Robust Temporal Logic Model Predictive Control

Sadra Sadraddini and Calin Belta

***Abstract*— Control synthesis from temporal logic specifications has gained popularity in recent years. In this paper, we use a model predictive approach to control discrete time linear systems with additive bounded disturbances subject to constraints given as formulas of signal temporal logic (STL). We introduce a (conservative) computationally efficient framework to synthesize control strategies based on mixed integer programs. The designed controllers satisfy the temporal logic requirements, are robust to all possible realizations of the disturbances, and optimal with respect to a cost function. In case the temporal logic constraint is infeasible, the controller satisfies a relaxed, minimally violating constraint. An illustrative case study is included.**

I. Introduction

Model predictive control (MPC), also known as receding horizon control (RHC), is a popular approach to generate (sub)optimal control strategies for systems with constraints [1],[2]. During the last decades, many theoretical aspects of MPC, such as stability and robustness, have been investigated [3],[4]. However, most works in the MPC literature focus on simple classes of performance objectives and constraints such as closeness to a reference point or trajectory. Recently, there has been a growing trend in control theory in considering a richer class of constraints that are described using rules and symbolism from formal methods such as temporal logics.

Temporal logics [5], such as linear temporal logic (LTL), computational tree logic (CTL), and signal temporal logic (STL), are able to describe a wide range of specifications. For example, satisfying disjoint sets of constraints infinitely often with specific deadlines for the satisfaction of each of them, (i.e., oscillatory behavior), can be naturally expressed in a temporal logic such as STL.

Temporal logic control based on finite abstractions [6],[7],[8] is a correct by construction control synthesis method that involves a finite state machine representation of the control system that is generally expensive to compute. To address this limitation, some works proposed receding horizon approaches to temporal logic control [9], [10]. Inspired by [11], the authors in [12],[13] have developed methods for translation of LTL constraints into mixed-integer constraints that are used in the controller synthesis algorithm. More recently, [14], [15] have extended this methodology to MPC from STL specifications by developing mixed-integer encodings of bounded time model checking.

In this paper, we use STL formulas over predicates in the state of the system to specify correctness requirements. We focus on discrete-time linear systems with additive bounded uncertainties. We propose an MPC approach to the synthesis problem with the goal of satisfying the STL specification globally (i.e., for all times) by characterizing the bounded propagation of uncertainties into STL constraints. We take a conservative approach that is computationally as tractable as STL control of deterministic systems. Furthermore, the notions introduced in this paper enable to treat STL constraints as *soft constraints* that may be violated if a feasible control policy does not exist. We are thus able to find minimally violating solutions in the presence of uncertainties and limited control actuation.

In the closest related work, the authors in [15] use a counter example guided approach to receding horizon control of disturbed systems. A major disadvantage of this approach is that the generation of counter examples may never terminate. Also, taking into account a large number of counter examples is computationally intractable. Furthermore, since there does not exist a global feasibility guarantee for STL MPC, the control algorithm in [15] may encounter infeasibility, an issue that we address in this paper.

This paper is organized as follows. First, we informally state the problem in Section II. Next, the receding horizon mechanism of STL control is explained in Section III. After providing technical details about robust prediction in Section IV, we formalize the problem as an optimization problem in Section V. Finally, a case study is presented in Section VI.

II. Problem Statement

We consider discrete-time dynamical systems of the form

$$x[t+1] = Ax[t] + Bu[t] + w[t], \tag{1}$$

$$y[t] = Cx[t] + Du[t] + e, \tag{2}$$

where $x \in \mathbb{R}^n$ is the state, $u \in \mathcal{U} \subseteq \mathbb{R}^m$ is the control restricted to an admissible set $\mathcal{U}$ and $t \in \mathbb{Z}_{\geq 0}$ is time. The exogenous inputs, or additive disturbances, $w[t] \in \mathcal{W} \subset \mathbb{R}^n$ are unknown but restricted to a known bounded set $\mathcal{W}$, which is assumed to be a polytope. Matrices A, B are fully known with appropriate dimensions. We also consider an associated stage cost function $J(x[t], u[t]), J : \mathbb{R}^n \times \mathcal{U} \to \mathbb{R}$, defined for each time step. The output y, and the corresponding C, D and e matrices, are defined from the predicates present in the temporal logic specification, as it will be shown below.

STL is a formal framework for describing a wide range of specifications in a convenient and compact form. The formal definition of STL is not presented in this paper, and the interested reader is referred to [16], [17]. Informally, STL formulas consist of boolean connectives $\neg$ (negation),

The authors are with the Department of Mechanical Engineering, Boston University, Boston, MA 02215 {sadra,cbelta}@bu.edu

This work was partially supported by the NSF under grants CPS-1446151 and CMMI-1400167

This is author's copy. Download the published version from doi:10.1109/ALLERTON.2015.7447084

$\wedge$ (conjunction), $\vee$ (disjunction), and bounded-time temporal operators $\mathcal{U}_{[a,b]}$ (until between a and b), $\Diamond_{[a,b]}$ (eventually between a and b) and $\Box_{[a,b]}$ (always between a and b) that operate on a finite set of numerical predicates over the underlying signals. In discrete time setting, we assume that interval bounds a and b are nonnegative integers, $b > a$.

Remark 1: STL was originally developed for monitoring dense-time (continuous-time) signals. The reason for restriction of control synthesis to discrete time systems is translating bounded time model checking to mixed integer constraints by using a finite number of integers, a method that has been developed in [14]. It should be noted that while discrete time approximations may be used to resemble a continuous time system, validity of a STL formula in discretized time, in general, does not indicate validity of the formula in the original continuous system.

Basically, an STL specification for a control system is intended to require certain behavior from the system. The predicates determine thresholds for functions of state and control values. For example, $(x_1 + x_2 \geq 1)$, is a predicate over state values x_1 and x_2. In this paper, each predicate, μ_i, is written in the form of:

$$\mu_i := (y_i[t] \geq 0) \qquad i = 1, \cdots, p, \tag{3}$$

where the set of outputs $y = (y_1, \cdots, y_p)^T \in \mathbb{R}^p$ are required to be linear functions over the state and control in the form of (2), and p is the number of predicates. Matrices C, D and vector e are appropriately defined with respect to the specification predicates. For instance, $(x_1 + x_2 \geq 1)$ corresponds to output $y = x_1 + x_2 - 1$ where the predicate appearing in the STL formula is $(y \geq 0)$. In this case, $C = (1\ 1), D = 0$ and $e = -1$.

Following the terminology from [17], we refer to the state and control as *primary* signals and to the scalars y_i as *secondary* signals. Throughout the terminology used in this paper, the specification, which naturally is over the state and control, is written over the secondary signals. The secondary signals notion provides a consistent and convenient format for analyzing the STL constraints later in the paper.

The validity of an STL formula is a function of the secondary signals. STL quantitive semantics defines a function called *robustness* that associates a scalar for the quality of satisfaction. The robustness function is recursively defined as follows [17]:

$$\begin{aligned}
&\rho^{\mu}_{y}[t] = y[t],\\
&\rho^{\neg\varphi}_{y}[t] = -\rho^{\varphi}_{y}[t],\\
&\rho^{\varphi\vee\psi}_{y}[t] = \max(\rho^{\varphi}_{y}[t], \rho^{\psi}_{y}[t]),\\
&\rho^{\varphi\wedge\psi}_{y}[t] = \min(\rho^{\varphi}_{y}[t], \rho^{\psi}_{y}[t]),\\
&\rho^{\Diamond_{[a,b]}\varphi}_{y}[t] = \max_{t'\in[t+a,t+b]} \rho^{\varphi}_{y}[t'],\\
&\rho^{\Box_{[a,b]}\varphi}_{y}[t] = \min_{t'\in[t+a,t+b]} \rho^{\varphi}_{y}[t'],\\
&\rho^{\varphi\ \mathcal{U}_{[a,b]}\psi}_{y}[t] = \max_{t'\in[t+a,t+b]} \left(\min(\rho^{\varphi}_{y}[t'], \min_{t''\in[t,t']} \rho^{\psi}_{y}[t''])\right),
\end{aligned} \tag{4}$$

where $\mu = (y \geq 0)$ is a predicate over scalar y and φ and ψ are STL formulas. Basically, $\rho^{\varphi}_{y}[t] > 0$ indicates that the formula φ is satisfied by secondary signals starting at time t whereas negative robustness indicates violation.

Remark 2: Zero robustness, which has measure zero in the continuous domain, does not indicate satisfaction nor violation. In this paper, however, algorithms are developed such that zero robustness is considered as satisfaction.

Example 1: Consider the STL formula $\varphi = \Box_{[0,2]}\mu_1 \wedge \Diamond_{[0,3]}\mu_2$ where $\mu_i = (y_i \geq 0), i = 1, 2$. In words, this formula requires that μ_1 is always true between time zero and two, *and*, μ_2 is eventually true between time zero and three. By applying the quantitative semantics in (4), the robustness function becomes:

$$\begin{aligned}
\rho^{\varphi}_{y}[t] = \min(&\min(y_1[t], y_1[t+1], y_1[t+2]),\\
&\max(y_2[t], y_2[t+1], y_2[t+2], y_2[t+3])).
\end{aligned}$$

Suppose the values of $y_i[t], t = 0, \cdots, 5$ are given in Table I:

TABLE I

$y_1[t]$ AND $y_2[t]$ FOR EXAMPLE 1

t	0	1	2	3	4	5
$y_1[t]$	-0.5	1.5	1	1	0.8	-0.5
$y_2[t]$	3	2	0.5	-1	-1.5	-1

The robustness values are $\rho^{\varphi}_{y}[0] = -0.5$, $\rho^{\varphi}_{y}[1] = 1$, $\rho^{\varphi}_{y}[2] = 0.5$. Note that the values of $\rho^{\varphi}_{y}[t], t \geq 3$ are not computable, in general, without knowing the values of $y_i[t], t \geq 6$.

■

Not that the robustness function depends on the future values of secondary signals. In this paper, we are interested in finding controls in a receding horizon manner that evolve the system such that an STL formula φ is *globally satisfied*, in the sense that the robustness function is always nonnegative, $\rho^{\varphi}_{y}[t] \geq 0\ \forall t \in \mathbb{Z}_{\geq 0}$. Global satisfaction can also be viewed as equivalent to satisfying $\Box_{[0,\infty]}\varphi$.

Now we are ready to informally state the problem. Our primary aim is to develop a receding horizon controller that is *correct*, *robust* and *optimal*. Note that, like other finite horizon controllers, suboptimal solutions are sought since the control synthesis algorithm depends on the size of the horizon. Furthermore, an issue in constrained controllers is infeasibility that often arises in highly disturbed or underactuated systems. In this paper, when encountering infeasibility, we soften the STL predicates using slack variables. For example, $(-2 \geq 0)$ is softened as $(-2 + \zeta \geq 0)$, which is satisfied by $\zeta \geq 2$. Details on optimality criterion and predicate softening are given in Section V.

Problem 1: Given a discrete-time system (1),(2), an STL formula φ over the predicates in the form of (3), a stage cost function $J(x[t], u[t])$, find a receding horizon controller that is:

1) Correct and Robust: The STL specification φ is globally satisfied for all realizations of disturbances,
2) Optimal: cost $J(x[t], u[t])$ cumulated over the receding horizon is optimized, and

3) Minimally Violating: in case the global (finite horizon) satisfaction of φ is infeasible, find controls such that the constraints are minimally violated.

After explaining necessary details in Section IV, the problem is formulated as an optimization problem in Section V.

III. STL Receding Horizon Control

In this section, we describe the receding horizon mechanism of STL control. As mentioned earlier, the robustness function depends on the future values of the secondary signals. The time bounds of the temporal operators determine the future times of the secondary signals that are necessary to compute robustness.

Definition 1: The *horizon* length of STL formula φ, denoted by h^φ, is defined as the largest τ such that that robustness $\rho_y^\varphi[t]$ depends on $y[t+\tau]$.

The formula horizon can be recursively computed as [18]:

$$\begin{aligned} &h^\mu = 0, \\ &h^{\neg\varphi} = h^\varphi, \\ &h^{\Diamond_{[a,b]}\varphi} = h^{\Box_{[a,b]}\varphi} = b + h^\varphi, \\ &h^{\varphi\wedge\psi} = h^{\varphi\vee\psi} = \max(h^\varphi, h^\psi), \\ &h^{\varphi\mathcal{U}_{[a,b]}\psi} = b + \max(h^\varphi, h^\psi), \end{aligned} \tag{5}$$

where μ is a numerical predicate and φ, ψ are STL formulas. In discrete time, robustness at a given time is a function of h^φ steps of secondary signals in future, i.e. $\rho_y^\varphi[t]$ is a function of $y[t], y[t+1], \cdots, y[t+h^\varphi]$. For instance, the horizon length of the specification in Example 1 is 3. It should be noted that the *horizon length* of an STL formula, which follows from the terminology used in [18], should not be confused with the *horizon length* of the control sequence that is used in receding horizon control.

At a given time t, the latest robustness that is computable is $\rho_y^\varphi[t - h^\varphi - 1]$ given the history of values of secondary signals $y^{his}[t] = \{y[t-h^\varphi-1], \cdots, y[t-1]\}$. The robustness values starting from $\rho_y^\varphi[t-h^\varphi]$ depend on the values of secondary signals starting from time t that are determined by the evolution of the system. We use the system model to predict robustness values and maintain satisfaction, i.e. nonnegative robustness, of the STL specification. Let $\rho_y^\varphi[t+h_p]$ be the most distant in future robustness value that is computed using the predicted values of $y[t], \cdots, y[t+h_p+h^\varphi]$, where h_p is the *prediction horizon* that is a user chosen integer. Consecutively, at time t, we enforce the following constraints to maintain the global STL satisfaction:

$$\begin{cases} \rho_y^\varphi[t-h^\varphi] & \geq 0, \\ \rho_y^\varphi[t-h^\varphi+1] & \geq 0, \\ \vdots & \\ \rho_y^\varphi[t+h_p] & \geq 0. \end{cases} \tag{6}$$

Note that when starting the control software, while $t \leq h^\varphi$, the set of constraints begin from $\rho_y^\varphi[0]$. The horizon of predictions for secondary signals is $H = h^\varphi + h_p$ and at each time, the control sequence that is searched for is:

$$u^H[t] = \{u^t[t], u^t[t+1], \cdots, u^t[t+H]\}. \tag{7}$$

Note that only the current time control is applied to the system and according to the new measurements, a new control sequence is found at next time step.

Remark 3: In case of $D = 0$ in (2), i.e. secondary signals only functions of state, we can reduce the number of constraints and variables for faster computation. At time t, the value of $y[t]$ is already determined hence $\rho_y^\varphi[t-h^\varphi]$ is fixed. Therefore, the set of constraints can be written as $\rho^\varphi[t-h^\varphi+1] \geq 0, \cdots, \rho^\varphi[t+h_p] \geq 0$ and since $y[t+H]$ is not dependent on $u^t[t+H]$, the finite horizon control sequence (7) is $u^H[t] = \{u^t[t], u^t[t+1], \cdots, u^t[t+H-1]\}$.

The following theorem establishes closed-loop soundness of the receding horizon controller.

Theorem 1: If $u^H[t]$ is found for all $t \in \mathbb{Z}_{\geq 0}$ such that (6) is satisfied, then the evolution of the system from applying the closed loop control sequence

$$u^0[0], u^1[1], \cdots, u^t[t], \cdots$$

globally satisfies φ.

Proof: At time t, constraint $\rho_y^\varphi[t-h_p] \geq 0$ is satisfied and $\rho_y^\varphi[t-h_p]$ is fixed since it is not dependent on the further evolution of the system. By applying $u^t[t]$, $\rho_y^\varphi[t-h_p]+1 \geq 0$ is guaranteed since $u^t[t]$ was found such that all constraints in (6) were satisfied. By induction, $\rho_y^\varphi[t] \geq 0 \ \forall t \in \mathbb{Z}_{\geq 0}$ and the specification φ is hence globally satisfied. ■

The theorem above does not state that $u^H[t]$ satisfying (6) always exist. Intuitively, larger values of prediction h_p may result in better performance due to longer horizon feasibility. However, in uncertain systems, larger horizon robust controllers are usually excessively conservative, and even infeasible since worst case predictions are made for a longer time. We address infeasibility issues in Sec. V.

Remark 4: The closed-loop synthesis approach of this paper differs from the approach in [15]. In the mentioned work, at each time step the control sequence $u^{2h^\varphi}[t]$ is found such that the set of constraints $\rho_y^\varphi[t] \geq 0, \rho_y^\varphi[t+1] \geq 0, \cdots \rho_y^\varphi[t+h^\varphi] \geq 0$ are satisfied and controls $u^t[t], \cdots, u^t[t+h^\varphi-1]$ are fixed by the values found in the previous iterations. Therefore, the synthesis algorithm is based on maintaining STL satisfaction in future whereas the past robustness values are fixed since the control values $u^t[t], \cdots, u^t[t+H-1]$ are not updated anymore, even though they are not yet applied to the system. The control strategy also involves a transient phase where control inputs for initial steps are computed by a slightly different algorithm. The receding horizon scheme in this paper, on the other hand, is based on both past and future satisfaction maintenance where by measuring the current state and storing the history of the system, the control sequence starting at the current time is updated at each time step. Therefore, our approach is more appropriate for online implementation since in the presence of uncertainties, given online measurements, the controller is able to find solutions that the control sequence stitching approach in [15] may not.

IV. Robust STL Satisfaction

In this section, we explain our approach to construct a robust MPC mechanism to find controls such that the system evolution satisfies the STL constraints for all possible realizations of disturbances. First, we introduce the notion of positive normal form STL which is important to characterize the behavior of STL constrains with respect to the changes in the secondary signals values. Next we provide the technical details on our approach to STL robust control.

Notation For two same-length vectors a and b, inequalities such as $a \geq b$ are interpreted element-wise. The notation $\underline{1}$ stands for appropriately sized vector of all ones. The notation $\otimes$ denotes the Kronecker product.

A. Positive Normal Form STL

The robustness function constructed from the quantitative semantics in (4) is a $\min / \max$ function that is piecewise linear and, in general, non-convex. The only scaling operation that appears in the quantitative semantics is multiplication by -1 from applying the semantics for negation operator.

Definition 2: An STL formula φ is in *positive normal form* if its robustness is a non-decreasing function with respect to the secondary signals, i.e. if y_1 and y_2 are two secondary signals such that $y_1[\tau] \geq y_2[\tau]\ \forall \tau \in [t, t+h^{\varphi}]$, then $\rho^{\varphi}_{y_1}[t] \geq \rho^{\varphi}_{y_2}[t]$.

By writing the STL specification in positive normal form, the constraints become monotonic with respect to the values of secondary signals, which enables us to characterize the bounds of propagation of uncertainties into STL constraints. Furthermore, in Section V, we exploit the positive normal form structure to soften the STL constraints by the means of addition of slack variables to the secondary signals.

Proposition 1: An STL specification that does not contain negation, i.e. only consisting of $\wedge, \vee, \mathcal{U}_{\mathcal{I}}, \Diamond_{\mathcal{I}}, \Box_{\mathcal{I}}$ where $\mathcal{I}$ is a time interval, is in positive normal form.

Proof: (sketch) By excluding the negation operator, the robustness function consists only of $\min$ and $\max$ operators over the secondary signal values without sign change. Therefore, an increase in secondary signal values will either increase robustness, or does not alter robustness. ■

Proposition 2: Every STL formula can be transformed into positive normal form.

Proof: We explain how to recursively transform a general STL formula to positive normal form using negation propagation into numerical predicates and modifying the sign of secondary signals:

1) Negation recursively propagates into numerical predicates:

$$\begin{aligned}
&\neg(\varphi_1 \wedge \varphi_2) = \neg\varphi_1 \vee \neg\varphi_2,\\
&\neg(\varphi_1 \vee \varphi_2) = \neg\varphi_1 \wedge \neg\varphi_2,\\
&\neg(\varphi_1 \mathcal{U}_{[a,b]} \varphi_2) = \neg\varphi_1 \mathcal{R}_{[a,b]} \neg\varphi_2,\\
&\neg\Diamond_{[a,b]}\varphi = \Box_{[a,b]}\neg\varphi,\\
&\neg\Box_{[a,b]}\varphi = \Diamond_{[a,b]}\neg\varphi,
\end{aligned}$$

where $\mathcal{R}$ is the *release* operator defined as $\varphi_1 \mathcal{R}_{[a,b]} \varphi_2 = \Box_{[a,b]}\varphi_2 \vee (\varphi_2 \mathcal{U}_{[a,b]}(\varphi_1 \wedge \varphi_2))$.

2) Negation of predicates: If $\neg\mu_i$ appears in the STL formula where $\mu_i = (y_i \geq 0)$ that $y_i = c_i x + d_i u + e_i$ is a secondary signal:
   - If only $\neg\mu_i$ appears in the STL formula: Redefine $y_i = -c_i x - d_i u - e_i$, thus $\neg$ is removed.
   - If both μ_i and $\neg\mu_i$ appear: Introduce new secondary signal $y_j = -c_i x - d_i u - e_i$, thus $\neg\mu_i$ is replaced by the new predicate $\mu_j = (y_j \geq 0)$.

■

Note that, in the worst case, the number of secondary signals is doubled during the construction of the positive normal form STL formula. As explained in Section V, the computational complexity of control synthesis is exponential with respect to the number of secondary signals.

B. Robust Prediction System

At time t, given the control sequence $u^H[t] = (u^t[t]^T, \cdots, u^t[t+H]^T)^T$ and the uncertain input sequence $w^H[t] = (w^t[t]^T, \cdots, w^t[t+H-1]^T)^T$, the prediction for secondary signals $y^H[t] = (y^t[t]^T, \cdots, y^t[t+H]^T)^T$ is: [1]

$$y^H[t] = \Phi_0^H x[t] + \Phi_1^H u^H[t] + \Phi_2^H w^H[t] + \underline{1} \otimes e, \qquad (8)$$

where the flow matrices Φ_0, Φ_1, Φ_2 are given by:

$$\Phi_0^H = \begin{pmatrix} C \\ CA \\ \vdots \\ CA^H \end{pmatrix},$$

$$\Phi_1^H = \begin{pmatrix} D & 0 & & \cdots & 0 \\ CB & D & & \cdots & 0 \\ CAB & CB & & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ CA^{H-1}B & CA^{H-2}B & \cdots & CB & D \end{pmatrix},$$

$$\Phi_2^H = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ C & 0 & \cdots & \vdots \\ CA & C & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ CA^{H-1} & CA^{H-2} & \cdots & C \end{pmatrix}.$$

Since the uncertain inputs belong to the polytope set $\mathcal{W}$, the admissible set of $y^H[t]$ is also a polytope in the finite horizon secondary signals space:

$$\mathcal{Y}_{u^H[t]} = \left\{ y^H[t] \middle| w^t[\tau] \in \mathcal{W}, \tau = t, \cdots, t+H-1 \right\}, \quad (9)$$

which consists of *uncertainty image* set, $\left\{ \Phi_2 w^H[t] \middle| w^t[\tau] \in \mathcal{W} \right\}$, which can be computed beforehand, plus an affine map of the control sequence $\Phi_0^H x[t] + \Phi_1^H u^H[t] + \underline{1} \otimes e$. The finite horizon robust prediction problem is finding controls $u^H[t]$ such that the set of constraints (6) is satisfied for all points in $\mathcal{Y}_{u^H[t]}$. Since the robustness function is non-convex, its

[1] With a slight abuse of notation, $w^H[t]$,$u^H[t]$ and $y^H[t]$ are interchangeably used for the described sequences and their corresponding representations as column vectors.

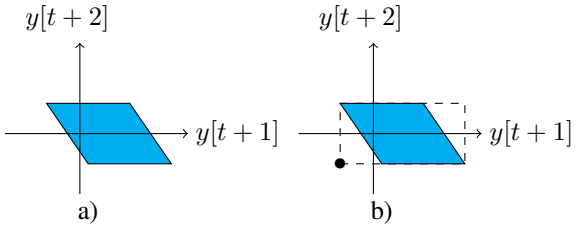


Fig. 1. a) An example representation of the admissible set of $y^H[t]$ in the finite horizon secondary signals space. b) The lower left corner of the axis-aligned minimum bounding box of the set.

extreme values do not necessarily lie on the vertices of $\mathcal{Y}_{u^H[t]}$. For example, consider a simple robustness function $\rho^{\varphi}_{y} = \max(y[t+1], y[t+2])$ and assume that for some $u^H[t]$, the set $\mathcal{Y}_{u^H[t]}$ is the shaded parallelogram illustrated in Fig. 1 a). It is observed that even though all the vertices of the parallelogram are in the positive robustness region, i.e. the first, second and fourth quadrants in Fig. 1 a), a small section lies in the third quadrant which corresponds to negative robustness.

In order to maintain robust satisfaction, we enforce the constraints (6) at a single point that is the lower left corner of the axis-aligned minimum bounding box of the uncertainty image set (See Fig. 1 b) for an illustration). As stated in Theorem 2 later in the paper, with STL formula being in positive normal form, the robustness is guaranteed to be greater or equal to the robustness at the lower left corner of the box since the secondary signals of the image are greater element-wise. Therefore, the robust *prediction* system is constructed based on the mentioned lower left corner. Note that this approach is, in general, conservative yet computationally manageable as the set of constraints (6) are imposed for a single point.

Definition 3: The lower left corner of the axis-aligned minimum bounding box of a bounded set $\mathcal{S} \subset \mathbb{R}^n$ is denoted by $\Omega(\mathcal{S})$, where the i'th element is given by:

$$\Omega_i(\mathcal{S}) = \inf_{s \in \mathcal{S}} \quad s_i, \ i = 1, \cdots, n. \tag{10}$$

A polytope set $\mathcal{P} \subset \mathbb{R}^n$ can be represented by the convex hull of its vertices. For a given polytope $\mathcal{P}$, we define the matrix P whose columns contain its vertices.

Definition 4: The function $\omega : \mathbb{R}^{n \times m} \to \mathbb{R}^n$ maps a $n \times m$ matrix to a n-dimensional vector where the i'th element is:

$$\omega_i(P) = \min_{j} \quad P_{ij}. \tag{11}$$

In words, the i'th element of the vector is the minimum value in the i'th row of the matrix.

Lemma 1: Let $\mathcal{P}$ be a polytope and matrix P whose columns contain its vertices. Then:

$$\Omega(\mathcal{P}) = \omega(P). \tag{12}$$

Proof: The value of i'th element is the solution to the following linear program:

$$\begin{aligned} \Omega_i(\mathcal{P}) = \min \quad & \sum_j P_{ij}\lambda_{jk}, \\ s.t. \quad & 0 \le \lambda_{jk} \le 1, \\ & \sum_k \lambda_{jk} = 1. \end{aligned}$$

It is straightforward to verify that the solution is:

$$\lambda_{jk} = \begin{cases} 1 & j = argmin\ P_{ij}, \\ 0 & otherwise. \end{cases}$$

Therefore, $\Omega_i(\mathcal{P}) = \omega_i(P)$ holds element-wise. ■

Lemma 2: If the open-loop control sequence $u^H[t]$ is applied to the system (1),(2), the following relation holds:

$$y^{\omega,H}[t] \le y^H[t], \tag{13}$$

where

$$y^{\omega,H}[t] = \Phi_0 x[t] + \Phi_1 u^H[t] + \omega\left(\Phi_2(\underline{1} \otimes W)\right) + \underline{1} \otimes e, \tag{14}$$

where W is the matrix whose columns are given by the polytope $\mathcal{W}$'s vertices.

Proof: The proof follows from the definition of Ω function where

$$y^{\omega,H}[t] = \Omega\left(\left\{\mathcal{Y}_{u^H[t]}\right\}\right) \le y^H[t]. \tag{15}$$

By applying Lemma 1 to (8) we arrive at (14). ■

Note that $\omega\left(\Phi_2(\underline{1} \otimes W)\right)$ is computed prior to starting the control synthesis optimization problem.

Theorem 2: Given a linear system (1), (2), STL specification φ in positive normal form and secondary signals history $y^{his}[t]$, for any control sequence $u^H[t]$ the following relations hold:

$$\begin{array}{lcl} \rho^{\varphi}_{y}[t - h^{\varphi}] & \ge & \rho^{\varphi}_{y^{pre}}[t - h^{\varphi}], \\ \rho^{\varphi}_{y}[t - h^{\varphi} + 1] & \ge & \rho^{\varphi}_{y^{pre}}[t - h^{\varphi} + 1], \\ \vdots & & \\ \rho^{\varphi}_{y}[t + h_p] & \ge & \rho^{\varphi}_{y^{pre}}[t + h_p], \end{array} \tag{16}$$

where y^{pre} is the prediction secondary signal that is composed from the stored values of y^{his} and the robust prediction values from (14).

Proof: The proof follows immediately from Lemma 2 and the definition of positive normal form STL. ■

Corollary 1: If the control sequence $u^H[t]$ is found such that the set of constraints

$$\begin{cases} \rho^{\varphi}_{y^{pre}}[t - h^{\varphi}] & \ge 0, \\ \rho^{\varphi}_{y^{pre}}[t - h^{\varphi} + 1] & \ge 0, \\ \vdots & \\ \rho^{\varphi}_{y^{pre}}[t + h_p] & \ge 0, \end{cases} \tag{17}$$

are satisfied, then the open-loop system response of $u^H[t]$ satisfies the set of original constraints (6).

Remark 5: The methodology of this paper can be easily extended to linear time variant (LTV) systems. The required modification is generalizing the flow matrices in (8) for time dependent matrices. In this case, the necessary assumption is that the time dependencies of the system matrices are known.

Remark 6: We have not assumed any restriction on the plant matrix A. In principle, an unstable A results in large entries in matrices in (8) that causes control decisions to be very conservative and may even cause infeasibility in longer horizon predictions. A well known technique in MPC literature is designing a control law in the form of $u[t] = Kx[t] + v[t]$, where K is a fixed state feedback gain. The closed loop matrix $A^{cl} = A + BK$ can be designed to possess some important properties such as stability and nilpotency (if the pair (A, B) is controllable). We have not investigated this approach since STL constraints, in general, are different from stability. We also remark that an analogous investigation of robust invariant sets [20] for STL MPC is an open problem.

V. Optimization Based Control

In the previous section, we explained our approach to the first objective of Problem 1. In this section, after explaining our approach to the formalization of the second and third objectives, i.e. optimality and minimality of violations, we formulate Problem 1 as an optimization problem. Finally we explain how to express the optimization problem as a mixed integer programming (MIP) problem that is solvable using standard solvers.

A. Optimization Problem

A performance criterion is required for selecting a control sequence from the robust open-loop control candidates $u^H[t]$. In principle, there exist two different approaches to define a cost criteria for a nondeterministic system. First, one can optimize the cost using predictions from the nominal system, where the disturbances are assumed to take a known *nominal* value. A more complicated alternative is optimizing the worst case cost that is admissible by the disturbance realizations. In this paper, we choose the nominal system cost since it is found to perform better in many classical MPC problems [20]. Furthermore, worst case cost approaches lead to optimization problems that are computationally more expensive. It should be noted that if the cost is only dependent on controls, the two approaches are identical.

We define $\hat{w}$ as the *nominal* disturbance, that may be given or may be chosen by some means such as finding the centroid of the polytope $\mathcal{W}$. Given the current state $x[t]$, the nominal system prediction is given by

$$\begin{aligned} \hat{x}[\tau+1] =& A\hat{x}[\tau] + Bu[\tau] + \hat{w}, \\ & t \le \tau \le t+H-1, \\ & \hat{x}[t] = x[t]. \end{aligned} \tag{18}$$

At time t, the finite horizon cost function is:

$$\sum_{\tau=t}^{t+H} J(\hat{x}[\tau], u[\tau]). \tag{19}$$

Effectively, within all control sequences that robustly satisfy STL constraints, we choose the one with the least finite horizon nominal evolution cost.

On the other hand, in systems with large disturbances or limited control actuation, it is possible that the STL constraints may be inevitably violated. If encountered with infeasibility, instead of terminating the control software, we find minimally violating solutions. With STL formula φ in positive normal form, a counterfeit increase in all the secondary signals values eventually restores the satisfaction of STL constraints. This method is similar to constraint softening method in [21]. We introduce softened secondary signals values as:

$$y^{his}_{soft} = y^{his} + \underline{1}\zeta, \qquad y^{\omega,H}_{soft} = y^{\omega,H} + \underline{1}\zeta, \tag{20}$$

where $\zeta \ge 0$ is the softening slack variable. Note that both robust prediction values and history values are softened to recover feasibility of (6). The artificial secondary signal composed from y^{his}_{soft} and $y^{\omega,H}_{soft}$ is denoted by y^{soft}. We desire that if the STL constraints are infeasible, ζ, the amount of softening, is minimized without optimizing the cost function. Finally, Problem 1 is formulated as the following optimization problem:

$$\begin{aligned} u^H[t] = \quad argmin \quad & \sum_{\tau=t}^{t+H} J(\hat{x}[\tau], u[\tau]) + M\zeta \\ s.t. \quad & \rho^{\varphi}_{y^{soft}}[t - h^{\varphi}] \ge 0 \\ & \vdots \\ & \rho^{\varphi}_{y^{soft}}[t + h_p] \ge 0, \\ & \text{Eqn. (14), (18), (20),} \\ & \zeta \ge 0, \end{aligned} \tag{21}$$

where M is a large penalizing positive number that unifies the separate optimization problems for cost optimality and violation minimality. In case the STL constraints are feasible, large M enforces $\zeta = 0$ and the cumulated cost is optimized. In case of infeasibility, effectively, ζ is minimized without optimization of the cumulated cost.

Proposition 3: The smallest ζ such that the constraints set in (21) is feasible is:

$$\zeta_{min} = \max\{0, -\min_{\tau=t-h^{\varphi},\cdots,t+h_p} \rho^{\varphi}_{y^{pre}}[\tau]\}. \tag{22}$$

The detailed proof is not included as it basically follows from the monotonicity of the robustness function of a positive normal form STL formula.

Remark 7: Eqn. (20) can be modified by using weights for softening different secondary signals. Multiple softening values for different secondary signals may also be used. In these cases, a practically efficient controller may require a tuning procedure to find the best softening strategy.

Remark 8: Removing the constraint $\zeta \ge 0$ results in a STL robustness maximization receding horizon policy. A negative ζ value can be viewed as *constraint tightening*, i.e. how much constraints can be tightened while keeping feasibility.

B. Mixed Integer Formulation

STL constraints can be written as mixed integer constraints. One can encode the robustness function by representing $\max$ and $\min$ operations in the quantitative semantics, Eqn. (4), by a set of mixed integer constraints. This method typically introduces a large number of integer

variables as the number of $\max / \min$ arguments may be large. An alternative approach, which is computationally more efficient, is binary encoding of quantitative semantics, which has been first introduced by the authors in [14]. In this section, we briefly explain this method.

The binary encoding is recursively executed. For a single predicate $\mu = (y \geq 0)$, a binary $z^{\mu}[t] \in \{0, 1\}$ indicates whether the predicate at time t is true, $z^{\mu}[t] = 1$, or false, $z^{\mu}[t] = 0$. The corresponding mixed integer constraints are:

$$\begin{cases} y[t] - Kz^{\mu}[t] & \leq 0, \\ y[t] + K(1 - z^{\mu}[t]) & \geq 0, \end{cases} \tag{23}$$

where K is a sufficiently large positive number. Overall, $p \times (H + h^{\varphi} + 1)$ number of binary variables is required to binary encode the secondary signals in (21). For encoding the STL formula, an additional number of variables are recursively defined as [14]:

- Conjunction: $\psi = \bigwedge_{i=1}^{m} \varphi_i$:

$$\begin{cases} z^{\psi}[t] \leq z^{\varphi_i} \\ z^{\psi}[t] \geq 1 - m + \sum_{i=1}^{m} z^{\varphi_i}[t] \end{cases} \tag{24}$$

- Disjunction: $\psi = \bigvee_{i=1}^{m} \varphi_i$:

$$\begin{cases} z^{\psi}[t] \geq z^{\varphi_i}[t] \\ z^{\psi}[t] \leq \sum_{i=1}^{m} z^{\varphi_i}[t] \end{cases} \tag{25}$$

- Eventually $\psi = \Diamond_{[a,b]} \varphi$

$$z^{\psi}[t] = \bigvee_{\tau=t+a}^{t+b} z^{\varphi_i}[\tau] \tag{26}$$

- Always $\psi = \Box_{[a,b]} \varphi$

$$z^{\psi}[t] = \bigwedge_{\tau=t+a}^{t+b} z^{\varphi_i}[\tau] \tag{27}$$

- Until $\psi = \varphi_1 \mathcal{U}_{[a,b]} \varphi_2$

$$z^{\psi}[t] = \bigvee_{\tau=t+a}^{t+b} (z^{\varphi_2}[\tau] \wedge \bigwedge_{\tau'=t}^{\tau} z^{\varphi_1}[\tau']) \tag{28}$$

Note that each $z^{\psi} \in [0, 1]$ is not required to be declared as an integer since is automatically enforced to take values from 0 or 1. Finally, the set of constraints (6) becomes binary encoded as:

$$\begin{cases} z^{\varphi}[t - h^{\varphi}] & = 1, \\ z^{\varphi}[t - h^{\varphi} + 1] & = 1, \\ \vdots & \\ z^{\varphi}[t + h_p] & = 1. \end{cases} \tag{29}$$

Depending on the cost function J, the optimization problem is a mixed integer linear programming (MILP) (in case of linear J), a mixed integer quadratic programming (MIQP) (in case of quadratic J) or a mixed integer nonlinear programming (MINLP) (in case of nonlinear J). Mixed integer programs are exponentially expensive with respect to the number of integer variables, therefore the real time applications of STL MPC are restricted to small systems.

VI. Case Study

We consider a linear system in the form (1), with

$$A = \begin{pmatrix} 1 & 0.5 \\ 0 & 0.8 \end{pmatrix}, B = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

The two dimensional state is $x[t] = (x_1[t], x_2[t])^T$ and control input is a scalar. This system represents a double integrator with energy loss, a type of system which is encountered in many engineering problems. The disturbance $w[t]$ is bounded to the two dimensional box $\|w[t]\|_{\infty} \leq w_0$, where w_0 is assigned multiple values as explained further. The stage cost function is $J(u[t]) = |u[t]|$, which penalizes the control effort.

We desire a STL specification that enforces x_1 to oscillate between $2 \leq x_1 \leq 4$ and $-4 \leq x_1 \leq -2$, with each interval being visited at least once within any five consecutive time steps. The specification, written in positive normal form is:

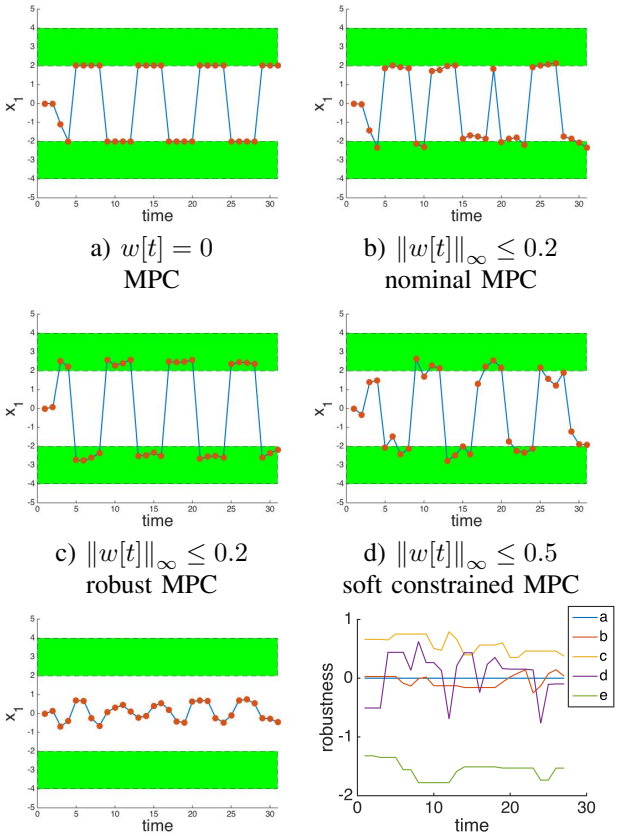
$$\begin{aligned} \varphi = &\left(\Diamond_{[0,4]}((y_1 \geq 0) \wedge (y_2 \geq 0))\right) \wedge \\ &\left(\Diamond_{[0,4]}((y_3 \geq 0) \wedge (y_4 \geq 0))\right), \end{aligned}$$

for which the corresponding matrices from (2) are:

$$C = \begin{pmatrix} 1 & 0 \\ -1 & 0 \\ -1 & 0 \\ 1 & 0 \end{pmatrix}, D = 0, e = \begin{pmatrix} -2 \\ 4 \\ -2 \\ 4 \end{pmatrix}.$$

We chose $h_p = 2$, which makes $H = h^{\varphi} + 2 = 6$. The initial state values are $x_1[0] = x_2[0] = 0$. The control admissible set is initially assigned as $|u| \leq 20$. We formulate the optimization problem given in (21) as a MILP and find solutions using the MATLAB standard optimization toolbox MILP solver. The assigned value of M in (21) is 10^5. We simulate the system for 30 time steps. The solution of each step takes less than 0.1s using a 2.8 GHz core i5 processor on an iMac computer. The uncertain values $w[t]$ are drawn randomly from a uniform distribution on $\mathcal{W}$. We investigated the following scenarios:

1) We observe that if the system was fully deterministic, $w_0 = 0$, the optimal-correct solution oscillates between $x_1 = 2$ and $x_1 = 4$ as illustrated in Fig. 2 a). The solution does not enter the mentioned regions as it is unnecessary and is associated with a larger control effort. The robustness function is always zero for this solution.
2) We consider a disturbed system $w_0 = 0.2$. We observe that the nominal MPC, i.e. neglecting the presence of disturbances, fails to satisfy the specification. The trajectory is shown in Fig. 2 b) and the robustness function values in this case are occasionally negative.
3) We now implement the robust MPC introduced in this paper. It is observed that the robust solution enters the regions to conservatively maintain STL satisfaction (See Fig. 2 c)). The robustness function is always above zero for this case.
4) We broaden the disturbances set to $\|w[t]\|_{\infty} \leq 0.5$. The controller fails to find a robust solution to this



a) $w[t] = 0$ MPC
b) $\|w[t]\|_\infty \leq 0.2$ nominal MPC
c) $\|w[t]\|_\infty \leq 0.2$ robust MPC
d) $\|w[t]\|_\infty \leq 0.5$ soft constrained MPC
e) $|u[t]| \leq 2$ minimally violating MPC
f) STL robustness function values

Fig. 2. Case study results

scenario, see Fig. 2 d), thus constraint softening is required. This is particularly due to the long horizon considered where the worst case predictions cause infeasibility. We observed that by decreasing the horizon length to $H = 5$ and $H = 4$, better solutions were found (results not shown).

5) We now limit the admissible control set to $|u| \leq 2$ with disturbances from the set $\|w[t]\|_\infty \leq 0.2$. It is impossible to satisfy STL constraints with such a limited control set. The implementation result, shown in Fig. 2 e), does not satisfy the STL specification, but nevertheless, the observed trajectory is maximally oscillating between the two regions in order to minimally violate the specification.

The robustness values as a function of time are shown in Fig. 2 f) for the five different scenarios.

VII. Conclusion and Future Work

In this paper, we focused on the connection between optimality and correctness for discrete time linear systems with additive bounded disturbances. Specifically, we proposed a model predictive control approach for the case when the correctness specifications are given as signal temporal logic formulas. We plan to extend these results to classes of discrete-time piecewise affine systems for which the monotonicity property stated in this paper holds, and apply them to controlling traffic networks. We are also working on extending this work to distributed model predictive control, where the additive disturbances of the component subsystems are used in assume-guarantee reasoning schemes. We believe that such techniques have the potential to impact temporal logic optimal control of large networks.

References

[1] C. E. Garcia, D. M. Prett, and M. Morari, "Model predictive control: theory and practice," *Automatica*, vol. 25, no. 3, pp. 335–348, 1989.

[2] E. F. Camacho and C. Bordons, *Model Predictive Control*. Springer Science & Business Media, 2010, vol. 57, no. 1.

[3] M. V. Kothare, V. Balakrishnan, and M. Morari, "Robust constrained model predictive control using linear matrix inequalities," *Automatica*, vol. 32, no. 10, pp. 1361–1379, Oct. 1996.

[4] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, June 2000.

[5] C. Baier and J.-P. Katoen, *Principles Of Model Checking*. The MIT Press, 2008, vol. 950.

[6] 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, no. 1, pp. 287–297, 2008.

[7] H. Kress-Gazit, G. E. Fainekos, and G. J. Pappas, "Temporal Logic based Reactive Mission and Motion Planning," *IEEE Transactions on Robotics*, vol. 25, no. 6, pp. 1370–1381, 2009.

[8] P. Tabuada and G. Pappas, "Linear Time Logic Control of Discrete-Time Linear Systems," *IEEE Transactions on Automatic Control*, vol. 51, no. 12, pp. 1862–1877, 2006.

[9] T. Wongpiromsarn, U. Topcu, and R. M. Murray, "Receding horizon control for temporal logic specifications." in *Hscc*. ACM, 2010, pp. 101–110.

[10] E. Aydin Gol, M. Lazar, and C. Belta, "Temporal logic model predictive control," *Automatica*, vol. 56, pp. 78–85, June 2015.

[11] a. Bemporad and M. Morari, "Control of systems integrating logics, dynamics, and constraints," *Automatica*, vol. 35, no. 3, pp. 407–427, 1999.

[12] S. Karaman, R. Sanfelice, and E. Frazzoli, "Optimal control of Mixed Logical Dynamical systems with Linear Temporal Logic specifications," in *2008 47th IEEE Conference on Decision and Control*. IEEE, 2008, pp. 2117–2122.

[13] E. M. Wolff, U. Topcu, and R. M. Murray, "Optimization-based Control of Nonlinear Systems with Linear Temporal Logic Specifications," in *Proc. of the International Symposium on Robotics Research (ISRR)*, 2014.

[14] 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 *CDC*, 2014.

[15] 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.

[16] 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.

[17] A. Donzé and O. Maler, *Robust satisfaction of temporal logic over real-valued signals*. Springer, 2010, vol. 6246 LNCS.

[18] A. Dokhanchi, B. Hoxha, and G. Fainekos, "On-Line Monitoring for Temporal Logic Robustness," in *Runtime Verification*. Springer, 2014, pp. 1–20.

[19] D. Bertsimas, J. N. Tsitsiklis, and J. Tsitsiklis, *Introduction to Linear Optimization*. Athena Scientific Belmont, MA, 1997, vol. 6.

[20] A. Bemporad and M. Morari, "Robust model predictive control: A survey," in *Robustness in identification and control*. Springer, 1999, vol. 245, pp. 207—-226.

[21] E. C. Kerrigan and J. M. Maciejowski, "Soft constraints and exact penalty functions in Model Predictive Control," in *Control 2000 Conference*, 2000.