IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

SIT-LMPC: Safe Information-Theoretic Learning Model Predictive Control for Iterative Tasks

Zirui Zang, Ahmad Amine, Nick-Marios T. Kokolakis, Truong X. Nghiem, Ugo Rosolia, and Rahul Mangharam

***Abstract*—Robots executing iterative tasks in complex, uncertain environments require control strategies that balance robustness, safety, and high performance. This paper introduces a safe information-theoretic learning model predictive control (SIT-LMPC) algorithm for iterative tasks. Specifically, we design an iterative control framework based on an information-theoretic model predictive control algorithm to address a constrained infinite-horizon optimal control problem for discrete-time nonlinear stochastic systems. An adaptive penalty method is developed to ensure safety while balancing optimality. Trajectories from previous iterations are utilized to learn a value function using normalizing flows, which enables richer uncertainty modeling compared to Gaussian priors. SIT-LMPC is designed for highly parallel execution on graphics processing units, allowing efficient real-time optimization. Benchmark simulations and hardware experiments demonstrate that SIT-LMPC iteratively improves system performance while robustly satisfying system constraints.**

I. Introduction

ITERATIVE tasks are ubiquitous in robotics, spanning applications such as quadrotor racing [1], quadruped motion planning [2], and surgical robotics [3]. The core challenge in these settings lies in improving performance over successive executions, leveraging data from prior attempts to refine future behavior [4]. To achieve this, a variety of approaches have been explored, including deep reinforcement learning (RL) [5], genetic algorithms [6], and optimal control [2, 7]. These iterative tasks usually involve additional complexities, such as navigating dynamic environments with obstacles [8] or ensuring safe human-robot interaction [9]. Thus, balancing performance with safety through constraint satisfaction during training is a key requirement.

Iterative learning control (ILC) improves system performance by using error information from previous task executions to refine future control signals [10]. *Learning model predictive control* (LMPC) [11] is a reference-free variant of ILC that iteratively constructs a controlled invariant terminal constraint set (safe set) and a terminal cost function within a model predictive control (MPC) framework, optimizing for solutions to constrained infinite-horizon optimal control problems over successive iterations. LMPC ensures safety by enforcing state constraints throughout the MPC horizon and enforcing the terminal state to reside within the safe set. The control method converges asymptotically to the optimal controller for deterministic linear systems with quadratic costs [12]. For stochastic systems, [13] specializes LMPC to linear systems with state noise by constructing robust safe sets from previous trajectories and learning a terminal cost function representing the value function associated with the control policies used for collecting data. These safe sets and terminal cost function can be approximated using a finite number of prior trajectories while ensuring that the worst-case iteration cost is non-increasing [14]. Recently, in adjustable boundary condition (ABC)-LMPC [15, 16], the LMPC framework is extended to stochastic nonlinear dynamical systems by using a sampling-based cross-entropy method (CEM) MPC to repeatedly sample trajectories until all samples satisfy the terminal set constraint and select the least-cost sampled trajectory [17]. Although ABC-LMPC theoretically extends the LMPC framework to stochastic nonlinear dynamical systems, the state constraints are encoded with a high constant cost in the cost function, which yields overly conservative control. Furthermore, it has been reported that the CEM sampling leads to infeasible solutions for stochastic systems and is susceptible to mode collapse in high-dimensional nonlinear systems [18].

Information-theoretic MPC or *model predictive path integral control* (MPPI) [19] is a sampling-based MPC algorithm for stochastic systems. MPPI synthesizes the optimal control by minimizing the Kullback-Leibler (KL) divergence between the optimal control distribution and the sampled control distribution [19]. Stochasticity is handled by sampling trajectories and optimizing over their expected cost without requiring gradient information. The sampling process can be parallelized on a graphics processing unit (GPU), enabling real-time control. Previous work has demonstrated that MPPI outperforms CEM-MPC in terms of safety and cost [20]. However, MPPI is an unconstrained optimal control method, and state constraints are typically incorporated through clamping in the dynamics or a high cost on violations [20]. Alternatively, state constraints can be satisfied by projecting unsafe sampled trajectories onto a feasible set for differentially flat systems [21] or by incorporating a control barrier function (CBF) into the cost function combined with a gradient-based local repair step [22]. These approaches can ensure safety, but rely on special assumptions about model dynamics or the existence of a valid CBF, which is hard to realize for real-world data-

Manuscript Received: June 26, 2025; Revised: September 28, 2025; Accepted: October 24, 2025.

This paper was recommended for publication by Editor Soon-Jo Chung upon evaluation of the Associate Editor and Reviewers' comments.

This work was partially supported by US DoT Safety21 National University Transportation Center and NSF grants CISE-2431569 and 2514584.

Z. Zang, A. Amine, N.-M. T. Kokolakis, and R. Mangharam are with the University of Pennsylvania, Philadelphia, PA 19104, USA. E-mail: {zzang, aminea, nmkoko, rahulm}@seas.upenn.edu.

T. X. Nghiem is with the Department of Electrical and Computer Engineering, University of Central Florida, Orlando, FL 32816, USA. E-mail: truong.nghiem@ucf.edu.

U. Rosolia is with Lyric. E-mail: ugo.rosolia@gmail.com.

Z. Zang and A. Amine contributed equally to this work.

Supplementary videos: https://sites.google.com/view/sit-lmpc/

Digital Object Identifier (DOI): see top of this page.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

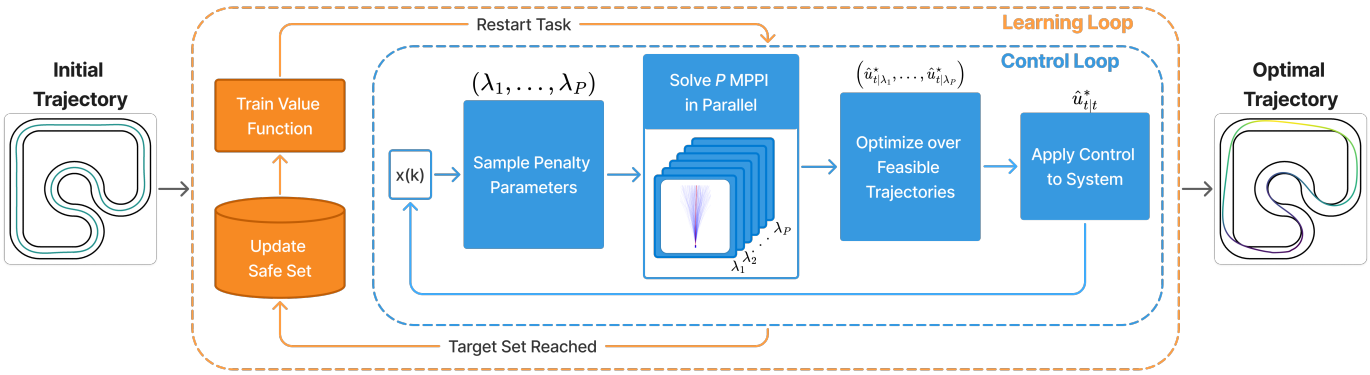


Fig. 1. SIT-LMPC architecture: starting from an initial trajectory, the algorithm iteratively updates the safe set and value function model (orange loop), while solving multiple MPPI problems in parallel (blue loop) to generate optimal trajectories.

based scenarios. To the best of our knowledge, there is no constrained MPPI that can handle general state constraints.

The main contribution of this paper is the development of a safe iterative learning control framework for general stochastic nonlinear systems by extending the LMPC formulation and solving the resulting optimization problem by designing a constrained information-theoretic MPC algorithm. Our proposed approach is general and does not rely on assumptions about the system's dynamics and state constraints. To efficiently and effectively balance optimality and safety, we develop an online sampling-based adaptive penalty method. We learn the value function using normalizing flows by leveraging trajectories from previous iterations, enabling richer uncertainty modeling than Gaussian priors. We provide a fully parallelized deployment of our method, enabling 100Hz+ real-time control on a scaled off-road vehicle with an NVIDIA Jetson Orin AGX. The architecture of the proposed SIT-LMPC framework is illustrated in Fig. 1.

II. Problem Formulation

In this section, we state the *constrained infinite horizon optimal control problem* for discrete-time nonlinear stochastic systems to characterize an admissible feedback controller that *i*) renders a set of admissible states and a target set robust controlled invariant, *ii*) robustly drives the system state to the target set asymptotically, and *iii*) optimizes the system performance.

Consider the discrete-time nonlinear stochastic dynamical system given by

$$x(k+1) = f\big(x(k), u(k), w(k)\big), \quad x(0) \overset{\text{a.s.}}{=} x_0, \tag{1}$$

where, for every $k \in \overline{\mathbb{Z}}_+ \triangleq \{0, 1, 2, \ldots\}$, $x(k) \in \mathbb{R}^{n_x}$ is a state vector, $u(k) \in \mathbb{R}^{n_u}$ is a control input, $w(k) \in \mathcal{W} \subset \mathbb{R}^{n_w}$ is an independent and identically distributed (i.i.d.) stochastic disturbance process with $\mathcal{W}$ being a bounded set, and $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathcal{W} \to \mathbb{R}^{n_x}$ is jointly continuous in x, u, and w. The initial condition $x(0)$ is assumed to be a constant vector, almost surely (a.s.) equal to x_0.

Let $\mathcal{X} \subset \mathbb{R}^{n_x}$ be a set of *admissible states* and let $\mathcal{U} \subset \mathbb{R}^{n_u}$ be a set of *admissible control inputs*. Furthermore, let $\mathcal{T} \subset \mathcal{X}$ be a set of *target states* and assume that $\mathcal{T}$ is a *robust controlled invariant set* [23] with respect to the discrete-time nonlinear stochastic dynamical system (1) and the admissible control inputs $\mathcal{U}$, that is, for every $x \in \mathcal{T}$, there exists $u(x) \in \mathcal{U}$ such that $f(x, u(x), w) \in \mathcal{T}$ for all $w \in \mathcal{W}$. In other words, the target set $\mathcal{T}$ is a robust controlled invariant set with respect to (1) and $\mathcal{U}$ if, for every initial condition $x_0 \in \mathcal{T}$, there exists an admissible feedback control law $u : \mathcal{T} \to \mathcal{U}$ such that the solution sequence $x(k)$, $k \in \overline{\mathbb{Z}}_+$, to (1) remains a.s. in $\mathcal{T}$ for all disturbance sequences $w(\cdot)$. A set $\mathcal{S} \subset \mathcal{X}$ is said to be *safe* if it is a robust controlled invariant set with respect to (1) and $\mathcal{U}$. Hence, $\mathcal{T}$ is safe.

To evaluate the performance of the discrete-time nonlinear stochastic dynamical system (1) over the infinite horizon, we define, for every $x_0 \in \mathbb{R}^{n_x}$ and control input $u(k)$, $k \in \overline{\mathbb{Z}}_+$, the performance measure

$$J\big(x_0, u(\cdot)\big) \triangleq \mathbb{E}\left[\sum_{k=0}^{\infty} h\big(x(k), u(k)\big)\right], \tag{2}$$

where $h : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}$ is the stage cost satisfying $h(x, u) = 0$ for every $(x, u) \in \mathcal{T} \times \mathbb{R}^{n_u}$ and $h(x, u) > 0$ for every $(x, u) \in (\mathbb{R}^{n_x} \backslash \mathcal{T}) \times \mathbb{R}^{n_u}$. That is, the target set $\mathcal{T}$ incurs zero cost.

The goal is the synthesis of an admissible feedback control law $u^\star : \mathcal{X} \to \mathcal{U}$ that renders $\mathcal{X}$ and $\mathcal{T}$ robust controlled invariant, robustly drives, for every $x_0 \in \mathcal{X}$, the state $x(k)$, $k \in \overline{\mathbb{Z}}_+$, to the target set $\mathcal{T}$ as $k \to \infty$, and minimizes the performance measure (2).

In light of the above, our control problem can be cast as a *constrained infinite horizon optimal control problem* given by

$$J^\star(x_0) \triangleq \min_{u(\cdot) \in \mathcal{F}(x_0)} J\big(x_0, u(\cdot)\big) \tag{3a}$$

subject to

$$x(k+1) = f\big(x(k), u(k), w(k)\big), \quad k \in \overline{\mathbb{Z}}_+, \tag{3b}$$

$$x(0) \overset{\text{a.s.}}{=} x_0 \in \mathcal{X}, \tag{3c}$$

$$x(k) \in \mathcal{X}, \quad u(k) \in \mathcal{U}, \quad k \in \overline{\mathbb{Z}}_+, \tag{3d}$$

where $\mathcal{F}(x_0)$ denotes the set of *admissible feedback stabilizing*

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

controllers, which is assumed to be nonempty and defined as

$$\mathcal{F}(x_0) \triangleq \big\{u : \mathcal{X} \to \mathcal{U} : x(\cdot) \text{ given by (1) satisfies a.s. } x(k) \in \mathcal{X},\ k \in \overline{\mathbb{Z}}_+,\ \lim_{k\to\infty} x(k) \in \mathcal{T}, \text{ and } J\big(x_0, u(\cdot)\big) < \infty\big\}.$$

III. Safe Information-Theoretic Learning Model Predictive Control

The constrained infinite-horizon optimal control problem (3) involves an optimization over infinite-dimensional function spaces, and hence, it is challenging to solve. In this section, we build on LMPC [11], MPPI [20], and penalty methods for constrained optimization [24] to develop the SIT-LMPC algorithm to address this problem. Specifically, we learn the solution to problem (3) by iteratively executing the constrained regulation task from the initial condition x_0. System trajectories associated with feasible iterations are stored and used to iteratively synthesize a sampling-based predictive control policy and learn a value function. Constraints are enforced by developing an adaptive penalty method while balancing optimality.

A. Stochastic LMPC Formulation

For every time step $k \in \overline{\mathbb{Z}}_+$, let $x^l(k)$, $u^l(k)$, and $w^l(k)$ be the system state, the control input, and the stochastic disturbance at the lth iteration. We assume that at every lth iteration, the closed-loop system trajectory starts a.s. from the initial state $x_0 \in \mathcal{X}$, namely, $x^l(0) \overset{\text{a.s.}}{=} x_0,\ l \in \mathbb{Z}_+$. An lth iteration is said to be *feasible* if $x^l(k) \in \mathcal{X}$ and $u^l(k) \in \mathcal{U},\ k \in \overline{\mathbb{Z}}_+$, and $\lim_{k\to\infty} x^l(k) \in \mathcal{T}$. In other words, the lth iteration is feasible if the system trajectory $x^l(\cdot)$ satisfies the state constraints and converges asymptotically to the target set $\mathcal{T}$ while $x^l(\cdot)$ is generated by an admissible control input $u^l(\cdot)$.

Let $\mathcal{I}^l \triangleq \{i \in [0,\ l] : i\text{th iteration is feasible}\}$ be the set of indices of feasible iterations up to the lth iteration. Define the *sampled safe set* $\mathcal{S}^l$ at the lth iteration as

$$\mathcal{S}^l \triangleq \left\{x^i(k) : i \in \mathcal{I}^l \text{ and } k \in \overline{\mathbb{Z}}_+\right\}, \tag{4}$$

which is a collection of all states along the system trajectories $x^i(\cdot),\ i \in \mathcal{I}^l$, of feasible iterations up to the lth iteration. Following [14], we define $\mathcal{CS}^l$ as the convex hull of $\mathcal{S}^l$, which is used as a *terminal constraint set* in our framework.

For every iteration $l \in \mathbb{Z}_+$, the lth *iteration cost* is defined by $J^l(x_0, u^l(\cdot)) \triangleq \sum_{k=0}^{\infty} h(x^l(k), u^l(k))$, which evaluates the performance of the controller $u^l(\cdot)$. Note that $J^l(x_0, u^l(\cdot))$ is a random variable depending on the realization of the stochastic disturbance $w(\cdot)$. The lth iteration *value function* $V^l(\cdot)$ is defined by $V^l(x_0) \triangleq \mathbb{E}[J^l(x_0, u^l(\cdot))],\ x_0 \in \mathcal{X}$, and serves as a *terminal cost function* in our SIT-LMPC algorithm.

Now, consider the *constrained finite-time optimal control problem*, for every iteration $l \in \mathbb{Z}_+$ and time $t \in \overline{\mathbb{Z}}_+$, given the terminal cost function $V^l(\cdot)$, the terminal constraint set $\mathcal{S}^{l-1}$, and a prediction horizon $T \in \mathbb{Z}_+$,

$$J^l_{\text{LMPC}}\left(x^l(t)\right) \triangleq \min_{u^l_{\cdot|t}} \mathbb{E}\left[\sum_{k=t}^{t+T-1} h(x^l_{k|t}, u^l_{k|t}) + V^l(x^l_{t+T|t})\right] \tag{5a}$$

subject to

$$x^l_{k+1|t} = f\big(x^l_{k|t}, u^l_{k|t}, w^l(k)\big), \quad k \in \{t, \dots, t+T-1\} \tag{5b}$$

$$x^l_{t|t} \overset{\text{a.s.}}{=} x^l(t), \tag{5c}$$

$$x^l_{k|t} \in \mathcal{X}, \quad u^l_{k|t} \in \mathcal{U} \quad \text{a.s.}, \quad k \in \{t, \dots, t+T-1\}, \tag{5d}$$

$$x^l_{t+T|t} \in \mathcal{CS}^{l-1} \quad \text{a.s.}, \tag{5e}$$

where, for every $k \in \{t, \dots, t+T\}$, $x^l_{k|t}$ and $u^l_{k|t}$ denote the predicted value of state $x^l(k)$ and input $u^l(k)$ given a measurement $x^l(t)$, and $u^l_{\cdot|t}$ denotes the predicted control sequence defined by $u^l_{\cdot|t} \triangleq [u^l_{t|t}, u^l_{t+1|t}, \dots, u^l_{t+T-1|t}]$.

Remark 1: The initial sampled safe set $\mathcal{S}^0$ is assumed to be nonempty and contains a collection of admissible state trajectories, generated by admissible controllers, that asymptotically converge to $\mathcal{T}$. For instance, $\mathcal{S}^0$ can be constructed from safe human demonstrations or low-performance safe controllers.

B. Safe MPPI via an Online Adaptive Penalty Method

Since MPPI does not consider state constraints, MPPI cannot be used to solve the constrained finite-time optimal control problem (5). To address this limitation, we convert the constrained finite-time optimal control problem (5) into an *unconstrained finite-time optimal control problem*. To measure the violation of state constraints for every $x \in \mathbb{R}^{n_x}$, we integrate the *exterior penalty functions* $d_{\mathcal{X}}(\cdot)$ and $d_{\mathcal{CS}^{l-1}}(\cdot),\ l \in \mathbb{Z}_+$, associated with the set of admissible states $\mathcal{X}$ and the safe set $\mathcal{CS}^{l-1}$ into the stage cost $h(\cdot,\cdot)$ and the terminal cost $V^l(\cdot)$, respectively.

Define the distance of a point $x \in \mathbb{R}^{n_x}$ to a closed set $C \subseteq \mathbb{R}^{n_x}$ in the Euclidean norm $\|\cdot\|_2$ as $\operatorname{dist}(x, C) \triangleq \inf_{y\in C}\{\|x-y\|_2\}$. The exterior penalty functions are defined as $d_{\mathcal{X}}(x) \triangleq \operatorname{dist}(x, \mathcal{X})$ and $d_{\mathcal{CS}^{l-1}}(x) \triangleq \operatorname{dist}(x, \mathcal{CS}^{l-1})$ with associated non-negative *penalty parameters* $\lambda_{\mathcal{X}}$ and $\lambda_{\mathcal{CS}^{l-1}}$. For every iteration $l \in \mathbb{Z}_+$ and time $t \in \overline{\mathbb{Z}}_+$, the *unconstrained finite-time optimal control problem* is given by

$$J^l_{\text{SIT-LMPC}}\left(x^l(t)\right) \triangleq \min_{u^l_{\cdot|t}} \mathbb{E}\Bigg[\sum_{k=t}^{t+T-1} \Big(h(x^l_{k|t}, u^l_{k|t}) + \lambda_{\mathcal{X}} d_{\mathcal{X}}(x^l_{k|t})\Big) + V^l(x^l_{t+T|t}) + \lambda_{\mathcal{CS}^{l-1}} d_{\mathcal{CS}^{l-1}}(x^l_{t+T|t})\Bigg] \tag{6a}$$

subject to

$$x^l_{k+1|t} = f\big(x^l_{k|t}, u^l_{k|t}, w^l(k)\big), \quad k \in \{t, \dots, t+T-1\}, \tag{6b}$$

$$x^l_{t|t} \overset{\text{a.s.}}{=} x^l(t), \tag{6c}$$

$$u^l_{k|t} \in \mathcal{U} \quad \text{a.s.}, \qquad k \in \{t, \dots, t+T-1\}. \tag{6d}$$

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

We use MPPI [19] to solve the stochastic optimal control problem (6) from an information-theoretic perspective by minimizing the KL divergence between the optimal predicted control distribution $\mathbb{Q}^{\star l}$ and the open-loop control distribution $\mathbb{Q}^{l}$. Hence, the optimal control is given by

$$u^{\star l}_{k|t} \triangleq \underset{u^{l}_{\cdot|t}}{\operatorname{argmin}} D_{\mathrm{KL}}(\mathbb{Q}^{\star l} \parallel \mathbb{Q}^{l}), \quad k \in \{t, \dots, t+T-1\}. \tag{7}$$

To optimize this KL divergence, we first sample N trajectories $x^{l_s}_{\cdot|t},\ s = 1, \dots, N$, starting from $x^l(t)$, which are generated by sampling control sequences $u^{l_s}_{\cdot|t},\ s = 1, \dots, N$, from a *truncated* normal distribution $\mathcal{N}_{\text{trunc}}(u^{\star l}_{\cdot|t-1}, \Sigma, \mathcal{U})$, where $u^{\star l}_{\cdot|t-1}$ is the mean value of the *parent* normal distribution, Σ is the user-prescribed covariance of the *parent* normal distribution, and $\mathcal{U}$ specifies the truncation range so that the input constraints (6d) are satisfied [25]. In practice, sampling of the control sequences $u^{l_s}_{\cdot|t}$ and simulation of the N trajectories generated by these control inputs are executed in parallel on a GPU. Then, we perform *importance sampling* on $u^{l}_{\cdot|t}$ to obtain the approximate optimal predicted control sequence as

$$\hat{u}^{\star l}_{k|t} = \sum_{s=1}^{N} w_s u^{l_s}_{k|t}, \quad k \in \{t, \dots, t+T-1\}, \tag{8}$$

with importance sampling weights w_s given by

$$w_s = \frac{\exp\left(-\frac{1}{\tau} J^{l}_{\text{sampled}}\big(x^l(t), x^{l_s}_{\cdot|t}, u^{l_s}_{\cdot|t}\big)\right)}{\sum_{s=1}^{N} \exp\left(-\frac{1}{\tau} J^{l}_{\text{sampled}}\big(x^l(t), x^{l_s}_{\cdot|t}, u^{l_s}_{\cdot|t}\big)\right)}, \tag{9}$$

where $s = 1, \dots, N$, $\tau > 0$ is a temperature parameter that tunes the sharpness of importance sampling, and $J^{l}_{\text{sampled}}\big(x^l(t), x^{l_s}_{\cdot|t}, u^{l_s}_{\cdot|t}\big)$ is a sampled cost function defined as

$$\begin{aligned} &J^{l}_{\text{sampled}}\big(x^l(t), x^{l_s}_{\cdot|t}, u^{l_s}_{\cdot|t}\big) \\ &\triangleq \sum_{k=t}^{t+T-1} \Big(h(x^{l_s}_{k|t}, u^{l_s}_{k|t}) + \lambda_{\mathcal{X}} d_{\mathcal{X}}(x^{l_s}_{k|t})\Big) + V^l(x^{l_s}_{t+T|t}) \\ &\quad + \lambda_{\mathcal{CS}^{l-1}} d_{\mathcal{CS}^{l-1}}(x^{l_s}_{t+T|t}), \quad s = 1, \dots, N. \end{aligned} \tag{10}$$

However, assuming that the penalty parameters $\lambda_{\mathcal{X}}$ and $\lambda_{\mathcal{CS}^{l-1}}$ are arbitrary large constants may yield conservative system behavior, where safety dominates performance. Hence, we develop an online sampling-based *adaptive penalty* (AP) method that generates the penalty parameters $\lambda_{\mathcal{X}}$ and $\lambda_{\mathcal{CS}^{l-1}}$ in real-time, allowing for a balance between *optimality* and *safety*. Specifically, we sample the penalty parameters $\lambda_{\mathcal{X}}$ and $\lambda_{\mathcal{CS}^{l-1}}$ from uniform distributions, that is, $\lambda_{\mathcal{X}} \sim \mathrm{Unif}\big(0, \lambda^{\max}_{\mathcal{X}}\big)$ and $\lambda_{\mathcal{CS}^{l-1}} \sim \mathrm{Unif}\big(0, \lambda^{\max}_{\mathcal{CS}^{l-1}}\big)$, with $\lambda^{\max}_{\mathcal{X}},\ \lambda^{\max}_{\mathcal{CS}^{l-1}} > 0$. Let $\lambda \triangleq [\lambda_{\mathcal{X}},\ \lambda_{\mathcal{CS}^{l-1}}]^{\mathrm{T}}$ be the augmented penalty parameter vector and let $\Lambda \subset \Omega \triangleq [0, \lambda^{\max}_{\mathcal{X}}] \times [0, \lambda^{\max}_{\mathcal{CS}^{l-1}}]$ be a finite set of samples drawn from the joint uniform distribution. For every $\lambda \in \Lambda$, we solve the optimal control problem (6) whose approximate solution (8) is now parametrized by λ and written as $\hat{u}^{\star l}_{\cdot|t,\lambda}$. We then construct the set of penalty parameters that generate feasible trajectories as

$$\begin{aligned} \mathcal{F}^l_t = \big\{\lambda \in \Lambda : \hat{x}^{\star l}_{k|t,\lambda} \in \mathcal{X},\ k \in \{t, \dots, t+T\}, \\ \hat{x}^{\star l}_{t+T|t,\lambda} \in \mathcal{CS}^{l-1}\big\}, \quad l \in \mathbb{Z}_+, \quad t \in \overline{\mathbb{Z}}_+. \end{aligned}$$

The optimal penalty parameter $\lambda^\star$ is given by

$$\lambda^\star = \begin{cases} \underset{\lambda \in \mathcal{F}^l_t}{\operatorname{argmin}} J^{l}_{\text{sampled}}\big(x^l(t), \hat{x}^{\star l}_{\cdot|t,\lambda}, \hat{u}^{\star l}_{\cdot|t,\lambda}\big), & \text{if } \mathcal{F}^l_t \neq \varnothing, \\ \underset{\lambda \in \Lambda}{\operatorname{argmin}} \left(\sum_{k=t}^{t+T-1} d_{\mathcal{X}}(\hat{x}^{\star l}_{k|t,\lambda}) + d_{\mathcal{CS}^{l-1}}(\hat{x}^{\star l}_{t+T|t,\lambda})\right), \\ \qquad\qquad \text{if } \mathcal{F}^l_t = \varnothing. \end{cases} \tag{11}$$

In other words, the optimal penalty parameter is chosen to minimize the sampled cost J^{l}_{sampled} over the set of feasible parameters, or, if this set is empty, to minimize the constraint violation. Note that sampling the penalty parameters λ and executing MPPI are performed in parallel, ensuring efficient computation.

Now, we apply the first component $\hat{u}^{\star l}_{t|t,\lambda^\star}$ of the approximate optimal predicted control sequence $\hat{u}^{\star l}_{\cdot|t,\lambda^\star}$ to the system (1), that is, $u^l(t) = \hat{u}^{\star l}_{t|t,\lambda^\star}$. Next, given the new observed state $x^l(t+1)$, the unconstrained finite-time optimal control problem (6) is solved at time $t+1$ with $x^l_{t+1|t+1} \overset{\text{a.s.}}{=} x^l(t+1)$, yielding a receding-horizon control strategy.

Remark 2: Interior penalty functions, such as log-barrier functions, would yield infinite costs for any trajectory with a state that violates the constraints, dominating the importance sampling.

IV. Practical Implementation

This section outlines two key challenges associated with implementing SIT-LMPC on a real-world platform.

A. Iteration Value Function Estimation using Normalizing Flows

Implementing SIT-LMPC requires approximating the lth iteration value function $V^l(x),\ x \in \mathcal{X}$, since a closed-form expression for $V^l(\cdot)$ is generally intractable. To this end, we use *normalizing flows* (NFs) to model the distribution of the lth iteration cost $J^l(x, u^l(\cdot)),\ x \in \mathcal{X}$, from the collected data. NFs [26] are a class of *generative models* that transform a simple latent probability distribution z into a complex target distribution via a sequence of invertible and differentiable mappings g_θ parameterized by θ. NFs are suitable for stochastic systems since they directly learn complex probability distributions, rather than relying on deterministic estimates or restrictive assumptions, such as Gaussian priors. We choose NFs over other probabilistic modeling methods due to their richer expressiveness [26] and computational efficiency for real-time deployment.

We use *neural spline flows* (NSFs) [27] to learn the conditional distribution of $J^l(x, u^l(\cdot))$ given a state $x \in \mathcal{X}$. In our implementation, we use an NSF with eight spline segments and four flow layers, each of which has two fully-connected layers with 96 hidden states. For each iteration $l \in \mathbb{Z}_+$, we train the NSF for 300 epochs with a learning rate of 10^{-4} on the dataset

$$\mathcal{D}^{l-1} \triangleq \Big\{\Big(x^i(k), J^i\left(x^i(k), u^i(\cdot)\right)\Big) : i \in \mathcal{I}^{l-1},\ k \in \overline{\mathbb{Z}}_+\Big\}, \tag{12}$$

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

which is initialized with $\mathcal{S}^0$ and subsequently built with system states and corresponding iteration costs collected from feasible iterations up to the $l-1$th iteration. Specifically, the estimated conditional distribution of $J^l(x, u^l(\cdot))$ is denoted by $\hat{J}^l$ and given by $\hat{J}^l = g_\theta^l(z, x)$, where g_θ is an invertible transformation, z is a sample from the latent distribution $\mathcal{N}(0, 1)$, and x is the system state used as context input. We train our model using stochastic gradient descent by minimizing a negative log-likelihood loss function given by

$$\mathcal{L}^l(\theta) = -\sum_{\mathcal{D}^{l-1}} \log\left(\frac{1}{\sqrt{2\pi}} e^{-\frac{\left(g_\theta^{l\,-1}(\hat{J}^l, x)\right)^2}{2}} \left|\frac{\partial g_\theta^{l\,-1}(\hat{J}^l, x)}{\partial \hat{J}^l}\right|\right),$$

where $g_\theta^{l\,-1}$ is the inverse function of g_θ^l. When performing inference on $\hat{J}^l$ for an arbitrary state $x \in \mathcal{X}$, we first sample a batch of latent variables z from the latent distribution, then evaluate the NSF conditioned on x, and finally compute the iteration value function estimate as the *expected value* of the iteration cost function estimates, that is, $V_\theta^l(x) = \mathbb{E}_z\left[g_\theta^l(z, x)\right]$. As shown in the ablation study in Section V-B, our NSF method for modeling the conditional distribution of the iteration cost function outperforms the Bayesian neural network (BNN) approach used in [16].

Algorithm 1 outlines our iterative learning procedure. At each iteration, the system executes a trajectory until either the target set is reached or feasibility is lost in steps 4 and 5, respectively. The approximate optimal control is computed in step 8 using Algorithm 2, which is the AP-MPPI scheme detailed in Section III-B. The trajectory states and the associated cost-to-go are collected into $\mathcal{D}^l$ in step 14, which is then used to update the safe set $\mathcal{S}^l$ and train the NF iteration cost function model g_θ^{l+1}.

Algorithm 1 SIT-LMPC

Require: N: number of trajectory samples, T: control horizon, P: number of λ samples, L: maximum iterations, $\mathcal{S}^0$: initial safe-set.

1: Initialize g_θ^1 with random weights
2: Sample $\Lambda \leftarrow \{\lambda_i\}$ for $i = 1, \ldots, P$ with $\lambda_i \sim \text{Unif}(\Omega)$
3: **for** $l = 1, \ldots, L$ **do**
4: **while** $x^l(t) \notin \mathcal{T}$ **do** ▷ *run until target set is reached*
5: **if** $x^l(t) \notin \mathcal{X}$ **then** ▷ *check for constraint violation*
6: **terminate**
7: **end if**
8: $\hat{u}^{\star l}_{t|t,\lambda^\star} \leftarrow$AP-MPPI$\big(x^l(t), \mathcal{S}^{l-1}, g_\theta^l, \Lambda, P, N, T\big)$
9: Apply $\hat{u}^{\star l}_{t|t,\lambda^\star}$ to system (1)
10: **end while**
11: **if** $\bigcup_{t=0}^{\infty}\left\{x^l(t)\right\} \subset \mathcal{X}$ **then** ▷ *if iteration is feasible*
12: $\mathcal{S}^l \leftarrow \mathcal{S}^{l-1} \cup \left(\bigcup_{t=0}^{\infty}\left\{x^l(t)\right\}\right)$
13: Build $\mathcal{D}^l$ using (12)
14: **end if**
15: Train g_θ^{l+1} on $\mathcal{D}^l$ ▷ *NF from Section IV-A*
16: **end for**

B. Parallelizing over Control and Penalty Parameter Sampling

SIT-LMPC is tailored to use parallelized computation to achieve low-latency control. Algorithm 2 implements the SIT-LMPC control loop. Solving the optimal control problem (6a) involves two layers of sampling: control sequences for MPPI and penalty parameters for the *online adaptive penalty method*. First, we sample control inputs and generate state trajectories in step 1-4. Then, we evaluate the cost function (10) for every sampled penalty parameter pair $(\lambda_{\mathcal{CS}^l}, \lambda_{\mathcal{X}})$ in step 7 and perform the importance sampling in step 8. To reduce computation, the sampled trajectories $x^{l_s}_{\cdot|t}$ are shared between the importance sampling processes associated with different penalty parameters. In step 11, we select the optimal solution as detailed in (11). With GPU parallelization, the latency of the SIT-LMPC is close to solving one MPPI process.

Algorithm 2 AP-MPPI$\big(x^l(t), \mathcal{S}^{l-1}, g_\theta^l, \Lambda, P, N, T\big)$

1: **parallel for** $s = 1, \ldots, N$
2: $u^{l_s}_{\cdot|t} \sim \mathcal{N}_{\text{trunc}}(u^{\star l}_{\cdot|t-1}, \Sigma, \mathcal{U})$ ▷ *sequence of length T*
3: $x^{l_s}_{\cdot|t} \leftarrow f(x^l(t), u^{l_s}_{\cdot|t}, 0)$ ▷ *rollout over horizon T*
4: **end parallel for**
5: $V_\theta^l(x^{l_s}_{t+T|t}) \leftarrow \mathbb{E}_z\left[g_\theta^l(z, x^{l_s}_{t+T|t})\right]$
6: **parallel for** $i = 1, \ldots, P$
7: Compute $J^l_{\text{sampled}}\big(x^l(t), x^{l_s}_{\cdot|t}, u^{l_s}_{\cdot|t}; \lambda_i, V_\theta^l(x^{l_s}_{t+T|t})\big)$ using (10)
8: Compute $\hat{u}^{\star l}_{\cdot|t,\lambda_i}$ using (8)
9: $\hat{x}^{\star l}_{\cdot|t,\lambda_i} \leftarrow f\big(x^l(t), \hat{u}^{\star l}_{\cdot|t,\lambda_i}, 0\big)$
10: **end parallel for**
11: Select $\lambda^\star$ using (11)
12: return $\hat{u}^{\star l}_{t|t,\lambda^\star}$

V. Experiments

To validate our approach, we conduct three experiments with progressively increasing system complexities. We start with a deterministic linear point-mass model to show that even for a deterministic linear system, SIT-LMPC still outperforms both LMPC and ABC-LMPC. Then, we move on to experiments with autonomous racing cars to showcase the advantages of SIT-LMPC when the system is stochastic, high-dimensional, nonlinear, and nonholonomic. To simulate stochasticity, truncated Gaussian noise is added to state observations and control inputs. Lastly, we validate our approach through real-world experiments with off-road autonomous racing on a $1/5$th scaled autonomous vehicle. Each experiment was conducted five times for each controller, with individual rollouts shown in light colors and the averaged performance shown in solid colors.

A. Point-mass Navigation

In this experiment, we consider the point mass system described in [28] with $x = [p_x, p_y, v_x, v_y]^{\mathrm{T}}$ where p_x, p_y are positions and v_x, v_y are velocities in Cartesian coordinates.

The control objective is to steer the system from the initial state $x_0 = [0, 0, 0, 0]^{\mathrm{T}}$ to the target state $\mathcal{T} = \{[60, 0, 0, 0]^{\mathrm{T}}\}$ in minimum time while avoiding a circular obstacle centered at $p_{obs} = [30, 0]^{\mathrm{T}}$ of radius 10 so that $\mathcal{X} = \{[p_x, p_y]^{\mathrm{T}} \in \mathbb{R}^2 : \|[p_x, p_y]^{\mathrm{T}} - p_{obs}\|_2 > 10\}$. The control input $u = [a_x, a_y]^{\mathrm{T}}$ is the acceleration in Cartesian coordinates and is assumed

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

to be constrained within the set $\mathcal{U} = \{u \in \mathbb{R}^2 : -1 \leq a_x \leq 1,\ -1 \leq a_y \leq 1\}$. The safe set is initialized with a hand-designed demonstration trajectory. The bottom part of Fig. 2 shows the layout of this experiment. As shown in the top part of Fig. 2, all three methods can iteratively reduce the time to reach the target without crashing into the obstacle, with SIT-LMPC outperforming in terms of convergence rate. This experiment shows that even for a deterministic 2D linear system with non-linear non-convex constraints, SIT-LMPC outperforms prior methods. Even though the system is linear, we solve a non-convex optimization problem due to the obstacle-avoidance constraint. Hence, it is expected that the LMPC, which uses a gradient-based solver, converges to a local minimum, and the sampling-based methods manage to escape the local minimum to converge to a better solution.

B. Vehicle Trajectory Optimization with Simulated Dynamics

The benefits of SIT-LMPC are accentuated with nonlinear, stochastic, and nonholonomic systems. To show this, we compare SIT-LMPC with ABC-LMPC on an autonomous race car tasked with minimizing lap time while staying within the track boundaries. Autonomous racing involves nonlinear dynamics, safety-critical constraints, and real-time decision-making, making it a strong benchmark for robotics [29]. Specifically, it requires a balance between safety and performance, as minimizing lap time demands pushing to the limits while remaining safe. In this experiment, we use the dynamic single-track model from [28], with parameters of vehicle ID:1, expressed in Frenet coordinates. The state of the system is given by $x = [p_x, p_y, v, \delta, \Psi, \dot{\Psi}, \beta]^{\mathrm{T}}$, where (p_x, p_y) is the position in cartesian coordinates, v is the velocity in the $x-$direction, δ is the steering angle of the front wheels, Ψ is the heading of the vehicle, $\dot{\Psi}$ is the yaw rate of the vehicle, and β is the side-slip angle of the vehicle. The control input $u = [a, \delta_v]^{\mathrm{T}}$ is the acceleration in the longitudinal direction and the steering speed. To define our initial state x_0, the target set $\mathcal{T}$, and the set of admissible states $\mathcal{X}$, we express the vehicle's position in Frenet coordinates using the transformation detailed in [30]. The resulting pose in Frenet frame is $[s, e_y, e_\Psi]^{\mathrm{T}}$, where s is the progress along the track, e_y is the lateral displacement, and e_Ψ is the yaw angle in vehicle frame. The initial state is the start line with $s = 0$ and the target set is $\mathcal{T} = \{x \in \mathbb{R}^7 : s \geq L_{\mathrm{t}}\}$, where L_{t} is the arc length of the track. The set of admissible states $\mathcal{X}$ is defined by $\{x \in \mathbb{R}^7 : -w \leq e_y \leq w\}$ where w is the width of the track, assumed to be uniform. The control objective is to minimize the stage cost function $h(x, u) \triangleq \mathbb{1}_{\mathcal{X}\setminus\mathcal{T}}(x)$, where $\mathbb{1}_{\mathcal{X}\setminus\mathcal{T}}(\cdot)$ is the indicator function of the set $\mathcal{X} \setminus \mathcal{T}$. In other words, this stage cost incurs a cost of 1 until the vehicle crosses the finish line to the target set $\mathcal{T}$.

The top part of Fig. 3 shows the lap time of the vehicle over successive iterations. ABC-LMPC frequently crashes halfway through the iterations, failing to satisfy the safe set and state constraints. We attribute this primarily to the selection of the single least-cost sample trajectory by CEM. Consequently, when this sample trajectory approaches a constraint boundary, system noise or perturbations can cause the vehicle to crash.

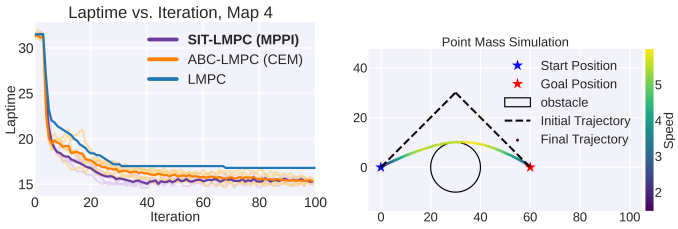


Fig. 2. Point mass experiment. (a) Convergence of lap time over iterations. (b) Layout of the experiment.

We also tested running ABC-LMPC with multiple CEM iterations, but this did not yield notable performance improvements despite the additional computational cost. In contrast, SIT-LMPC converges to a lower lap time while ensuring safety throughout all 150 iterations. Note that when LMPC failed to finish more than five iterations without crashing, as LMPC is *not* designed for stochastic nonlinear systems.

Ablation Study: Using the simulated racing environment, we perform ablation studies on the key components of SIT-LMPC. As shown in Fig. 4(a), using CEM always yields infeasibility. We believe this is due to CEM's susceptibility to noise perturbation and mode collapse. For both CEM and MPPI, comparing the experiments using the NFs and the BNNs, the NF-based modeling yields better performance. To show the effect of the proposed AP method, we tested SIT-LMPC with a fixed penalty. Fig. 4(b) shows that a fixed high penalty (blue) yields suboptimal performance, whereas a fixed low penalty (green) makes the system unsafe. We also implemented each combination with and without the AP method. Our results show that AP improves the learning process, even when used for ABC-LMPC (red in Fig. 4(a)), but is most effective as an integral part of the SIT-LMPC.

C. Real-world Experiment with 1/5th Scaled Vehicle

To show the effectiveness of our approach on real robotic systems, we deployed SIT-LMPC on a $1/5$th scale autonomous off-road race car shown in Fig. 5(a), comparing with ABC-LMPC as a baseline. The initial state x_0, the target set $\mathcal{T}$, the set of admissible states $\mathcal{X}$, and the stage cost $h(x, u)$ for this experiment are identical to those of V-B. The localization is provided by a Fixposition Vision-RTK 2 GNSS system in an off-road grassy outdoor environment. The racetrack used for this experiment is shown in Fig. 5(b). The initial safe set is created using an MPPI controller tracking the centerline of the track at a reference speed of 2 m/s with an initial lap time of 52.13 seconds. For both the MPPI controller and SIT-LMPC, we use the same dynamic model as in section V-B. The parameters of this model were empirically identified, and are not entirely accurate. Fig. 5(c) shows that SIT-LMPC improves lap time (shown in blue) over iterations. During these iterations, the average velocity increased by 75% up to 3.5 m/s, limited by the acceleration bound of the car. Comparatively, ABC-LMPC does not converge without crashing. SIT-LMPC remains safe while outperforming ABC-LMPC by achieving a lap time 31.43% faster. This shows that SIT-LMPC improves safety and performance for a real-world stochastic system with model mismatch, imperfect localization, and control noise from the off-road environment.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

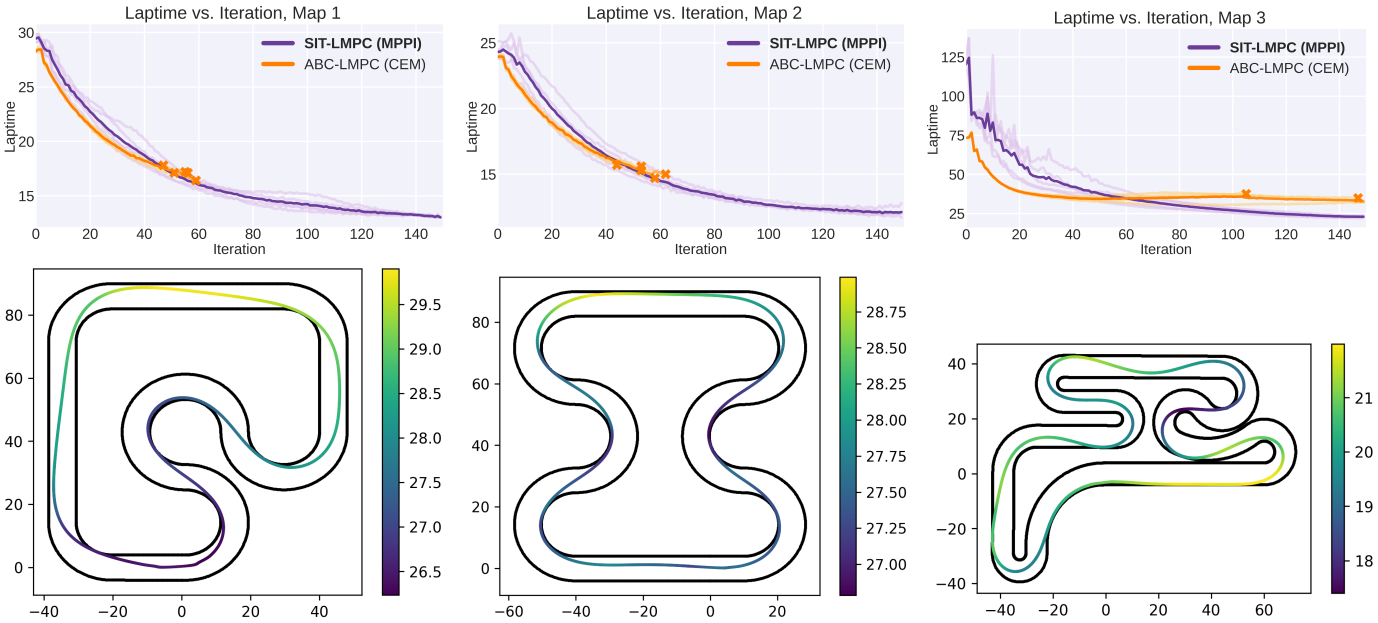
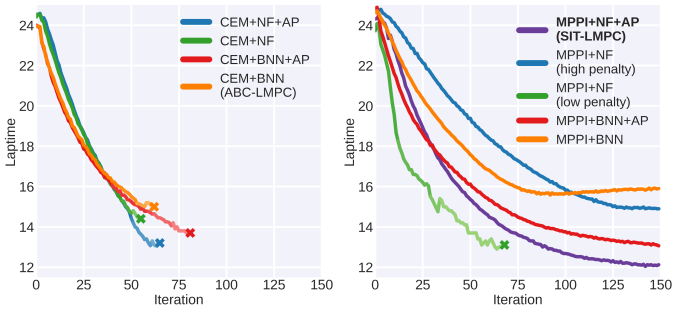
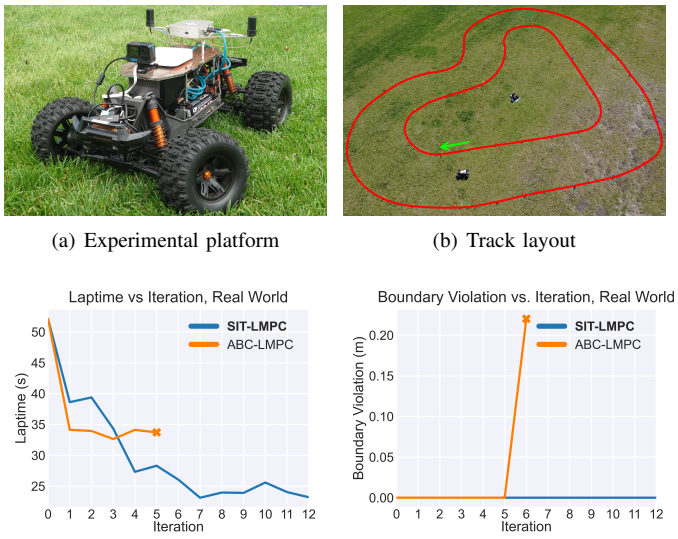


Fig. 3. Top: Convergence of lap time for three simulated experiments (five independent rollouts in light color and averages in dark color; × denotes out-of-track). Bottom: Fastest lap trajectories with velocity colorbar.



(a) Ablations with CEM (b) Ablations with MPPI

Fig. 4. Ablation study of key SIT-LMPC components for a CEM controller (left) and an MPPI controller (right), comparing NF and BNN value function models, with and without the AP method. Plotted are averages of five rollouts with × denoting out-of-track.



(a) Experimental platform (b) Track layout

(c) Laptime per iteration (d) Boundary violations

Fig. 5. Experimental setup and results with a real vehicle. (a) Platform. (b) Track. (c) Laptime per iteration. (d) Boundary violations.

VI. Conclusion

This paper presented SIT-LMPC, an extension of the LMPC formulation to stochastic nonlinear systems. The resulting constrained receding-horizon stochastic optimization problem was solved by developing a constrained information-theoretic MPC algorithm using exterior penalty functions with online adaptive penalty parameters. The proposed method does not rely on prior assumptions about system dynamics or state constraints and is fully parallelizable by construction. Additionally, learning the value function using normalizing flows allowed for richer uncertainty modeling and yielded better performance compared to BNNs. Simulations and real-world experiments demonstrated that SIT-LMPC generates safer and higher-performance trajectories than LMPC and ABC-LMPC. Future research will focus on applying SIT-LMPC to a wider range of robotic platforms and addressing systems with unknown dynamics.

References

[1] E. Kaufmann, L. Bauersfeld, A. Loquercio, M. Müller, V. Koltun, and D. Scaramuzza, "Champion-level drone racing using deep reinforcement learning," *Nature*, vol. 620, no. 7976, pp. 982–987, 2023.

[2] J. Ding, M. A. van Löben Sels, F. Angelini, J. Kober, and C. Della Santina, "Robust jumping with an articulated soft quadruped via trajectory optimization and iterative learning," *IEEE Robotics and Automation Letters*, vol. 9, no. 1, pp. 255–262, 2023.

[3] J. Van Den Berg, S. Miller, D. Duckworth, H. Hu, A. Wan, X.-Y. Fu, K. Goldberg, and P. Abbeel, "Su-

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

perhuman performance of surgical tasks by robots using iterative learning from human-guided demonstrations," in *2010 IEEE international conference on robotics and automation*. IEEE, 2010, pp. 2074–2081.

[4] Z. Zhao, S. Cheng, Y. Ding, Z. Zhou, S. Zhang, D. Xu, and Y. Zhao, "A survey of optimization-based task and motion planning: From classical to learning approaches," *IEEE/ASME Transactions on Mechatronics*, 2024.

[5] A. Fishman, A. Murali, C. Eppner, B. Peele, B. Boots, and D. Fox, "Motion policy networks," in *conference on Robot Learning*. PMLR, 2023, pp. 967–977.

[6] Y. V. Pehlivanoglu and P. Pehlivanoglu, "An enhanced genetic algorithm for path planning of autonomous uav in target coverage problems," *Applied Soft Computing*, vol. 112, p. 107796, 2021.

[7] A. P. Schoellig, F. L. Mueller, and R. D'andrea, "Optimization-based iterative learning for precise quadrocopter trajectory tracking," *Autonomous Robots*, vol. 33, pp. 103–127, 2012.

[8] U. Rosolia, M. Ahmadi, R. M. Murray, and A. D. Ames, "Time-optimal navigation in uncertain environments with high-level specifications," in *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, 2021, pp. 4287–4294.

[9] M. A. Goodrich, A. C. Schultz *et al.*, "Human–robot interaction: a survey," *Foundations and Trends® in Human–Computer Interaction*, vol. 1, no. 3, pp. 203–275, 2008.

[10] D. A. Bristow, M. Tharayil, and A. G. Alleyne, "A survey of iterative learning control," *IEEE control systems magazine*, vol. 26, no. 3, pp. 96–114, 2006.

[11] U. Rosolia and F. Borrelli, "Learning model predictive control for iterative tasks. a data-driven control framework," *IEEE Transactions on Automatic Control*, vol. 63, no. 7, pp. 1883–1896, 2018.

[12] U. Rosolia, Y. Lian, E. T. Maddalena, G. Ferrari-Trecate, and C. N. Jones, "On the optimality and convergence properties of the iterative learning model predictive controller," *IEEE Transactions on Automatic Control*, vol. 68, no. 1, pp. 556–563, 2022.

[13] U. Rosolia, X. Zhang, and F. Borrelli, "Robust learning model-predictive control for linear systems performing iterative tasks," *IEEE Transactions on Automatic Control*, vol. 67, no. 2, pp. 856–869, 2021.

[14] U. Rosolia and F. Borrelli, "Sample-based learning model predictive control for linear uncertain systems," in *2019 IEEE 58th Conference on Decision and Control (CDC)*, 2019, pp. 2702–2707.

[15] B. Thananjeyan, A. Balakrishna, U. Rosolia, F. Li, R. McAllister, J. E. Gonzalez, S. Levine, F. Borrelli, and K. Goldberg, "Safety augmented value estimation from demonstrations (saved): Safe deep model-based rl for sparse cost robotic tasks," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3612–3619, 2020.

[16] B. Thananjeyan, A. Balakrishna, U. Rosolia, J. E. Gonzalez, A. Ames, and K. Goldberg, "Abc-lmpc: Safe sample-based learning mpc for stochastic nonlinear dynamical systems with adjustable boundary conditions," in *Algorithmic Foundations of Robotics XIV: Proceedings of the Fourteenth Workshop on the Algorithmic Foundations of Robotics 14*. Springer, 2021, pp. 1–17.

[17] R. Y. Rubinstein and D. P. Kroese, *The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation and machine learning*. Springer Science & Business Media, 2004.

[18] G. Williams, N. Wagener, B. Goldfain, P. Drews, J. M. Rehg, B. Boots, and E. A. Theodorou, "Information theoretic mpc for model-based reinforcement learning," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 1714–1721.

[19] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Aggressive driving with model predictive path integral control," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 1433–1440.

[20] ——, "Information-theoretic model predictive control: Theory and applications to autonomous driving," *IEEE Transactions on Robotics*, vol. 34, no. 6, pp. 1603–1622, 2018.

[21] F. Rastgar, H. Masnavi, B. Sharma, A. Aabloo, J. Swevers, and A. K. Singh, "Priest: Projection guided sampling-based optimization for autonomous navigation," *IEEE Robotics and Automation Letters*, 2024.

[22] J. Yin, C. Dawson, C. Fan, and P. Tsiotras, "Shield model predictive path integral: A computationally efficient robust mpc method using control barrier functions," *IEEE Robotics and Automation Letters*, 2023.

[23] M. Rungger and P. Tabuada, "Computing robust controlled invariant sets of linear systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 7, pp. 3665–3670, 2017.

[24] R. M. Freund, "Penalty and barrier methods for constrained optimization," *Lecture Notes, Massachusetts Institute of Technology*, 2004.

[25] J. Burkardt, "The truncated normal distribution," *Department of Scientific Computing Website, Florida State University*, vol. 1, no. 35, p. 58, 2014.

[26] G. Papamakarios, E. Nalisnick, D. J. Rezende, S. Mohamed, and B. Lakshminarayanan, "Normalizing flows for probabilistic modeling and inference," *Journal of Machine Learning Research*, vol. 22, no. 57, pp. 1–64, 2021.

[27] C. Durkan, A. Bekasov, I. Murray, and G. Papamakarios, "Neural spline flows," *Advances in neural information processing systems*, vol. 32, 2019.

[28] M. Althoff, M. Koschi, and S. Manzinger, "Commonroad: Composable benchmarks for motion planning on roads," in *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 719–726.

[29] A. G. Puigjaner, M. Prajapat, A. Carron, A. Krause, and M. N. Zeilinger, "Performance-driven constrained optimal auto-tuner for mpc," *IEEE Robotics and Automation Letters*, vol. 10, no. 5, pp. 4698–4705, 2025.

[30] A. Micaelli and C. Samson, "Trajectory tracking for unicycle-type and two-steering-wheels mobile robots," Ph.D. dissertation, Inria, 1993.