Safety-aware Model Predictive Path Integral Control with Signal Temporal Logic

Yiqi Zhao[1], Taekyung Kim[2], Hideki Okamoto[3], Bardh Hoxha[3], Jyotirmoy V. Deshmukh[1], Lars Lindemann[4], Georgios Fainekos[3]

***Abstract*— Safety-aware motion planning remains a challenge in robotics, especially when missions are time-critical and are under complex specifications. In this paper, we propose safety-aware-stl-mppi, a computationally efficient sampling-based receding-horizon planning framework designed to promote satisfaction of constraints expressed in Signal Temporal Logic (STL). Our approach encodes discrete-time STL formulas into candidate time-varying control barrier functions (CBF), which are integrated into a model predictive path integral (MPPI) controller. Our method inherits the benefits of low computational cost from an efficiently parallelizable sampling based planner and utilizes CBF for constraints expressed in STL. We compare against several MPPI baselines using four artificial Mars Rover planning case studies with a diverse environment and cost setups, where we show our method consistently achieving high safety and efficiency. We show a quadcopter planning experiment with NVIDIA Isaac Lab.**

I. Introduction

Signal temporal logic (STL) [1], [2] arises as a commonly-used formal language that can express a rich set of real-time spatio-temporal safety and mission level requirements. For instance, STL has been used as a specification language for the responsibility sensitive safety model [3] for autonomous vehicles. Planning in STL with rigorous safety guarantees has been studied in [4], however with a high computational cost from solving mixed integer programs. Realizing this, [5] proposes control barrier functions for STL specifications over continuous-time systems, which lead to convex programs given quadratic cost functions. Encodings for discrete-time stochastic systems have been studied in [6]. In real-life applications, however, cost functions are generally nonconvex. Realizing the computational difficulty in such planning problems, sampling-based planners such as Model Predictive Path Integral are studied for general motion planning.

Model predictive path integral (MPPI) has been proposed [7], [8] to solve optimal control problems by sampling the control inputs from a normal distribution and executing a weighted average over the rollouts based on performances. MPPI is easily parallelizable and has optimality guarantees [9]. The works in [10]–[12] took the first step in using STL specifications as the cost function in MPPI design. Ensuring satisfaction of general STL formulas remains unresolved.

In this paper, we view safety as satisfaction of a user-defined STL formula. We propose **safety-aware-stl-mppi** (Figure 1), a safety-aware sampling-based receding-horizon control strategy under STL specification with general cost. Particularly, given a specification in STL, we first construct a time-varying control barrier function adapting the approach of [5]. We incorporate the CBF in weighting the samples from our planner and design an efficient projection of the MPPI weighted rollout onto the CBF zero-superlevel set to promote STL satisfaction. We summarize our contributions:

- We propose safety-aware-stl-mppi, a sampling-based receding-horizon planner that incorporates STL specifications as CBF-informed constraints while minimizing a user-defined cost function.
- We adapt the time-varying control barrier function encoding for STL formulas from [5] to discrete-time and design a lightweight projection toward satisfaction of the associated CBF constraints.
- We present numeric case studies and a quadcopter simulation in Section V. The empirical results show that our algorithm achieves high safety, not attained with other MPPI baselines. We additionally demonstrate the computational efficiency of our approach. We release codes associated with the algorithm at https://zhaoy37.github.io/safety-aware-stl-mppi/.

A. Related Literature

Planning with Temporal Logic. Temporal logics are extensions of propositional and predicate logics to describe timed system behaviors. Linear temporal logic (LTL) [13] and STL [1], [2] are rich specification languages widely used for planning. Abstraction-based temporal logic planning [14], [15] constructs discrete abstraction over systems and uses graph-based planning. Optimization-based approaches use mixed integer programs [4], which are expensive to run online. Control barrier functions are used to encode STL specifications [5], leading to convex programs when the cost functions are restricted to be convex. Computationally efficient sampling-based methods are proposed in [16], [17]. Recently, diffusion models [18], vision-language-action model based approaches [19], and neural predictive models are used in [20] for STL-based trajectory generation and planning but without safety guarantee. Different from these,

[1]Yiqi Zhao and Jyotirmoy V. Deshmukh are with the Thomas Lord Department of Computer Science, University of Southern California, Los Angeles, CA 90089, USA. {yiqizhao, jdeshmuk}@usc.edu

[2]Taekyung Kim is with the Department of Robotics, University of Michigan, Ann Arbor, MI 48109, USA. taekyung@umich.edu

[3] Hideki Okamoto, Bardh Hoxha, and Georgios Fainekos are with the Toyota Motor North America R&D, Ann Arbor, MI 48105, USA. {hideki.okamoto, bardh.hoxha, georgios.fainekos}@toyota.com

[4]Lars Lindemann is with the Automatic Control Laboratory, ETH Zürich, Zürich, Switzerland. llindemann@ethz.ch

This work was primarily carried out during the summer internships of Yiqi Zhao and Taekyung Kim at Toyota Motor North America R&D.

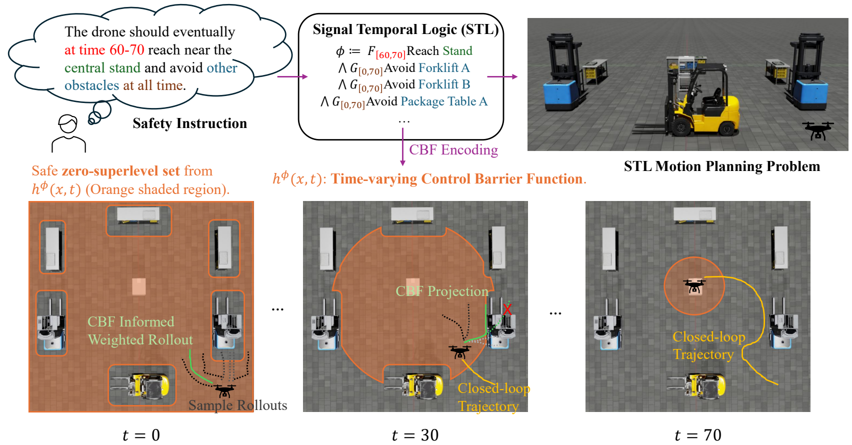


Fig. 1: Safety-aware-stl-mppi: The top row highlights the STL motion planning problem with an STL formula ϕ. Our solution is on the second row, where we rely on a time-varying CBF h^{ϕ}, which encodes the region consistent with the STL constraint with its time-varying zero-super level set (shown in orange). We plan with MPPI using CBF informed weighted rollout, and project any weighted rollout (dashed green) violating the CBF into the safety region (solid green). The planning works in receding-horizon with a closed-loop trajectory (in yellow) satisfying ϕ. The drawings in the figure are hypothetical. We show drawings in a case study in Section V.

we focus on addressing satisfaction in STL on an MPPI planner with customized cost functions that can differ from the STL constraints.

Model Predictive Path Integral. Model predictive path integral (MPPI) was proposed in [7], [8] as a sampling-based receding-horizon controller. Reach-avoid specifications are considered in [12], [21] where time-invariant barrier functions are considered to ensure safety. Path-integral based offline policy learning methods in continuous-time are proposed in [22]–[24] for STL tasks but are not designed to be optimized in closed-loop and guarantee satisfaction. Differentiable STL robustness is used as the objective of a Stein variational path-integral optimizer for long-horizon open-loop trajectory synthesis [25]. The work closest to ours are [10], [11], [26]–[28], which augment STL satisfaction as part of costs in MPPI rollouts. Existing cost-only MPPI methods do not provide a direct connection to a formal sufficient condition for STL satisfaction. Our method is grounded in an exact STL-CBF condition, while its nominal real-time implementation approximates this condition. We introduce the baselines in detail in Section V.

II. Preliminaries

Consider a control system of the form $x_{t+1} = f(x_t, u_t) := f_0(x_t) + g(x_t)u_t$ with $t \in \mathbb{Z}_{\geq 0}$, where $x_t \in \mathbb{R}^n$ and $u_t \in \mathcal{U} \subseteq \mathbb{R}^m$ are the state and input with actuation limit $\mathcal{U}$, assumed to be box constraints (i.e., the constraints are in the form $e_i^T u_t \leq u_{\max,i}$ or $-e_i^T u_t \leq -u_{\min,i}$, where $i \in \{1, \dots, m\}$, e_i is the i-th Euclidean basis vector, and $u_{\min,i}, u_{\max,i} \in \mathbb{R}$). Assume $f_0 : \mathbb{R}^n \to \mathbb{R}^n$ and $g : \mathbb{R}^n \to \mathbb{R}^{n \times m}$ are Lipschitz continuous and differentiable.

We consider in this paper a fragment of signal temporal logic (STL) specifications following [5]. We refer readers interested in the general syntax and semantics to [1], [29]. The STL syntax is defined recursively as follows

$$\begin{aligned} &\varphi := \top \mid \pi^{\mu} \mid \neg\pi^{\mu} \mid \varphi_1 \wedge \varphi_2, \\ &\phi := G_{[a,b]}\varphi \mid F_{[a,b]}\varphi \mid \varphi_1 U_{[a,b]} \varphi_2 \mid \phi_1 \wedge \phi_2 \mid \varphi \end{aligned} \tag{1}$$

where φ_1 and φ_2 are of type φ and ϕ_1 and ϕ_2 are of type ϕ. We denote the Boolean True and False with $\top$ and $\bot$ respectively. We emphasize that the atomic elements of an STL formula are the predicates $\pi^{\mu} : \mathbb{R}^n \to \{\top, \bot\}$, which are defined via a predicate function $\mu : \mathbb{R}^n \to \mathbb{R}$ such that $\pi(x_t) := \top$ if $\mu(x_t) \geq 0$ and $\pi(x_t) := \bot$ otherwise. Let us define $\sigma(\tau_1, \tau_2) := [\tau_1, \tau_2] \cap \{0, \dots, H\}$, where we assume H is sufficiently large to capture the finite-length specification (see [30] for computing lower bounds of H). We say that x satisfies an STL formula ϕ at time τ_0 if $(x, \tau_0) \models \phi$ and

$$\begin{aligned} (x, \tau_0) \models \pi^{\mu} &\Leftrightarrow \mu(x_{\tau_0}) \geq 0, \\ (x, \tau_0) \models \neg\pi^{\mu} &\Leftrightarrow \neg((x, \tau_0) \models \pi^{\mu}), \\ (x, \tau_0) \models \varphi_1 \wedge \varphi_2 &\Leftrightarrow (x, \tau_0) \models \varphi_1 \wedge (x, \tau_0) \models \varphi_2, \\ (x, \tau_0) \models G_{[a,b]}\varphi &\Leftrightarrow \forall \tau \in \sigma(\tau_0 + a, \tau_0 + b), (x, \tau) \models \varphi, \\ (x, \tau_0) \models F_{[a,b]}\varphi &\Leftrightarrow \exists \tau \in \sigma(\tau_0 + a, \tau_0 + b), (x, \tau) \models \varphi, \\ (x, \tau_0) \models \varphi_1 U_{[a,b]} \varphi_2 &\Leftrightarrow \exists \tau_1 \in \sigma(\tau_0 + a, \tau_0 + b), (x, \tau_1) \models \varphi_2 \\ &\quad \wedge \forall \tau_2 \in \sigma(\tau_0, \tau_1), (x, \tau_2) \models \varphi_1, \\ (x, \tau_0) \models \phi_1 \wedge \phi_2 &\Leftrightarrow (x, \tau_0) \models \phi_1 \wedge (x, \tau_0) \models \phi_2. \end{aligned}$$

We assume in this paper that $a, b \geq 0$. STL has a robust semantics $\rho^{\phi}(x, \tau_0) \in \mathbb{R}$ which measures how robustly a formula ϕ is satisfied by x at τ_0. Importantly, $(x, \tau_0) \models \phi$ if $\rho^{\phi}(x, \tau_0) > 0$ [2]. We refer the readers to the definition of robust semantics in [2], [31].

III. Problem Formulation

In receding-horizon control, we are given a user-defined planning horizon $\mathcal{H} \leq H - t$. We are also given user-defined cost functions $q : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ denoting the stage cost and $E : \mathbb{R}^n \to \mathbb{R}$ denoting the terminal cost. At each current time $t \in \{0, \dots, H-1\}$ during planning, we are asked to find an optimal sequence of control inputs $v_t^{*,(t)}, \dots, v_{t+\mathcal{H}-1}^{*,(t)}$ via

Formula	Encoding Rule (which we follow when constructing the CBF)
π^μ	$\forall x \in D, h^{\pi^\mu}(x,\tau_0) \geq 0$ implies $(x,\tau_0) \models \pi^\mu$.
$G_{[a,b]}\pi^\mu$	$\forall t' \in \sigma(a,b), \forall x \in D, h^{G_{[a,b]}\pi^\mu}(x,t') \geq 0$ implies $\mu(x) \geq 0$.
$F_{[a,b]}\pi^\mu$	$\exists t' \in \sigma(a,b), \forall x \in D, h^{F_{[a,b]}\pi^\mu}(x,t') \geq 0$ implies $\mu(x) \geq 0$.
$\mu_1 U_{[a,b]}\mu_2$	$h^{\mu_1 U_{[a,b]}\mu_2}(x,t) := \widetilde{\min}(h_1(x,t), h_2(x,t))$ where $\exists t' \in \sigma(a,b), \forall x \in D, h_2(x,t') \geq 0$ implies $\mu_2(x) \geq 0$ and $\forall t'' \in \sigma(0,t'), \forall x \in D, h_1(x,t'') \geq 0$ implies $\mu_1(x) \geq 0$.
$G_{[a,b]}\mu_1 \wedge \mu_2$	$h^{G_{[a,b]}\mu_1\wedge\mu_2}(x,t) := \widetilde{\min}(h_1(x,t), h_2(x,t))$ where $\forall t' \in \sigma(a,b), \forall x \in D, h_1(x,t') \geq 0$ implies $\mu_1(x) \geq 0$ and $h_2(x,t') \geq 0$ implies $\mu_2(x) \geq 0$.
$F_{[a,b]}\mu_1 \wedge \mu_2$	$h^{F_{[a,b]}\mu_1\wedge\mu_2}(x,t) := \widetilde{\min}(h_1(x,t), h_2(x,t))$ where $\exists t' \in \sigma(a,b), \forall x \in D, h_1(x,t') \geq 0$ implies $\mu_1(x) \geq 0$ and $h_2(x,t') \geq 0$ implies $\mu_2(x) \geq 0$.
$\mu_1 \wedge \mu_2 U_{[a,b]}(x,t)\mu_3 \wedge \mu_4$	$h^{\mu_1\wedge\mu_2 U_{[a,b]}(x,t)\mu_3\wedge\mu_4}(x,t) := \widetilde{\min}(h_1(x,t),\dots,h_4(x,t))$ where $\exists t' \in \sigma(a,b), \forall x \in D, h_3(x,t') \geq 0$ implies $\mu_3(x) \geq 0$ and $h_4(x,t') \geq 0$ implies $\mu_4(x) \geq 0$ and $\forall t'' \in \sigma(0,t'), \forall x \in D, h_1(x,t'') \geq 0$ implies $\mu_1(x) \geq 0$ and $h_2(x,t'') \geq 0$ implies $\mu_2(x)$.
$\bigwedge_{i=1}^p \phi_i$	$h^{\wedge_{i=1}^p \phi_i}(x,t) := \widetilde{\min}(h_1(x,t),\dots,h_p(x,t))$ where $h_i(x,t)$ is the CBF for ϕ_i following the recursive definition.

TABLE I: Candidate CBF functions for STL encoding

solving an open-loop optimal control problem (OCP)

$$v_t^{*,(t)},\dots,v_{t+\mathcal{H}-1}^{*,(t)} := \underset{u_t,\dots,u_{t+\mathcal{H}-1}\in\mathcal{U}}{\arg\min}\{E(x_{t+\mathcal{H}}^{(t)}) + \sum_{\tau=t}^{t+\mathcal{H}-1} q(x_\tau^{(t)}, u_\tau) \text{ s.t. } x_{\tau+1}^{(t)} := f(x_\tau^{(t)}, u_\tau)\}. \quad (2)$$

Note that we use the superscript (t) here to denote the current time when we solve the OCP and the subscript to denote the predictive steps. After solving the OCP at time t, we plan in closed-loop by executing $v_t^{*,(t)}$ and increment the current time t to solve the OCP again[1]. Through this procedure, we attain x^*, which corresponds to the closed-loop trajectory from the receding-horizon procedure. Namely, $x_{t+1}^* := f(x_t^*, v_t^{*,(t)})$ with the closed-loop solution $v^* := (v_0^{*,(0)},\dots,v_{H-1}^{*,(H-1)})$. Receding-horizon control allows trajectories robust against system disturbances.

Problem 1: Given a system f, the initial state x_0, a planning horizon $\mathcal{H}$, and cost functions q and E, design a feedback control s.t. $(x^*, 0) \models \phi$, where ϕ is a user-defined STL specification. At each time $t \in \{0,\dots,H-1\}$, we would like to optimize the OCP defined in (2).

IV. Safety-Aware-STL-MPPI

We propose safety-aware-stl-mppi to approach Problem 1. The algorithm relies on MPPI [8], a sampling-based method proposed to solve OCPs without constraint satisfaction guarantee. In an attempt to enable hard constraint satisfaction, time-invariant CBFs were first considered in MPPI-CBF [32], Shield-MPPI [33], and BR-MPPI [21] as a safety filter to enforce non-STL safety constraints. Inspired by these works, we propose to incorporate time-varying CBF in MPPI to satisfy STL constraints. We first introduce how time-varying CBF can be used to encode STL formulas, which we adapt to discrete-time from [5], [34] where continuous-time CBFs are discussed. This design choice is naturally motivated by our formulation in Problem 1.

A. Time-varying Control Barrier Function for STL

Consider the system f and a function $h : D \times \mathbb{Z}_{\geq 0} \to \mathbb{R}$ where $D \subseteq \mathbb{R}^n$ denotes the domain in which the system operates. We assume h to be differentiable in x and consider a time-varying set $\mathcal{C}_t := \{x \in D \mid h(x,t) \geq 0\}$.

Definition 1: **Time-varying Control Barrier Function.** h is a candidate time-varying control barrier function (CBF) to f if $C_t \neq \emptyset, \forall t \in \{0,\dots,H\}$. A candidate CBF h is valid if $\forall t \in \{0,\dots,H-1\}, \forall x \in C_t, \exists u \in \mathcal{U}$ such that

$$h(f(x,u), t+1) - h(x,t) \geq -\alpha h(x,t), \quad (3)$$

where $\alpha \in (0,1]$ is a positive constant.

Consider $S(x,t) := \{u \in \mathcal{U} \mid (3)\}$. We present Theorem 1, whose proof can be found in Appendix VIII-A

Theorem 1: **Forward Invariance.** Let $x_0 \in \mathcal{C}_0$ and consider a sequence of inputs $(u_0,\dots,u_{H-1})$ where $u_t \in S(x_t,t)$. Then, $x_t \in C_t, \forall t \in \{0,\dots,H\}$.

Given an STL formula ϕ from fragment (1), the goal now is to design a CBF $h^\phi(x,t)$ such that $h^\phi(x,t) \geq 0, \forall t \in \{0,\dots,H\}$ ensures $(x,0) \models \phi$. We follow the recursive semantics and start by considering the predicate. Given a predicate π^μ, we consider any function h^{π^μ} such that $h^{\pi^\mu}(x,\tau_0) \geq 0 \implies (x,\tau_0) \models \pi^\mu$. A natural choice for h^{π^μ} is μ if μ is differentiable. Otherwise, we consider a smooth construction (e.g. $h^{\pi^\mu}(x,t) := a^2 - \|x\|_2^2$ with scalar $a > 0$ is used to encode π^μ where $\mu := a - \|x\|_2$). Since negation is only defined over predicates, ϕ can be assumed negation-free without loss of generality [30] and thus the encoding for negation is ignored. We can then design CBFs for more complex formulas recursively following the encoding rules listed in Table I, where the rows 2-4 encode temporal operators over predicates, rows 5-7 encode temporal operators over conjunctions, and the last row encodes conjunctions over temporal operators. We emphasize on the use of a smooth underapproximation of the piecewise minimum operator $\min_{i\in\{1,\dots,p\}} h_i(x,t) \geq \widetilde{\min}(h_1(x,t),\dots,h_p(x,t)) := -\ln(\sum_{i=1}^p \exp(-h_i(x,t)))$. See concrete examples of constructed CBFs in Section V. We emphasize that constructive synthesis procedures for continuous-time CBFs are discussed in [35].

Note that the CBF encoded following Table I does not necessarily guarantee $\mathcal{C}_t \neq \emptyset$ for all $t \in \{0,\dots,H\}$, required for the candidacy in Definition 1. In fact, a deletion mechanism can be considered to reduce conservatism [5],

[1] As standard practice, $\mathcal{H}$ is shrunk when $t + \mathcal{H} > H$.

which we summarize here for discrete-time: For any ϕ with temporal operators, the general form of the constructed CBF after encoding is $h(x,t) = \widetilde{\min}(h_1(x,t), \dots, h_p(x,t))$ where each h_i corresponds to a temporal formula $\phi_i = \mathcal{T}_{[t_i, t_{i+1}]}\varphi$ or $\phi_i = \varphi_1 \mathcal{T}_{[t_i, t_{i+1}]} \varphi_2$ where $\mathcal{T}$ is a temporal operator. Note that after t_{i+1}, ϕ_i no longer needs to be enforced and thus h_i can be deleted from the set of arguments to $\widetilde{\min}$ after t_{i+1}. We consider the following deletion strategy: Let $\mathcal{A}_{t_0}(x,t) := \{h_1(x,t), \dots, h_p(x,t)\}$ and $\mathcal{A}_{t_{i+1}+1}(x,t) := \mathcal{A}_{t_i} \setminus \{h_i(x,t)\}$ and we set $h(x,t) := \widetilde{\min}(\mathcal{A}_{t_i}(x,t))$ when $t_i \le t \le t_{i+1}$. When several functions h_j corresponds to the same switching time, the deletion strategy can remove all such terms simultaneously. We present Theorem 2, whose proof is in Appendix VIII-B

Theorem 2: **Safety from STL CBF.** Given a formula ϕ following (1). Let $x_0 \in \mathcal{C}_0$. Suppose there exists a valid CBF in the sense of (3) constructed satisfying the conditions in Table I. Then, enforcing (3) in Theorem 1 implies $(x,0) \models \phi$.

Theorem 2 applies to a controller that enforces the exact constraint (3). It does not directly establish STL satisfaction for Algorithm 1, which replaces the constraint with a first order approximation, which we discuss later in (6). Consider an example with $x \in \mathbb{R}$, $x_0 = -2$, and $\phi := G_{[0,10]} x \ge -4 \wedge F_{[0,5]} x \ge 0$. We construct a candidate CBF following the rules of Table I where $h^\phi(x,t) := \widetilde{\min}(x+4, \kappa(x,t))$ with $\kappa(x,t) := x + 5 - t$ removed from the arguments of $\widetilde{\min}$ at $t = 6$. Note that our construction has no guarantee that h^ϕ is valid. We make Assumption 1, standard for CBF in the MPPI literature [21], [33].

Assumption 1: $S(x_t, t) \neq \emptyset, \forall t \in \{0, \dots, H-1\}$.

It is easy to show that $S(x_t,t)$ is nonempty if $\mathcal{U} = \mathbb{R}^m$ and f is feedback-equivalent to a single integrator $x_{t+1} = x_t + u_t$ when C_t is nonempty. Our algorithm requires $\frac{\partial h^\phi}{\partial x} g$ to be not identically zero (i.e., the CBF is of 1st order). For formulas that result in high-order CBFs, we reduce the order by substituting the forward dynamics following [21]. See example in Case D of Section V-A.

B. Model Predictive Path Integral

MPPI is a receding-horizon sampling-based strategy, which we use to solve for v^* in Problem 1. Given a fixed horizon $\mathcal{H}$, at each time t, we sample K control input perturbations $\omega^{k,(t)} = (\omega_t^{k,(t)}, \dots, \omega_{t+\mathcal{H}-1}^{k,(t)})$ where $\omega^{k,(t)} \in \mathbb{R}^m \times \mathcal{H}$ for $k \in \{1, \dots, K\}$, where K is a sampling size. Further, we require $\omega_\tau^{k,(t)} \sim \mathcal{N}(0, \Sigma_\omega)$ to be sampled from a zero-centered Gaussian distribution with a noise covariance of Σ_ω. Based on each sample, we can obtain a sample rollout x^k up to time $t+\mathcal{H}$ using a sample control sequence $u^{k,(t)} = v^{(t)} + \omega^{k,(t)} \in \mathbb{R}^{m \times \mathcal{H}}$ where $v^{(t)}$ is a nominal control input. We can then compute a sample cost

$$J^{k,(t)} := E(x_{t+\mathcal{H}}^{k,(t)}) + \sum_{\tau=t}^{t+\mathcal{H}-1} [q(x_\tau^{k,(t)}, u_\tau^{k,(t)}) + \gamma (v_\tau^{(t)})^T \Sigma_\omega^{-1} \omega_\tau^{k,(t)}], \tag{4}$$

where[2] $\gamma \ge 0$ is a regularization factor. Note that inherently $J^{k,(t)}$ does not contain any safety information pertinent to ϕ. As inspired by [33], we incorporate the CBF information in the cost function by considering the safety augmented cost,

$$J_\phi^{k,(t)} := J^{k,(t)} + \eta \sum_{\tau=t}^{t+\mathcal{H}-1} C_{\text{CBF}}(x^{k,(t)}, \tau),$$

where $C_{\text{CBF}}(x^{k,(t)}, \tau) := \max\{0, (1-\alpha) h^\phi(x_\tau^{k,(t)}, \tau) - h^\phi(x_{\tau+1}^{k,(t)}, \tau+1)\}$ denotes the CBF constraint violation. We remark here $\eta \ge 0$ is a tunable hyperparameter. We also remark that one can consider the robust semantics ρ^ϕ as a part of the cost (as considered in [10]), but for general STL this requires a shrinking-horizon rollout where $\mathcal{H} := H - t$, which has a high computational cost. Standard in literature, the next step is to assign a weight on each of the sample cost based on the performance using $W^{k,(t)} := \exp[-\frac{1}{\lambda}(J_\phi^{k,(t)} - \min_{i \in \{1,\dots,K\}} J_\phi^{i,(t)})]$, with $\lambda \ge \gamma$, and the optimal control sequence is expressed as a weighted rollout $v^{+,(t)} = \sum_{k=1}^K W^{k,(t)} u^{k,(t)} / \sum_{k=1}^K W^{k,(t)}$. Despite C_{CBF}, $v^{+,(t)}$ is not guaranteed to satisfy the CBF constraint. Standard procedure [33] considers a safety filter as adapted below for our setting of time-varying CBF,

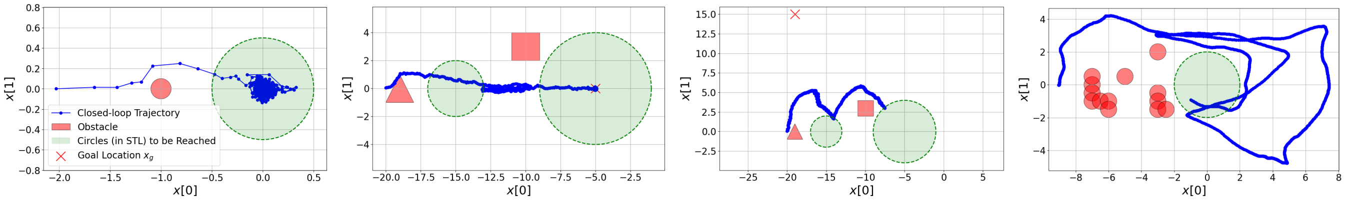
$$\begin{aligned} v_\tau^{s*,(t)} &:= \arg\min_{v_\tau^s \in \mathcal{U}} \|v_\tau^{+,(t)} - v_\tau^s\| \qquad (5) \\ \text{s.t. } & h^\phi(f(x_\tau^{(t)}, v_\tau^s), \tau+1) - h^\phi(x_\tau^{(t)}, \tau) \ge -\alpha h^\phi(x_\tau^{(t)}, \tau), \end{aligned}$$

where $\tau \in \{t, \dots, t+N\}$ and $0 \le N \le \mathcal{H} - 1$. Let $v_\tau^{s*,(t)} = v_\tau^{+,(t)}$ for $\tau > t+N$. One can forward $v_t^{s*,(t)}$ to the dynamics and plan in receding-horizon with MPPI. One can use ${v_{t+1}^s}^{*,(t)}, \dots, {v_{t+\mathcal{H}-1}^s}^{*,(t)}$ as a warm-start for time $t+1$. Note that (5) is generally nonconvex in discrete-time. The standard procedure proposed in [33] to approach (5) is to solve the following through a first-order optimizer ${v_{t:t+N}^s}^{*,(t)} \approx \arg\max \sum_{\tau=t}^{t+N} \min\{0, h^\phi(x_{\tau+1}^{(t)}, \tau+1) + (\alpha - 1) h^\phi(x_\tau^{(t)}, \tau)\}$ which guarantees safety when the cost is zero and when $\mathcal{U} = \mathbb{R}^m$. Usually this is not run until convergence in real-time. Moreover, the solution is not guaranteed optimal to (5) even after convergence. Alternatively, a closed-form solution is proposed in [21] for time-invariant CBF with $\mathcal{U} = \mathbb{R}^m$ using first-order Taylor expansion. Motivated by [21], we propose an approximated quadratic program solved in closed-form when $\mathcal{U} = \mathbb{R}^m$.

C. Solving the CBF Constraint with Approximation

Since $h^\phi(x,t)$ is differentiable in x, we have by first-order Taylor expansion around x_τ that $h^\phi(f(x_\tau^{(t)}, v_\tau^s), \tau+1) \approx h^\phi(x_\tau^{(t)}, \tau+1) + \frac{\partial h^\phi}{\partial x}^T\Big|_{(x_\tau^{(t)}, \tau+1)} [f(x_\tau^{(t)}, v_\tau^s) - x_\tau^{(t)}]$. Suppose $A(x_\tau^{(t)}, \tau) := \frac{\partial h^\phi}{\partial x}\Big|_{(x_\tau^{(t)}, \tau+1)} \in \mathbb{R}^n$. Given the input-affine

[2]Note that some work [21], [33] replace $\gamma(v_\tau^{(t)})^T \Sigma_\omega^{-1} \omega_\tau^{k,(t)}$ with other action perturbation cost functions in (4). We use (4) for its consistency with the original formulation in MPPI [8] and use (4) consistently across baselines in Section V for fair comparison.



(a) Case A **(b)** Case B **(c)** Case C **(d)** Case D

Fig. 2: Example Trajectories Planned with safety-aware-stl-mppi (Limited Actuation).

Algorithm 1 Safety-Aware-STL-MPPI.

We are given ϕ, $\mathcal{H}$, f, E, and q.
We have constructed h^ϕ following Section IV-A before the closed-loop planning.
Suppose the current time is t. We are given hyperparameters $\Sigma_\omega, K, \eta, \gamma, \lambda, N$.
for $k \in \{1, \dots, K\}$ in parallel **do**
  Sample $\omega^{k,(t)} \leftarrow (\omega_t^{k,(t)}, \dots, \omega_{t+\mathcal{H}-1}^{k,(t)})$,
  where $\omega_\tau^{k,(t)} \sim \mathcal{N}(0, \Sigma_\omega)$.
  $u^{k,(t)} = v^{(t)} + \omega^{k,(t)}$; $J_\phi^{k,(t)} \leftarrow 0$.
  for $\tau \in \{t, \dots, t+\mathcal{H}-1\}$ **do**
    $x_{\tau+1}^{k,(t)} \leftarrow f(x_\tau^{k,(t)}, u_\tau^{k,(t)})$.
    $$J_\phi^{k,(t)} \leftarrow J_\phi^{k,(t)} + q(x_\tau^{k,(t)}, u_\tau^{k,(t)}) + \gamma\,(v_\tau^{(t)})^T \Sigma_\omega^{-1} \omega_\tau^{k,(t)} + \eta\, C_{\text{CBF}}(x^{k,(t)}, \tau).$$
  end for
  $J_\phi^{k,(t)} \leftarrow J_\phi^{k,(t)} + E(x_{t+\mathcal{H}}^{k,(t)})$.
end for
$W^{k,(t)} := \exp[-\frac{1}{\lambda}(J_\phi^{k,(t)} - \min_{i \in \{1,\dots,K\}} J_\phi^{i,(t)})]$
$v^{+,(t)} = \sum_{k=1}^K W^{k,(t)} u^{k,(t)} / \sum_{k=1}^K W^{k,(t)}$.
$\{\hat{v}_t^{s,(t)}, \dots, \hat{v}_{t+N}^{s,(t)}\} \leftarrow$ (8) or (9).
Warm start with the new weighted rollout.
Execute $\hat{v}_t^{s,(t)}$.

dynamics f, we can approximate the constraint in (5) with

$$h^\phi(x_\tau^{(t)}, \tau+1) + A(x_\tau^{(t)}, \tau)^T[(f_0(x_\tau^{(t)}) - x_\tau^{(t)}) + g(x_\tau^{(t)}) v_\tau^s] \geq (1-\alpha) h^\phi(x_\tau^{(t)}, \tau), \tag{6}$$

which is affine in v_τ^s. We can therefore attain

$$\hat{v}_\tau^{s,(t)} := \arg\min_{v_\tau^s \in \mathcal{U}} \{\|v_\tau^{+,(t)} - v_\tau^s\| \text{ subject to (6)}\}, \tag{7}$$

where $v_\tau^{s*,(t)} \approx \hat{v}_\tau^{s,(t)}$. Consider all active actuation limits $e_i^T v_\tau^s \leq u_{\max,i}$ and $-e_j^T v_\tau^s \leq -u_{\min,j}, \forall (i,j) \in \mathcal{A}$ where $\mathcal{A} \subseteq \{1, \dots, m\}^2$. Let $\Pi := (a_{\text{cbf}}^T(x_\tau^{(t)}, \tau), e_i^T, \dots, -e_j^T, \dots)^T$ and $b := (b_{\text{cbf}}(x_\tau^{(t)}, \tau), u_{\max,i}, \dots, -u_{\min,i}, \dots)$ where

$$a_{\text{cbf}}^T(x_\tau^{(t)}, \tau) := -A(x_\tau^{(t)}, \tau)^T g(x_\tau^{(t)});$$
$$b_{\text{cbf}}(x_\tau^{(t)}, \tau) := A(x_\tau^{(t)}, \tau)^T (f_0(x_\tau^{(t)}) - x_\tau^{(t)}) + h^\phi(x_\tau^{(t)}, \tau+1) + (\alpha-1) h^\phi(x_\tau^{(t)}, \tau).$$

Then, the constraints in (6) is equivalent to $\Pi v_\tau^s \leq b$. We can therefore write (7) in standard Quadratic Program (QP)

$$\hat{v}_\tau^{s,(t)} = \arg\min\{\frac{1}{2}(v_\tau^s - v_\tau^{+,(t)})^T P (v_\tau^s - v_\tau^{+,(t)}) \mid \Pi v_\tau^s \leq b\}, \tag{8}$$

where $P := I_m$ is the m dimensional identity matrix. The QP in (8) can be solved efficiently at runtime and admits a closed form solution when $\mathcal{U} = \mathbb{R}^m$, in which case we have

$$\hat{v}_\tau^{s,(t)} = v_\tau^{+,(t)} - a_{\text{cbf}}(x_\tau^{(t)}, \tau)[\max\{0, a_{\text{cbf}}^T(x_\tau^{(t)}, \tau) v_\tau^+ - b_{\text{cbf}}(x_\tau^{(t)}, \tau)\} / \|a_{\text{cbf}}(x_\tau^{(t)}, \tau)\|_2^2]. \tag{9}$$

The derivation from (8) to (9) when under unlimited actuation is standard, but we present it for completeness in Appendix VIII-C. We summarize the safety-aware-stl-mppi in Algorithm 1, which we call in closed-loop. Naturally, along the closed-loop trajectory, we assume $\hat{S}(x_\tau, \tau) \neq \emptyset$ where $\hat{S}(x, \tau) := \{v \in U \mid a_{\text{cbf}}(x, \tau)^T v \leq b_{\text{cbf}}(x, \tau)\}$ and that $a_{\text{cbf}}(x_\tau^{(t)}, \tau) \neq 0$ at every state where (9) is applied. Due to the approximation error, perfect safety is not guaranteed theoretically. We discuss how one can address this by optional robustification in Theorem 3 in the Appendix.

V. Evaluation

We evaluate safety-aware-stl-mppi via four artificial Mars rover path planning case studies and a photorealistic simulation with a quadcopter. In the rover case studies, we compare the algorithm with four state-of-the-art baseline models (detailed in Section V-A) and discuss our advantage in safety and low computation cost. We then simulate a drone inspection task in Nvidia Isaac Lab [36], [37] with safety-aware-stl-mppi. All experiments are run on a Dell G16 7630 Model with a 13th Gen Intel(R) Core(TM) i9 processor.

A. Mars Rover Case Studies

We consider artificial case studies where we simulate safety-critical missions of a Mars rover. In each mission, the rover is subject to a safety constraint defined in STL (e.g., to return to home base before the battery dies out and to avoid hazardous regions at all time). We consider four case studies with different dynamics, goals, and safety specifications, which we evaluate with and without actuation limit. In each of cases A, B, and C, we consider a goal location to inspect, incentivized by the terminal cost $E(x_{t+\mathcal{H}}^{k,(t)}) := \|x_{t+\mathcal{H}}^{k,(t)} - x_g\|_2$ where $x_g \in \mathbb{R}^2$ denotes the goal location. In case D, we simply set $E(x_{t+\mathcal{H}}^{k,(t)}) := 0$. Let $q(x_\tau^{(t)}, u_\tau) := 0$ for all cases. The purpose of the hypothetical goal locations is to

(a) Limited actuation

Method	Case A		Case B		Case C		Case D	
	SR ↑	Timing ↓	SR ↑	Timing ↓	SR ↑	Timing ↓	SR ↑	Timing ↓
Safety-Aware-STL-MPPI (Ours)	100%	5.26	100%	8.80	100%	8.84	100%	17.93
Vanilla MPPI	100%	1.35	100%	3.32	0%	3.33	0%	5.97
Reach-avoid MPPI	100%	5.92	100%	7.31	0%	7.32	25%	17.05
Robustness-based MPPI	100%	190.35	95%	90.74	90%	90.84	55%	1060.55
Penalty-based MPPI	100%	191.16	80%	91.21	85%	91.34	60%	1057.51

(b) Unlimited actuation

Method	Case A		Case B		Case C		Case D	
	SR ↑	Timing ↓	SR ↑	Timing ↓	SR ↑	Timing ↓	SR ↑	Timing ↓
Safety-Aware-STL-MPPI (Ours)	100%	3.80	100%	7.40	100%	7.37	100%	16.39
Vanilla MPPI	100%	1.55	100%	3.31	0%	3.31	0%	5.90
Reach-avoid MPPI	100%	3.47	100%	5.19	0%	5.21	25%	14.55
Robustness-based MPPI	100%	202.87	100%	90.32	100%	90.31	55%	1057.29
Penalty-based MPPI	95%	203.45	100%	90.65	95%	90.59	60%	1057.07

TABLE II: Section V-A Evaluation Results.

show that safety-aware-stl-mppi is robust to adversarial (and non-adversarial) cost objectives. We expect safety-aware-stl-mppi to respect the safety constraint even if the goal is not achievable in due time. All unit of time are seconds, which we highlight with the s postfix in the metric interval of the STL specifications. Since the experiment runs in real-time, we set a sampling time of $\Delta t := 0.01s$ between the two time index t and $t+1$. In all cases, the running costs are 0. For validation, we draw initial positions randomly as specified below and show examples of successful trajectories from safety-aware-stl-mppi for all cases (with actuation limits) in Figure 2[3].

- Case A: Consider a nonlinear dynamics $x_{t+1} = \sin(x_t) + u_t$ where $x_t, u_t \in \mathbb{R}^2$ and $x_0 \sim (U(-3,-2), 0)^T$ where $U(a,b)$ denotes the uniform distribution over $[a,b)$ and $x_g := (0,0)$. We are given the safety specification $\phi_A := G_{[0s,10s]}\|x-(-1,0)^T\|_2 \geq 0.1 \wedge F_{[0s,10s]}0.5 \geq \|x-(0,0)^T\|_2$, specifying the robot to at all time avoid a circle of radius 0.1 centered at $(-1,0)^T$ and reach a circle of radius 0.5 centered at the origin between 0 and 10 seconds. The actuation limit is $(-5,-5)^T \preccurlyeq u_t \preccurlyeq (5,5)^T$.
- Case B: Consider a single integrator dynamics $x_{t+1} = x_t + u_t$ where $x_t, u_t \in \mathbb{R}^2$, $x_0 \sim (-20, U(0,2))^T$ and a goal location $x_g := (-5,0)^T$. We are given the following safety specification, $\phi_B := G_{[0s,10s]}T(x) \geq 0 \wedge G_{[0s,10s]}S(x) \geq 0 \wedge F_{[0s,7s]}4 - [(x[1]^2 + (x[0]+15)^2] \geq 0 \wedge F_{[0s,10s]}16 - [x[1]^2 + (x[0]+5)^2] \geq 0$, where $T(x) = \widetilde{\max}(x[1]+2x[0]+37, -1-x[1], -2x[0]+x[1]-39)$, $S(x) := \widetilde{\max}(-11-x[0], x[0]+9, x[1]-4, 2-x[1])$, and $\widetilde{\max}(a_1,a_2,\ldots) := \frac{1}{\Omega}\log(\exp(\Omega a_1)+\exp(\Omega a_2)+\ldots)$ with $\Omega > 0$ denotes smooth maximum. We set $\Omega := 1000$. The formula specifies that the robot should always avoid a triangle with vertices $(-20,-1)^T, (-19,1)^T, (-18,-1)^T$ and a square with vertices $(-11,2)^T, (-9,2)^T, (-11,4)^T, (-9,4)^T$. We require the robot to reach a circle of radius 2 centered at $(-15,0)^T$ in time interval $[0s,7s]$ and a circle of radius 4 centered at $(-5,0)^T$ in time interval $[0s,10s]$. The actuation limit is $(-0.08,-0.08)^T \preccurlyeq u_t \preccurlyeq (0.08,0.08)^T$.
- Case C: Case C is identical to Case B but with the goal location $x_g := (-19,15)^T$.
- Case D: We consider a double integrator dynamics $x_{t+1} = \begin{bmatrix} p_{t+1} \\ v_{t+1} \end{bmatrix} = \begin{bmatrix} I_2 & I_2 \\ 0 & I_2 \end{bmatrix}\begin{bmatrix} p_t \\ v_t \end{bmatrix} + \begin{bmatrix} 0 \\ I_2 \end{bmatrix} u_t$, where I_2 is a 2−dimensional identity matrix. We consider $x_0 \sim (U(-9.5,-9), 0, 0, 0)^T$ and 13 circular obstacles, shown in Figure 2d. Let the centers of each obstacle i be c^i, and we consider the safety specification $\phi_D := (\wedge_{i=1}^{13} G_{[0s,10s]}\|p-c^i\| \geq 0.5) \wedge F_{[9s,10s]}2 - \|p-(0,0)^T\| \geq 0$, which means that the rover needs to avoid all obstacles at all time and reach a circle of radius 2 centered at the origin in time interval $[9s,10s]$. The actuation limit is $(-20,-20)^T \preccurlyeq u_t \preccurlyeq (20,20)^T$.

We construct candidate CBFs:

$$h^{\phi_A}(x,t) := \widetilde{\min}((x_t[0]+1)^2 + x_t[1]^2 - 0.1^2, -\frac{7}{8}t\Delta t + 9 - (x_t[0]^2 + x_t[1]^2)),$$

$$h^{\phi_B}(x,t) := \widetilde{\min}(\widetilde{\min}(T(x_t), S(x_t)), \widetilde{\min}(-\frac{32}{7}t\Delta t + 36 - x_t[1]^2 - (x[0]+15)^2, -24t\Delta t + 256 - (x_t[0]+5)^2 - x_t[1]^2)),$$

$$h^{\phi_D}(x,t) := \widetilde{\min}(\widetilde{\min}_{i=1}^{12}((x_{t+1}[0]-c^i[0])^2 + (x_{t+1}[1]-c^i[1])^2 - 0.5^2, (x_{t+1}[0]-c^{i+1}[0])^2 + (x_{t+1}[1] - c^{i+1}[1])^2 - 0.5^2), 100 - \frac{96t\Delta t}{10-\Delta t} - (x_{t+1}[0]^2 + x_{t+1}[1]^2)),$$

where $\widetilde{\min}_{i=1}^{12}$ means that $\widetilde{\min}$ is applied 12 times. We switch h^{ϕ_B} to $\widetilde{\min}(\widetilde{\min}(T(x_t), S(x_t)), -24t\Delta t + 256 - (x_t[0]+5)^2 - x_t[1]^2)$ at $t\Delta t = 7$[4]. Note that $h^{\phi_C} := h^{\phi_B}$. Case C is more difficult than case B in that x_g may not be achievable given the specification, and the robot needs to obey the safety

[3]We emphasize that in Figure 2b, we plot exact traingle and squares instead of smooth approximations.

[4]The reported results instantiate the switch at $t\Delta t = 7$ rather than $7+\Delta t$; this affects only the final discretization step of the $F_{[0s,7s]}$ component. Reported robustness values are computed by an independent monitor under the exact closed-interval semantics and are therefore unaffected.

constraint even if it may not reach close to the goal location.

Baseline Models. We compare our results with four other baseline models, all adapted to PyTorch [38] for consistent comparison. In **vanilla MPPI** [7], we consider constant penalties respectively for each violation of obstacle avoidance and for when the robot does not satisfy the reach requirement at time τ, where $\tau \in \{t_1, \dots, t_2\}$ with $[t_1, t_2]$ denoting the STL metric interval for each time operator in sampling index[5]. In **reach-avoid MPPI** [12], we consider a cost function $c_r(x_\tau^k) := k_r(\|p_t - p_r\|^2 - r_r^2)$ for each center of the circles to be reached p_r with a sufficient radius r_r and a cost function $c_o(x_\tau^k) := k_o/(\max(\|p_t - p_o\|, \epsilon_o)$ for each obstacle center p_o in the STL specifications. Note that p_r is not to be confused with x_g. A composite cost function $Q(x_{t:t+\mathcal{H}}) := \max(Q_{r_1}, Q_{r_2}, \dots, Q_{o_1}, Q_{o_2})$, where $\{r_1, \dots\}$ and $\{o_1, \dots\}$ denote the centers of circles to be reached and of obstacles in the STL, is considered with $Q_o := \max\{c_o(x_{t_1}), \dots, c_o(x_{t_2})\}$ and $Q_r := \min\{c_r(x_{t_1}), \dots, c_r(x_{t_2})\}$, where $[t_1, t_2]$ are metric intervals for the time operators[6]. We add the composite cost function to the rollout costs. We use $\nabla_x h(f(x_t, u_t))(f(x_t, u_t) - x_t) \geq -\alpha h(x_t)$ as a time-invariant CBF filter. In **penalty-based MPPI** (i.e. MPPI-STL) [10], a penalty function, $I(x^k)$, for the violation of the safety specification based on STL robustness is augmented to the rollout costs. Before calling MPPI, the paper relies on a hybrid A^* guiding prior factored into the cost function. We disable this pre-planning step to compare the core MPPI algorithms under identical priors. The paper did not specify the exact penalty, so we choose $I(x^k) := -\rho^\phi(x^k)1(x^k \not\models \phi)$, where 1 is an indicator function and ρ^ϕ is the robust semantics over ϕ. An open loop path integral optimization method, which we here denote as **robustness-based MPPI**, is proposed in [11] with $I(x^k) := -\rho^\phi(x^k)$. The paper also proposes a hyperparameter tuning procedure, which we have to disregard for a real-time closed-loop implementation. Note that both penalty-based and robustness-based MPPIs require the computation of STL robust semantics, which must be executed in shrinking-horizon (i.e., $\mathcal{H} := H - t + 1$ adapts online). The augmented costs for each method are rescaled with weights[7]. We use the same regularization and weighting schemes across baselines. We specifically emphasize that none of the above methods are designed for providing safety guarantee with STL under arbitrary cost functions, and to the best our knowledge we are the first work to address such problem with MPPI.

Results. For each case, we evaluate the experiment via three metrics. **Success Rate (SR)** measures the percentage of closed-loop planned trajectories over the 20 trials with nonnegative robust semantics. For negation-free discrete-time STL formulas with bounded metric intervals (which we have in evaluation), a trajectory satisfies the STL formula if and only if its robust semantics are nonnegative. **Actuation Limit Satisfaction Rate (ALSR)** measures the percentage of trajectories satisfying the actuation limit at all time[8]. Lastly, **timing**, in seconds, measures the overall computation time for the closed-loop planning (i.e. the time needed to solve for x^* in problem 1) averaged across the 20 trials. We show the results across all baselines in Table II[9]. All methods achieved an ALSR of 100% across all cases. Importantly, our method demonstrates 100% empirical STL satisfaction rate in the evaluated trials and computation **efficiency** under **diverse cost settings and environments**. Vanilla MPPI is efficient but violates safety constraints frequently. Reach-avoid MPPI cannot generalize to complex STL tasks with tight timed constraints. Robustness and penalty-based MPPIs suffer from high computation costs due to the long-horizon rollouts and do not have safety guarantee.

B. *Quadcopter Simulation*

We consider a drone planning simulation in an artificial factory environment in Isaac Lab with a Crazyflie quadcopter. The environment consists of seven obstacles including three package tables, one two-level stand and three forklifts. We estimate the obstacles with infinite cylinder bounding volumes, which we denote with C_i where $i \in \{1, \dots, 7\}$. We first consider a planning stage (with $200HZ$ planning frequency) where we model the quadcopter with a double integrator dynamics (see case D from Section V-A but extended to 3 dimensions), where $x_t := (p_t, v_t)$ with $p_t := (p^x, p^y, p^z)^T \in \mathbb{R}^3$ and $v_t \in \mathbb{R}^3$. We impose an actuation limit of $(-0.2, -0.2, -0.2)^T \preccurlyeq u_t \preccurlyeq (0.2, 0.2, 0.2)^T$. The goal of the task is to safely navigate near $(0, 0, 1)^T$ without collision with the obstacles nor the ground. Formally, we consider the safety specification $\phi := \wedge_{i=1}^7 G_{[0s,5s]} S(p, C_i) \geq 0 \wedge G_{[0s,5s]}(p^z - 0.3) \geq 0 \wedge F_{[5s,5s]} \|p - (0,0,1)^T\| \leq 1$, where $S(p, C)$ denotes the signed distance between p and set C. We consider 50 experimental trials, where in each experiment, we use safety-aware-stl-mppi to plan a trajectory. To allow a smooth trajectory converging towards the goal with minimal actuation expense, a stable height, and a small velocity, we consider a stage cost of $q(x_\tau^{k,(t)}, u^{k,(t)}) := \|u^{k,(t)}\|_2 + |p_\tau^{k,(t),z} - 1| + \|v_\tau^{k,(t)}\|_2$ and a terminal cost of $E(x_{t+H}^{k,(t)}) := 0$. See Figure 3a for an example of a successfully planned trajectory, denoted by the red waypoints. We show an example of the rollouts during planning in Figure 3b, where the gray waypoints denote the rollouts x^k with $K := 100$ and the yellow waypoints are the weighted rollout. We record the average time of 15.21 seconds from planning across the 50 trials, with all executions satisfying the actuation limit during planning. In the second

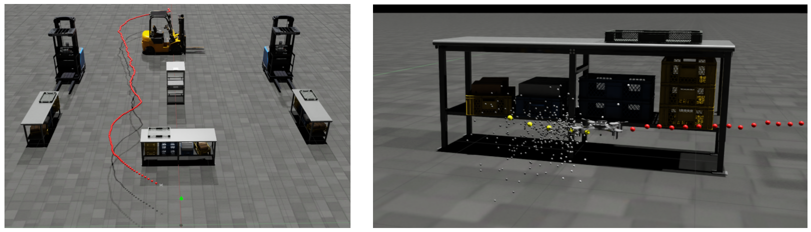
[5] In Case D, obstacle-avoidance predicates for Vanilla MPPI are evaluated on a one-step-ahead kinematic surrogate, consistent with the CBF treatment for higher-order dynamics.

[6] For numerical robustness in discrete-time, we adopt a one-sample temporal buffer when $t_1 > 0$, evaluating Q_r over $[t_1 - \Delta t, t_2]$.

[7] For all methods that do not use a CBF projection, as standard practice [39], we impose an action filter in the rollouts to bound the actions within actuation limit during the rollouts.

[8] We treat actuation as within limit if it lies within $[u_{min,i} - \epsilon, u_{max,i} + \epsilon]$ for all entries i with $\epsilon := 10^{-5}$ to prevent floating-point issues.

[9] In Case A with limited actuation when evaluating safety-aware-stl-mppi, one of the 20 trials terminated with an infeasible CBF-QP (Assumption 1 fails for that initial condition under the tight actuation bound); the planner reports infeasibility rather than executing an unsafe action. SR and timing are computed over the 19 completed trials.



(a) Example Planned Trajectory **(b)** Example Planned Rollouts

Fig. 3: Quadcopter Simulation Example

stage, we track the planned waypoints with position update through Isaac Sim API calls. We record that 100% of the tracked trajectories satisfy the safety specification.

VI. Conclusion

In this paper, we propose safety-aware-stl-mppi, a receding-horizon sampling-based planner under STL safety specifications by integrating a time-varying CBF. We conducted a series of case studies with Mars Rover planning and a quadcopter simulation, which demonstrate that our method maintains high safety and efficiency outcomes. The next step will focus on extending the results to multi-agent tasks and dynamics with uncertainty.

VII. Acknowledgement

We would like to thank Hardik Parwana for discussing solutions for higher-order CBFs and Sung Woo Choi for discussing use cases and experiments with our algorithm. We would like to thank Paul Lutkus for ideas on visualizing the MPPI rollouts in the quadcopter study and suggestions in presentation. We acknowledge the adaptation of [39] for our implementation of MPPI. This work was partially supported by the National Science Foundation through the following grants: CAREER award (SHF-2048094), IIS-SLES-2417075, and funding by Toyota R&D through the USC Center for Autonomy and AI.

References

[1] O. Maler and D. Nickovic, "Monitoring temporal properties of continuous signals," in *FTRTFT*, pp. 152–166, Springer, 2004.

[2] 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.

[3] M. Hekmatnejad, S. Yaghoubi, A. Dokhanchi, H. B. Amor, A. Shrivastava, L. Karam, and G. Fainekos, "Encoding and monitoring responsibility sensitive safety rules for automated vehicles in signal temporal logic," in *MEMOCODE*, pp. 1–11, 2019.

[4] V. Raman, M. Maasoumy, and A. Donzé, "Model predictive control from signal temporal logic specifications: A case study," in *Proc.of the 4th ACM SIGBED International Workshop on Design, Modeling, and Evaluation of Cyber-Physical Systems*, pp. 52–55, 2014.

[5] L. Lindemann and D. V. Dimarogonas, "Control barrier functions for signal temporal logic tasks," *L-CSS*, vol. 3, no. 1, pp. 96–101, 2018.

[6] A. B. Kordabad, M. Charitidou, D. V. Dimarogonas, and S. Soudjani, "Control barrier functions for stochastic systems under signal temporal logic tasks," in *ECC*, pp. 3213–3219, IEEE, 2024.

[7] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Aggressive driving with model predictive path integral control," in *ICRA*, pp. 1433–1440, IEEE, 2016.

[8] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Information-theoretic model predictive control: Theory and applications to autonomous driving," *T-RO*, vol. 34, no. 6, pp. 1603–1622, 2018.

[9] H. Homburger, F. Messerer, M. Diehl, and J. Reuter, "Optimality and suboptimality of mppi control in stochastic and deterministic settings," *L-CSS*, 2025.

[10] F. Baldini, F. M. Tariq, S. Bae, and D. Isele, "Don't get stuck: A deadlock recovery approach," *arXiv preprint arXiv:2408.10167*, 2024.

[11] P. Halder, H. Homburger, L. Kiltz, J. Reuter, and M. Althoff, "Trajectory planning with signal temporal logic costs using deterministic path integral optimization," *arXiv preprint arXiv:2503.01476*, 2025.

[12] H. Parwana, M. Black, G. Fainekos, B. Hoxha, H. Okamoto, and D. Prokhorov, "Model predictive path integral methods with reach-avoid tasks and control barrier functions," *arXiv preprint arXiv:2407.13693*, 2024.

[13] A. Pnueli, "The temporal logic of programs," in *sfcs 1977*, pp. 46–57, ieee, 1977.

[14] G. E. Fainekos, A. Girard, H. Kress-Gazit, and G. J. Pappas, "Temporal logic motion planning for dynamic robots," *Automatica*, vol. 45, no. 2, pp. 343–352, 2009.

[15] X. Ding, S. L. Smith, C. Belta, and D. Rus, "Optimal control of markov decision processes with linear temporal logic constraints," *TAC*, vol. 59, no. 5, pp. 1244–1257, 2014.

[16] C.-I. Vasile, V. Raman, and S. Karaman, "Sampling-based synthesis of maximally-satisfying controllers for temporal logic specifications," in *IROS*, pp. 3840–3847, IEEE, 2017.

[17] Y. Kantaros and M. M. Zavlanos, "Stylus*: A temporal logic optimal control synthesis algorithm for large-scale multi-robot systems," *IJRR*, vol. 39, no. 7, pp. 812–836, 2020.

[18] Z. Zhong, D. Rempe, D. Xu, Y. Chen, S. Veer, T. Che, B. Ray, and M. Pavone, "Guided conditional diffusion for controllable traffic simulation," *arXiv preprint arXiv:2210.17366*, 2022.

[19] C. S. Wang, Y. Zhao, J. Ye, Y. Wang, and J. V. Deshmukh, "Logic-vla: A temporal logic conditioned vision-language-action model," *arXiv preprint arXiv:2608.20556*, 2026.

[20] Y. Meng and C. Fan, "Signal temporal logic neural predictive control," *RA-L*, vol. 8, no. 11, pp. 7719–7726, 2023.

[21] H. Parwana, T. Kim, K. Long, B. Hoxha, H. Okamoto, G. Fainekos, and D. Panagou, "Br-mppi: Barrier rate guided mppi for enforcing multiple inequality constraints with learned signed distance field," *arXiv preprint arXiv:2506.07325*, 2025.

[22] P. V. Dimos and D. V. Dimarogonas, "A learning framework for versatile stl controller synthesis," in *CDC*, pp. 4596–4600, IEEE, 2019.

[23] P. Varnai and D. V. Dimarogonas, "Prescribed performance control guided policy improvement for satisfying signal temporal logic tasks," in *ACC*, pp. 286–291, IEEE, 2019.

[24] P. Varnai and D. V. Dimarogonas, "Guided policy improvement for satisfying stl tasks using funnel adaptation," *arXiv preprint arXiv:2004.05653*, 2020.

[25] H. Zheng, Z. Zang, A. Amine, C. I. Vasile, and R. Mangharam, "Stl-svpio: Signal temporal logic guided stein variational path integral optimization," *arXiv preprint arXiv:2603.13333*, 2026.

[26] P. Halder, L. Kiltz, H. Homburger, J. Reuter, and M. Althoff, "Lexicographic minimum-violation motion planning using signal temporal logic," *arXiv preprint arXiv:2604.20428*, 2026.

[27] J. Han, C. Li, X. Hou, D. Pi, G. Yin, J. Zhao, P. K. Wong, and Y. Wang, "Signal temporal logic constrained motion control of electric vehicles under slippery roads using an mppi approach," *IEEE Transactions on Transportation Electrification*, 2026.

[28] T. Bouzid, S. Qi, M. Lazar, and S. Haesaert, "Autonomous driving with priority-ordered stl specifications under multimodal uncertainty," *arXiv preprint arXiv:2606.20336*, 2026.

[29] E. Bartocci, J. Deshmukh, A. Donzé, G. Fainekos, O. Maler, D. Ničković, and S. Sankaranarayanan, "Specification-based monitoring of cyber-physical systems: a survey on theory, tools and

applications," in *Lectures on Runtime Verification: Introductory and Advanced Topics*, pp. 135–175, Springer, 2018.
[30] S. Sadraddini and C. Belta, "Robust temporal logic model predictive control," in *Allerton*, pp. 772–779, IEEE, 2015.
[31] A. Donzé and O. Maler, "Robust satisfaction of temporal logic over real-valued signals," in *FORMATS*, pp. 92–106, Springer, 2010.
[32] C. Tao, H. Kim, H. Yoon, N. Hovakimyan, and P. Voulgaris, "Control barrier function augmentation in sampling-based control algorithm for sample efficiency," in *ACC*, pp. 3488–3493, IEEE, 2022.
[33] J. Yin, C. Dawson, C. Fan, and P. Tsiotras, "Shield model predictive path integral: A computationally efficient robust mpc method using control barrier functions," *RA-L*, vol. 8, no. 11, pp. 7106–7113, 2023.
[34] L. Lindemann and D. V. Dimarogonas, *Formal Methods for Multi-Agent Feedback Control Systems*. MIT Press, 2025.
[35] G. Marchesini, S. Liu, L. Lindemann, and D. V. Dimarogonas, "Sampling-based planning under stl specifications: A forward invariance approach," *IEEE Transactions on Automatic Control*, 2026.
[36] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, *et al.*, "Isaac gym: High performance gpu-based physics simulation for robot learning," *arXiv preprint arXiv:2108.10470*, 2021.
[37] M. Mittal, C. Yu, Q. Yu, J. Liu, N. Rudin, D. Hoeller, J. L. Yuan, R. Singh, Y. Guo, H. Mazhar, A. Mandlekar, B. Babich, G. State, M. Hutter, and A. Garg, "Orbit: A unified simulation framework for interactive robot learning environments," *RA-L*, vol. 8, no. 6, pp. 3740–3747, 2023.
[38] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *NeurIPS*, vol. 32, 2019.
[39] U. o. M. Autonomous Robotic Manipulation Lab, "pytorch_mppi." https://github.com/UM-ARM-Lab/pytorch_mppi, 2024. Accessed: 2025-08-17.

VIII. Appendix

A. Proof for Theorem 1

Proof: The theorem can be proven by induction (similar to the proof of Property III.1 in [33]): We assumed $x_0 \in \mathcal{C}_0$. At step t, knowing $\alpha \in (0,1]$, we assume $x_t \in \mathcal{C}_t$ and $u_t \in S(x_t,t)$ and thus $h(f(x_t,u_t),t+1) \geq h(x_t,t) - \alpha h(x_t,t) \geq 0$, which implies $f(x_t,u_t) \in \mathcal{C}_{t+1}$. ■

B. Proof for Theorem 2

Proof: We are given the condition $x_0 \in C_0$. From Theorem 1, we know $x_{t_i} \in \mathcal{C}_{t_i} \implies x_t \in \mathcal{C}_t$ for $t \in \{t_i, \ldots, t_{i+1}\}$. Since the exponential function is strictly positive and logarithm is monotonic, we have $h(x,t_{i+1}) \underset{\sim}{\leq} h^+(x,t_{i+1})$ for all $x \in D$, where $h^+(x,t_{i+1}) := \widetilde{\min}(\mathcal{A}_{t_{i+1}+1}(x,t))$ denotes the deletion. The update is forward invariant ($h(x,t_{i+1}) \geq 0 \implies h^+(x,t_{i+1}) \geq 0$). Since $\widetilde{\min}$ lower-bounds each argument, $h^\phi(x_t,t) \geq 0$ implies $h_i(x_t,t) \geq 0$ for all $t \in \{t_i, \ldots, t_{i+1}\}$, including the endpoint t_{i+1}; by the conditions of Table I this implies satisfaction of each ϕ_i, and hence $(x,0) \models \phi$ by induction on the structure of the formula: For an atomic predicate π^μ, the implication follows directly from the first row of Table I. For $G_{[a,b]}$, Table I gives the predicate at every $t \in \sigma(\tau_0 + a, \tau_0 + b)$. For $F_{[a,b]}$, Table I gives a witness $t' \in \sigma(\tau_0 + a, \tau_0 + b)$. For $U_{[a.b]}$, Table I gives a witness for the right operand and the left operand up to that witness. Conjunction follows because the smooth minimum being nonnegative implies every constituent barrier is nonnegative. ■

C. Derivation of Equation (9)

For simple notation, let $\hat{v}^s := \hat{v}_\tau^{s,(t)}, v^+ := v_\tau^{+,(t)}, a := a_{\text{cbf}}(x_\tau^{(t)},\tau)$, and $b := b_{\text{cbf}}(x_\tau^{(t)},\tau)$. We assumed $a \neq 0$ in Section IV. Assume $\mathcal{U} := \mathbb{R}^m$. Then, we can write (8) as:

$$\hat{v}_\tau^s = \arg\min_{v \in \mathbb{R}^m} \{\frac{1}{2}\|v - v^+\|^2 \mid a^T v \leq b\}.$$

Suppose $a^T v^+ \leq b$, it is intuitive to have $\hat{v}^s = v^+$ because this corresponds to an already safe action, and the optimal objective cost is 0 and any $\hat{v}^s \neq v^+$ will result in a straight positive cost.

On the other hand, suppose $a^T v^+ > b$. We have the Lagrangian $\mathcal{L}(v,\mu) = \frac{1}{2}\|v - v^+\|^2 + \mu(a^T v - b)$. By the KKT conditions, the stationarity states $\hat{v}^s = v^+ - \mu a$ and the complementary slackness states $\mu(a^T \hat{v}^s - b) = 0$. Since $a^T v^+ > b$, the optimum must lie on the boundary and thus $a^T \hat{v}^s = b$. By substitution, we have $a^T(v^+ - \mu a) = b$ and thus it must hold that $\mu := (a^T v^+ - b)/\|a\|_2^2$. Therefore, $\hat{v}^s = v^+ - a[(a^T v^+ - b)/\|a\|_2^2]$. Note that here the KKT conditions are sufficient and necessary since (8) is strictly convex with affine feasible region containing nonempty interior.

Combining the above two cases, one can derive the closed form solution $\hat{v}^s = v^+ - a[\max(0, a^T v^+ - b)/\|a\|_2^2]$.

D. Theorem 3

Theorem 3: Suppose $\nabla h^\phi(\cdot,\tau+1)$ is $L_{x,\tau}$-lipschitz on a set K_x containing every segment $\{x + s(f(x,v) - x) : s \in [0,1]\}$ where $v \in \mathcal{U}$ (i.e., $\|\nabla_x h^\phi(x_1,\tau+1) - \nabla_x h^\phi(x_2,\tau+1)\| \leq L_{x,\tau}\|x_1 - x_2\|, \forall x_1, x_2 \in K_x$). Assume also $\Delta_x := \sup_{v \in \mathcal{U}} \|f(x,v) - x\| < \infty$. Then (6) defines a subset of the feasibility region to (5) if the right-hand side of (6) is incremented[10] by $\epsilon_{x,\tau} := \frac{L_{x,\tau}}{2}\Delta_x^2$.

Proof: Let us denote $P(\cdot) := h^\phi(\cdot,\tau+1)$ and $d_v = f(x,v) - x$. We can show the local version of the descent lemma. Per assumption, ∇P is $L_{x,\tau}$-lipschitz on $\{x + sd_v : s \in [0,1], \forall v \in \mathcal{U}\}$. Suppose $g_v(s) = P(x + sd_v)$, then

$$\begin{aligned}
&P(f(x,v)) - P(x) \\
&= \int_0^1 \nabla P(x + sd_v)^T d_v ds \\
&= \nabla P(x)^T d_v + \int_0^1 (\nabla P(x + sd_v) - \nabla P(x))^T d_v ds.
\end{aligned}$$

We then have by Cauchy-Schwarz and our lipschitz assumption that

$$\begin{aligned}
&(\nabla P(x + sd_v) - \nabla P(x))^T d_v \\
&\geq -\|\nabla P(x + sd_v) - \nabla P(x)\|\|d_v\| \\
&\geq -L_{x,\tau} s \|d_v\|^2.
\end{aligned}$$

Therefore, we have

$$P(f(x,v)) - P(x) \geq \nabla P(x)^T d_v - \frac{L_{x,\tau}}{2}\|d_v\|^2$$

[10] The norms in Theorem 3 are Euclidean norms.

which holds for all $v \in \mathcal{U}$. By assumption, we have $\|d_v\| = \|f(x,v) - x\| \leq \Delta_x, \forall v \in \mathcal{U}$. We thus have

$$P(x) + \nabla P(x)^T d_v - \frac{L_{x,\tau}}{2}\Delta_x^2 \leq P(f(x,v))$$

Hence, if

$$\begin{aligned} & P(x) + \nabla P(x)^T d_v - \frac{L_{x,\tau}}{2}\Delta_x^2 \\ \coloneqq & h^\phi(x,\tau+1) + \nabla_x h^\phi(x,\tau+1)[(f_0(x) - x) \\ & + g(x)v] - \frac{L_{x,\tau}}{2}\Delta_x^2 \geq (1-\alpha)h^\phi(x,\tau), \end{aligned}$$

then it holds that

$$h^\phi(f(x,v),\tau+1) \geq (1-\alpha)h^\phi(x,\tau).$$

■

By Theorem 3, we can derive a robustified algorithm (which we call r-safety-aware-stl-mppi), where we replace $b_{\text{cbf}}(x_\tau^{(t)},\tau)$ in (8) and (9) with $b^r_{\text{cbf}}(x_\tau^{(t)},\tau) \coloneqq b_{\text{cbf}}(x_\tau^{(t)},\tau) - \epsilon_{x,\tau}$ in Algorithm 1.

Computing $L_{x,\tau}$ is generally nontrivial, and one can use a heuristic sampling-based estimation

$$L_{x,\tau} \approx \max_{x_i,x_j \in K_x, i\neq j} \frac{\|\nabla_x h^\phi(x_i,\tau+1) - \nabla_x h^\phi(x_j,\tau+1)\|}{\|x_i - x_j\|}. \tag{10}$$

However, for control affine dynamics $f(x,u) = f_0(x) + g(x)u$ and boxed input constraints $\mathcal{U}$, it is easy to show that $\Delta_x = \max_{v\in \text{vert}(\mathcal{U})} \|f_0(x) - x + g(x)v\|$ where vert$(\cdot)$ denote the set of box vertices. Since $\mathcal{U}$ is a convex hull of its vertices $w_1,\ldots,w_k$, for any $v \in U$, we can express $v = \sum_{i=1}^k \lambda_i w_i$ where $\lambda_i \geq 0$ are coefficients such that $\sum_i \lambda_i = 1$. Suppose $p(v) = \|(f_0(x) - x) + g(x)v\|_2$. Since p is convex, we have

$$p(v) = p(\sum_{i=1}^k \lambda_i w_i) \leq \sum_{i=1}^k \lambda_i p(w_i) \leq \max_i p(w_i)$$

and we can conclude that **in this setting**

$$\Delta_x = \max_{v\in\text{vert}(\mathcal{U})} \|f_0(x) - x + g(x)v\|_2.$$

We remark on the conservatism of this robustification. Also, compared to safety-aware-stl-mppi, the robustified method is more time consuming due to the computation of $\epsilon_{x,\tau}$.