

Learning-based Predictive Path Following Control for Nonlinear Systems Under Uncertain Disturbances

Rui Yang¹, Lei Zheng², Jiesen Pan¹, Hui Cheng¹

Abstract—Accurate path following is challenging for autonomous robots operating in uncertain environments. Adaptive and predictive control strategies are crucial for a nonlinear robotic system to achieve high-performance path following control. In this paper, we propose a novel learning-based predictive control scheme that couples a high-level model predictive path following controller (MPFC) with a low-level learning-based feedback linearization controller (LB-FBLC) for nonlinear systems under uncertain disturbances. The low-level LB-FBLC utilizes Gaussian Processes to learn the uncertain environmental disturbances online and tracks the reference state accurately with a probabilistic stability guarantee. Meanwhile, the high-level MPFC exploits the linearized system model augmented with a virtual linear path dynamics model to optimize the evolution of path reference targets, and provides the reference states and controls for the low-level LB-FBLC. Simulation results illustrate the effectiveness of the proposed control strategy on a quadrotor path following task under unknown wind disturbances.

Index Terms—Machine Learning for Robot Control, Control Architectures and Programming, Motion Control

I. INTRODUCTION

INCREASINGLY wide applications of autonomous mobile robots, such as in package delivery, electrical lines supervision and industrial inspection, require the controller to handle unpredictable and potentially adverse outdoor conditions and maintain high trajectory control performance. In particular, robots should be accurately steered along a predefined path in the presence of uncertain environmental disturbances. These non-negligible uncertainties are usually hard to model, such as the aerodynamic effects of flying vehicles. This makes it difficult for high-performance trajectory control using model-based controllers. Besides, it is also not realistic to tune the controller parameters manually for each specific operating condition. Desired trajectory controllers therefore should exhibit high control accuracy, adapt online to the uncertain environment, be robust to the uncertain disturbances, and take input constraints into account.

The trajectory control problem, defined as steering a robot to follow a predefined path, can be solved using trajectory

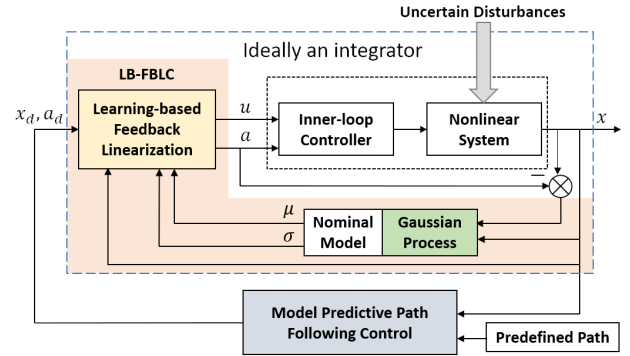


Fig. 1. Architecture diagram of the proposed strategy for nonlinear system path following under uncertain disturbances. The GPs are used to learn the disturbances online with uncertainty bounds. With the estimation and the uncertainty bound, the low-level LB-FBLC forces the nonlinear system to behave like an integrator and tracks the reference state $x_d = [x_{1d}, x_{2d}]^T$. The linear integrator is used as a predictive model in the high-level MPFC, which optimizes the reference target evolution along the path and provides reference states x_d and reference controls a_d for the LB-FBLC.

tracking or path following [1]. Compared with the trajectory tracking approach, where a timed parameterized reference is tracked, the path following approach removes the time dependence and regards it as a degree of freedom in the controller design. It steers the robot along a geometric path while prioritizing closeness to the desired reference with an attempt to satisfy a dynamic specification, such as a speed assignment along the path. This feature brings in some advantages to control performance and exhibits robustness to disturbances [1]. For example, if the robot loses track of the path under irresistible environmental disturbances, it will slow down to come back to the path rather than attempt to align with the time-dependent reference.

Typical path following approaches usually do this by separating the control scheme into an outer-loop guidance law for the generation of the reference motions along the path and then an inner-loop controller to track those motions [2]. Among these methods, geometric methods, such as Carrot-chasing [3] and nonlinear guidance law [4], are widely used in nonlinear mobile robots path following tasks [2]. The reference target is chosen based on geometric methods and can not be optimized according to the path following error. Model predictive path following control (MPFC) shapes the path following problem into an optimal control problem of a model predictive control (MPC) framework [5]. It uses the nonlinear system dynamics augmented with a virtual path dynamics model to optimize the evolution of the reference

Manuscript received: October, 15, 2020; Revised January, 12, 2021; Accepted February, 4, 2021.

This paper was recommended for publication by Editor Dana Kulic upon evaluation of the Associate Editor and Reviewers' comments.

¹R. Yang, J. Pan, H. Cheng are with the School of Computer Science and Engineering, Sun Yat-Sen University, Guangzhou, China. Corresponding author: H. Cheng chengh9@mail.sysu.edu.cn.

²L. Zheng is with the School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou, China.

Digital Object Identifier (DOI) 10.1109/LRA.2021.3062805.

target and reference controls. It can significantly improve the control performance by looking ahead to account for changes in the path [6], but the resulting MPC for the nonlinear system is subject to nonlinear models. The model error due to the uncertain disturbances may accumulate in multi-step prediction in MPC.

For feedback linearizable nonlinear systems [7], feedback or feedforward linearization techniques can be used to convert the nonlinear dynamics into a linear integrator model, which can be used as the predictive model in MPC. Exploiting differential flatness property, [8] combines the MPC with feedforward linearization control to tackle the trajectory tracking problem for a quadrotor. The MPFC technique combined with nonlinear dynamic inversion acceleration controller is designed for multirotor path following in [9]. However, the performance of these model-based methods is still limited by the discrepancy between the nominal model and the actual system in the presence of unknown environmental disturbances.

To solve this problem, the uncertain disturbances should be estimated and compensated in controller design. In [10], external wind gusts and ground effects of the quadrotor are estimated using the Kalman filter and compensated in the predictive model of MPC. Based on [9], model reference adaptive control is introduced to the acceleration controller in [11], but the adaptive scheme is limited to the rotational dynamics for a quadrotor.

One alternative solution is to use data-driven machine learning methods to learn environmental disturbances and use the learned model to design the controller. In [12], a deep neural network with spectral normalization is designed to learn the aerodynamic effects of a quadrotor in a feedback linearization controller to improve the quadrotor position control. Since only limited data can be obtained to approximate the uncertain disturbances, Gaussian Processes (GPs) [13] can be utilized to estimate the uncertain disturbances and capture both the epistemic uncertainty of the estimation due to the lack of data and the aleatoric uncertainty inherent in the environment [14]. In [15] [16], the estimation using GPs and the corresponding confidence bounds are used to design a robust controller for Euler-Lagrange systems. While these proposed control strategies provide tracking control stability, control constraints are not considered in the controller. In [17], GPs are used in a feedback linearization controller to compensate the effect of wind disturbances. The controller reduces the trajectory tracking error for the quadrotor but without providing a control stability guarantee.

To eliminate the effect of environmental disturbances and maintain the predictive capability of MPC, a nonlinear MPC is designed using GPs to update the model error of the nominal model for multi-step predictions [18]. In [19], an \mathcal{L}_1 adaptive controller is designed to force the system to behave in a predefined linear model. A higher-level linear MPC uses this reference model to calculate the optimal reference input for the \mathcal{L}_1 controller for trajectory tracking problem.

The limitations of the mentioned path following approaches and the advantages of learning-based control techniques show an urgent need for designing an adaptive and predictive path following control strategy that achieves accurate path follow-

ing for nonlinear robotic systems under uncertain disturbances.

In this paper, we propose a novel learning-based predictive path following control strategy, as illustrated in Fig. 1, for nonlinear systems to accurately follow the predefined paths under environmental disturbances. The control strategy couples an MPFC with an LB-FBLC which uses GPs to learn the environmental disturbances model online. The LB-FBLC forces the nonlinear system under disturbances to behave like an integrator and tracks the reference states. The high-level MPFC with the linear integrator model optimizes the evolution of the reference targets, and provides reference states and controls for the low-level LB-FBLC.

The main contributions of the paper include:

- A novel online LB-FBLC combining first-principle and data-driven design methods is derived to enable accurate tracking control under uncertain environmental disturbances. The theoretical analysis of probabilistic stability is provided.
- A novel adaptive predictive path following control scheme coupling a high-level MPFC with the low-level LB-FBLC is designed to tackle path-following problems for the nonlinear system under unknown environmental disturbances.
- The effectiveness of the proposed control scheme is validated on the quadrotor path following task under uncertain wind disturbances via simulation.

The remainder of this paper is organized as follows. Problem statement and necessary preliminary are presented in Section II. The proposed methodology is introduced in Section III. In Section IV, the effectiveness of the proposed method is validated via simulations on a quadrotor. The conclusion is reached in Section V.

II. PROBLEM STATEMENT AND PRELIMINARY

A. Problem Statement

Consider a nonlinear system with dynamics:

$$\dot{x}_1 = x_2, \quad \dot{x}_2 = f(x) + G(x)u, \quad (1)$$

where the state $x = [x_1, x_2]^T \in X \subset \mathbb{R}^{2n}$, $x_1, x_2 \in \mathbb{R}^n$, the state space X is compact, and the controls $u \in U \subset \mathbb{R}^n$. A wide range of robots such as quadrotor and car-like vehicles can be transformed into this form. It is noted that the analysis is restricted to this form, but the results can be extended to the systems of higher relative degree [7]. In general, f and G may not be fully known for real systems. We make the following assumption:

Assumption 1: The function $f : X \rightarrow \mathbb{R}^n$ is unknown but has a bounded reproducing kernel Hilbert space (RKHS) norm under a known kernel k . The function $G : X \rightarrow \mathbb{R}^{n \times n}$ is known and differentiable, satisfying $\text{Rank}(G(x)) = n$, $\forall x \in X$.

The assumption of f limits the irregularity under the induced norm of RKHS, which can be used as a measure for smoothness [13]. It is common for practical systems, such as quadrotors and robotic manipulators, that the assumption of G holds such that $G(x)$ is invertible for $x \in X$.

The goal of this work is to design a novel, learning-based path following control strategy for nonlinear system (1), that satisfies the following desired objectives:

- (R1) *High-Accuracy Following*: For a given geometric path, the nonlinear system (1) moves forward along the path and achieves a high-accuracy path following performance.
- (R2) *Adaptability*: The proposed strategy can leverage online learning to estimate and adapt to the uncertain environmental disturbances.
- (R3) *Robustness to Disturbances*: The proposed strategy can optimize the reference target along the path and drive the nonlinear system back to the reference path after irresistible disturbances.

B. Gaussian Process Regression

A Gaussian process (GP) [13] is a stochastic process that can be used as a nonparametric regression model to approximate a nonlinear dynamical function, $\delta_i : X \rightarrow \mathbb{R}$, with a fidelity estimation. The GP assumes that function values, associated with different inputs, are random variables and any finite number of them have a joint Gaussian distribution. The approximation of δ_i can be denoted by

$$\bar{\delta}_i(x) \sim \mathcal{N}(\mu_i(x), k_i(x, x')), \quad (2)$$

which is fully specified by a mean function $\mu_i(x) : X \rightarrow \mathbb{R}$ and a kernel $k_i(x, x') : X \times X \rightarrow \mathbb{R}$ estimating the similarity between states x and x' . It is common practice to set the prior mean function to zero and use squared-exponential kernel

$$k_i(x, x') = \sigma_f^2 \exp\left(-\frac{1}{2}(x - x')^T L^{-2}(x - x')\right), \quad (3)$$

which is characterized by hyperparameters of the length scale diagonal matrix L and the prior variance σ_f^2 . Since (2) represents only functions with a scalar output, n independent GPs can be utilized to model the nonlinear function $\delta : X \rightarrow \mathbb{R}^n$,

$$\bar{\delta}(x) = \begin{cases} \bar{\delta}_1(x) \sim \mathcal{N}(\mu_1(x), k_1(x, x')) \\ \dots \\ \bar{\delta}_n(x) \sim \mathcal{N}(\mu_n(x), k_n(x, x')) \end{cases}. \quad (4)$$

We assume that the training set D is available to employ the GPs for regression.

Assumption 2: The state x and the function value $\delta(x)$ can be measured with noises over a finite time horizon to make up a training set with N data pairs

$$D = \left\{ \left(x^{(i)}, y^{(i)} \right) \right\}_{i=1}^N, \quad y^{(i)} = \delta \left(x^{(i)} \right) + w_i, \quad (5)$$

where w_i are i.i.d. noises $w_i \sim N(0, \sigma_{noise}^2 I_N)$, $\sigma_{noise} \in \mathbb{R}$.

Given this training dataset D , GPs can be used to predict function value y^* at any query input x^* . The j -th component of the inferred output y^* is jointly Gaussian distributed with the training set

$$\begin{bmatrix} y_j^* \\ y_j \end{bmatrix} \sim \mathcal{N} \left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} k_j^* & k_j^T \\ k_j & K_j + \sigma_{noise}^2 I_N \end{bmatrix} \right), \quad (6)$$

where $k_j^* = k_j(x^*, x^*) \in \mathbb{R}$, $y_j = [y_j^{(1)}, \dots, y_j^{(N)}]^T \in \mathbb{R}^N$, $k_j = [k_j(x^{(1)}, x^*), \dots, k_j(x^{(N)}, x^*)]^T \in \mathbb{R}^N$, and $K_j \in \mathbb{R}^{N \times N}$ is covariance matrix with the entry $[K_j]_{(l,m)} = k_j(x^{(l)}, x^{(m)})$, $l, m = 1, \dots, N$.

Conditioning on the test input x^* and the training data D , the distribution on $\bar{\delta}(x^*)$ is $\bar{\delta}(x^*) \sim \mathcal{N}(\mu(x^*), \sigma^2(x^*))$ with the j -th component of the mean and variance:

$$\begin{aligned} \mu_j(x^*) &= k_j^T (K_j + \sigma_{noise}^2 I_N)^{-1} y_j, \\ \sigma_j^2(x^*) &= k_j^* - k_j^T (K_j + \sigma_{noise}^2 I_N)^{-1} k_j. \end{aligned} \quad (7)$$

With the function $\bar{\delta}$ approximated by GPs, the following result allows us to quantify the upper bound of the difference between the true function $\delta(x)$ and the inferred mean $\mu(x)$ with a reliable confidence interval.

Lemma 1 ([20]): For any compact set $X \subset \mathbb{R}^{2n}$ and a probability $\varsigma \in (0, 1)$ holds

$$Pr\{\|\mu(x) - \delta(x)\| \leq \|\beta\| \|\sigma(x)\|, \forall x \in X\} \geq (1 - \varsigma)^{2n}, \quad (8)$$

where Pr denotes probability, $\beta = [\beta_1, \dots, \beta_n]^T$, $\beta_j = (2\|\delta_j\|_{k_j}^2 + 300\gamma_j \ln^3(\frac{N+1}{\delta}))^{\frac{1}{2}}$, $j = 1, \dots, n$, γ_j is the maximum information gain under the kernel k_j : $\gamma_j = \max_{\{x^{(1)}, \dots, x^{(N)}\} \in X} \frac{1}{2} \log(\det(I_N - \sigma_{noise}^{-2} K_j(x, x')))$, $x, x' \in \{x^{(1)}, \dots, x^{(N)}\}$.

Lemma 1 allows us to make high probability statements on the maximum modeling error between the true function δ and the inferred mean μ , and it will be utilized in the analysis and synthesis of the proposed control scheme.

III. METHODOLOGY

To achieve the three goals in II-A, we propose a hierarchical control scheme composed of a high-level MPFC coupled with an underlying LB-FBLC, as shown in Fig. 1. We first introduce the LB-FBLC in Section III-A and the high-level MPFC in Section III-B.

A. Learning-based Feedback Linearization Controller (LB-FBLC)

Suppose we are given a bounded reference state $x_d(t) = [x_{1d}(t), x_{2d}(t)]^T \in X$ and a control $a_d(t) \in \mathcal{A} \subset \mathbb{R}^n$ from the high-level MPFC. Since the MPFC uses an integrator as the predictive model, these references satisfy

$$\dot{x}_{1d}(t) = x_{2d}(t), \quad \dot{x}_{2d}(t) = a_d(t). \quad (9)$$

For simplicity, we omit the time index in the following.

A common method to track the reference state for nonlinear systems (1) is feedback linearization control. Since $f(x)$ is not known exactly, let $\hat{f}(x)$ be a nominal model of $f(x)$. We formulate the feedback linearization control law u , with pseudo-control component a ,

$$u = G(x)^{-1} (a - \hat{f}(x)) \quad (10)$$

to convert the nonlinear system (1) into an approximately linear integrator model

$$\dot{x}_1 = x_2, \quad \dot{x}_2 = a + \delta(x), \quad (11)$$

where $\delta(x) = f(x) - \hat{f}(x) \in \mathbb{R}^n$ is the modeling error resulting from environmental disturbances. If the nominal model matches the actual model, then $\delta(x) = 0$ and (11) becomes a double integrator. However, it is difficult to get an accurate model in advance for practical robotic systems under uncertain environmental disturbances.

To achieve precise tracking control given reference state x_d and control a_d , we design the pseudo-control a as

$$a = a_d + K_P(x_{1d} - x_1) + K_D(x_{2d} - x_2) + r, \quad (12)$$

where $K_P \in \mathbb{R}^{n \times n}$, $K_D \in \mathbb{R}^{n \times n}$ are the proportional and derivative matrices of PD control law [7], respectively, and $r \in \mathbb{R}^n$ is an added vector to be designed to compensate the disturbances and achieve tracking stability.

Define the tracking error $e = x - x_d$. Then, it can be shown from (11) and (12) that the tracking error dynamics can be written as

$$\dot{e} = Ae + B(r + \delta(x)), \quad (13)$$

where $A = \begin{bmatrix} 0 & I_n \\ -K_P & -K_D \end{bmatrix} \in \mathbb{R}^{2n \times 2n}$, and $B = \begin{bmatrix} 0 \\ I_n \end{bmatrix} \in \mathbb{R}^{2n \times n}$. The control gain matrix K_P , K_D should be chosen to make A a Hurwitz matrix [7]. Let $P \in \mathbb{R}^{2n \times 2n}$ be the unique positive definite matrix, satisfying $A^T P + PA = -Q$, where $Q \in \mathbb{R}^{2n \times 2n}$ is a positive definite matrix.

We now discuss how to design the adaptive control vector r . Intuitively, the environmental disturbances can be totally compensated when they are known exactly, i.e. $r = -\delta(x)$. However, only an approximation $\bar{\delta}$ of $\delta(x)$ can be obtained from limited data. Thus, the approximation methods should provide the estimation of $\delta(x)$ and capture the uncertainty of the estimation. To this end, GPs are utilized to predict the environmental disturbances and quantify the uncertainty based on its predictions, where the output of the GPs is $\bar{\delta}(x) \sim \mathcal{N}(\mu(x), \sigma(x))$ as illustrated in Section II. To train the GPs, N data points can be collected online to makeup the training dataset $D = \{(x^{(i)}, (\hat{x}_2 - a)^{(i)})\}_{i=1}^N$.

With the estimated disturbance $\bar{\delta}(x)$, the adaptive control vector r can be designed as

$$r = -\mu(x) - k_c B^T P e, \quad (14)$$

where $k_c \in \mathbb{R}$ is an adjustable control parameter.

Lemma 2: Consider the system (1) with a bounded desired state x_d . Suppose that **Assumptions 1** and **Assumptions 2** hold. Then, the proposed learning-based control strategy in (10), (12) and (14) with the condition

$$k_c \|B^T P e(x)\| - \|\beta\| \|\sigma(x)\| \geq 0, \forall x \in X, \quad (15)$$

ensures that the tracking error e semi-globally asymptotically converges to zero with probability at least $(1 - \varsigma)^n$ for $e \in \mathcal{E}$, where \mathcal{E} is a compact set $\mathcal{E} = \{e \in \mathbb{R}^{2n} | e^T P e \leq e(0)^T P e(0)\}$.

Proof: Consider a candidate Lyapunov function $V(e) = e^T P e$. Denote $w = w^T = B^T P e$. It can be shown from (13) that $\dot{V}(e) = -e^T Q e + 2w(r + \delta)$. With control law (14), we have

$$\begin{aligned} \dot{V}(e) &= -e^T Q e + 2w(\delta - \mu - k_c w) \\ &\leq -e^T Q e + 2w(\delta - \mu) - 2k_c \|w\|^2 \\ &\leq -e^T Q e + 2\|w\| \|\delta - \mu\| - 2k_c \|w\|^2, \end{aligned} \quad (16)$$

where the inequality comes from the Cauchy-Schwarz inequality. Employing **Lemma 1**, we have $Pr\{\dot{V}(e) \leq -e^T Q e + 2\|w\|(\|\beta\| \|\sigma(x)\| - k_c \|w\|), \forall e \in \mathcal{E}\} \geq (1 - \varsigma)^n$. It yields $Pr\{\dot{V}(e) < 0, \forall e \in \mathcal{E} \setminus \{0\}\} \geq (1 - \varsigma)^n$ under the condition (15). This strict inequality holds because $-e^T Q e < 0$ with Q

a positive definite matrix. In addition, $\dot{V}(0) = 0$ holds. ■

Nonlinear system (1) is forced to behave like an integrator (9) leveraging the control law (14). It is noted that if the variance $\sigma(x) = 0$, i.e. the prediction of disturbances $\bar{\delta}(x)$ via GPs matches the true value $\delta(x)$ ideally, the tracking error e will asymptotically converge to zero and the system (1) can be transformed exactly into an integrator.

Additionally, considering the control constraints, we can leverage the control Lyapunov function to construct a quadratic programming (QP) to obtain the parameter k_c :

$$\begin{aligned} k_c^* &= \arg \min_{k_c} \|k_c w\|_2^2 + k_c \epsilon^2, \\ \text{s.t. } & H_{clf} k_c + b_{clf} \leq \epsilon, \quad (\text{Stability Constraints}) \\ & H_u k_c + b_u \leq 0, \quad (\text{Control Constraints}) \end{aligned} \quad (17)$$

where $H_{clf} = -2\|w\|^2$, $b_{clf} = -e^T Q e + 2\|w\| \|\beta\| \|\sigma(x)\|$, $H_u = [G(x)^{-1} w, -G(x)^{-1} w]^T$, $b_u = [G(x)^{-1}(\mu - a_d - a_{pd} + \hat{f}(x)) + u_{min}, G(x)^{-1}(-\mu + a_d + a_{pd} - \hat{f}(x)) - u_{max}]^T$, $a_{pd} = K_P(x_{1d} - x_1) + K_D(x_{2d} - x_2)$, $\epsilon \in \mathcal{R}$ is a slack variable to ensure the QP is feasible.

Remark. Note that the optimization (17) is not sensitive to the k_c parameter as long as it is large enough (e.g. 10^{20}), such that stability constraints violation is heavily penalized.

B. Model Predictive Path Following Control (MPFC)

With the integrator (9) as the predictive model, an MPFC is designed to optimize the reference target along the path, and provide reference state x_d and reference control a_d for the LB-FBLC. As a high-level controller, it accounts for disturbances by optimizing the speed of reference target evolution along the path in a framework of nonlinear model predictive control (NMPC).

Given a geometric path \mathcal{P} :

$$\mathcal{P} = \{x_{ref} \in X \subset \mathbb{R}^{2n} | x_{ref} = P(\theta), \theta \in \Theta\}, \quad (18)$$

which is described by a projection $P : \Theta \rightarrow X$, from a parameter interval $\Theta = [\theta_0, \theta_{end}] \subset \mathbb{R}$ to the state space X . This parameter can be regarded as an additional degree of freedom to be optimized in the NMPC. However, changing $\theta(t)$ directly can result in undesired jumps along the path. To avoid this effect, a virtual dynamics model of reference target is designed as another integrator:

$$\dot{\theta} = \theta_{vel}, \quad \dot{\theta}_{vel} = \theta_{acc}, \quad (19)$$

in which $\theta_{vel} > 0$, $\theta_{acc} \in \mathcal{A}_\Theta \subset \mathbb{R}$ is the control input to the virtual system (19) and \mathcal{A}_Θ is a compact set containing the origin in its interior. This control input θ_{acc} can be regarded as the acceleration of the reference target evolution. The relative degree of virtual system is designed corresponding to the linearized system (11) and can provide smooth evolution of θ by optimizing its acceleration. [5] gives a differential algebraic and geometric explanation of this kind of virtual reference target dynamics.

Based on the linearized system dynamics (9) and the linear virtual target dynamics (19), a continuous time sampled-data NMPC scheme is constructed. As commonly in the MPC, the system input is obtained by repetitively solving a finite-time optimal control problem (OCP). Denote the sampling instance

$t_k = t_0 + k \cdot dt$, where $k \in \mathbb{N}$, t_0 is the initial time instant, and dt is the control period. Specifically, at each sampling instance t_k , the following OCP is solved:

$$\min_{\bar{a}(t), \bar{\theta}_{acc}(t)} J(x(t_k), \theta(t_k), \bar{a}(t), \bar{\theta}_{acc}(t)), \quad (20)$$

$$\text{s.t. } \dot{\bar{x}}_1(t) = \bar{x}_2(t), \quad \dot{\bar{x}}_2(t) = \bar{a}(t), \quad (21)$$

$$\dot{\bar{\theta}}(t) = \bar{\theta}_{vel}(t), \quad \dot{\bar{\theta}}_{vel}(t) = \bar{\theta}_{acc}(t), \quad (22)$$

$$\bar{\theta}_{vel}(t) > 0, \quad (23)$$

$$\bar{x}(t_k) = x(t_k), \quad \bar{\theta}(t_k) = \theta(t_k), \quad (24)$$

$$\bar{x}(t) \in X, \quad \bar{a}(t) \in \mathcal{A}, \quad (25)$$

$$\bar{\theta}(t) \in \Theta, \quad \bar{\theta}_{acc}(t) \in \mathcal{A}_\Theta, \quad (26)$$

where the superscript $\bar{\cdot}$ denotes the predicted state or control variables.

The objective function J at t_k can be designed as follows.

$$J(x(t_k), \theta(t_k), \bar{a}(t), \bar{\theta}_{acc}(t)) = \int_{t_k}^{t_k + H \cdot dt} \|\bar{x}(t) - P(\bar{\theta}(t))\|_{Q_{mpc}}^2 + \|\bar{a}(t)\|_{R_a}^2 + R_\theta \bar{\theta}_{acc}(t)^2 dt, \quad (27)$$

where H is the predictive steps, positive semi-definite matrix $Q_{mpc} \in \mathbb{R}^{2n \times 2n}$ weights the tracking error between the system states and reference targets, and positive definite matrix $R_a \in \mathbb{R}^{n \times n}$ and $R_\theta > 0$ ensure regularization of the inputs. Notice that constraint (23) guarantees forward moving of reference target along the path.

The solution to the OCP (20)-(26) contains reference target $P(\bar{\theta}^*(t))$, state $\bar{x}^*(t)$ and control input $\bar{a}^*(t)$, $\forall t \in [t_k, t_k + dt]$. The reference and control are applied to the LB-FBLC as $x_d(t)$ and $a_d(t)$. At the next sampling instant $t_{k+1} = t_k + dt_{mpc}$, the OCP (20)-(26) will be solved again with new measured states served as an initial condition. As discussed in [5], the path convergence and recursive feasibility in the presence of system constraints can be ensured by carefully adding an end penalty and a terminal constraint to the OCP (20)-(26).

IV. APPLICATION TO QUADROTOR AND SIMULATION RESULTS

In this section, the proposed approach is applied to a quadrotor to follow different predefined paths in the presence of unknown wind disturbances. Predictive following performance and adaptability to the uncertain disturbances of the proposed framework are verified via simulation.

A. Quadrotor Dynamics and Control

The quadrotor is a well-modeled dynamic 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. Thus, the attitude rotation matrix $R \in SO(3)$ from the body frame B to the world frame W can be written as

$$R = \begin{bmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\phi c\psi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix}, \quad (28)$$

where s and c represents \sin and \cos , respectively [21].

Given quadrotor states as position $p \in \mathbb{R}^3$ and velocity $v \in \mathbb{R}^3$ in the world frame W , we consider the following translational dynamics:

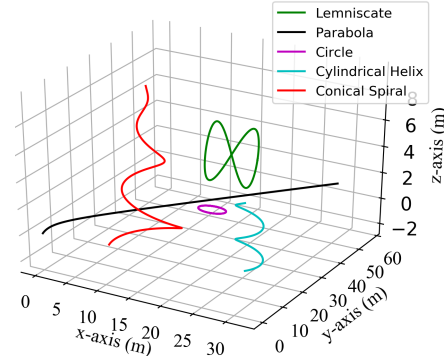


Fig. 2. The five different paths used to test the proposed path following control scheme.

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

$$\dot{v} = -ge_3 + \frac{1}{m}Rf_u + \frac{1}{m}f_a, \quad (29)$$

where m is the mass of the quadrotor, $e_3 = [0, 0, 1]^T$ is the unit vector, g is the gravitational acceleration, and $f_u = [0, 0, f_T]^T$ with f_T the total thrust generated from four rotors. The wind disturbance is $f_a = K_{drag}(v_w - v)$, where $v_w \in \mathbb{R}^3$ is the velocity of wind disturbances in W and $K_{drag} \in \mathbb{R}^{3 \times 3}$ is a drag coefficient diagonal matrix. We define in the dynamics equation (1) the state $x = [x_1, x_2]^T = [p, v]^T$, with $x_1 = p = [p_x, p_y, p_z]^T \in \mathbb{R}^3$ and $x_2 = v = [v_x, v_y, v_z]^T \in \mathbb{R}^3$, and define the total desired rotor force as control $u = (Rf_u)_d \in \mathbb{R}^3$ following [12].

We assume that the attitude controller is provided by a commercial quadrotor with the control interface (ϕ_{cmd} , θ_{cmd} , ψ_{cmd} , f_{Tcmd}). Given geometric paths as described in IV-B, reference states x_d and controls a_d are provided by the MPFC. With these references, the control $u = [u_x, u_y, u_z]^T$ as well as $a = [a_x, a_y, a_z]^T$ are computed using (10), (12) and (14) with k_c computed by solving (17). As shown in Fig. 1, an inner-loop controller leveraging the differential flatness property of the quadrotor [22] converts the computed controls to the attitude and thrust commands. Specifically, these commands can be computed as follows [23].

$$f_{Tcmd} = \|u\|, \quad \theta_{cmd} = \text{atan2}(\beta_a, \beta_b), \quad (30)$$

$$\phi_{cmd} = \text{atan2}\left(\beta_c, \sqrt{\beta_a^2 + \beta_b^2}\right),$$

where $\beta_a = -a_x \cos \psi_{cmd} - a_y \sin \psi_{cmd}$, $\beta_b = -a_z + g$, and $\beta_c = -a_x \sin \psi_{cmd} + a_y \cos \psi_{cmd}$. The command of yaw is set as $\psi_{cmd} = 0$.

B. Simulation Setup

We create a simulation platform using Python 3.7 to numerically validate the performance of the proposed control methodology for quadrotor following paths under unknown wind disturbances. The quadrotor model corresponds to the commercial quadrotor Crazyflie 2.1 with mass $m = 0.036 \text{ kg}$. The control constraints of u are set $u_{min} = m \cdot [-2, -2, -2 + g]^T (N)$ and $u_{max} = m \cdot [2, 2, 2 + g]^T (N)$ [24]. The drag coefficient is set $K_{drag} = \text{diag}[0.02, 0.02, 0.02]$. The proposed path following controller runs at 100 Hz with a control period $dt = 0.01 \text{ s}$. The internal prediction steps of the MPFC is $H = 20$. The weights in the objective function (27) of NMPC are set as

TABLE I
AVERAGED PATH FOLLOWING RMSES (IN METER) OVER THE FIVE PATHS
FOR DIFFERENT CONTROL SCHEMES

High-level	Low-level	Without Disturbances	Uncertain Disturbances
MPFC	FBLC	0.0190	0.0346
MPFC	LB-FBLC	0.0144	0.0162
MPFC	FFLC	0.0353	0.0376
MPFC	LB-FFLC	0.0380	0.1102
MPFC	ROBUST [15]	0.0168	0.0368
Carrot-chasing	LB-FBLC	0.1580	0.1653
NLGL	LB-FBLC	0.1692	0.1785

TABLE II
MAXIMUM FOLLOWING ERRORS (IN METER) OF A QUADROTOR
FOLLOWING 5 PATHS WITH DIFFERENT HIGH-LEVEL CONTROLLERS

High-level	Lemniscate	Parabola	Circle	CH	CS
MPFC	0.5667	0.2221	0.1384	0.8190	0.4397
Carrot-chasing	0.6050	0.3299	0.3637	1.4188	0.5945
NLGL	0.6020	0.3290	0.3868	1.3200	0.6638

$Q_{mpc} = \text{diag}[10, 10, 10, 1, 1, 1]$, $R_a = \text{diag}[0.1, 0.1, 0.1]$ and $R_\theta = 0.1$.

The control gain matrix in PD control is $K_p = \text{diag}[2, 2, 2]$ and $K_d = \text{diag}[1, 1, 1]$. The Q matrix in the Lyapunov function is set as $Q = \text{diag}[1, 1, 1, 1, 1, 1]$. The penalty coefficients of the slack variable in the QP are set $k_\epsilon = 1e20$. The QP is solved with the OSQP solver [25]. We use the scikit-learn Python package [26] to build 3 GPs to estimate unknown wind disturbances f_a . Each GP uses the same squared-exponential kernel with parameters $L = 10$ and $\sigma_f = 1$. The GPs update with the past $N = 5$ data pairs as the training set for prediction. We set $\beta = 3$ in (8) for each GP. The computing time for GP learning is $0.1s$ and for inference is below $0.003s$ on average on a 2.10GHz Intel Xeon CPU. Solving the NMPC in the simulation takes $0.07s$ on average using the CasADi [27] with the IPOPT solver [28] in Python. These codes have not been optimized for speed and can be much accelerated in C++.

A set of five paths, i.e., lemniscate, parabola, circle, cylindrical helix (CH) and conical spiral (CS), are used to test the path following performance, as shown in Fig. 2. These paths are parameterized by $\theta \in [0, 20]$ with nominal velocity profiles by setting $\theta_{vel}(t) = 1s^{-1}$. We start the reference target and initial position of the quadrotor on the path at $P(\theta = 0)$. To quantify the following performance, we test for $T = 20s$ and compare the root mean square error (RMSE) of the minimum distance from the quadrotor position to the path:

$$\mathcal{E} = \sqrt{\frac{1}{M} \sum_{k=1}^M \|d_{min}\|^2}, \quad (31)$$

where the minimum distance is calculated as $d_{min} = \min_\theta (x_1(t_k) - P(\theta))$, and $M = T/dt = 2000$ represents the total control steps in the discrete control.

To evaluate the algorithm performance under unknown disturbances, a wind model in [29] is utilized, consisting of a constant component v_c , a turbulent component v_t and a wind gust component v_g , i.e., wind velocity $v_w = v_c + v_t + v_g$. The horizontal constant wind v_c is randomly set from $3m/s$ to $10m/s$. The turbulence wind uses the von Kármán velocity model defined analytically in the specification MIL-F-8785C [30], with the low-altitude model in the specification for the

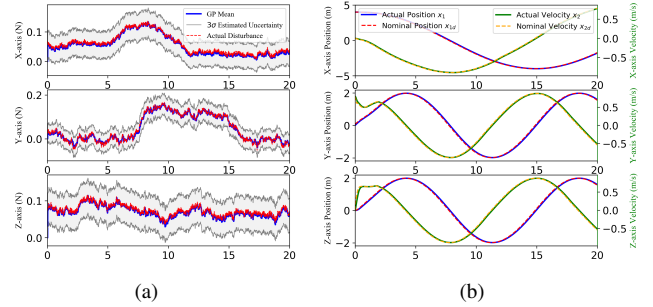


Fig. 3. (a) The estimated disturbances in three axes with GPs when following the lemniscate path. (b) The nominal trajectory followed by the ideal system (9) is compared to the actual trajectory followed by the quadrotor.

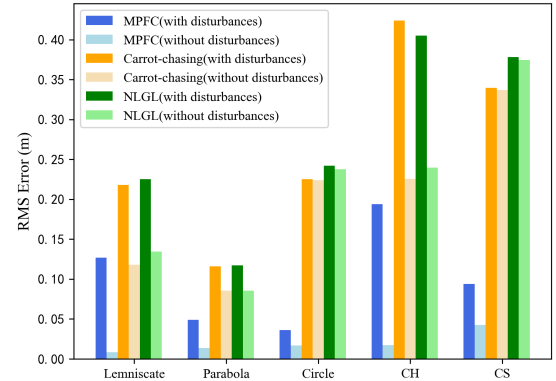


Fig. 4. The RMSEs of the quadrotor following five different paths using different high-level controllers without disturbances and under the same wind disturbances.

model parameters. The turbulent wind component v_t can be generated following [31] and [32] with the model. The gusting profile v_g is defined as a $1 - \cos$ model in the specification. We randomly generate 10 sets of parameters of the wind model to evaluate the algorithm performance.

C. Results

1) *Adaptability*: In this subsection, we verify that (i) the actual nonlinear system remains close to the ideal integrator model, and (ii) the adaptive performance benefits from the designed LB-FBLC.

White noises are added to the acceleration measurements to simulate the real sensor condition. The noise level on the measurements is considered in the GPs (5)-(7), corresponding to the noise level parameter α in the scikit-learn package of GPs. Table III shows the following errors for the lemniscate path. When the parameter α in the GPs are properly chosen, the proposed method is robust to the measurement noises unless the measurement noises increase beyond a certain range. In the following numerical validation, white noise $w_i \sim N(0, 0.005)$ and $\alpha = 5e-4$ are set.

To validate (i), we compare the actual trajectory of the quadrotor following the lemniscate path with the reference trajectory generated by MPFC, as shown in Fig. 3(b). The reference states satisfy an ideal integrator (9). The actual system under unknown wind disturbances does behave close to the ideal integrator using the LB-FBLC. Fig. 3(a) shows the estimations of wind disturbances f_a on quadrotor using GPs. It can be seen that the actual disturbances lie in the uncertainty bound of the estimations.

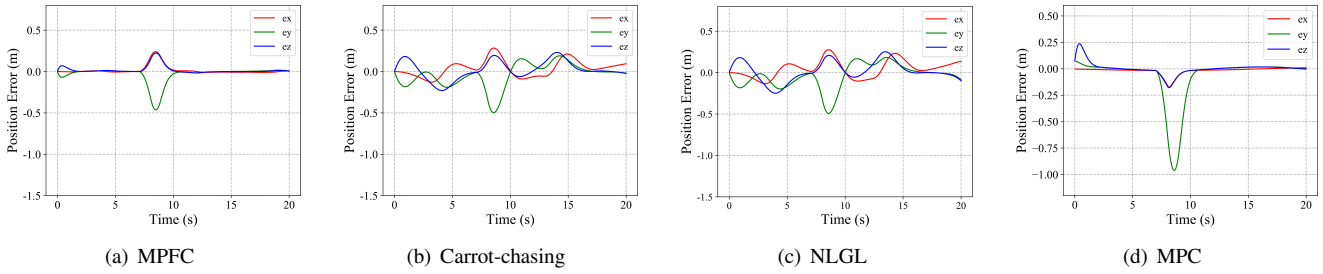


Fig. 5. Position error of quadrotor following the aggressive lemniscate path under wind disturbances. Note that the deviation at the beginning is caused by wind disturbances, which has not been estimated and compensated by GPs with limited data. The jumps at around 7s are caused by an irresistible wind gust. The proposed control scheme with MPFC as the high-level controller outperforms the other three baseline controllers in terms of position error and overshooting.

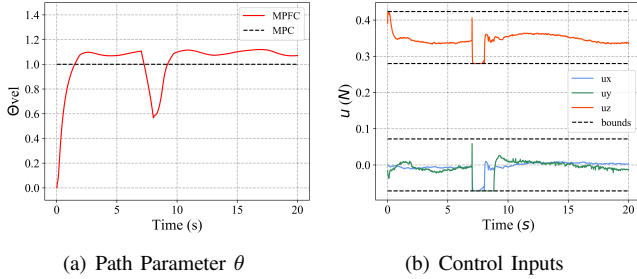


Fig. 6. (a) The evolution of parameter θ in the aggressive lemniscate path. (b) Control constraints are satisfied when following the lemniscate path with the proposed method.

TABLE III

THE RMSES (IN METER) WITH DIFFERENT LEVELS OF MEASUREMENT NOISES AND NOISE SETTINGS α IN GPs.

Measurement noise	$\alpha = 0.0005$	0.005	0.05	0.5
0.005	0.00852	0.01346	0.01346	0.01345
0.05	0.01226	0.01362	0.01498	0.01518
0.1	0.01645	0.01423	0.01500	0.01518
0.5	0.02847	0.28544	0.38680	0.01518

We keep the high-level MPFC the same to fairly compare the designed low-level LB-FBLC with five controllers: a) a nominal feedback linearization controller (FBLC) with no GPs; b) a nominal feedforward linearization controller (FFLC) with no GPs; c) a learning-based feedforward linearization controller using GPs (LB-FFLC), which removes the PD feedback control from the LB-FBLC; d) the robust control method proposed in [15] (ROBUST). For the ROBUST method, we selected the parameter $\epsilon = 0.1$. The Lyapunov function and PD control gain matrix in the ROBUST method are set the same as those in our method. The settings on GPs among the learning-based methods are the same for fairness of comparison.

The path following RMSEs shown in Table.I are averaged over five paths with or without disturbances. The proposed low-level LB-FBLC achieves the minimum RMSE in all cases. Both the LB-FBLC and LB-FFLC use GPs to learn the uncertain disturbances and achieve lower tracking error than the nominal FBLC and FFLC methods under disturbances. It shows that the proposed approach benefits from using GPs to compensate the unknown disturbances. Besides, the LB-FBLC/FBLC achieves a smaller RMSE compared with the LB-FFLC/FFLC. It demonstrates that the feedback PD control design in the proposed approach is effective. Table.I also shows that the proposed approach can achieve lower RMSE compared to the ROBUST method. The results show that the designed adaptive control law, with the form and the condition of stability different from the ROBUST method, can

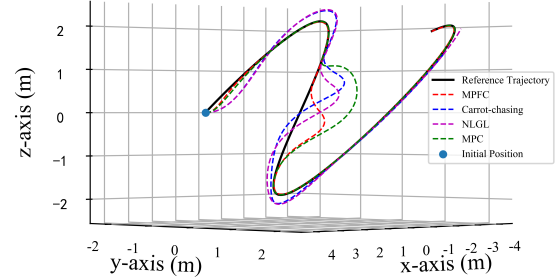


Fig. 7. Trajectories of the quadrotor following the aggressive lemniscate path. The proposed control scheme with MPFC as the high-level controller can achieve lower following error due to the predictive control of the system and reference evolution. Best viewed in color.

effectively reduce the path following error under the unknown wind disturbances.

2) *Predictivity*: The benefits of the predictive control inherent in the designed MPFC are demonstrated in this part. For a fair comparison, we keep the LB-FBLC as the low-level controller and compare the high-level MPFC against two baseline path following control algorithms: a) Carrot-chasing, b) Nonlinear Guidance Law (NLGL), and a trajectory control algorithm: c) MPC. In the Carrot-chasing method, the reference target is chosen at a constant distance D_1 ahead from the path point which is computed by projecting the quadrotor position to the path. In the NLGL, the reference target is calculated as the point on the path, which is at a distance D_2 from the quadrotor. The D_1 and D_2 are set for the best following performance. Since the velocity profile of the path is predefined with the evolution velocity of the parameter $\theta_{vel}(t) = 1s^{-1}$, we can compare the trajectory control performance with the predefined time-parameterized reference trajectory $\mathcal{P}(\theta(t)) = \mathcal{P}(t)$, to illustrate the advantage of path following described in Section I. The MPC uses the same double integrator predictive model and parameter settings as in the MPFC.

As shown in Table.I, the proposed method with MPFC as the high-level controller can largely reduce the averaged RMSE compared with the Carrot-chasing and NLGL method in all cases. To assess the robustness to irresistible disturbances, a large wind gust of around $20m/s$ is added to the wind v_w from 7s to 8s to push the quadrotor away from the path. The proposed approach reduces both the maximum following error and the RMSE under disturbances, as shown in table II and Fig.4, respectively, compared with the Carrot-chasing and the NLGL method. Without disturbances, the following errors can also be reduced largely with the MPFC as the high-

level controller. To obtain an intuitive view on the control performance under disturbances, Fig. 7 shows the trajectories of the quadrotor following the aggressive lemniscate path using different high-level controllers, with the position errors shown in Fig. 5. It can be seen that the quadrotor is blown away but soon converges back to the path rapidly with smaller deviation and obviously less overshooting using the proposed method compared with the baseline NLGL and Carrot-chasing methods. Lower following errors can also be observed in the sharp turns.

Figure 6(a) shows the evolution velocity of the reference parameter θ . The MPC tracks the time-parameterized trajectory with higher tracking errors after the quadrotor deviates from the path as shown in Fig. 5(d). With MPFC as the high-level controller, the evolution of the reference target slows down when the quadrotor deviates from the path at around 7s and sharp turns. It reveals that the path following error can be largely reduced with the help of optimization of reference target evolution in the MPFC. Figure 6(b) illustrates the control constraints of u can be satisfied with the designed learning-based control scheme.

V. CONCLUSIONS

In this paper, a novel learning-based predictive control scheme is presented for nonlinear systems to accurately follow paths under uncertain environmental disturbances. A low-level LB-FBLC with GPs is designed to track the reference states accurately under disturbances with a probabilistic stability guarantee. A high-level MPFC exploits an integrator system model and a virtual linear path dynamics model to simultaneously optimize the reference target revolution, and provides the reference states and controls for the LB-FBLC. Simulation results show that the proposed control scheme can successfully drive a quadrotor to accurately follow various geometric paths under different unknown wind disturbances. Both the maximum and the mean following errors are shown to be effectively reduced using the proposed method. The quadrotor with the proposed control scheme exhibits predictive ability when following aggressive paths and robustness to the wind disturbances. In future work, estimation delay and limited update frequency will be considered, as well as hardware experiments under real conditions.

REFERENCES

- [1] J. Matschek, T. B athge, T. Faulwasser, and R. Findeisen, "Nonlinear predictive control for trajectory tracking and path following: An introduction and perspective," in *Handbook of Model Predictive Control*. Springer, 2019, pp. 169–198.
- [2] B. Rub , R. P rez, and B. Morcego, "A survey of path following control strategies for uavs focused on quadrotors," *J. Intell. Robot. Syst.*, vol. 98, no. 2, pp. 241–265, 2020.
- [3] A. Micaelli and C. Samson, "Trajectory tracking for unicycle-type and two-steering-wheels mobile robots," *INRIA, Sophia-Antipolis, Tech. Rep. 2097*, 1993.
- [4] S. Park, J. Deyst, and J. P. How, "Performance and lyapunov stability of a nonlinear path following guidance method," *J. Guid. Control Dyn.*, vol. 30, no. 6, pp. 1718–1728, 2007.
- [5] T. Faulwasser and R. Findeisen, "Nonlinear model predictive control for constrained output path following," *IEEE Trans. Autom. Control*, vol. 61, no. 4, pp. 1026–1039, 2015.
- [6] T. Faulwasser, T. Weber, P. Zometa, and R. Findeisen, "Implementation of nonlinear model predictive path-following control for an industrial robot," *IEEE Trans. Control Syst. Technol.*, vol. 25, no. 4, pp. 1505–1511, 2016.
- [7] H. K. Khalil and J. W. Grizzle, *Nonlinear systems*. Prentice hall Upper Saddle River, NJ, 2002, vol. 3.
- [8] M. Greeff and A. P. Schoellig, "Flatness-based model predictive control for quadrotor trajectory tracking," in *Proc. IEEE/R SJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 6740–6745.
- [9] P. Niermeyer, V. S. Akkinapalli, M. Pak, F. Holzapfel, and B. Lohmann, "Geometric path following control for multirotor vehicles using nonlinear model predictive control and 3d spline paths," in *Proc. int. Conf. Unmanned Aircr. Syst.* IEEE, 2016, pp. 126–134.
- [10] D. Hentzen, T. Stastny, R. Siegart, and R. Brockers, "Disturbance estimation and rejection for high-precision multirotor position control," in *Proc. IEEE/R SJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 2797–2804.
- [11] V. S. Akkinapalli, P. Niermeyer, B. Lohmann, and F. Holzapfel, "Adaptive nonlinear design plant uncertainty cancellation for a multirotor," in *Proc. int. Conf. Unmanned Aircr. Syst.* IEEE, 2016, pp. 1102–1110.
- [12] G. Shi, X. Shi, M. O'Connell, R. Yu, K. Azizzadenesheli, A. Anandkumar, Y. Yue, and S.-J. Chung, "Neural lander: Stable drone landing control using learned dynamics," in *Proc. IEEE int. Conf. Robot. Autom.*, 2019, pp. 9784–9790.
- [13] C. E. Rasmussen, "Gaussian processes in machine learning," in *Summer School on Machine Learning*. Springer, 2003, pp. 63–71.
- [14] A. Urbina, S. Mahadevan, and T. L. Paez, "Quantification of margins and uncertainties of complex systems in the presence of aleatoric and epistemic uncertainty," *Reliab. Eng. Syst. Saf.*, vol. 96, no. 9, pp. 1114–1125, 2011.
- [15] M. K. Helwa, A. Heins, and A. P. Schoellig, "Provably robust learning-based approach for high-accuracy tracking control of lagrangian systems," *IEEE Robot. Autom. Lett.*, vol. 4, no. 2, pp. 1587–1594, 2019.
- [16] T. Beckers, D. Kuli , and S. Hirche, "Stable gaussian process based tracking control of euler–lagrange systems," *Automatica*, vol. 103, pp. 390–397, 2019.
- [17] L. Wang, E. A. Theodorou, and M. Egerstedt, "Safe learning of quadrotor dynamics using barrier certificates," in *Proc. IEEE int. Conf. Robot. Autom.*, 2018, pp. 2460–2465.
- [18] M. Mehndiratta and E. Kayacan, "Gaussian process-based learning control of aerial robots for precise visualization of geological outcrops," in *Proc. Eur. Control Conf.* IEEE, 2020, pp. 10–16.
- [19] K. Pereida and A. P. Schoellig, "Adaptive model predictive control for high-accuracy trajectory tracking in changing conditions," in *Proc. IEEE/R SJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 7831–7837.
- [20] J. Umlauf, L. P hler, and S. Hirche, "An uncertainty-based control lyapunov approach for control-affine systems modeled by gaussian process," *IEEE. Control. Syst.*, vol. 2, no. 3, pp. 483–488, 2018.
- [21] M. Hehn and R. D'Andrea, "Real-time trajectory generation for quadcopters," *IEEE Trans. Robot.*, vol. 31, no. 4, pp. 877–892, 2015.
- [22] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2011, pp. 2520–2525.
- [23] D. Zhou and M. Schwager, "Vector field following for quadrotors using differential flatness," in *Proc. IEEE int. Conf. Robot. Autom.*, 2014, pp. 6567–6572.
- [24] M. W. Mueller and R. D'Andrea, "A model predictive controller for quadcopter state interception," in *Proc. Eur. Control Conf.* IEEE, 2013, pp. 1383–1389.
- [25] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "OSQP: An operator splitting solver for quadratic programs," *Math. Program. Comput.*, 2020. [Online]. Available: <https://doi.org/10.1007/s12532-020-00179-2>
- [26] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.
- [27] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "Casadi: a software framework for nonlinear optimization and optimal control," *Math. Program. Comput.*, vol. 11, no. 1, pp. 1–36, 2019.
- [28] L. T. Biegler and V. Zavala, "Large-scale nonlinear programming using ipopt: An integrating framework for enterprise-wide dynamic optimization," *Comput. Chem. Eng.*, vol. 33, pp. 575–582, 2009.
- [29] K. Cole and A. M. Wickenheiser, "Reactive trajectory generation for multiple vehicles in unknown environments with wind disturbances," *IEEE Trans. Robot.*, vol. 34, no. 5, pp. 1333–1348, 2018.
- [30] D. Moorhouse and R. Woodcock, "Us military specification mil–f–8785c," *US Department of Defense*, 1980.
- [31] M. Shinozuka, "Monte carlo solution of structural dynamics," *Computers & Structures*, vol. 2, no. 5-6, pp. 855–874, 1972.
- [32] G. Deodatis, "Simulation of ergodic multivariate stochastic processes," *J. Eng. Mech.*, vol. 122, no. 8, pp. 778–787, 1996.