{(x)} \in \mathbb{R}^n$, $c_{k,i,j}^{(u)} \in \mathbb{R}^n$, and $\sigma \in \mathbb{R}$ are constants obtained by solving the matrix exponentials in (3) and carrying out the subsequent matrix-vector calculations.

From a computational standpoint, the main difficulty in enforcing (14) is the fact that the inequality needs to hold on an entire interval of τ. An equivalent constraint could be obtained by taking the minimum of $\zeta(t)$ over the same interval, and then enforcing the inequality on this minimum. To obtain the minimum, we decompose the sum in (14) into the following terms:

$$\zeta_{k,i,j}^{(x)}(t) = c_{k,i,j}^{(x)T} x(t_0) e^{\lambda_i t} t^j, \quad \zeta_{k,i,j}^{(u)}(t) = c_{k,i,j}^{(u)T} u_0 e^{\lambda_i t} t^j,$$
$$i = 1, \ldots, \kappa, j = 0, \ldots, s(i) - 1, \qquad (15)$$

and we introduce a set of slack variables $\beta_{ij}^{(x)}$, $\beta_{ij}^{(x)}$, $i = 1, \ldots, \kappa, j = 0, \ldots, s(i) - 1$ such that

$$\sum_{i=1}^{\kappa} \sum_{j=0}^{s(i)-1} \beta_{k,i,j}^{(x)} + \beta_{k,i,j}^{(u)} = \sigma. \qquad (16)$$

We then substitute (14) with the following inequalities:

$$\zeta_{k,i,j}^{(x)}(t) + \beta_{k,i,j}^{(x)} \geq 0, \ \zeta_{k,i,j}^{(u)}(t) + \beta_{k,i,j}^{(u)} \geq 0,$$
$$i = 1, \ldots, \kappa, j = 0, \ldots, s(i) - 1, \forall t \in [t_k, t_{k+1}]. \qquad (17)$$

To simplify the notation, we will drop the subscript k for the remainder of this section. The transformation of the constraints is justified by the following proposition.

Proposition 1: There exist a set of $\{\beta_{i,j}^{(x)}, \beta_{i,j}^{(u)}\}$ such that (16) and (17) hold if and only if inequality (14) holds.

Proof: To prove that (17) implies (14), we can simply sum all the inequalities in (17) over i and j, and then simplify the summation of the β's using (16).

To prove that (14) implies (17), we first define the "excess" quantity

$$\delta = \sum_{i=1}^{\kappa} \sum_{j=0}^{s(i)-1} \left(\zeta_{i,j}^{(x)}(t) + \zeta_{i,j}^{(u)}(t)\right), \tag{18}$$

and then construct the β's by splitting δ and σ evenly as follows:

$$\begin{gathered}\beta_{i,j}^{(x)} = -\,\zeta_{i,j}^{(x)}(t) + \frac{\delta + \sigma}{2\sum_{i=1}^{n}(s(i)-1)},\\ \beta_{i,j}^{(u)} = -\,\zeta_{i,j}^{(u)}(t) + \frac{\delta + \sigma}{2\sum_{i=1}^{n}(s(i)-1)}.\\ i \in \{1, \dots, \kappa\}, j \in \{0, \dots, s(i)-1\},\end{gathered}$$

We first verify that these β's satisfy the summation constraint (16):

$$\begin{aligned}\sum_{i=1}^{\kappa} \sum_{j=0}^{s(i)-1} \beta_{i,j}^{(x)} + \beta_{i,j}^{(u)} &= -\sum_{i=1}^{\kappa} \sum_{j=0}^{s(i)-1} (\zeta_{i,j}^{(x)}(t) + \zeta_{i,j}^{(u)}(t))\\ &+ 2\sum_{i=1}^{\kappa} \sum_{j=0}^{s(i)-1} \frac{\delta + \sigma}{2\sum_{i=1}^{\kappa}(s(i)-1)} = \sigma \end{aligned} \tag{19}$$

To show that the constructed β's also satisfy (17), first notice that by substituting (18) into (14), we have $\delta + \sigma \geq 0$; then we have

$$\zeta_{ij}^{(x)}(t) + \beta_{i,j}^{(x)} = \frac{\delta + \sigma}{2\sum_{i=1}^{n}(s(i)-1)} \geq 0, \tag{20}$$

for all i, j, and with an analogous expression for $\zeta_{ij}^{(u)}(t)$, $\beta_{i,j}^{(u)}$. This completes the proof.

■

B. CBF lower bound through mixed-integer encoding

As briefly anticipated in the previous section, the constraints in (17) need to hold for every time instant in a given interval, resulting in an infinite number of constraints. In order to include such constraints in the MIQP formulation, we need to drop the dependency on t while maintaining linearity in terms of x and u. We perform one additional transformation by defining new variables that capture lower bounds (over time) of the expressions in (17):

$$\begin{gathered}\zeta_{k,i,j,\min}^{(x)} = \min_{t \in [t_k, t_{k+1}]} \zeta_{k,i,j}^{(x)}(t),\\ \zeta_{k,i,j,\min}^{(u)} = \min_{t \in [t_k, t_{k+1}]} \zeta_{k,i,j}^{(u)}(t),\\ i \in \{1, \dots, \kappa\}, j \in \{0, \dots, s(i)-1\}.\end{gathered} \tag{21}$$

Then, (17) can be exactly replaced by

$$\begin{gathered}\zeta_{k,i,j,\min}^{(x)}(t) + \beta_{k,i,j}^{(x)} \geq 0,\ \zeta_{k,i,j,\min}^{(u)}(t) + \beta_{k,i,j}^{(u)} \geq 0,\\ i \in \{1, \dots, \kappa\}, j \in \{0, \dots, s(i)-1\}.\end{gathered} \tag{22}$$

There is a finite number of such constraints, as they do not depend on the continuous time anymore. We incorporate them in our MIQP formulation in two steps.

The first step is to use the Big-M encoding method to remove all the terms that are either monotonically increasing or bounded below by zero, and so they cannot be active at the current solution. We define sets of binary variables $z_{k,i,j}^{(x)}, z_{k,i,j}^{(u)} \in \{0, 1\}$ for each one of the inequalities in (22). We then associate desired values of $\zeta_{k,i,j}^{(x)}(t)$ or $\zeta_{k,i,j}^{(u)}(t)$ according to the following rules:

$$z_{k,i,j}^{(x)} = \begin{cases} 0, & c_{k,i,j}^{(x)T} x[t_k] \geq 0 \wedge \lambda_i \geq 0 \\ 0, & c_{k,i,j}^{(x)T} x[t_k] \geq 0 \wedge \lambda_i \leq 0 \wedge \sigma \geq 0 \\ 1, & \text{otherwise} \end{cases} \tag{23}$$

$$z_{k,i,j}^{(u)} = \begin{cases} 0, & c_{k,i,j}^{(u)T} u[t_k] \geq 0 \wedge \lambda_i \geq 0 \\ 0, & c_{k,i,j}^{(u)T} u[t_k] \geq 0 \wedge \lambda_i \leq 0 \wedge \sigma \geq 0 \\ 1, & \text{otherwise} \end{cases} \tag{24}$$

These rules are motivated by the fact that when $z_{k,i,j} = 0$, the corresponding inequality in (17) is automatically satisfied, and hence it can be ignored. The rules are transformed into mixed-integer linear constraints using the big-M method.

For example, if we want to enforce $c_{k,i,j}^{(x)T} x[t_k] \geq 0 \iff z_{k,i,j}^{(x)} = 0$ and $c_{k,i,j}^{(u)T} u[t_k] \geq 0 \iff z_{k,i,j}^{(u)} = 0$, the following mixed integer encoding are used: $c_{k,i,j}^{(x)T} x[t_k] \leq M(1 - z_{k,i,j}^{(x)})$, $-c_{k,i,j}^{(x)T} x[t_k] \leq M z_{k,i,j}^{(x)}$, $c_{k,i,j}^{(u)T} u[t_k] \leq M(1 - z_{k,i,j}^{(u)})$, $-c_{k,i,j}^{(u)T} u[t_k] \leq M z_{k,i,j}^{(u)}$. For $\lambda_i \geq 0 \iff z_{k,i,j}^{(x,u)} = 0$, we have $\lambda_i \leq M(1 - z_{k,i,j}^{(x,u)})$,$-\lambda_i \leq M z_{k,i,j}^{(x,u)}$, where M is a sufficiently large number. For all terms such that $z_{k,i,j}^{(x,u)} = 1$, we need to ensure $\zeta_{k,\min}^{(x)}(x[t_k], \tau)$ and $\zeta_{k,\min}^{(u)}(u[t_k], \tau)$ are positive. Consider the CBF lower bound (21), for $\forall t \in [t_k, t_{k+1}]$. The idea is that $\zeta_k(t)$ converges to a value monotonically when $j = 0$ (simple eigenvalue) and has a minimum stationary point when $j \geq 1$ (Jordan block of dimension greater than one). In this form, we can compute such lower bounds analytically; the results are collected in Table II.

Note that the minimum values $\zeta_{k,\min}$ are linear in the optimization variables $x[t_k]$, $u[t_k]$. Therefore, these lead to linear constraints in our optimization problem.

C. CBF constraints for STL predicates

The forward invariance property from the CBF can be carried over to ensure STL satisfaction in continuous time.

Condition	Minimum Value
$c_{k,i,j}^{(x)T} x[t_k] < 0, c_{k,i,j}^{(u)T} u[t_k] < 0$ $\lambda_i \geq 0, j \geq 1$	$\zeta_{k,\min}^{(x)}(x[t_k], \tau) = \zeta_{k,i,j}^{(x)}(\tau)$ $\zeta_{k,\min}^{(u)}(u[t_k], \tau) = \zeta_{k,i,j}^{(u)}(\tau)$
$c_{k,i,j}^{(x)T} x[t_k] < 0, c_{k,i,j}^{(u)T} u[t_k] < 0$ $\lambda_i > 0, j = 0$	$\zeta_{k,\min}^{(x)}(x[t_k], \tau) = c_{k,i,j}^{(x)T} x[t_k] + \sigma + \beta_k^{(x)}$ $\zeta_{k,\min}^{(u)}(u[t_k], \tau) = c_{k,i,j}^{(u)T} u[t_k] + \beta_k^{(u)}$
$c_{k,i,j}^{(x)T} x[t_k] < 0, c_{k,i,j}^{(u)T} u[t_k] < 0$ $\lambda_i < 0, j \geq 1$	$\zeta_{k,\min}^{(x)}(x[t_k], \tau) = \zeta_{k,i,j}^{(x)}(-\frac{j}{\lambda_i})$ $\zeta_{k,\min}^{(u)}(u[t_k], \tau) = \zeta_{k,i,j}^{(u)}(-\frac{j}{\lambda_i})$
$c_{k,i,j}^{(x)T} x[t_k] < 0, c_{k,i,j}^{(u)T} u[t_k] < 0$ $\lambda_i < 0, j = 0$	$\zeta_{k,\min}^{(x)}(x[t_k], \tau) = c_{k,i,j}^{(x)T} x[t_k] + \sigma + \beta_k^{(x)}$ $\zeta_{k,\min}^{(u)}(u[t_k], \tau) = c_{k,i,j}^{(u)T} u[t_k] + \beta_k^{(u)}$

TABLE II: CBF lower bound encoding

In short, we would like to enforce CBF constraints on all subformulae containing **G** (always) temporal operator, such that the continuous state trajectory satisfies the subformulae. More specifically, we first use the mixed-integer method from II-F to ensure the trajectory satisfies the formula at sampling instants $t_k, k = 1, ..., N$. The idea is to ensure the state trajectory will stay within the set defined by predicate $\mu, \forall t \in [a, b]$ by defining the predicate as a safety set i.e., $h(x) := \mu$ and enforce continouse time CBF lower bound at t_k.

D. Asynchronous STL and control updates

Recall that we discretize system (1) with constant sampling interval τ. If a time boundary of a temporal operator (i.e., a or b in $[a, b]$) coincides with some sampling time instant t_k, we can directly encode the CBF constraints at t_k using the methods from sections IV-A and II-F. However, temporal operators could have time boundaries falling between sampling intervals (i.e., $a, b \in (t_k, t_{k+1})$ for some k). For example, we have STL formula $\phi = \mathbf{G}_{[0.63s,0.80s]}\mu$ for some predicate μ. Assume that the system can only sample at the $t_k \in \{0, 0.2, ..., 2.0\}$ time instants. In this case, we cannot encode the STL specification directly due to the mismatch between the system sampling instants and the time intervals from STL predicate. We approach this issue by using two time scales, namely the *Simulated System*, $\{t^{sim}_{k_s}\}^{N_s-1}_{k_s=0}$, and the *Real System*, $\{t^{real}_{k_r}\}^{N_r-1}_{k_r=0} \subset \{t^{sim}_{k_s}\}$. we define the two control sequences as $\{u^{sim}[t_{k_s}]\}$ and $\{u^{real}[t_{k_r}]\}$ accordingly.

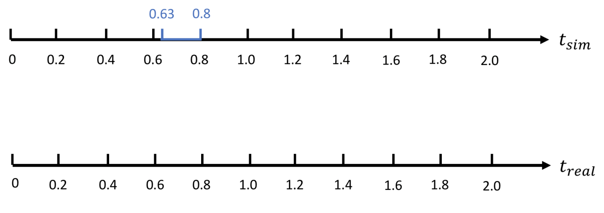

Fig. 1: Time Scales for Real and Simulated Systems

Notice that under the system update constraint, we can only control the system at the real sampling times (e.g., $t = 0.2s, 0.4s, ..., 2.0s$). To solve the mismatch, we formulate the MIQP with the simulated control sequence $\{u_{sim}[t_{k_s}]\}$, but then add additional constraint $u^{sim}[t_{k_s}] = u_{sim}[t'_{k_r}]$, for all t_{k_s} not appearing in $\{t^{real}_{k_r}\}$, and where t'_{k_r} is picked to be the closest preceding real sampling time; for instance, in our example we set $u^{sim}[t = 0.63] = u_{sim}[t = 0.6]$. In this way, u^{sim} can be applied to the real system while satisfying the required time discretization. In example V-C, we show an example of solving this type of problem.

E. Optimization Problem

To solve Problem 1, we formulate the following MIQP:

$$\begin{aligned} \min_{\mathbf{u},\mathbf{x}} \quad & u(t)^T u(t) \\ \text{s.t.} \quad & x[t_{k+1}] = A_k x[t_k] + B_k u[t_k], \\ & x(t) \models \varphi, \\ & z_\varphi, z^{(x)}_{k,i,j}, z^{(u)}_{k,i,j} \in \{0, 1\}, \\ & u_l \leq u_k \leq u_u, \\ & k = 0, ..., N-1 \\ & t \in [0, t_f], i = [1, ..., \kappa], j = [1, ..., s(i)]. \end{aligned} \tag{25}$$

N is the total number of controller updates and t_f is the horizon of the formula φ. The decision variables for the MIQP are $x[t_k]$ and $u[t_k]$ that are evaluated on the simulated time sequence, which is defined in IV-D. The u_l and u_u are the lower bound and upper control bounds, respectively. To ensure $x(t) \models \varphi$, we enforce the mixed integer constraints that are defined in II-F and IV-B. A_k and B_k are defined using the discretization method in Section II-B.

V. Numerical Examples

In Example 1, we enforce safety constraints over velocity on an one dimensional double integrator system, while its position is required to oscillate between some time intervals; we impose safety velocity constraints as an intersection of two safety sets in the form of (4). In Example 2, we demonstrate safety constraints over positions as a union of the safety sets on a 2-dimensional double integrator system. In Example 3, given the same system from Example 1, we demonstrate that our method can be used for the case where we have asynchronous time scales between $\{[t_{k_s}]_{sim}\}$ and $\{[t_{k_r}]_{real}\}$. All examples were formulated as MIQPs using Gurobi [25] and solved on an i9-9980HK CPU.

A. Example 1: STL based planning with safety constraints

Consider a double integrator system

$$\begin{bmatrix} \dot{x_1}(t) \\ \dot{x_2}(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u, \tag{26}$$

where x_1 is the position and x_2 is the velocity. Given the following STL formula with horizon $t_f = 2s$:

$$\begin{aligned} \varphi_1 = & F_{[0.2s,0.8s]}(x_1(t) <= -2) \\ & \wedge F_{[1s,1.4s]}(x_1(t) >= 2) \\ & \wedge F_{[1.6s,2s]}(x_1(t) <= -2), \\ & \wedge G_{[0s,2s]}(x_2(t) <= 10 \wedge x_2(t) >= -10), \end{aligned} \tag{27}$$

which requires state $x_1(t)$ to oscillate between -2 and 2 within some real time interval. The velocity is required to satisfy the safety requirement: $-10 < x_2(t) < 10, \forall t \in [0, t_f]$. To achieve the safety requirement, we define two safety sets with $h_1(x) = x_2 - 10$ and $h_2(x) = -x_2 + 10$ in the form of (4) as follows: $\mathcal{C}_1 = \{x_2 \in \mathbb{R} | h_1(x) \geq 0\}, \mathcal{C}_2 = \{x_2 \in \mathbb{R} | h_2(x) \geq 0\}$.We would like o ensure system trajecotry stays within the intersection of the two safety sets, i.e., $\mathcal{C}_1 \cap \mathcal{C}_2$.

To achieve continuous-time satisfaction, we enforce the following CBF constraints based on the encoding method from IV-B as $\zeta_{\min,h_1}(x_2[t_k], u[t_k], \tau) \geq 0, \zeta_{\min,h_2}(x_2[t_k], u[t_k], \tau) \geq 0, \tau = t_{k+1} - t_k, k = 0, ..., N-1$. The elapsed time for solving the MIQP is 0.02 seconds. The hyper parameters are $x(t_0) = [1, -1]^T$, $k = 20$, $t_f = 2.0s$, $[u_l, u_u] = [-40, 40]$ and $N = 10$.

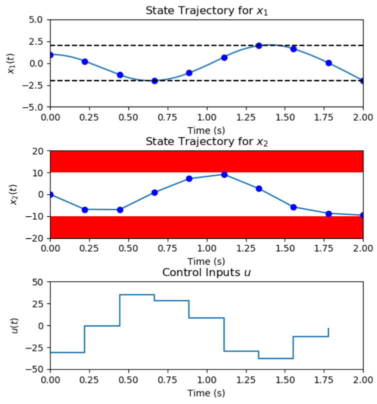

Fig. 2: Velocity constraints using CBFs

B. Example 2: STL predicates with disjunction

Consider a two-dimensional double-integrator system with $x = [x_1, x_2, x_3, x_4]$, where x_1, x_3 are positions and x_2, x_4 are velocities and control $u = [u_1, u_2]^T$ are the accelerations. Given the following STL formula with horizon $t_f = 1s$:

$$\begin{aligned}\varphi_2 :=& \mathbf{F}_{[0.1s,0.6s]}(x_1(t) \leq -0.5 \wedge x_3(t) \geq 0.5)\\ &\wedge \mathbf{F}_{[0.7s,1s]}(x_1(t) \geq 1 \wedge x_3(t) \geq 1)\\ &\wedge \mathbf{G}_{[0s,1s]}(x_1(t) \geq 0 \vee x_3(t) \geq 0),\\ & t \in [0, t_f].\end{aligned} \tag{28}$$

Again, we define two new safety sets with $h_3(x) = x_1$ and $h_4(x) = x_3$ as $\mathcal{C}_3 = \{x_1 \in \mathbb{R} | h_3(x) \geq 0\}, \mathcal{C}_4 = \{x_3 \in \mathbb{R} | h_4(x) \geq 0\}$. In this example, we demonstrate that a formula with an *always* temporal operator $\mathbf{G}_{[0s,1.0s]}x_1(t) \geq 0 \vee x_3(x) \geq 0$ can be expressed as a union of the two safety sets, i.e., $\mathcal{C}_3 \cup \mathcal{C}_4$. Note that, since h_3 and h_4 have relative degrees of $r_b = 2$, we define ECBF constraints: $\zeta_{h_3}(t), \zeta_{h_4}(t)$ based on (8). Next, we obtain the CBF lower bounds $\zeta_{\min,h_3}$, $\zeta_{\min,h_4}$ using our proposed mixed-integer encoding method from Section IV-B and enforce the following condition:

$$\begin{aligned}&\zeta_{\min,h_3}(x_1[t_k], u[k], \tau) \geq 0 \vee \zeta_{\min,h_4}(x_3[t_k], u[t_k], \tau) \geq 0\\ &t_k \leq t \leq t_{k+1}, k = 0, ..., N-1.\end{aligned}$$

Note we can encode logical OR ($\vee$) using Big-M encoding to achieve $\mathcal{C}_3 \cup \mathcal{C}_4$.

In Case 1, STL formula φ_2 is encoded with the mixed integer encoding method from Section II-F. Note that the discrete state trajectory satisfies φ_2 in Figure 3, but the continuous state trajectory clearly violates the formula. In Case 2, the CBF constraints are used in the MIQP and the resulting trajectory (Figure 4) satisfies φ_2 in continuous

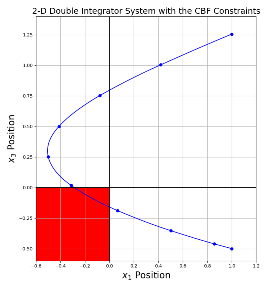

Fig. 3: Case 1 no CBF

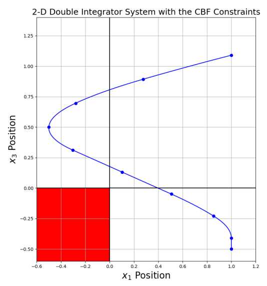

Fig. 4: Case 2 with CBF

time. The optimization problems in Case 1 and Case 2 are solved in 0.063s and 0.12s respectively. The hyper parameters are $x(t_0) = [1, 0, -0.5, 0]^T$, $[k_1, k_2] = [30, 30]$, $t_f = 1.0s$,$[u_l, u_u] = [-40, 40]$ and $N = 10$.

C. Example 3: Continuous-time STL under asynchronous time lines

Finally, we illustrate that a given STL formula can be satisfied under the asynchronous time scales between the real system and the simulated system in IV-D.

Consider the following example with the same double integrator system (26). The system can only update at the time instants $[t_{k_r}]_{real} = \{0, 0.2, ..., 2.0\}$ with $N_r = 11$. We would like the system trajectory to satisfy the following STL formula:

$$\varphi_3 = \mathbf{G}_{[0.63s,0.8s]}(x_2(t) >= 3) \wedge \mathbf{F}_{[1.4s,2s]}(x_2(t) <= -4). \tag{29}$$

We can directly encode $\mathbf{F}_{[1.4s,2s]}(x_2(t) <= -4)$ with the mixed-integer method from Section II-F because the time bound $[1.4, 2]$ is defined on the sampling instants $[t_{k_r}]_{real}$. However, we cannot enforce $\mathbf{G}_{[0.63s,0.8s]}(x_2(t) >= 3)$ using the previous method due to the asynchronous time scales. To resolve this issue, we define a simulated time scale based on φ_3 with $N_s = 12$, $[t_3]_{sim} = 0.6$, $[t_4]_{sim} = 0.63$ and safety set with $h_5(x) = x_2 - 3$ as $\mathcal{C}_5 = \{x_2 \in \mathbb{R} | h_5(x) \geq 0\}$. To ensure the system stays within $\mathcal{C}_5$ at $t_{real} = 0.63s$, the following CBF constraint is applied at $[t_4]_{sim}$:

$$\zeta_{\min,h_5}(x_2[t_4]_{sim}, u_{sim}[t_4], \tau) \geq 0, \tag{30}$$

with $\tau = 0.8s - 0.63s = 0.17s$ and additional constraint $u[t_4]_{sim} = u[t_3]_{sim}$. Finally, we can directly apply the synthesized control sequence $\{u[t_{k_s}]_{sim}\}$ in t_{real} by setting $u[t_3]_{real} := u[t_4]_{sim}$.

In Fig 5, we demonstrate that $\mathbf{G}_{[0.63s,0.80s]}$ is satisfied by defining an unsafe region (Red) using Eqn. (30). The resulting trajectory from the real system still satisfies φ_3 in continuous time. The MIQP is solved in 0.042s. The hyper parameters are $x(t_0) = [1, -1]^T$, $k = 5$, $t_f = 2.0s$, $[u_l, u_u] = [-40, 40]$, $N_s = 12$, $N_r = 11$.

D. Example 4: Nested Temporal Operators

In this example, we perform motion planning under an STL specification with nested temporal operators: $\varphi_4 =$

$\mathbf{F}_{[0,10]}\mathbf{G}_{[0,1]}(x_1 \geq 3) \wedge \mathbf{F}_{[0,10]}(x_1 \leq -2)$ for system (26). In other words, we would like the state x_1 to eventually reach $x_1 \geq 3$ hold there for 1s, and eventually reach $x_1 \leq -2$ between the entire interval $[0, 10]$ in seconds. The total number of control updates $N = 20$ and control is bounded with $-1 \leq u \leq 1$. The MIQP is solved in 0.0459s.

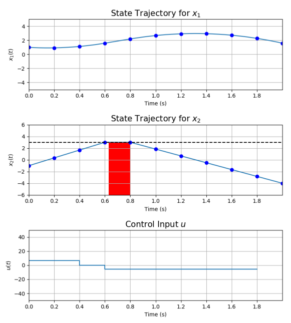

Fig. 5: Asynchronous Updates

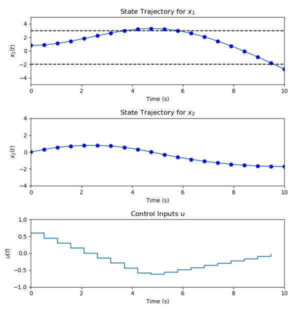

Fig. 6: Nested operator

VI. Conclusion

In this paper, we developed a novel framework for trajectory planning under real-valued space and time constraints from a STL formula. Motivated by the fact that current control approaches using STL cannot guarantee satisfaction in between discrete-time instants, we introduce an encoding of the *always* temporal operator by using the lower bound of a CBF. We further ensure the satisfaction of safety properties through STL formulation to be satisfied in continuous-time by adding CBF constraints. For future work, we will investigate adaptive control frameworks, where the control update intervals vary based on system states and CBF constraints, such that the STL formula is satisfied in continuous-time but with minimum number of controller updates.

VII. Acknowledgement

This work was partially supported by the National Science Foundation under grants IIS-1723995 and CMMI-1728277.

References

[1] K. Leahy, D. Zhou, C.-I. Vasile, K. Oikonomopoulos, M. Schwager, and C. Belta, "Persistent surveillance for unmanned aerial vehicles subject to charging and temporal logic constraints," *Autonomous Robots*, vol. 40, no. 8, pp. 1363–1378, 2016.
[2] S. Sadraddini and C. Belta, "Model predictive control of urban traffic networks with temporal logic constraints," in *2016 American Control Conference (ACC)*. IEEE, 2016, pp. 881–881.
[3] Z. Serlin, K. Leahy, R. Tronl, and C. Beita, "Distributed sensing subject to temporal logic constraints," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 4862–4868.
[4] C. Baier and J.-P. Katoen, *Principles of model checking*. MIT press, 2008.
[5] A. Pnueli, "The temporal logic of programs," in *18th Annual Symposium on Foundations of Computer Science (sfcs 1977)*. IEEE, 1977, pp. 46–57.
[6] E. M. Clarke and E. A. Emerson, "Design and synthesis of synchronization skeletons using branching time temporal logic," in *Workshop on Logic of Programs*. Springer, 1981, pp. 52–71.
[7] C.-I. Vasile, D. Aksaray, and C. Belta, "Time window temporal logic," *Theoretical Computer Science*, vol. 691, pp. 27 – 54, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0304397517305509
[8] O. Maler and D. Nickovic, "Monitoring temporal properties of continuous signals," in *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems*. Springer, 2004, pp. 152–166.
[9] R. Koymans, "Specifying real-time properties with metric temporal logic," *Real-time systems*, vol. 2, no. 4, pp. 255–299, 1990.
[10] G. E. Fainekos and G. J. Pappas, "Robustness of temporal logic specifications for continuous-time signals," *Theoretical Computer Science*, vol. 410, no. 42, pp. 4262–4291, 2009.
[11] A. Donzé and O. Maler, "Robust satisfaction of temporal logic over real-valued signals," in *International Conference on Formal Modeling and Analysis of Timed Systems*. Springer, 2010, pp. 92–106.
[12] V. Raman, A. Donzé, M. Maasoumy, R. M. Murray, A. Sangiovanni-Vincentelli, and S. A. Seshia, "Model predictive control with signal temporal logic specifications," in *53rd IEEE Conference on Decision and Control*. IEEE, 2014, pp. 81–87.
[13] V. Raman, A. Donzé, D. Sadigh, R. M. Murray, and S. A. Seshia, "Reactive synthesis from signal temporal logic specifications," in *Proceedings of the 18th international conference on hybrid systems: Computation and control*. ACM, 2015, pp. 239–248.
[14] S. Sadraddini and C. Belta, "Robust temporal logic model predictive control," in *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 2015, pp. 772–779.
[15] P. Wieland and F. Allgöwer, "Constructive safety using control barrier functions," *IFAC Proceedings Volumes*, vol. 40, no. 12, pp. 462–467, 2007.
[16] Q. Nguyen and K. Sreenath, "Exponential control barrier functions for enforcing high relative-degree safety-critical constraints," in *American Control Conference (ACC), 2016*. IEEE, 2016, pp. 322–328.
[17] W. Xiao and C. Belta, "Control barrier functions for systems with high relative degree," *preprint arXiv:1903.04706*, 2019.
[18] A. D. Ames, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs with application to adaptive cruise control," in *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*. IEEE, 2014, pp. 6271–6278.
[19] U. Borrmann, L. Wang, A. D. Ames, and M. Egerstedt, "Control barrier certificates for safe swarm behavior," *IFAC-PapersOnLine*, vol. 48, no. 27, pp. 68–73, 2015.
[20] L. Wang, A. Ames, and M. Egerstedt, "Safety barrier certificates for heterogeneous multi-robot systems," in *2016 American Control Conference (ACC)*. IEEE, 2016, pp. 5213–5218.
[21] S.-C. Hsu, X. Xu, and A. D. Ames, "Control barrier function based quadratic programs with application to bipedal robotic walking," in *American Control Conference (ACC), 2015*. IEEE, 2015, pp. 4542–4548.
[22] A. Ghaffari, I. Abel, D. Ricketts, S. Lerner, and M. Krstić, "Safety verification using barrier certificates with application to double integrator with input saturation and zero-order hold," in *2018 Annual American Control Conference (ACC)*. IEEE, 2018, pp. 4664–4669.
[23] G. Yang, C. Belta, and R. Tron, "Self-triggered control for safety critical systems using control barrier functions," in *2019 American Control Conference (ACC)*, July 2019, pp. 4454–4459.
[24] L. Lindemann and D. V. Dimarogonas, "Control barrier functions for signal temporal logic tasks," *IEEE control systems letters*, vol. 3, no. 1, pp. 96–101, 2019.
[25] L. Gurobi Optimization, "Gurobi optimizer reference manual," 2018. [Online]. Available: http://www.gurobi.com