Engineering Applications of Artificial Intelligence 110 (2022) 104731

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Safe learning-based gradient-free model predictive control based on cross-entropy method

Lei Zheng [a], Rui Yang [b], Zhixuan Wu [b], Jiesen Pan [b], Hui Cheng [b,*]

[a] School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou 510006, China
[b] School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou 510006, China

ARTICLE INFO

Keywords:
Model predictive control
Learning-based control
Cross-entropy method
Minimal intervention controller

ABSTRACT

In this paper, a safe and learning-based control framework for model predictive control (MPC) is proposed to optimize nonlinear systems with a non-differentiable objective function under uncertain environmental disturbances. The control framework integrates a learning-based MPC with an auxiliary controller in a way of minimal intervention. The learning-based MPC augments the prior nominal model with incremental Gaussian Processes to learn the uncertain disturbances. The cross-entropy method (CEM) is utilized as the sampling-based optimizer for the MPC with a non-differentiable objective function. A minimal intervention controller is devised with a control Lyapunov function and a control barrier function to guide the sampling process and endow the system with high probabilistic safety. The proposed algorithm shows a safe and adaptive control performance on a simulated quadrotor in the tasks of trajectory tracking and obstacle avoidance under uncertain wind disturbances.

1. Introduction

Robots and autonomous systems are increasingly widely applied to solve complex tasks in highly uncertain and dynamic environments (Siciliano and Khatib, 2016). Operating in such environments requires sophisticated control methods that can adapt to the uncertain environmental disturbances, plan and execute trajectories utilizing the full dynamics and complete the predefined tasks safely. Model predictive control (MPC) (Mayne, 2014) provides a general framework to consider the system constraints naturally, anticipate future events and take control actions accordingly to complete complex tasks which are encoded in the objective function. However, designing differentiable objective functions corresponding to all requirements of the task is nontrivial (Schaal and Atkeson, 2010). For example, it is challenging to design a differentiable objective function accounting for simultaneously avoiding unexpectedly detected obstacles and aggressively tracking trajectory for a safety-critical quadrotor in a clustered obstacle field. The trade-off between safety and tracking control performance is hard to mediate using a simplified differentiable form. It is more convenient for the designer to compose multiple high-level, simple but possibly non-differentiable terms in the objective function, but the resulting non-differentiable objective function brings difficulties to the optimization (Williams et al., 2018). Obtaining the solutions to the optimization problem using regular gradient-based optimizers is difficult due to the nonlinear dynamics constraints and non-differentiable objective function.

There are some optimization methods such as nonconvex alternating direction method of multipliers (ADMM) (Wang et al., 2019; O'Donoghue et al., 2013), sequential quadratic programming, and interior point methods (Wright, 1997; Andersson et al., 2019) for optimizing the system with nonconvex or even sparse non-differentiable cost function. However, either linear system or derivative information is required, making it difficult to directly apply to the optimization with nonlinear dynamics constraints and arbitrary non-differentiable objective function. Sampling-based methods such as random shooting (Nagabandi et al., 2018), path integral control (Williams et al., 2017), and cross-entropy method (CEM) (Boer et al., 2005) are effective approaches to solve the optimization problem with a non-differentiable objective function. The CEM is initially devised to estimate the probability of a rare event and later employed as a general optimization framework. The optimal sampling distribution in CEM is recursively approximated to guide the sampled trajectories towards lower cost regions until converging to a delta distribution (Boer et al., 2005). As with most randomized methods, there is no guarantee that this would be the global optimum but with high likelihood, the local optimum can be approached if enough samples are generated. It has been demonstrated to effectively optimize complex nonlinear systems with a non-differentiable objective function (Kobilarov, 2012; Chua et al., 2018). An adaptive sampling strategy is developed in Tan et al. (2018) using receding-horizon cross-entropy trajectory optimization with Gaussian Process upper confidence bound (Srinivas et al., 2009).

* Corresponding author.
E-mail addresses: zhenglei5@mail2.sysu.edu.cn (L. Zheng), chengh9@mail.sysu.edu.cn (H. Cheng).

https://doi.org/10.1016/j.engappai.2022.104731
Received 27 February 2021; Received in revised form 31 January 2022; Accepted 31 January 2022
Available online 14 February 2022

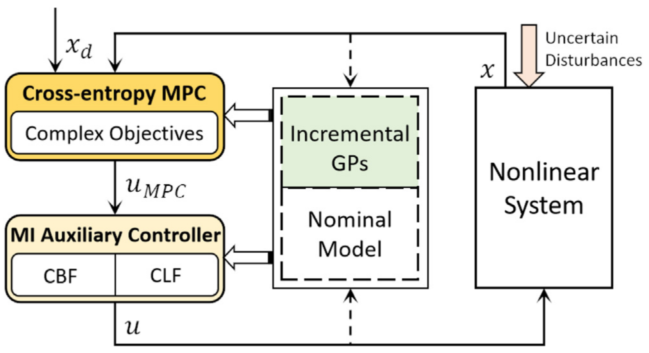


Fig. 1. Diagram of the proposed control scheme for a nonlinear system under uncertain disturbances. Taking in the desired state x_d and current state x, the MPC optimizes the nonlinear system with a simple non-differentiable objective function using CEM. The output u_{MPC} of MPC is modified by an auxiliary controller in a way of minimal intervention, which guides the sampling process and preserves system safety. The environmental uncertainties are learned using incremental GPs. The learned model is combined with a prior nominal model to serve in both the CEMPC and the MI(Minimal Intervention) auxiliary controller.

In Bharadhwaj et al. (2020), the planning problem of high-dimensional systems is solved by interleaving CEM and gradient descent optimization. Actually, as a local search method, the CEM is essentially prone to the model errors caused by unexpected disturbances, which may make the sampling get stuck in a bad region of state space and even diverge, resulting in dangers to safety-critical systems (Williams et al., 2018). To address this problem, decent initialization and gradient signal are required in this case to lead sampling distribution back to the low-cost region safely. To illustrate, it is common for robotics systems that the actual next state is close to the predicted one in a high-frequency control scheme. Thus it is reasonable to initialize the sampling distribution of next iteration with information from the last iteration as a type of warm start. On the other hand, the accuracy of the predictive model under uncertain disturbances should be improved.

To guide the sampling distribution safely in case of external disturbances while keeping the control proactive, a sampling-based Tube-MPC method is augmented in Williams et al. (2018) with an iterative linear quadratic Gaussian controller for sampling distribution guidance and disturbance rejection. Model predictive path integral control (Williams et al., 2017) is integrated with $\mathcal{L}1$ adaptive control in Pravitra et al. (2020) to achieve both fast model predictive trajectory planning and robust trajectory tracking. Besides, the minimum intervention principle has shown an increasing interest in the safe control domain of semi-autonomous vehicles. A minimal intervention (MI) mechanism is designed in Leung et al. (2020) to infuse reachability-based safety assurance within the planning frameworks. It essentially projects the desired trajectory into a set of safety-preserving controls whenever safety is threatened, so as to ensure safety and meanwhile keep the behavior as proactive as possible. Similarly, the control barrier function (CBF) is utilized to design the safety barrier certificates in Luo et al. (2020) and Hirshberg et al. (2020) to minimally modify an existing controller to formally satisfy collision avoidance constraints via a constrained quadratic program (QP). Multiple obstacles situations can be handled based on CBF (Wang et al., 2018; Deits and Tedrake, 2015). While robust control under bounded uncertainties achieves safety and meanwhile preserves the control performance of the existing controller, the adaptation to disturbances is not considered in Luo et al. (2020) and Hirshberg et al. (2020). The CBF and control Lyapunov function (CLF) are integrated into a QP to achieve both safety and tracking stability in Zheng et al. (2020), but without a minimal intervention scheme.

On the other hand, the control performance of MPC depends on the accuracy of the predictive model (Mayne, 2014). However, it is generally difficult to obtain precise system models for real-world nonlinear systems. This issue of model discrepancy has been addressed through various adaptation approaches. Some of the existing adaptive MPC approaches assume a structured system model with uncertain parameters that can be estimated online with an estimator such as extended Kalman filter (Fukushima et al., 2007; Aswani et al., 2013). However, it is limited to treat all model errors as parameters to estimate, especially when the system is subject to complex and external uncertain disturbances (Desaraju and Michael, 2016).

Considering model uncertainties resulting from uncertain disturbances, learning-based techniques can also be applied to learn the predictive model in MPC. In the model-based reinforcement learning (RL), the dynamics model is learned from data using a deep neural network (NN) (Nagabandi et al., 2018) or Gaussian Processes (GPs) (Urbina et al., 2011; Cao et al., 2017). The effectiveness of CEM for model-based RL is demonstrated in Chua et al. (2018) with Bayesian neural network ensemble as the predictive model. While avoiding the need for manual controller design, these works do not well utilize the prior knowledge on systems and control design (Kober et al., 2013), which can be used to improve the learning efficiency and provide safety for system design. Taking advantage of prior knowledge of the system, semi-structured approaches augment a prior nominal model with machine learning models to capture the model errors for MPC (Hewing et al., 2020; Desaraju and Michael, 2016). GPs are utilized to learn the model errors, since they can simultaneously capture the uncertainty of the estimation due to the lack of data and the noises inherent in the environment (Urbina et al., 2011). It is combined with a nominal model as the predictive model for MPC (Hewing et al., 2018; Ostafew et al., 2016; Mehndiratta and Kayacan, 2020), as a kind of robust or stochastic MPC (Mayne, 2016), where the control performance is shown to greatly improve with a more accurate predictive model. While the safety regarding state constraints can be satisfied in the form of chance constraint (Hewing et al., 2020), the propagation of state variance along the prediction horizon brings much computation and the optimization of the resulting nonlinear MPC requires gradient information of the objective function.

The limitations of the sampling-based MPC and the advantages of safe learning-based control techniques show an urgent need for designing an adaptive, high-performance and safe control strategy for nonlinear system optimization with a non-differentiable objective function. Safety should be enforced to the sampling-based optimization while the proactive control ability should be maintained for the nonlinear system under environmental disturbances. To solve this problem, we propose a safe and adaptive predictive control architecture as shown in Fig. 1.

Our main contributions are summarized as follows.

- A novel learning-based CEM-based MPC (CEMPC) framework is proposed for system optimization with a non-differentiable objective function. The CEM is utilized as the optimizer to solve the non-differentiable MPC based on a prior predictive model and incremental GPs (IGPs) for additive state disturbance estimation.
- A minimal intervention (MI) auxiliary controller based on CBF and CLF is devised to intervene in the sampling-based MPC, endowing the system with safety and guiding the sampling distribution to low-cost regions when necessary.
- The proposed control methodology is validated on a quadrotor tracking aggressive trajectory and simultaneously avoiding detected obstacles under uncertain time-varying wind disturbances in simulation.

The rest of this paper is organized as follows. The problem statement is presented in Section 2. The IGPs, learning-based CEMPC and the MI auxiliary controller are described in Section 3. Numerical simulation of the proposed algorithm on a quadrotor system is shown in Section 4. Finally, a conclusion is drawn in Section 5.

2. Problem statement

Consider a nonlinear control affine system with dynamics

$$\dot{x} = f(x) + G(x)u, \tag{1}$$

where $x \in \mathcal{X} \subset \mathbb{R}^n$ denotes the system state and $u \in \mathcal{U} \subset \mathbb{R}^m$ is the control input. A wide range of robots such as quadrotors and car-like vehicles can be transformed into a control affine system in this form. Though our analysis is restricted to this form, the results can be extended to the systems of higher relative degrees. Assume that the function $f : \mathcal{X} \to \mathbb{R}^n$ is partially unknown but Lipschitz continuous and has a bounded reproducing kernel Hilbert space (RKHS) norm under a known kernel, and the function $G : X \to \mathbb{R}^{n\times m}$ is known and Lipschitz continuous. The partially unknown $f(x)$ consists of a known nominal model $\hat{f}(x)$ and the uncertain disturbances $d(x)$,

$$f(x) = \hat{f}(x) + d(x). \tag{2}$$

If there is no disturbance, then the nominal model matches the actual one and $d(x) = 0$. However, it is difficult to get an accurate model in advance for practical nonlinear systems under uncertain disturbances, e.g. for a quadrotor under uncertain wind disturbances.

The **goal** is to optimize the system (1) to accomplish specified complex tasks safely under uncertain environmental disturbances. The specified tasks can be described with the **cost function** of the form:

$$\mathcal{L}(x,u) = \iota(x)^T Q \iota(x) + \sum_{i=1}^{N} w_i \mathbb{I}_{C_i}(x) + u^T R_u u, \tag{3}$$

where $\iota(x)$ extracts features from the state, Q and R_u are a positive semidefinite and a positive definite weight matrix, respectively. N is the number of simple encodings of task descriptions, w_i is the corresponding weight coefficient, $\mathbb{I}_{C_i}$ is an indicator function for the set C_i, which is 1 if $x \in C_i$ and 0 otherwise.

The first portion of the cost function (3) is to encode the main task for the system, e.g. tracking a trajectory. The second term encodes specific tasks, which could be sparse, non-differentiable and hard to rewrite into a typically differentiable form like the first term, e.g. to avoid the unexpectedly detected obstacles for a quadrotor with a limited sensing range. The last term regularizes the control inputs. With this form of the cost function, the tasks can be easily encoded with several interpretable terms and weighted differently according to the importance of task requirements in convenience.

Remark 1. Note that though the non-differentiable terms are technically soft constraints, the penalty is obtained immediately once the state falls into the specified sets. Besides, they have the advantage that the importance of different requirements can be delineated by setting different weight coefficients w_i.

3. Methodology

In this section, the proposed control scheme for the nonlinear system (1) is described, as shown in Fig. 1. With a predefined objective function consisting of simple and non-differentiable encodings of task descriptions, the CEMPC optimizes the nonlinear system with a predictive model. This predictive model is composed of a nominal model from prior knowledge and a discrepancy model learned via GPs. To guide the sampling distribution back towards the low-cost regions in case of disturbances and preserve system safety, the control inputs computed by the CEMPC are modified with a designed MI auxiliary controller in an efficient QP framework.

The GPs for learning the uncertain disturbances (2) and the incremental implementation to reduce the computational complexity are first introduced in Section 3.1. With the prior and learned model, the CEMPC is then presented in Section 3.2 and the MI auxiliary controller is described in Section 3.3.

3.1. Increment Gaussian processes for disturbance learning

A GP is an efficiently nonparametric regression method to estimate complex functions and their uncertain distribution (Rasmussen and Nickisch, 2010). It assumes that function values associated with different inputs are random variables, and any finite number of them have a joint Gaussian distribution. The uncertain model errors $d(x)$ resulted from uncertain disturbances in (2) can be learned using GPs with the data collected from the system during operation. Similar to Ostafew et al. (2016), we train n separate GPs to model the disturbances $d(x)$ with the output of n dimensions based on the following assumption.

Assumption 1. The unknown disturbances $d(x)$ in (2) are uncorrelated.

The approximation of disturbances $d(x)$ can be denoted by $\widetilde{d}$. To make the problem tractable, similar to the Assumption 1 in Berkenkamp et al. (2016), the following assumption is considered.

Assumption 2. The unknown disturbances $d(x)$ has a bounded norm in the associated Reproducing Kernel Hilbert Space (RKHS) (Schölkopf et al., 2002), corresponding to a continuously differentiable kernel k.

This assumption can be interpreted as a requirement for the smoothness of the disturbances $d(x)$. Its boundedness implies that $d(x)$ is regular with respect to the kernel k (Srinivas et al., 2012). It is common practice to use the squared-exponential kernel $k(x,\ x') = \sigma_f^2 \exp(-\frac{1}{2}(x-x')^{\mathrm{T}} L^{-2}(x-x'))$ to estimate the similarity between states x and $x\prime$, which is characterized by hyperparameters of the length scale diagonal matrix L and the prior variance σ_f^2. Besides, we assume that the training set D is available to the GPs for regression.

Assumption 3. The predefined state x and the function value $d(x)$ can be measured with noises over a finite time horizon to make up a training set with n data pairs

$$D = \left\{\left(x^{(i)},\ y^{(i)}\right)\right\}_{i=1}^{n},\ y^{(i)} = d\left(x^{(i)}\right) + v_i, \tag{4}$$

where v_i are i.i.d. noises $v_i \sim \mathcal{N}\left(0, \sigma_{noise}^2 I_n\right)$, $\sigma_{noise}^2 \in \mathbb{R}$.

Given the dataset D, the mean and variance of $\widetilde{d}(x_*)$ at the query state x_* can be given by (Rasmussen and Nickisch, 2010)

$$\mu(x_*) = k_n^T (K_\sigma + \sigma_{noise}^2 I)^{-1} y_n, \tag{5}$$

$$\sigma^2(x_*) = k(x_*, x_*) - k_n^T (K_\sigma + \sigma_{noise}^2 I)^{-1} k_n, \tag{6}$$

respectively, where $y_n = [y(x_1), y(x_2), \dots, y(x_n)]^T$ is the observed vector, $K_\sigma \in \mathbb{R}^{n\times n}$ is the covariance matrix with entries $[K_\sigma]_{(i,j)} = k(x_i, x_j)$, $i,j \in \{1,\dots,n\}$, $k_n = [k(x_1,x_*), k(x_2,x_*), \dots, k(x_n,x_*)]$ is the vector with kernel function $k(x_i,x_j)$, $I_n \in \mathbb{R}^{n\times n}$ is the identity matrix. The variance of noise σ_{noise} can be selected and set according to the actual noise level of the disturbances and the accuracy of measurements.

A high probability confidence interval $\mathcal{D}(x)$ on $\widetilde{d}(x)$ can then be obtained (Berkenkamp et al., 2016)

$$\mathcal{D}(x) = \{\widetilde{d} \mid \mu(x) - c_\delta \sigma(x) \le \widetilde{d} \le \mu(x) + c_\delta \sigma(x)\}, \tag{7}$$

where c_δ is a parameter designed to get a confidence interval of $(1-\delta)$, $\delta \in (0,1)$. For instance, 95.5% and 99.7% confidence of the uncertainty bound can be achieved at $c_\delta = 2$ and $c_\delta = 3$, respectively.

The computational complexity of GPs is $O(n^3)$ due to the matrix inverse of K, $K = K_\sigma + \sigma^2 I$. It brings non-negligible challenges for practical applications. An incremental implementation of GPs is devised to reduce the time of inference and learning by fixing the size of the dataset and recursively computing the matrix inverse. Specifically, the IGP considers two stages as follows.

(1) Adding New Data: Given a predefined size n of the dataset, a new data point is added into the dataset at each time step until the size of the dataset reaches the predefined size. Assume there has been i data

points in the dataset, $0 < i < n$. Denote the computed kernel matrix as $K_{old} \in \mathbb{R}^{i \times i}$ and the corresponding matrix inverse as K_{old}^{-1} based on the old dataset. When a new data point (x_{i+1}, y_{i+1}) is added, the new kernel matrix $K_{new} \in \mathbb{R}^{(i+1)\times(i+1)}$ can be computed based on K_{old}:

$$K_{new} = \begin{bmatrix} K_{old} & \mathbf{k}_{i+1} \\ \mathbf{k}_{i+1}^T & k_{i+1} + \sigma_{noise}^2 I \end{bmatrix}, \tag{8}$$

where $\mathbf{k}_{i+1} = [k(x_1, x_{i+1}), k(x_2, x_{i+1}), \ldots, k(x_i, x_{i+1})]^\intercal$ and $k_{i+1} = k(x_{i+1}, x_{i+1})$. The incremental update of the matrix inverse K_{new}^{-1} can be computed by

$$K_{new}^{-1} = \begin{bmatrix} K_{old}^{-1} + \xi \Gamma \cdot \Gamma^T & -\xi \Gamma \\ -\xi \Gamma^T & \xi \end{bmatrix} \tag{9}$$

where $\Gamma = K_{old}^{-1} \cdot \mathbf{k}_{i+1}$ and $\xi = (k_{i+1} + \sigma_{noise}^2 I - \mathbf{k}_{i+1}^T \cdot \Gamma)^{-1} \in \mathbb{R}^{1\times 1}$.

Remark 2. The inversion operation only needs to perform on a scalar ξ rather than on the entire matrix $K_{new} \in \mathbb{R}^{(i+1)\times(i+1)}$. Matrix multiplication in the method takes $O(i^2)$ complexity. Thus, this method can reduce the computation burden largely by saving and reusing the computed matrix inverse result, especially when the predefined size n of the dataset is quite large.

(2) Replacing Data: If the size of the dataset reaches the predefined size n, a new data point will be added to the dataset at each timestep while the oldest one will be deleted. Denote the old kernel matrix in the form of a block matrix as

$$K_{old} = \begin{bmatrix} k_0 & \mathbf{k}_0^T \\ \mathbf{k}_0 & \Upsilon \end{bmatrix}, \tag{10}$$

where $\mathbf{k}_0$ and k_0 are the covariance vector and variance value of the oldest data point in the dataset, and $\Upsilon \in \mathbb{R}^{(n-1)\times(n-1)}$ are the sub-matrix at the right bottom corner. The inverse matrix of K_{old} can be computed as

$$K_{old}^{-1} = \begin{bmatrix} \rho & q^T \\ q & \Xi \end{bmatrix}, \tag{11}$$

where ρ, q and Ξ is the sub-matrices, $\Xi \in \mathbb{R}^{(n-1)\times(n-1)}$. With the obtained new data point, the oldest data point is removed from the dataset, and variance k_{i+1} and covariance vector $\mathbf{k}_{i+1}$ are computed. The new kernel matrix can be obtained based on the old kernel matrix K_{old}:

$$K_{new} = \begin{bmatrix} \Upsilon & \mathbf{k}_{i+1} \\ \mathbf{k}_{i+1}^T & k_{i+1} + \sigma_{noise}^2 I \end{bmatrix}, \tag{12}$$

and the inverse matrix can be computed by

$$K_{new}^{-1} = \begin{bmatrix} \Lambda + (\Lambda \mathbf{k}_{i+1})(\Lambda \mathbf{k}_{i+1}^T) l & -(\Lambda \mathbf{k}_{i+1}) l \\ -(\Lambda \mathbf{k}_{i+1}^T) l & l \end{bmatrix}, \tag{13}$$

where $\Lambda = \Xi - q \cdot q^T \rho^{-1}$, $l = (k_{i+1} + \sigma_{noise}^2 I - \mathbf{k}_{i+1}^T \cdot \Lambda \cdot \mathbf{k}_{i+1})^{-1}$.

Remark 3. Note that there are only 4 times of matrix multiplications in this method. Taking several matrix addition and transposition into consideration, it takes $O(n^2)$ computational complexity.

3.2. Learning-based Model Predictive Control with Cross-Entropy Method (CEMPC)

With the learned disturbances $\widetilde{d}$ via the IGPs, the predictive model for the MPC can be obtained based on the prior model. Considering the non-differentiable objective function, a sampling-based MPC scheme is designed with CEM to provide the nominal controls for the nonlinear systems (1). The MPC is based on the following open-loop finite horizon optimal control problem given the measured state $x(t_k)$ at each sampling time $t_k = k \cdot T_s$ with a control period T_s, $k \in \mathbb{N}^+$.

$$u^* = \underset{\bar{u}(t) \in \mathcal{PC}([t_k, t_k+T], \mathcal{U})}{\arg\min} \quad \Phi(\bar{x}(t+T)) + \int_{t_k}^{t_k+T} \mathcal{L}(\bar{x}(\tau), \bar{u}(\tau)) d\tau., \tag{14}$$

$$\text{s.t.} \quad \dot{\bar{x}}(t) = \hat{f}(\bar{x}(t)) + g(\bar{x}(t))\bar{u}(t) + \widetilde{d}(\bar{x}(t)), \tag{15}$$

$$\bar{x}(t_k) = x(t_k), \tag{16}$$

where $\mathcal{PC}([t_k, t_k+T], \mathcal{U})$ represents the set of all piece-wise continuous functions $\varphi : [t_k, t_k+T] \to \mathcal{U}$, $\bar{x}$ denotes the state predicted based on the system model (15) given candidate controls $\bar{u}(t)$ over the prediction horizon $T > 0$, Φ is a terminal cost function, $\mathcal{L}$ is the non-differentiable running cost (3), and $\widetilde{d}$ is the approximation to the actual environmental uncertainties $d(x)$. The Eq. (16) is the state initialization of the finite horizon optimal control problem. The input constraints are taken into account by bounding the control input samples in the control space and limiting the initial mean and variance of the sampling distribution.

Optimization and execution take place alternatively. With the current state $x(t_k)$ fedback at any sampling time t_k, the problem (14)–(16) is solved to obtain the optimal open-loop control inputs $u^*(t), \forall t \in [t_k, t_k+T]$, and only the first-step input $u_{MPC}(t) = u^*(t), \forall t \in [t_k, t_k+T_s]$ is applied to the nonlinear system (1). The overall process is repeated at the next sampling time t_{k+1}.

Algorithm 1 Learning-based CEMPC

```
Require:
    $N$: Number of iterations,
    $M$: Sample numbers per iteration,
    $H$: Predictive time steps of the MPC,
    $K$: Size of the elite set,
    $\Sigma_{min}$: A minimum variance bound for optimization,
    $\beta$: Update rate,
    $\mathcal{N}(O_{0:H-1}^{(0)}, \Sigma_{0:H-1}^{(0)})$: Initial sampling distribution, where $O_{0:H-1}^{(0)}$
    and $\Sigma_{0:H-1}^{(0)}$ denotes the mean and covariance matrix of $H$ separate
    initial multivariate Gaussian distributions, respectively.
Ensure:
    $\mu_*$ : Optimized mean value of the control input sampling
    distribution,
1: while Task is not completed do
2:    Measure current state $x_k$.
3:    $i = 0$.
4:    while $i < N$ and $max(\Sigma_{0:H-1}) > \Sigma_{min}$ do
5:       Sample $\{(u_0^{(i)}, ..., u_{H-1}^{(i)})_j\}_{j=0}^{M-1} \sim \mathcal{N}(O_{0:H-1}^{(i)}, \Sigma_{0:H-1}^{(i)})$.
6:       Score the samples according to (18), (15) and (16).
7:       Sort them in an ascending order, $\mathcal{J}_0^{(i)} \leq ... \leq \mathcal{J}_{M-1}^{(i)}$.
8:       Choose $K$ sequences according to $\mathcal{J}_0^{(i)} \leq ... \leq \mathcal{J}_{K-1}^{(i)}$.
9:       Update sampling distribution using the elite set
         $O_{0:H-1}^{(i+1)} \leftarrow Mean(\{(u_0^{(i)}, ..., u_{H-1}^{(i)})_j\}_{j=0}^{K-1})$,
         $\Sigma_{0:H-1}^{(i+1)} \leftarrow Var(\{(u_0^{(i)}, ..., u_{H-1}^{(i)})_j\}_{j=0}^{K-1})$.
10:      $(O_{0:H-1}^{(i+1)}, \Sigma_{0:H-1}^{(i+1)}) \leftarrow (1-\beta)(O_{0:H-1}^{(i+1)}, \Sigma_{0:H-1}^{(i+1)}) + \beta(O_{0:H-1}^{(i)}, \Sigma_{0:H-1}^{(i)})$
11:      $i \leftarrow i + 1$
12:   end while
13:   $u_{MPC} \leftarrow \{(u_0^{(N-1)})_j\}_{j=0}$.
14:   $u_k \leftarrow$ MIControlSheme($x_k, u_{MPC}$) in Algorithm 2.
15:   $x_{k+1} \leftarrow$ Apply $u_k$ to the system (1).
16:   Collect data point $\{x_k, u_k, x_{k+1}\}$ and Update $GP$ in (5) and (6).
17:   Reinitialize sampling distribution $O_{0:H-2}^{(0)} \leftarrow O_{1:H}^{(N-1)}$, $\Sigma_{0:H-2}^{(0)} \leftarrow$
      $\Sigma_{1:H}^{(N-1)}$.
18: end while
```

It is hard to obtain a closed-form solution or apply a gradient-based optimizer to the optimization problem (14)–(16), since the dynamics (15) is nonlinear, and the objective function (14) is non-differentiable. The CEM is adopted as an adaptive sampling-based method to solve the MPC, which is widely applied as a general optimization framework to solve complex optimization problems (Finn and Levine, 2017; Kobilarov, 2012; Chua et al., 2018). It treats the optimization problem as an estimation problem of the probability of a rare event, with the distribution parameters to be estimated. The control space is sampled repeatedly and the sampling distributions of the controls are optimized via the importance sampling technique.

To ease the understanding, the optimization process with CEM for (14)–(16) is described in discrete control. Denote the time variable $(t+k{\cdot}T_s)$ with the time step subscript k. At the ith CEM iteration, multiple M random control sequences with H time steps are sampled

$$\{(u_0^{(i)},\dots,u_{H-1}^{(i)})_j\}_{j=0}^{M-1} \sim \mathcal{N}(O_{0:H-1}^{(i)},\Sigma_{0:H-1}^{(i)}), \tag{17}$$

where $\mathcal{N}(O_{0:H}^{(i)},\Sigma_{0:H}^{(i)})$ denotes H separate multivariate Gaussian distributions, from which control input sequences are sampled. The mean and covariance matrix of the distributions are set initially $O_{0:H}^{(0)}=\{0\}_{0:H}$ and $\Sigma_{0:H}^{(0)}=\{\Sigma_{init}\}_{0:H}$, respectively, where $\Sigma_{init}=(\frac{u_{max}-u_{min}}{2})^2$. With these control sequences, the accumulated costs $\mathcal{J}_j^{(i)}$ of the jth control sequences are evaluated based on the cost function $\mathcal{L}$ (3):

$$\mathcal{J}_j^{(i)}=\Phi(x_H)+\sum_{k=0}^{H-1}\mathcal{L}(x_k,u_k),\forall\ j=0,\dots,M-1. \tag{18}$$

With the estimated costs $\{J_j^{(i)}\}_{j=0}^{M-1}$, an elite set of K control sequences with the K lowest costs are chosen out from the M sequences (17). The elite set is used to update the sampling distribution $\mathcal{N}(O_{0:H}^{(i+1)},\Sigma_{0:H}^{(i+1)})$ for the next $i+1$ iteration:

$$O_{0:H}^{(i+1)}\leftarrow(1-\beta)Mean(\{(u_0^{(i)},\dots,u_{H-1}^{(i)})_j\}_{j=0}^{K-1})+\beta O_{0:H}^{(i)}, \tag{19}$$

$$\Sigma_{0:H}^{(i+1)}\leftarrow(1-\beta)Var(\{(u_0^{(i)},\dots,u_{H-1}^{(i)})_j\}_{j=0}^{K-1})+\beta\Sigma_{0:H}^{(i)}. \tag{20}$$

where β is a smoothness coefficient to adjust the update of the distribution parameters. The smaller it is, the more we trust the new distribution parameters. We usually set its value by experience and test different values in practice.

The sampling distribution is updated towards the regions with lower cost, as the above process iterates for N times or the maximum variance drops below a minimum variance bound Σ_{min}. The remaining control sequence $\{(u_0^{(N)},\dots,u_{H-1}^{(N)})_j\}_{j=0}$ is returned with the lowest cost and the first-step control input $u_{MPC}=\{(u_0^{(N)})_j\}_{j=0}$ is applied as the output of the CEMPC. Modified by the auxiliary controller when necessary, the control input is applied to the system. The data of the system evolution is then collected in the dataset to update the GPs. Algorithm 1 details the proposed learning-based CEMPC.

Remark 4. Note that the prediction of mean in (5) takes $O(n^2H+MnH)$ computational complexity when predicting M samples over H timesteps in (17). The calculation of the cost function in (18) is applied for M samples and H timesteps for each sample, which takes $O(MH)$ complexity.

3.3. *Minimal Intervention (MI) auxiliary controller*

In the learning-based CEMPC, task requirements and state constraints can be encoded in the cost functions with different penalty weights. Though the design process is simplified in this way, system safety regarding state constraints is not guaranteed. Besides, the sampling distribution should be guided to the low-cost regions in the case of unexpected disturbances. To solve the problem, an auxiliary controller is designed in this part to minimally intervene in the CEMPC.

3.3.1. *Barrier-enforced safety scheme*

Safety under environmental uncertainties should be considered carefully before the optimized control inputs are applied to the system. It can be enforced with safety constraints via CBF, which can quantify the system safety regarding state constraints (Ames et al., 2019).

The *safety set* $\mathcal{S}$ of the system (1) can be defined by

$$\mathcal{S}:=\{x\in\mathcal{X}|h(x)\geq 0\}, \tag{21}$$

where $h:\mathbb{R}^n\to\mathbb{R}$ is a continuously differentiable function related to the state constraints.

Definition 1. The set $\mathcal{S}$ is called *forward invariant*, if for every $x_0\in\mathcal{S}$, $x(t,x_0)\in\mathcal{S}$ for all $t\in\mathbb{R}_0^+$.

To ensure forward invariance of $\mathcal{S}$, e.g. quadrotors stay in the collision-free safety set at all times, we consider the following definition.

Definition 2 (*Definition 5 of Ames et al. (2017)*)**.** For the dynamical system (1), given a set $\mathcal{S}\subset\mathbb{R}^n$ defined by (21) for a continuously differentiable function $h:\mathbb{R}^n\to\mathbb{R}$, the function h is called a *Zeroing Control Barrier Function (ZCBF)* defined on the set $\mathcal{E}$ with $\mathcal{S}\subseteq\mathcal{E}\subset\mathbb{R}^n$, if there exists an extended class $\mathcal{K}$ function κ ($\kappa(0)=0$ and strictly increasing) such that

$$\sup_{u\in\mathcal{U}}[L_fh(x)+L_gh(x)u+\kappa(h(x))]\geq 0,\forall x\in\mathcal{E}, \tag{22}$$

where L represents the Lie derivatives.

To be more specific,

$$L_fh(x)=\frac{\partial h(x)}{\partial x}f(x),\ \ L_gh(x)=\frac{\partial h(x)}{\partial x}g(x). \tag{23}$$

ZCBF is a special control barrier function that comes with asymptotic stability (Xu et al., 2015). The existence of a ZCBF implies the asymptotic stability and forward invariance of $\mathcal{S}$ as proved in Xu et al. (2015).

Based on the ZCBF defined in Definition 2, a safety barrier for safety-critical systems can be constructed. Concretely, we aim to design a safety barrier for the uncertain system (1) to keep the state x in the forward invariant safety set $\mathcal{S}$, which requires to hold $\dot{h}(x)\geq-\kappa(h(x))$.

With the learned disturbances $\tilde{d}(x)$ via IGPs and the high confidence interval $\mathcal{D}$ (7) for the uncertain dynamical system (1), the following safe control space K_{rzbf} is formulated as shown in our previous work (Zheng et al., 2020)

$$K_{rzbf}(x)=\{u\in\mathcal{U}|\inf_{d\in\mathcal{D}(x)}[\dot{h}(x)+\kappa(h(x))]\geq 0\}. \tag{24}$$

where $h(x)$ is a ZCBF, $\dot{h}(x)=\frac{\partial h(x)}{\partial x}\dot{x}=L_{\hat{f}}h(x)+L_gh(x)u+L_{\tilde{d}}h(x)$, where the $L_{\hat{f}}h(x)$ and $L_{\tilde{d}}h(x)$ denotes the Lie derivative of h with respect to the known nominal model $\hat{f}$ of f and the learned disturbances $\tilde{d}(x)$, respectively.

Lemma 1. *Given a set $\mathcal{S}\subset\mathbb{R}^n$ defined by (21) with an associated ZCBF $h(x)$, the control input $u\in K_{rzbf}$ has a probability of at least $(1-\delta)$, $\delta\in(0,1)$, to guarantee the forward invariance of the set $\mathcal{S}$ for the uncertain dynamical system (1)*

Proof. From (7), there is a probability of at least $(1-\delta)$ such that the bounded model uncertainty $d(x)\in\mathcal{D}(x)$ for all $x\in\mathcal{X}$. Since the control input $u\in\mathcal{U}$ of the safe control space K_{rzbf} satisfies the constraint in (24), the following result holds with a probability of at least $(1-\delta)$:

$$\dot{h}(x)+\kappa(h(x))\geq 0,\forall x\in\mathcal{S}. \tag{25}$$

As a result, the control input $u\in\mathcal{U}$ of the safe control space K_{rzbf} has a probability of at least $(1-\delta)$ to guarantee the forward invariance of the set $\mathcal{S}$ for the uncertain dynamical system (1) as proven in Ames et al. (2017). □

For convenience, with the estimated high confidence interval $\mathcal{D}$ (7) via GPs, the constraint in (24) can be equivalently expressed as

$$L_{\hat{f}}h(x)+L_gh(x)u+L_{\mu}h(x)-c_{\delta}|L_{\sigma}h(x)|\geq-\kappa(h(x)), \tag{26}$$

where $L_{\mu}h(x)$ and $L_{\sigma}h(x)$ denote the Lie derivatives of $h(x)$ with respect to μ and σ, respectively.

Remark 5. Note that with more informative data collected, the bounded uncertainty σ (6) will gradually decrease. The proof of such conclusion can be obtained using the partitioned matrix equations, see Appendix. An alternative information theoretic argument is given in Srinivas et al. (2012) and Williams and Vivarelli (2000). Thus, the probability rendering $\mathcal{S}$ forward invariant is much higher than $(1-\delta)$ in most cases.

3.3.2. Sampling guidance scheme

In this section, we develop a sampling guidance scheme for CEMPC under uncertain disturbances based on the stability property. Sampling distribution could easily deviate from the low-cost regions in the case of external disturbances. Based on a deviated distribution, the optimization may get stuck in the local minimums. The Lyapunov methods have proven to be an efficient way to improve sampling performance for nonlinear systems in model-based RL (Berkenkamp et al., 2016, 2017). To guide the sampling distribution of CEMPC to the low-cost region in the case of uncertain disturbances, the stability constraints with regard to the main task, corresponding to the first term in the cost function (3), can be constructed based on the CLF $V(x)$: $\dot{V}(x) \le -\alpha V(x)$, $\alpha > 0$ (Khalil and Grizzle, 2002).

With learned disturbances $\widetilde{d}$ estimated by the IGPs, CLF can be utilized to construct a stability control set K_{rclf} (Zheng et al., 2020)

$$K_{rclf}(x) = \{u \in \mathcal{U} | \sup_{d \in D(x)} [\dot{V}(x) + \alpha V(x)] \le 0\}, \tag{27}$$

where $\dot{V}(x) = \frac{\partial V(x)}{\partial x}\dot{x} = L_{\hat{f}}V(x) + L_g V(x)u + L_{\widetilde{d}}V(x)$ and $\alpha > 0$.

With the estimated high confidence interval $\mathcal{D}$ (7) via GPs, the constraint in (27) can be simplified as:

$$L_{\hat{f}}V(x) + L_g V(x)u + L_\mu V(x) + c_\delta |L_\sigma V(x)| \le -\alpha V(x), \tag{28}$$

where $L_\mu V(x)$ and $L_\sigma V(x)$ denote the Lie derivatives of $V(x)$ with respect to μ and σ, respectively.

3.3.3. Auxiliary controller

Considering safety constraint (26) and stability constraint (28) with slack variables directly in the LB-CEMPC controller, one needs to compute the variance of the predicted states and propagate the variances along the prediction horizon. As a result, it can lead to a heavy computation burden.

We design a MI auxiliary controller to integrate constraints (26) and (28) to minimally modify the outputs of the MPC and formally satisfy these constraints. The proposed control scheme decouples the requirements of predictive and adaptive control performance in a way of minimal intervention. The safety requirements are considered as soft constraints in the non-differentiable objective function of LB-CEMPC, while they are enforced in a form of CBF in the MI controller. The MI auxiliary controller considers the uncertainty of the disturbance and forms an efficient QP controller to minimally modify the control outputs.

With conditions (26) and (28), a QP (29)–(32) can be constructed to modify the control inputs u_{MPC} of the learning-based CEMPC in a way of minimal intervention. The nonlinear system under environmental uncertainties can be guaranteed safe and guided to a low-cost region of the main task in a high probability by solving the following QP.

$$u^*(x) = \underset{(u,\epsilon,\eta)\in\mathbb{R}^{m+1}}{\arg\min} \; \|u - u_{MPC}\|^2 + \lambda_\epsilon \epsilon^2 + \lambda_\eta \eta^2 \tag{29}$$

$$\text{s.t.} \quad A_{cbf}u + b_{cbf} \le \epsilon, \tag{30}$$

$$A_{clf}u + b_{clf} \le \eta, \tag{31}$$

$$u_{min} \le u \le u_{max}, \tag{32}$$

where $u_{min}, u_{max} \in \mathcal{U}$ are the lower and upper bound of the control inputs, respectively. $\lambda_\epsilon, \lambda_\eta \in \mathbb{R}^+$ are penalty coefficients of the slack variables $\epsilon \in \mathbb{R}$ and $\eta \in \mathbb{R}$, respectively. $A_{cbf} = -L_g h(x)$, $b_{cbf} = -L_f h(x) - L_\mu h(x) + c_\delta |L_\sigma h(x)| - \kappa(h(x))$, $A_{clf} = L_g V(x)$, $b_{clf} = L_f V(x) + L_\mu V(x) + c_\delta |L_\sigma V(x)| - \alpha V(x)$. The feasibility of the QP (29)–(32) can be ensured with the slack variables, while the violations of safety constraints can be heavily penalized as long as the corresponding coefficients are large enough.

The designed MI auxiliary control scheme is shown in Algorithm 2. We can always directly apply the control outputs u_{MPC} of the learning-based CEMPC (Algorithm 1, line 10) if the u_{MPC} satisfies the constraints (26) and (28). Otherwise, it is modified by solving the QP (29)–(32).

Algorithm 2 MI Auxiliary Control Scheme

```
Require:
    $x_k$: Current state,
    $u_{MPC}$: Control inputs from learning-based CEMPC.
1: $u_k \leftarrow u_{MPC}$
2: if (26) and (28) Infeasible then
3:     $u_k \leftarrow$ Solve QP (29)–(32) with $x_k$ and $u_{MPC}$
4: end if
```

Remark 6. Note that the optimization (29) is not sensitive to the parameters λ_ϵ and λ_η. The violation of the safety (26) and stability constraints (28) can be heavily penalized as long as the λ_ϵ and λ_η are large enough (e.g. $\lambda_\epsilon = 10^{30}$, $\lambda_\eta = 10^{20}$). Besides, λ_ϵ can be set extremely larger than λ_η to make the safety constraints much stricter.

4. Simulation studies

In this section, the proposed control architecture is verified on a task of simultaneous trajectory tracking and obstacle avoidance of a quadrotor. For a safety-critical quadrotor with a limited sensing range, avoiding an uncertain number of detected obstacles around or on the trajectory can be conveniently encoded as some non-differentiable running-cost terms into the objective function. It is difficult to describe such a complex task and trade off the tracking and safety well with a simplified differentiable objective function. Besides, a quadrotor is prone to uncertain wind disturbances, which are hard to be accurately modeled. Uncertain wind disturbances not only pose a critical challenge to achieve accurate tracking, but also may cause the quadrotor to collide in a cluttered obstacle field.

4.1. Quadrotor dynamics and control

The quadrotor is a well-modeled dynamical system with torques and forces generated by four rotors and gravity. The Euler angles (roll ϕ, pitch θ, and yaw ψ) are defined with the ZYX convention. The attitude rotation matrix $R \in SO(3)$ from the body frame $\mathcal{B}$ to the global frame $\mathcal{W}$ can be written as (Hehn and D'Andrea, 2015):

$$R = \begin{bmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\phi c\psi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix}, \tag{33}$$

where s and c denote sin and cos, respectively.

The nonlinear quadrotor system can be modeled as following (Shi et al., 2019):

$$\dot{p} = v, \tag{34}$$

$$m\dot{v} = mge_3 + Rf_u + d_w, \tag{35}$$

$$\dot{R} = RS(\omega), \tag{36}$$

where m and g denote the mass and the gravity acceleration, respectively. $e_3 = [0,0,1]^T$ is the unit vector, $p = [p_x, p_y, p_z]^T$ and $v = [v_x, v_y, v_z]^T$ denote translational position and velocity in $\mathcal{W}$, respectively. $f_u = [0,0,f_T]^T$ with f_T the total thrust generated from the four rotors in $\mathcal{B}$, and $S(\cdot)$ is skew-symmetric mapping. The uncertain wind disturbances acting on the quadrotor dynamics is represented as $d_w = K_{drag}(v_w - v)$, where $v_w \in \mathbb{R}^3$ is the velocity of wind disturbances in $\mathcal{W}$ and $K_{drag} \in \mathbb{R}^{3\times3}$ is the drag coefficient diagonal matrix. We define in the dynamics Eq. (1) the state $x = [p_x, p_y, p_z, v_x, v_y, v_z, \phi, \theta, \psi]^T$ and the control input $u = [f_T, \omega^T]^T$, where $\omega = [\omega_x, \omega_y, \omega_z]^T$ is the body rotational rates. It is assumed that the body rotational rates are directly controllable through the fast response onboard controller of commercial quadrotors.

Table 1
Parameter table for simulation setup.

Parameter	Value
The mass of quadrotor m	0.08 kg
The arm length from the center of mass to each motor	0.11 m
The maximum total thrust $f_{T_{max}}$	1.3 N
The maximum body rotational rates ω_{max}	$[3.49, 3.49, 5.24]^T$
The minimum body rotational rates ω_{min}	$[-3.49, -3.49, -5.24]^T$
The detection range for the obstacle	2 m
The simulation time	20 s
The control frequency	50 Hz
The hyperparameters L of GP kernel	1
The hyperparameters σ_f of GP kernel	1
The max size of IGP dataset n	20
The confidence parameter c_δ	3
The predictive time steps H of CEMPC per iteration	20
The number of the samples M for CEMPC per iteration	100
The size of the elite set K for CEMPC per iteration	10
The number of iterations N of CEMPC	5
The minimum variance bound for optimization Σ_{min}	0.001
The update rate β	0.25
The slack variable λ_ϵ in the QP	10^{30}
The slack variable λ_η in the QP	10^{20}

For the task of simultaneous trajectory tracking and obstacle avoidance, the cost function $\mathcal{L}$ in the objective function (18) can be defined as

$$\mathcal{L}(x) = \iota(x - x_d)^T Q \iota(x - x_d) + \sum_{i}^{N_0} w_i \frac{\mathbb{I}_{C_i}}{r_i} \tag{37}$$

$$C_i = \{x \mid \|r_i\| < 0.8\}, \tag{38}$$

where $\iota = \mathrm{diag}(1, 1, 1, 1, 1, 1, 0, 0, 0)$ extracts the position and velocity states from the state x, the weight coefficient matrix $Q = \mathrm{diag}(8.5, 8.5, 8.5, 1.5, 1.5, 1.5)$, N_0 denotes the number of the detected obstacles, and the weight coefficient w_i is a positive constant, $\forall\ i = 1, \dots, N_0$. $x_d = [p_d^T, \dot{p}_d^T, \phi_d, \theta_d, \psi_d]^T$ is the desired state, where $\phi_d, \theta_d, \psi_d \in \mathbb{R}$ are the desired attitudes, r_i is the shortest Euler distance from the quadrotor to the ith detected obstacle, and $\mathbb{I}_{C_i}$ is an indicator function that will be turned on if the state is in the set C_i.

Remark 7. Note that the second term in (37) is designed to show the predictivity inherent in the CEMPC for obstacle avoidance. The trade-off between the safety and tracking performance can be adjusted by the weight w_i in (37).

Remark 8. Note that designing a differentiable cost function for this task would be nontrivial! Besides, composing a cost function with different nonlinear cost terms may lead to local minima, which brings difficulties to the optimization. In this case, specifying a non-differentiable cost function with several interpretable terms and different weights can be convenient according to the importance of task requirements.

4.2. Simulation setup

A simulation platform is created using Python 3.6 on an Intel Xeon X5675 CPU with 3.07 GHz clock frequency. A quadrotor model of Blade mQX quadrotor is used with the parameters set referred to Cabecinhas et al. (2014). The main simulation parameter values are shown in Table 1.

A wind model in Cole and Wickenheiser (2018) is utilized to evaluate the algorithm performance. Wind velocity consists of a constant component v_c and a turbulent component v_t, i.e., $v_w = v_c + v_t$. The turbulence wind uses the von Kármán velocity model defined analytically in the specification MIL-F-8785C (Moorhouse and Woodcock, 1980), with the specific low-altitude model for the model parameters. The drag coefficient diagonal matrix $K_{drag} = \mathrm{diag}(0.03, 0.03, 0.03)^T$. Four magnitudes of constant wind components are used to validate the trajectory tracking performance, as shown in Table 2.

Three GPs are built to estimate the effects of the unknown wind disturbances d_w. Each GP uses the same squared-exponential kernel.

The quadrotor with a limited sensing range is required to track a reference trajectory while avoiding obstacles under varying wind disturbances in **three scenarios**. The initial state of the quadrotor is set as the initial position of the reference trajectory with zero velocity and attitude.

Scenario 1 is designed to validate the effectiveness of the proposed learning-based CEMPC method with the MI controller. The reference trajectory is given as a spiral curve $p_d(t) = [2sin(0.5t), 2{-}2cos(0.5t), 0.2t]^T$ and $\psi_d(t) = 0$. It lies in a dense cluttered obstacle field under time-varying wind disturbances, where a moving obstacle flies along the reference trajectory to the quadrotor with a speed of 0.8 m/s. The weight coefficients are set $w_i = 10, \forall\ i = 1, \dots, N_0$ in (37).

Scenario 2 is studied to further validate the safety and tracking performance trade-off in the design of the non-differentiable cost function (37), where the weight coefficient w_i is set to different orders of magnitude, $\forall\ i = 1, \dots, N_0$. The difference to **Scenario 1** is that there are only one static obstacle and one dynamic obstacle along the reference trajectory. It is devised to clearly show the proactive ability inherent in the CEMPC framework for obstacle avoidance.

Scenario 3 is studied to illustrate the efficiency of LB-CEMPC with different prediction horizons and sample sizes. There is no obstacle in **Scenario 3** and the reference velocity is set 8 m/s. A trade-off between efficiency and tracking performance can be obtained from the simulation results of **Scenario 3**.

The barrier function h can be constructed with the distance from the quadrotor to the obstacles within its sensing region. This distance can be obtained with the largest ellipsoidal region of obstacle-free space, which can be efficiently computed using the IRIS algorithm (Deits and Tedrake, 2015) through semi-definite programming. Specifically, an ellipsoid can be represented as an image of the unit ball:

$$E(C, \zeta) = \{Co + \zeta \mid \|o\| = 1\}, \tag{39}$$

where $o \in \mathbb{R}^{3\times1}$, $o^T o = 1$ denotes a unit ball, the mapping matrix $C \in \mathbb{R}^{3\times3}$ and the offset vector $\zeta \in \mathbb{R}^{3\times1}$ can be obtained using the IRIS algorithm. For the vector ϵ on the ellipsoid $\epsilon \in E$, we have $(\epsilon - \zeta)^T {C^{-1}}^T C^{-1}(\epsilon - \zeta) = 1$. The CBF can be constructed to enforce the quadrotor to stay within the safety ellipsoid region as

$$h(x) = 1 - (\iota_p x - \zeta)^T {C^{-1}}^T C^{-1}(\iota_p x - \zeta). \tag{40}$$

where $\iota_p = \mathrm{diag}(1, 1, 1, 0, 0, 0, 0, 0, 0)$ extracts the position from the state x.

For the main task of trajectory tracking, the CLF is designed as

$$V(x) = (\iota_p x - p_d)^T Q_p(\iota_p x - p_d) + (\iota_v x - \dot{p}_d)^T Q_v(\iota_v x - \dot{p}_d), \tag{41}$$

where $Q_p = \mathrm{diag}(0.8, 0.8, 0.8)$, $Q_v = \mathrm{diag}(0.2, 0.2, 0.2)$ and $\iota_v = \mathrm{diag}(0, 0, 0, 1, 1, 1, 0, 0, 0)$ extracts the velocity from the state x. The extended $\mathcal{K}$ class function κ is chosen as $\kappa(h(x)) = 10h(x)$, and the positive constant $\alpha = 0.1$. The QP is solved with CVXOPT solver (Andersen et al., 2013).

4.3. Results

To validate the effectiveness of the proposed control scheme, an ablation study is conducted to assess that the proposed method is able to: (1) learn and adapt to the uncertain environmental disturbances, (2) handle the task with a non-differentiable objective function, (3) achieve safe control with a low tracking error, and (4) provide a way to trade off between safety and tracking performance. We compare the following four methods:

- **CEMPC**: A CEM-based MPC without the IGPs for learning the uncertain disturbances.
- **LB-CEMPC**: A learning-based CEMPC without the MI auxiliary controller.
- **LB-CEMPC-CBF**: The proposed learning-based CEMPC without the sampling guidance scheme (28) in the MI auxiliary controller.
- **LB-CEMPC-MI**: The proposed learning-based CEMPC with the designed MI auxiliary controller.

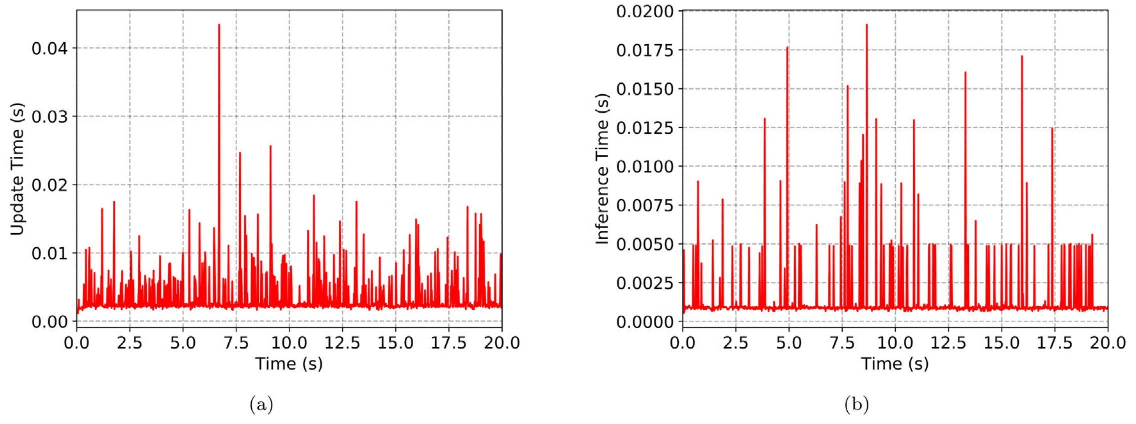


Fig. 2. (a) The time of learning, and (b) the time of inference of the IGP under the Wind-4.

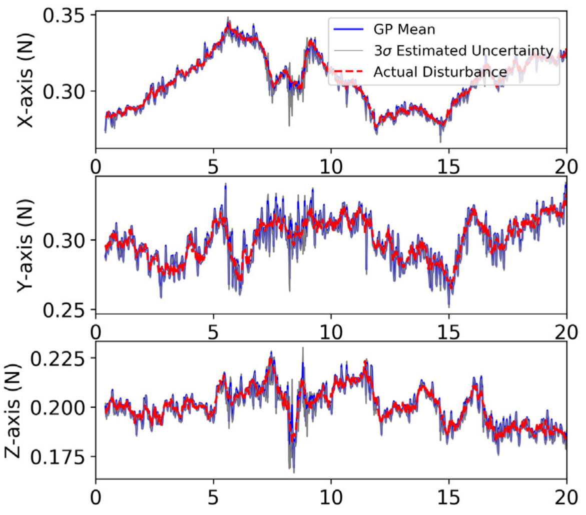


Fig. 3. The wind disturbances estimated by IGPs in three axes on the spiral trajectory under the Wind-4.

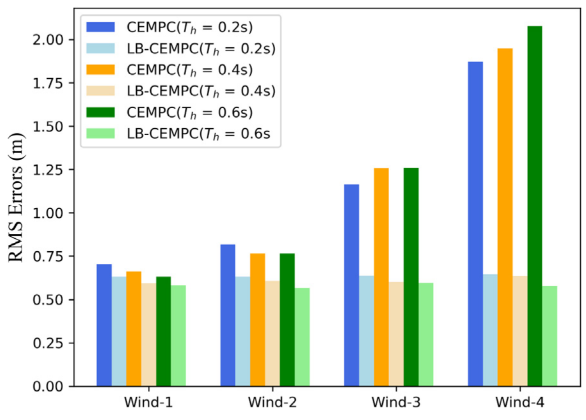


Fig. 4. The RMS errors of the quadrotor trajectory tracking using the LB-CEMPC and the CEMPC controllers with different prediction horizons under wind disturbances. The proposed LB-CEMPC outperforms the baseline CEMPC in four settings of time-varying wind disturbances.

4.3.1. *Learning performance*

Figs. 2(a) and 2(b) show the time of learning and inference with the IGP at each iteration under Wind-4. The learning time is less than 0.02 s most of the time and the inference time keeps below 0.02 s. It shows that the IGP can be used as an online learning technique with fast learning and inference time. Note that the code in Python has not been optimized for speed and can be accelerated in a C++ implementation.

The uncertain wind disturbances in three axes modeled by GPs are shown in Fig. 3. It can be seen that the IGPs can estimate well the actual wind disturbances with turbulence. The actual disturbances lie within the uncertainty bounds of the estimations.

4.3.2. *Trajectory tracking performance*

As shown in Table 2 and Fig. 4, the CEMPC with a longer horizon achieves smaller tracking RMS error under mild Wind-1 or Wind-2, due to the robustness from the predictivity and receding horizon optimization inherent in the MPC (Mayne, 2014). However, under larger wind disturbances, e.g. Wind-3 and Wind-4, the RMS error of the CEMPC instead increases as the prediction horizon gets longer, since the accumulation of the model error along the multi-step predictions degrades the control performance. The tracking errors of the CEMPC and LB-CEMPC with a prediction horizon of T_h = 0.6 s under Wind-1 and Wind-4 are shown in Fig. 5. It can be seen that the tracking errors of the CEMPC are similar to that of the LB-CEMPC under Wind-1, while the tracking errors of the CEMPC get quite high without the IGPs under the stronger Wind-4. These results indicate that the LB-CEMPC benefits from the IGPs learning and compensating the wind disturbances.

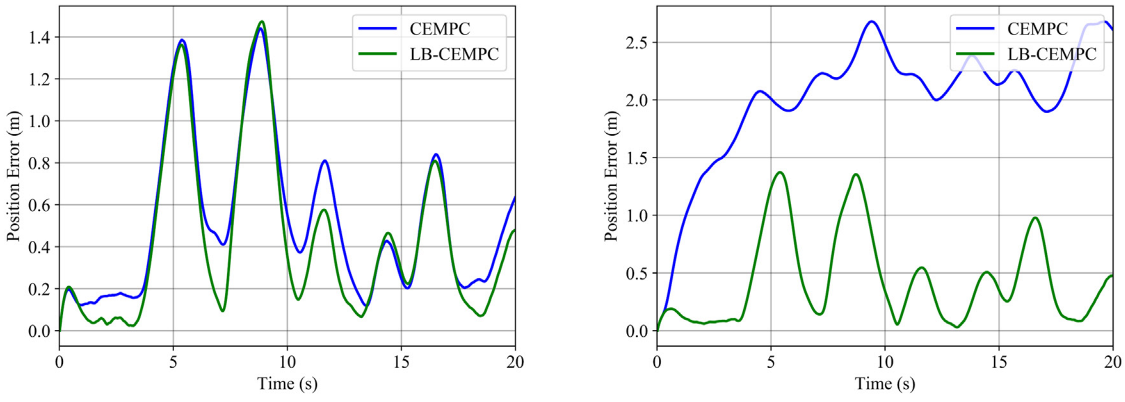
The proposed LB-CEMPC-MI controller achieves the lowest tracking errors among the four controllers under different all four magnitudes of wind disturbances, as shown in Table 2. Figs. 6, 7 and 8 show the control performance with our proposed LB-CEMPC-MI control framework on the quadrotor under the Wind-4. At about 4.3 and 7.5 s, the quadrotor deviates from the reference trajectory with increasing tracking errors. It has changed its trajectory to safely avoid the collision with the gray static obstacle on the trajectory. Fig. 7(b) shows that the barrier function keeps positive and safety is ensured when avoiding the obstacle. Besides, compared with the other three baseline controllers, the LB-CEMPC-MI controller can achieve smoother obstacle avoidance behavior as shown in Fig. 6(a). Moreover, Figs. 6(a) and 8 illustrate that the MI auxiliary controller can effectively help the main task of trajectory tracking by guiding the sampling distribution. The quadrotor can be guided quickly back to the references when deviating from the desired trajectories due to the unexpected obstacle avoidance. These results demonstrate that the LB-CEMPC-MI control scheme can drive the system to track the reference trajectory stably when the trajectory is safe and can relax the tracking to avoid obstacles safely when it is to collide with the detected obstacles.

4.3.3. *Trade-off between safety and tracking performance*

To validate the trade-off between safety and tracking performance, **Scenario 2** with Wind-2 is fixed in this part. To demonstrate the

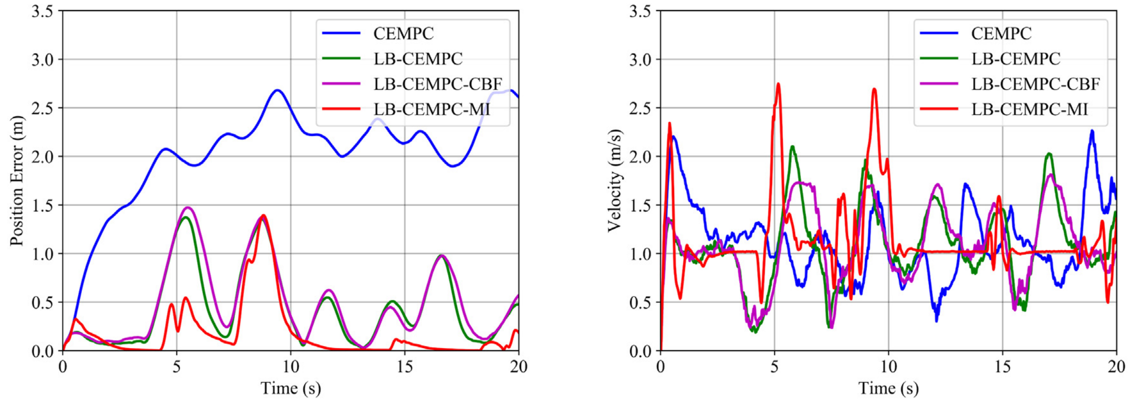
Table 2
Statistics of RMS errors (in meter) with different control schemes and configurations.

Prediction horizon T_h	High-level	Scheme	Wind-1 $v_c = 5$ m/s	Wind-2 $v_c = 8$ m/s	Wind-3 $v_c = 10$ m/s	Wind-4 $v_c = 12$ m/s
0.2 s	CEMPC	–	0.704	0.818	1.164	1.872
0.2 s	LBCEMPC	–	0.630	0.631	0.636	0.644
0.2 s	LBCEMPC	CBF	0.632	0.638	0.640	0.636
0.2 s	**LB-CEMPC**	**MI**	**0.354**	**0.195**	**0.246**	**0.388**
0.4 s	CEMPC	–	0.662	0.765	1.257	1.950
0.4 s	LBCEMPC	–	0.593	0.608	0.601	0.636
0.4 s	LBCEMPC	CBF	0.578	0.625	0.601	0.618
0.4 s	**LB-CEMPC**	**MI**	**0.215**	**0.124**	**0.214**	**0.349**
0.6 s	CEMPC	–	0.631	0.737	1.260	2.077
0.6 s	LBCEMPC	–	0.582	0.566	0.596	0.579
0.6 s	LBCEMPC	CBF	0.556	0.583	0.604	0.625
0.6 s	**LB-CEMPC**	**MI**	**0.170**	**0.178**	**0.271**	**0.341**



(a) Wind-1. (b) Wind-4.

Fig. 5. Evolution of trajectory tracking errors using the CEMPC and the LB-CEMPC under the mild Wind-1 and the strong Wind-4 disturbances.

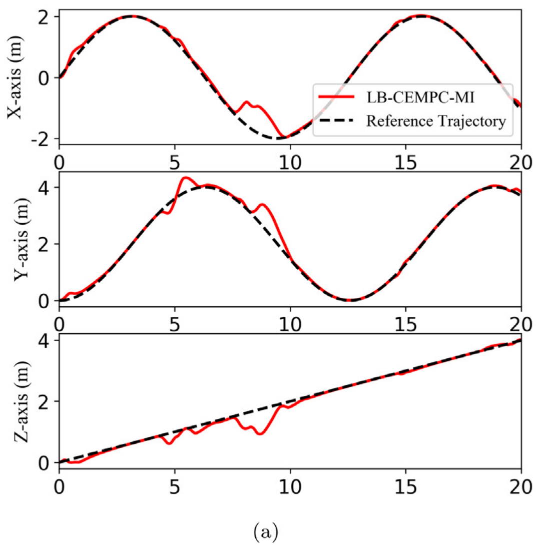


(a) Tracking error. (b) Velocity

Fig. 6. Safe trajectory tracking in **Scenario 2**. (a) Tracking error with different controllers, and (b) Tracking velocity with different controllers under Wind-4 disturbances.

Table 3
Trajectory tracking control and safety performance with different weight coefficients.

w_i	MI scheme	Time to avoid obstacle 1 (in s)	Time to avoid obstacle 2 (in s)	Minimum Barrier Value (in m)	Max tracking error (in m)	RMS error (in m)	Collide or not
10	–	3.283	11.010	0.990	1.952	0.630	No
1	–	–	–	–	–	–	Yes
0.1	–	–	–	–	–	–	Yes
0.01	–	–	–	–	–	–	Yes
0	–	–	–	–	–	–	Yes
10	CBF	3.283	11.010	0.994	1.957	0.632	No
1	CBF	3.737	11.136	0.427	1.321	0.311	No
0.1	CBF	3.737	11.364	0.617	0.772	0.214	No
0.01	CBF	4.289	11.710	0.485	0.703	0.167	No
0	CBF	4.293	11.717	0.481	0.723	0.190	No



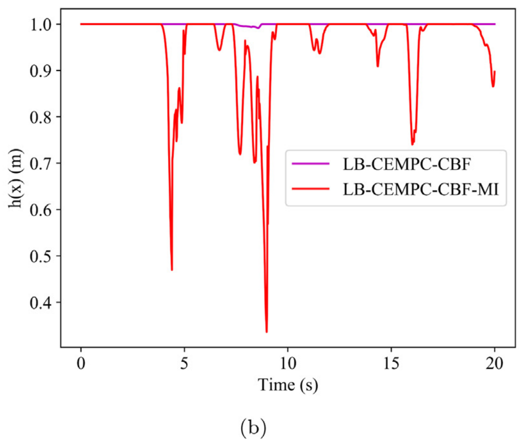


Fig. 7. Safe trajectory tracking in **Scenario 2**. (a) Position of the quadrotor with the proposed LB-CEM-MI control scheme, and (b) values of CBF in the LB-CEMPC-CBF and the LB-CEMPC-MI under Wind-4 disturbances.

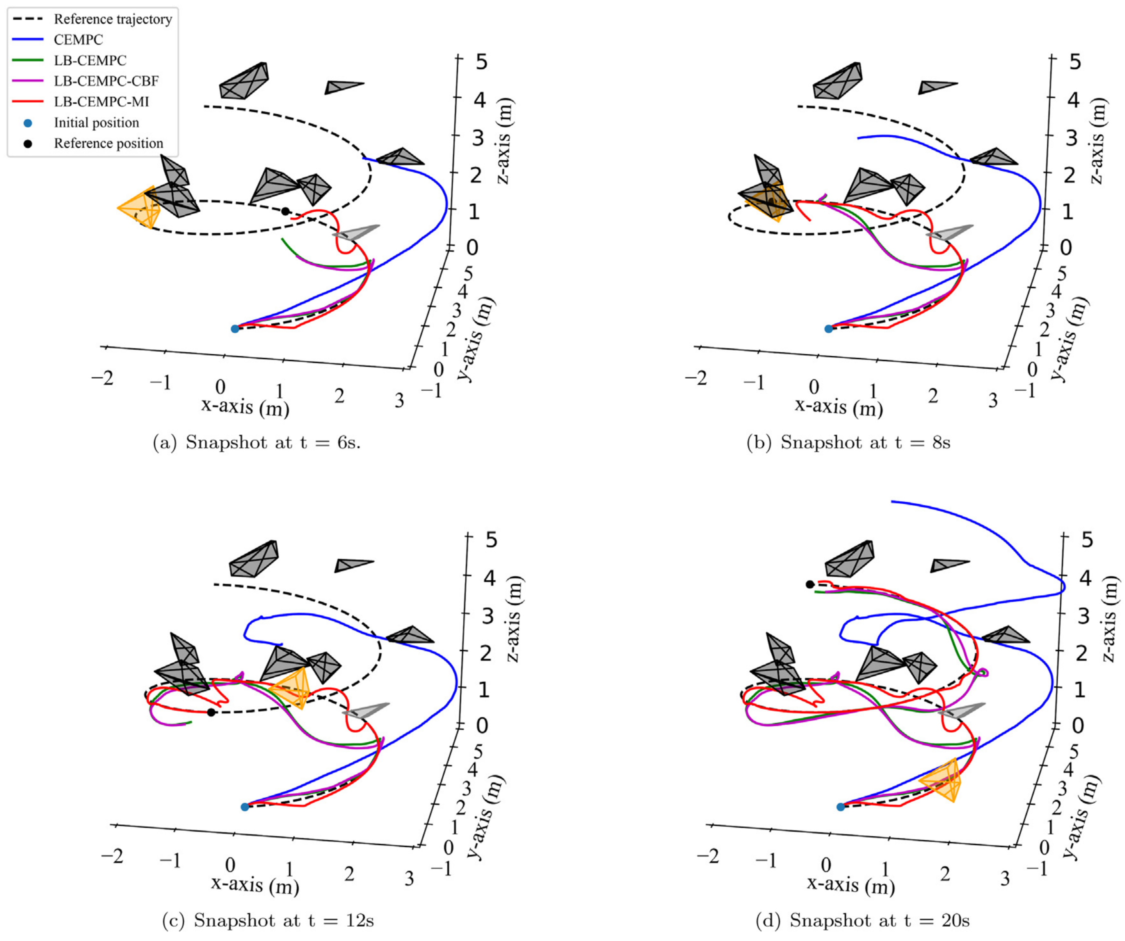


Fig. 8. Numerical validation of the trajectory tracking in **Scenario 1**, where a quadrotor flies through a densely cluttered obstacle field under uncertain Wind-4 disturbances. Snapshots of the simulation are shown in 8(a)–8(d). The black dashed line denotes the reference trajectory. The irregular polyhedron in orange and gray denote the dynamic and static obstacles crossing the reference trajectory, respectively.

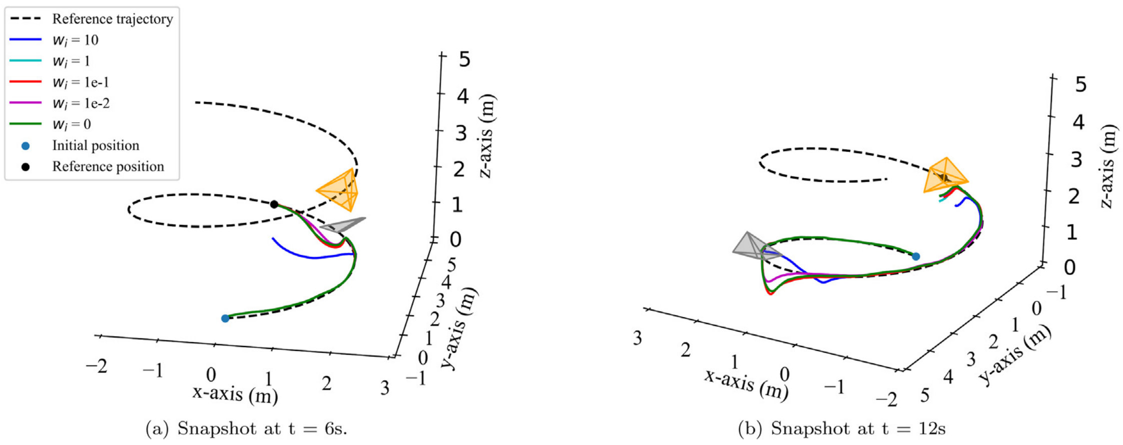
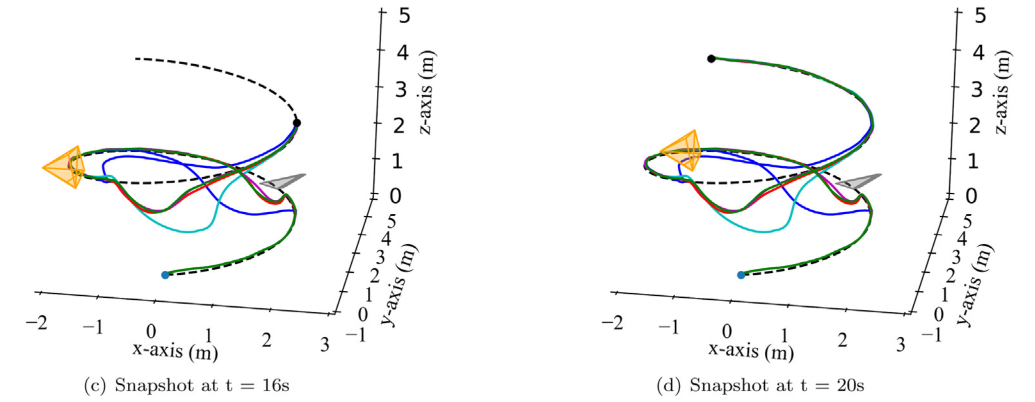



Fig. 9. Numerical validation of the trajectory tracking in the **Scenario 2**, where a quadrotor tracks the reference spiral trajectory under uncertain Wind-2 disturbances. Snapshots of the simulation are shown in 9(a)–9(d). The black dashed line denotes the reference trajectory $p_d(t)$. The irregular polyhedron in orange and gray denote the dynamic and static obstacles crossing the reference trajectory, respectively. The quadrotor tracks the reference trajectory and strictly guarantees non-collision with the obstacles.

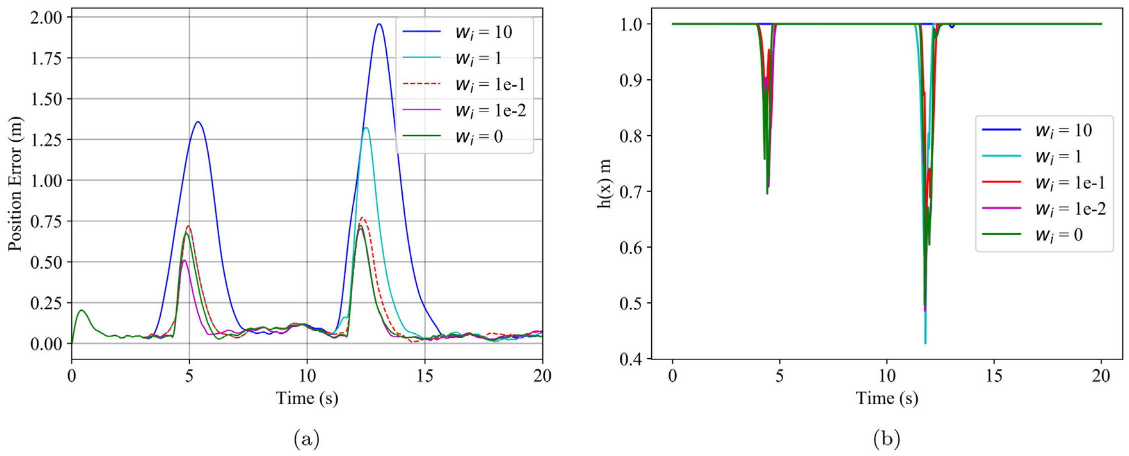


Fig. 10. Simulation results in **Scenario 2**. The prediction horizon is $T_h = 0.6$ s. (a) The evolution of tracking errors with different weight coefficients w_1, where the dashed red line is used to distinguish the tracking error using the controller with w_1=1e−1; (b) The values of the CBF.

effects of predictivity inherent in the MPC and the CBF-enforced safety scheme, we only keep the CBF (40) in the MI scheme without the CLF-based sampling guidance scheme and design the LB-CEMPC algorithm with different coefficients w_i in (37) for obstacle avoidance. Controllers without MI scheme are also compared with controllers with CBF scheme, as shown in Table 3.

Statistic of the simulations is listed in Table 3 with the trajectories graphically shown in Fig. 9 and tracking errors shown in Fig. 10(a). We highlight four key takeaways from these results. Firstly, the controllers with $w_i = 10$ predict to avoid the unexpectedly gray static obstacle and the orange dynamic obstacle in advance at 3.28 s and 11.01 s , respectively, while the controllers with $w_i = 0$ avoid these two obstacles at later 4.29 s and 11.72 s. It illustrates the proposed method can keep the predictive ability for obstacle avoidance even under time-varying environmental disturbances. Secondly, the RMS and maximum tracking errors increase with a larger positive weight w_i, while the time for obstacle avoidance decreases as observed in Table 3. This indicates that there is a trade-off between safety margin and tracking

Table 4
Running time and tracking performance of CEMPC with different parameters.

Predictive horizon T_h (s)	Number of samples M	CEMPC optimization time (ms)	RMS error (m)
0.2	100	38.960	1.808
0.2	200	47.607	1.783
0.2	300	52.106	1.714
0.4	100	73.575	1.165
0.4	200	88.966	1.007
0.4	300	103.984	0.950
0.6	100	114.629	0.968
0.6	200	127.892	0.792
0.6	300	151.410	0.722

performance, which can be adjusted with the weight coefficient w_i in the cost function (37). When we need a smaller w_i for better tracking performance, the CBF scheme is necessary to avoid collisions. Thirdly, when safety constraints are already satisfied, that is, $\omega_i = 10$, it can be seen from Table 3 that the time to avoid the static and dynamic obstacles are the same whether the CBF scheme exists or not. And the max tracking error and RMS error are also similar. This reflects the minimal intervention of the MI scheme. Besides, to obtain an intuitive view on the safety performance under disturbances, Fig. 10(b) shows the safety performance of the quadrotor with CBF scheme tracking the reference trajectory. It can be seen that the values of the CBF keep positive, which indicates that the position of the quadrotor always stays within the safe obstacle-free ellipsoid region.

4.3.4. Real-time performance of CEMPC

The time consumption and tracking performance of CEMPC are compared under different parameters in **Scenario 3**, as shown in Table 4. As the iteration number is set the same, the running time increases with larger sample size or prediction horizon, which corresponds with the theoretical analysis of the algorithm complexity. From Table 4, it can be seen that increasing the number of samples helps less for tracking performance than increasing the predictive horizon. In addition, the improvement of tracking performance by increasing the predictive horizon is also limited, as increasing T_h for 0.2 s to 0.4 s decreases the RMS errors much more than increasing T_h for 0.4 s to 0.6 s. Therefore, the predictive horizon is required to be chosen according to the circumstances and the sample size can be fixed for better real-time performance. Note that the code and computation are not been optimized for speed.

4.4. Discussion

The predictive nature of the MPC brings proactivity to the control scheme, which improves the trajectory tracking accuracy and introduces conservation in terms of safety to the system, as shown in Section 4.3. It results in safer behaviors of the system but also degrades the control performance of the main task, e.g. trajectory tracking accuracy. Such trade-off could be considered by adjusting the weight coefficients w_i in the cost function (37) according to the requirements of practical applications. For example, if larger weight coefficients w_i are set to achieve better foresight to avoid obstacles, there will be a larger tracking RMS error, given a determined prediction horizon T_h. In contrast, if we choose small weight coefficients w_i to achieve low tracking RMS error, there will be poor foresight for obstacle avoidance. Actually, it is flexible for the designer to set the weight coefficients w_i according to their considerations. The designed MI scheme reserves the safety and guides the optimization, which enables the customized and convenient design of the cost function for high-level tasks.

5. Conclusion

In this paper, a safe learning-based MPC architecture is designed to optimize the nonlinear system with a non-differentiable objective function under uncertain environmental uncertainties. Our proposed approach allows the convenient design of objective function using simple but non-differentiable running-cost terms. The IGPs are utilized to estimate model uncertainties with a low computational burden and augment the prior predictive model in the MPC. Solved with the sampling-based CEM, the proposed CEMPC is augmented by an auxiliary controller based on the control Lyapunov function and the CBF to guide the sampling process and theoretically endows the system with safety in a way of minimal intervention. We provide numerical simulation results comparing the CEMPC, LB-CEMPC, LB-CEMPC-CBF, and the LB-CEMPC-MI algorithms on a quadrotor trajectory tracking and obstacle avoidance task under different wind disturbances in two different scenarios. The results show that the proposed LB-CEMPC-MI algorithm can successfully and safely optimize the quadrotor system with a conveniently designed non-differentiable objective function, achieving accurate tracking performance and safe obstacle avoidance under uncertain wind disturbances. In future work, the proposed learning-based MPC framework will be verified in hardware platforms under realistic conditions, and extended to solve a navigation problem considering uncertain environmental disturbances.

CRediT authorship contribution statement

Lei Zheng: Conceptualization, Methodology, Software, Formal analysis, Investigation, Writing – original draft, Writing – review & editing. **Rui Yang:** Conceptualization, Methodology, Writing – original draft. **Zhixuan Wu:** Methodology, Software, Writing – review & editing. **Jiesen Pan:** Methodology, Software. **Hui Cheng:** Conceptualization, Resources, Supervision, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix. Proof of Remark 5

Let $\sigma^2_{n+1}(x_*)$ be the predictive variance of a Gaussian process regression model at x_* given a dataset of size $n+1$. The corresponding predictive variance using a dataset of only the first n training points is denoted $\sigma^2_n(x_*)$. Then $\sigma^2_{n+1}(x_*) \le \sigma^2_n(x_*)$.

Proof. Firstly we have

$$\sigma^2_n(x_*) = k(x_*, x_*) - \mathbf{k}^{\mathsf{T}}_{n,*}(K_{\sigma,n} + \sigma^2_{noise} I)^{-1}\mathbf{k}_{n,*}, \tag{A.1}$$

$$\sigma^2_{n+1}(x_*) = k(x_*, x_*) - \mathbf{k}^{\mathsf{T}}_{n+1,*}(K_{\sigma,n+1} + \sigma^2_{noise} I)^{-1}\mathbf{k}_{n+1,*} \tag{A.2}$$

where

$$K_{\sigma,n+1} = \begin{bmatrix} K_{\sigma,n} + \sigma^2_{noise} I & \gamma \\ \gamma^{\mathsf{T}} & k(x_{n+1}, x_{n+1}) \end{bmatrix},$$

$$\mathbf{k}^{\mathsf{T}}_{n,*} = [k(x_1, x_*), k(x_1, x_*), \dots, k(x_n, x_*)]^{\mathsf{T}},$$

$$\mathbf{k}^{\mathsf{T}}_{n+1,*} = [k(x_1, x_*), k(x_1, x_*), \dots, k(x_{n+1}, x_*)]^{\mathsf{T}},$$

$$\gamma^{\mathsf{T}} = [k(x_1, x_{n+1}), k(x_1, x_{n+1}), \dots, k(x_{n+1}, x_{n+1})]^{\mathsf{T}}.$$

For simplicity, let $K_n = K_{\sigma,n} + \sigma^2_{noise} I$, $K_{n+1} = K_{\sigma,n+1} + \sigma^2_{noise} I$, $b = k(x_{n+1}, x_{n+1})$ and $e = k(x_{n+1}, x_*)$. Then we have $K_{n+1} = \begin{bmatrix} K_n & \gamma \\ \gamma^{\mathsf{T}} & b \end{bmatrix}$, $\mathbf{k}_{n+1,*} = \begin{bmatrix} \mathbf{k}_{n,*} \\ e \end{bmatrix}$ and

$$\sigma^2_n(x_*) = k(x_*, x_*) - \mathbf{k}^{\mathsf{T}}_{n,*} K_n^{-1} \mathbf{k}_{n,*}, \tag{A.3}$$

$$\sigma_{n+1}^2(x_*) = k(x_*, x_*) - \begin{bmatrix}\mathbf{k}_{n,*}^\intercal & e\end{bmatrix} \begin{bmatrix} K_n & \gamma \\ \gamma^\intercal & b \end{bmatrix}^{-1} \begin{bmatrix}\mathbf{k}_{n,*} \\ e\end{bmatrix}. \tag{A.4}$$

Notice that K_n^{-1}, K_{n+1}^{-1} are symmetric as K_n and K_{n+1} are symmetric, the inversion of the partitioned matrix in (A.4) is

$$K_{n+1}^{-1} = \begin{bmatrix} K_n & \gamma \\ \gamma^\intercal & b \end{bmatrix}^{-1} = \begin{bmatrix} \tilde{K}_n & \tilde{\gamma} \\ \tilde{\gamma}^\intercal & \tilde{b} \end{bmatrix} = \begin{bmatrix} K_n^{-1} + K_n^{-1}\gamma\tilde{b}\gamma^\intercal K_n^{-1} & -K_n^{-1}\gamma\tilde{b} \\ -\tilde{b}\gamma^\intercal K_n^{-1} & \tilde{b} \end{bmatrix}, \tag{A.5}$$

where $\tilde{b} = (b - \gamma^\intercal K_n^{-1}\gamma)^{-1}$. It is obvious that $\tilde{b} > 0$ as $\begin{bmatrix} K_n & \gamma \\ \gamma^\intercal & b \end{bmatrix}$ and K_n are positive definite. Then

$$\begin{aligned}
\sigma_n^2(x_*) - \sigma_{n+1}^2(x_*) &= \begin{bmatrix}\mathbf{k}_{n,*}^\intercal & e\end{bmatrix} \begin{bmatrix} K_n & \gamma \\ \gamma^\intercal & b \end{bmatrix}^{-1} \begin{bmatrix}\mathbf{k}_{n,*} \\ e\end{bmatrix} - \mathbf{k}_{n,*}^\intercal K_n^{-1}\mathbf{k}_{n,*} \\
&= \begin{bmatrix}\mathbf{k}_{n,*}^\intercal & e\end{bmatrix} \begin{bmatrix} \tilde{K}_n & \tilde{\gamma} \\ \tilde{\gamma}^\intercal & \tilde{b} \end{bmatrix} \begin{bmatrix}\mathbf{k}_{n,*} \\ e\end{bmatrix} - \mathbf{k}_{n,*}^\intercal K_n^{-1}\mathbf{k}_{n,*} \\
&= \mathbf{k}_{n,*}^\intercal \tilde{K}_n \mathbf{k}_{n,*} + \mathbf{k}_{n,*}^\intercal \tilde{\gamma} e + e\tilde{\gamma}^\intercal \mathbf{k}_{n,*} + e\tilde{b}e - \mathbf{k}_{n,*}^\intercal K_n^{-1}\mathbf{k}_{n,*} \\
&= \mathbf{k}_{n,*}^\intercal (K_n^{-1} + K_n^{-1}\gamma\tilde{b}\gamma^\intercal K_n^{-1})\mathbf{k}_{n,*} - \mathbf{k}_{n,*}^\intercal K_n^{-1}\gamma\tilde{b}e \\
&\quad - e\tilde{b}\gamma^\intercal K_n^{-1}\mathbf{k}_{n,*} + e\tilde{b}e - \mathbf{k}_{n,*}^\intercal K_n^{-1}\mathbf{k}_{n,*} \\
&= \mathbf{k}_{n,*}^\intercal K_n^{-1}\gamma\tilde{b}\gamma^\intercal K_n^{-1}\mathbf{k}_{n,*} - \mathbf{k}_{n,*}^\intercal K_n^{-1}\gamma\tilde{b}e - e\tilde{b}\gamma^\intercal K_n^{-1}\mathbf{k}_{n,*} + e\tilde{b}e \\
&= \tilde{b}(\mathbf{k}_{n,*}^\intercal K_n^{-1}\gamma - e)(\gamma^\intercal K_n^{-1}\mathbf{k}_{n,*} - e) \\
&= \tilde{b}(\mathbf{k}_{n,*}^\intercal K_n^{-1}\gamma - e)^2 \\
&\geq 0.
\end{aligned} \tag{A.6}$$

Then we have

$$\sigma_{n+1}^2(x_*) \leq \sigma_n^2(x_*). \quad \square \tag{A.7}$$

References

Ames, Aaron D., Coogan, Samuel, Egerstedt, Magnus, Notomista, Gennaro, Sreenath, Koushil, Tabuada, Paulo, 2019. Control barrier functions: Theory and applications. In: 2019 18th European Control Conference (ECC). IEEE, pp. 3420–3431.

Ames, Aaron D., Xu, Xiangru, Grizzle, Jessy W., Tabuada, Paulo, 2017. Control barrier function based quadratic programs for safety critical systems. IEEE Trans. Autom. Control 62 (8), 3861–3876.

Andersen, Martin S., Dahl, Joachim, Vandenberghe, Lieven, 2013. Cvxopt: A python package for convex optimization. abel.ee.ucla.edu/cvxopt.

Andersson, Joel A.E., Gillis, Joris, Horn, Greg, Rawlings, James B., Diehl, Moritz, 2019. Casadi: a software framework for nonlinear optimization and optimal control. Math. Program. Comput. 11 (1), 1–36.

Aswani, Anil, González, Humberto, Shankar Sastry, S., Tomlin, Claire J., 2013. Provably safe and robust learning-based model predictive control. Automatica 49 (5), 1216–1226.

Berkenkamp, Felix, Moriconi, Riccardo, Schoellig, Angela P., Krause, Andreas, 2016. Safe learning of regions of attraction for uncertain, nonlinear systems with gaussian processes. In: 2016 IEEE 55th Conference on Decision and Control (CDC). pp. 4661–4666.

Berkenkamp, Felix, Turchetta, Matteo, Schoellig, Angela P., Krause, Andreas, 2017. Safe model-based reinforcement learning with stability guarantees. In: NIPS.

Bharadhwaj, Homanga, Xie, Kevin, Shkurti, Florian, 2020. Model-predictive control via cross-entropy and gradient-based optimization. arXiv preprint arXiv:2004.08763.

Boer, Pieter-Tjerk De, Kroese, Dirk P., Mannor, Shie, Rubinstein, Reuven Y., 2005. A tutorial on the cross-entropy method. Ann. Oper. Res. 134 (1), 19–67.

Cabecinhas, David, Cunha, Rita, Silvestre, Carlos, 2014. A globally stabilizing path following controller for rotorcraft with wind disturbance rejection. IEEE Trans. Control Syst. Technol. 23 (2), 708–714.

Cao, Gang, Lai, Edmund M.-K, Alam, Fakhrul, 2017. Gaussian process model predictive control of an unmanned quadrotor. J. Intell. Robot. Syst. 88 (1), 147–162.

Chua, Kurtland, Calandra, Roberto, McAllister, Rowan, Levine, Sergey, 2018. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In: Advances in Neural Information Processing Systems. pp. 4754–4765.

Cole, Kenan, Wickenheiser, Adam M., 2018. Reactive trajectory generation for multiple vehicles in unknown environments with wind disturbances. IEEE Trans. Robot. 34 (5), 1333–1348.

Deits, Robin, Tedrake, Russ, 2015. Computing large convex regions of obstacle-free space through semidefinite programming. In: Algorithmic Foundations of Robotics XI. Springer, pp. 109–124.

Desaraju, Vishnu R., Michael, Nathan, 2016. Experience-driven predictive control. In: Robot Learning and Planning (RLP 2016). p. 29.

Finn, Chelsea, Levine, Sergey, 2017. Deep visual foresight for planning robot motion. In: 2017 IEEE International Conference on Robotics and Automation (ICRA). IEEE, pp. 2786–2793.

Fukushima, Hiroaki, Kim, Tae-Hyoung, Sugie, Toshiharu, 2007. Adaptive model predictive control for a class of constrained linear systems based on the comparison model. Automatica 43 (2), 301–308.

Hehn, Markus, D'Andrea, Raffaello, 2015. Real-time trajectory generation for quadrocopters. IEEE Trans. Robot. 31 (4), 877–892.

Hewing, Lukas, Liniger, Alexander, Zeilinger, Melanie N., 2018. Cautious NMPC with gaussian process dynamics for autonomous miniature race cars. In: 2018 European Control Conference (ECC). IEEE, pp. 1341–1348.

Hewing, Lukas, Wabersich, Kim P., Menner, Marcel, Zeilinger, Melanie N., 2020. Learning-based model predictive control: Toward safe learning in control. Annu. Rev. Control Robot. Auton. Syst. 3, 269–296.

Hirshberg, Tom, Vemprala, Sai, Kapoor, Ashish, 2020. Safety considerations in deep control policies with safety barrier certificates under uncertainty. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS). pp. 6245–6251.

Khalil, Hassan K., Grizzle, Jessy W., 2002. Nonlinear Systems, Vol. 3. Prentice Hall, Upper Saddle River, NJ.

Kober, Jens, Bagnell, J. Andrew, Peters, Jan, 2013. Reinforcement learning in robotics: A survey. Int. J. Robot. Res. 32 (11), 1238–1274.

Kobilarov, Marin, 2012. Cross-entropy randomized motion planning. In: Robotics: Science and Systems, Vol. 7. pp. 153–160.

Leung, Karen, Schmerling, Edward, Zhang, Mengxuan, Chen, Mo, Talbot, John, Christian Gerdes, J., Pavone, Marco, 2020. On infusing reachability-based safety assurance within planning frameworks for human–robot vehicle interactions. Int. J. Robot. Res. 39 (10–11), 1326–1345.

Luo, Wenhao, Sun, Wen, Kapoor, Ashish, 2020. Multi-robot collision avoidance under uncertainty with probabilistic safety barrier certificates. In: Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems.

Mayne, David Q., 2014. Model predictive control: Recent developments and future promise. Automatica 50 (12), 2967–2986.

Mayne, David, 2016. Robust and stochastic model predictive control: Are we going in the right direction? Annu. Rev. Control 41, 184–192.

Mehndiratta, Mohit, Kayacan, Erdal, 2020. Gaussian process-based learning control of aerial robots for precise visualization of geological outcrops. In: 2020 European Control Conference (ECC). IEEE, pp. 10–16.

Moorhouse, D., Woodcock, R., 1980. US Military Specification Mil–F–8785c. US Department of Defense.

Nagabandi, Anusha, Kahn, Gregory, Fearing, Ronald S., Levine, Sergey, 2018. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In: 2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE, pp. 7559–7566.

O'Donoghue, Brendan, Stathopoulos, Giorgos, Boyd, Stephen, 2013. A splitting method for optimal control. IEEE Trans. Control Syst. Technol. 21 (6), 2432–2442.

Ostafew, Chris J., Schoellig, Angela P., Barfoot, Timothy D., 2016. Robust constrained learning-based nmpc enabling reliable mobile robot path tracking. Int. J. Robot. Res. 35 (13), 1547–1563.

Pravitra, Jintasit, Ackerman, Kasey A., Cao, Chengyu, Hovakimyan, Naira, Theodorou, Evangelos A., 2020. L1-adaptive mppi architecture for robust and agile control of multirotors. arXiv preprint arXiv:2004.00152.

Rasmussen, Carl Edward, Nickisch, Hannes, 2010. Gaussian processes for machine learning (GPML) toolbox. J. Mach. Learn. Res. 11, 3011–3015.

Schaal, Stefan, Atkeson, Christopher G., 2010. Learning control in robotics. IEEE Robot. Autom. Mag. 17 (2), 20–29.

Schölkopf, Bernhard, Smola, Alexander J., Bach, Francis, et al., 2002. Learning with Kernels: Support Vector Machines, Regularization, Optimization, and beyond. MIT Press.

Shi, Guanya, Shi, Xichen, O'Connell, Michael, Yu, Rose, Azizzadenesheli, Kamyar, Anandkumar, Animashree, Yue, Yisong, Chung, Soon-Jo, 2019. Neural lander: Stable drone landing control using learned dynamics. In: 2019 International Conference on Robotics and Automation (ICRA). IEEE, pp. 9784–9790.

Siciliano, Bruno, Khatib, Oussama, 2016. Springer Handbook of Robotics. Springer.

Srinivas, Niranjan, Krause, Andreas, Kakade, Sham M., Seeger, Matthias, 2009. Gaussian process optimization in the bandit setting: No regret and experimental design. arXiv preprint arXiv:0912.3995.

Srinivas, Niranjan, Krause, Andreas, Kakade, Sham M., Seeger, Matthias W., 2012. Information-theoretic regret bounds for gaussian process optimization in the bandit setting. IEEE Trans. Inform. Theory 58 (5), 3250–3265.

Tan, Yew Teck, Kunapareddy, Abhinav, Kobilarov, Marin, 2018. Gaussian process adaptive sampling using the cross-entropy method for environmental sensing and monitoring. In: 2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE, pp. 6220–6227.

Urbina, Angel, Mahadevan, Sankaran, Paez, Thomas L., 2011. Quantification of margins and uncertainties of complex systems in the presence of aleatoric and epistemic uncertainty. Reliab. Eng. Syst. Saf. 96 (9), 1114–1125.

Wang, Li, Han, Dongkun, Egerstedt, Magnus, 2018. Permissive barrier certificates for safe stabilization using sum-of-squares. In: 2018 Annual American Control Conference, ACC. IEEE, pp. 585–590.

Wang, Yu, Yin, Wotao, Zeng, Jinshan, 2019. Global convergence of admm in nonconvex nonsmooth optimization. J. Sci. Comput. 78 (1), 29–63.

Williams, Grady, Aldrich, Andrew, Theodorou, Evangelos A., 2017. Model predictive path integral control: From theory to parallel computation. J. Guid. Control Dyn. 40 (2), 344–357.

Williams, Grady, Goldfain, Brian, Drews, Paul, Saigol, Kamil, Rehg, James M., Theodorou, Evangelos A., 2018. Robust sampling based model predictive control with sparse objective information. In: Robotics: Science and Systems.

Williams, Christopher K.I., Vivarelli, Francesco, 2000. Upper and lower bounds on the learning curve for Gaussian processes. Mach. Learn. 40, 77–102.

Wright, Stephen J., 1997. Primal–Dual Interior-Point Methods. SIAM.

Xu, Xiangru, Tabuada, Paulo, Grizzle, Jessy W., Ames, Aaron D., 2015. Robustness of control barrier functions for safety critical control. IFAC-PapersOnLine 48 (27), 54–61.

Zheng, Lei, Yang, Rui, Pan, Jiesen, Cheng, Hui, Hu, Haifeng, 2020. Learning-based safety-stability-driven control for safety-critical systems under model uncertainties. In: 2020 International Conference on Wireless Communications and Signal Processing (WCSP). IEEE, pp. 1112–1118.