Received: 20 June 2022 | Revised: 14 January 2023 | Accepted: 31 January 2023

DOI: 10.1002/rob.22165

RESEARCH ARTICLE

WILEY

Learning-based model predictive control for improved mobile robot path following using Gaussian processes and feedback linearization

Jie Wang | Michael T. H. Fader | Joshua A. Marshall

Ingenuity Labs Research Institute & Department of Electrical and Computer Engineering, Queen's University, Kingston, Ontario, Canada

Correspondence
Jie Wang, Ingenuity Labs Research Institute, Queen's University, Kingston, ON K7L 3N6, Canada.
Email: jwangjie@outlook.com

Funding information
Natural Sciences and Engineering Research Council of Canada

Abstract

This paper proposes a high-performance path following algorithm that combines Gaussian processes (GP) based learning and feedback linearization (FBL) with model predictive control (MPC) for ground mobile robots operating in off-road terrains, referred to as GP-FBLMPC. The algorithm uses a nominal kinematic model and learns unmodeled dynamics as GP models by using observation data collected during field experiments. Extensive outdoor experiments using a Clearpath Husky A200 mobile robot show that the proposed GP-FBLMPC algorithm's performance is comparable to existing GP learning-based nonlinear MPC (GP-NMPC) methods with respect to the path following errors. The advantage of GP-FBLMPC is that it is generalizable in reducing path following errors for different paths that are not included in the GP models training process, while GP-NMPC methods only work well on exactly the same path on which GP models are trained. GP-FBLMPC is also computationally more efficient than the GP-NMPC because it does not conduct iterative optimization and requires fewer GP models to make predictions over the MPC prediction horizon loop at every time step. Field tests show the effectiveness and generalization of reducing path following errors of the GP-FBLMPC algorithm. It requires little training data to perform GP modeling before it can be used to reduce path-following errors for new, more complex paths on the same terrain (see video at https://youtu.be/tC09jJQ0OXM).

KEYWORDS

field robotics, machine learning, path planning of mobile robot, predictive control, wheeled robotics

1 | INTRODUCTION

Path following is a fundamental functionality for autonomous vehicles and mobile robots, where the vehicle is required to track a prescribed path irrespective of time. High-accuracy path following of wheeled robots operating in off-road terrains is challenging because of the complex interaction dynamics between the robot and terrains. Traditionally, evaluating robot–terrain interaction models (also known as terramechanics) is based on the identification of terrain parameters that either requires many in-place measurements prior

robots being deployed or needs to restrict robot operating speeds to lower speeds for online parameters estimation (Iagnemma et al., 2004). More recently, instead of trying to develop accurate mathematical models using first principles based on physics, learning-based or data-driven modeling for controllers has become a popular field of study (Brunke et al., 2021). Machine learning techniques are being applied to learn system models from data, possibly together with a prior nominal model representing known knowledge about the system, and the learned models (possibly together with a prior model) are used as the system process models for motion controllers.

In this work, we propose a Gaussian processes (GP) based feedback linearized model predictive control (GP-FBLMPC) algorithm for path-following control of wheeled mobile robots operating in challenging terrains. In preliminary work (Fader, 2020) we studied a path-following algorithm that combines model predictive control (MPC) with feedback linearization (FBL) for wheeled-vehicle path-following applications, referred to as FBLMPC. The FBLMPC algorithm eliminates the need for iterative nonlinear optimization, and thus is computationally efficient and easy to implement. In this paper, we further improve path-following performance by integrating GP models with a fixed kinematic vehicle model. The GP models are used to learn the discrepancies between a prior kinematic model and the true system model (the observed real-world system behaviors) as disturbance models. The GP models are functions of the robot state and input variables from the observed real-world experiences to learn system disturbances that include the unmodeled dynamics of the robot, wheel-terrain interactions, and other disturbances. While learning-based control for mobile robots has been studied extensively, learning for FBL-based controllers is limited (Spitzer & Michael, 2021). To our knowledge, the GP-FBLMPC algorithm is the first application of GP learning-based MPC combined with FBL for wheeled mobile robot path following, and we show that it is generalizable for the purposes of reducing path following errors on paths different from those used for training.

The main contributions of this work are as follows: (i) A high-performance path-following algorithm (GP-FBLMPC) that combines a nominal kinematic model with GP learning-based dynamics models as the system process model; (ii) extensive outdoor field experiments using a Husky A200 mobile robot (see Figure 1) to validate the GP-FBLMPC algorithm is able to reduce path following errors effectively; and (iii) comparison to a GP learning-based nonlinear MPC method to show that the GP-FBLMPC algorithm is more computationally efficient and generalizable to reducing path following errors across a variety of different paths due to integrating both FBL and GP into MPC.

In this paper, Section 2 reviews the recent research in this field. Section 3 describes the mathematical concepts behind FBLMPC, GP models, and GP learning-based MPC. Section 4 explains the implementation details of the GP-FBLMPC algorithm. Section 5 describes the experimental results of the field tests that validate the effectiveness of the GP-FBLMPC algorithm. Section 6 provides concluding remarks.

2 | RELATED WORK

MPC has been widely used for mobile robot path following applications (Prado et al., 2020; Yu et al., 2021). MPC uses a vehicle model to predict future path-following errors over a finite prediction time horizon given a finite sequence of computed control inputs. By minimizing a cost function that is usually a weighted sum of the predicted errors and the sequence of control inputs, a new optimal sequence of control inputs is computed to minimize the predicted errors. The performance of the resulting MPC design is highly dependent on the accuracy of the model, hence a reasonably accurate process model of the system is usually required. The process models of wheeled robots, whether they are kinematic (Tang et al., 2020), dynamic (Prado et al., 2020), or even learned vehicle models (Kabzan et al., 2019), are nonlinear and result in a non-convex MPC cost function that must be optimized using computationally expensive nonlinear solving techniques (Amer et al., 2016; Ostafew et al., 2016). MPC is being actively researched in the path following control of wheeled robots; see Yu et al. (2021) for a comprehensive review. This section will focus on reviewing MPC methods that combine FBL or/and (Gaussian process) learning-based methods for the path following control of mobile robots.

FBL is a powerful approach to nonlinear control design (Oriolo et al., 2002). As opposed to the conventional linearization based on Taylor series approximations (i.e., Jacobian linearization) of the dynamics, FBL linearizes the nonlinear system by exact state transformations and feedback (Slotine & Li, 1991). Because the FBL technique exactly linearizes the nonlinear system by a suitable state space change of coordinates, MPC methods combing with FBL are different from the standard linear MPC methods that linearize the system model approximately. Combining FBL with MPC in a way that trades the nonlinear optimization for a linear one has been shown via simulations to reduce the computational burden (Caldwell & Marshall, 2021; Greeff & Schoellig, 2020; Wu et al., 2022) and field experiments (Fader, 2020; Greeff et al., 2022; Oriolo et al., 2002). However, implementing FBL requires a precise model (exact knowledge

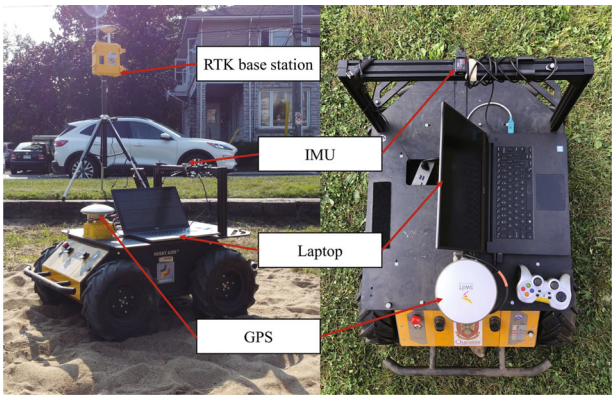


FIGURE 1 Configuration of the Clearpath Husky A200 mobile robot used in this work for field experiments in sand and grass terrains.

of kinematic and dynamic parameters) of the system to ensure an exact cancellation of all nonlinearities. For wheeled robots operating in off-road terrains it can be very challenging, sometimes even impossible, to identify precise dynamics models of robot–terrain interaction from first principles based on physics. Therefore, significant work on learning-based or data-driven approaches for robotics control has been conducted during the past decade (Brunke et al., 2021).

Existing research on learning-based control combined with FBL is limited as it relates to mobile robots (Spitzer & Michael, 2021). The most popular learning strategy applied to FBL-based control is to learn the forward model of the nonlinear terms of the robot system (Spitzer & Michael, 2021; Umlauft et al., 2017). Because FBL requires an exact cancellation of the system's nonlinearities, learning forward models contributes to this need directly. Other methods learn transformed nonlinear terms, nonlinear mismatch (Greeff & Schoellig, 2020), which is defined as the model error between the nominal FBL controller and nonlinear term. In Helwa et al. (2019), a forward model of the nonlinear mismatch is learned by GP models to generate a bound indicating how well the nominal FBL controllers can linearize the system. The bound is used in a linear outer-loop controller. The continuing work in Greeff and Schoellig (2020) further learns an inverse model of the nonlinear mismatch by GP models to cancel the nonlinear mismatch, and is thus able to improve the accuracy of the FBL step. In contrast to the above model-based approaches that learn a model to cancel the system's nonlinearities, Westenbroek et al. (2020) propose a model-free method that learns linearizing controllers for nonlinear systems with unknown dynamics directly by using policy search algorithms of reinforcement learning techniques. Optimal parameters of the linearizing controller that best linearize the system are obtained by formatting a continuous-time optimization problem over the parameters of the learned linearizing controller. This method tries to learn a linearizing controller without learning a model, but Westenbroek et al. (2020) propose the approach should be used in combination with simple nominal dynamics models with parameters that are easy to identify in future work. However, with extensive field tests on three different robots, Ostafew et al. (2016) show by using the same kinematic model as the nominal model, the GP-NMPC algorithm was able to be applied to wheeled robots with significantly different designs. Thus utilizing a kinematic model as the nominal model is simpler to implement and can be easier to be used on different wheeled robots.

There are several works that combine nonlinear MPC (NMPC) and GP learning for path-following applications of wheeled robots. These methods model the robot as a sum of a fixed kinematic model and GP learning-based models (Hewing et al., 2020). GP models are trained with previous experience data to estimate the unmodeled dynamics of the robot and wheel–terrain interactions due to operating in off-road environments (Ostafew et al., 2016) or operating at high speeds (Hewing et al., 2019; Kabzan et al., 2019). Our work is most inspired by Ostafew et al. (2016), who proposed a GP learning-based NMPC algorithm for a path-repeating wheeled robot operating in challenging terrains. The algorithm, referred to herein as GP-NMPC, used a fixed kinematic model as the nominal model and disturbance GP models in the prediction horizon loop to reduce path-tracking errors. The GP-NMPC uses previous experience to estimate disturbance GP models. Similar to Hewing et al. (2019) and Ostafew et al. (2016), this work uses a fixed kinematic model as the nominal model, and the unmodeled system dynamics learned by GP models are utilized to update the process model in the prediction horizon loop of the proposed GP-FBLMPC algorithm. The essential difference is the combination of both GP and FBL with MPC. Its advantages include:

- unlike NMPC, it does not require iterative optimization due to combining FBL with MPC;
- unlike standard linear MPC, it exactly linearizes the nonlinear model by FBL rather than using an approximate linearization;
- unlike GP-NMPC, which is only able to reduce path-following errors on the same path, it is generalizable to reduce path-following errors across a variety of different paths due to integrating both FBL and GP into MPC.

3 | MATHEMATICAL FORMULATION

We investigate the application of GP-FBLMPC on a Clearpath Husky A200 skid-steer mobile robot, as shown in Figure 1. Figure 2 presents a block diagram of the path following control by using the GP-FBLMPC algorithm. The block diagram serves as an outline to present the necessary mathematical formulations of the GP-FBLMPC algorithm.

3.1 | Feedback linearized model predictive control

The kinematics of a skid-steer robot can be approximately modeled by a simple "unicycle" vehicle model (see Figure 3). The configuration of the model can be described by the coordinates $\mathbf{q} = (x, y, \theta) \in \mathbb{R}^2 \times \mathbb{S}^1$, where x and y are the position of the geometric center and θ is the orientation of the vehicle with respect to the global frame $[X, Y]$, respectively. The continuous-time nonholonomic kinematic model is

$$\dot{\mathbf{q}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \underbrace{\begin{bmatrix} v \\ \omega \end{bmatrix}}_{\text{input}}, \tag{1}$$

where the input (v, ω) comprises the commanded forward velocity v (m/s) and yaw rate ω (rad/s). For the sample time $0 < T \ll 1$ and $\mathbf{q}_k \equiv \mathbf{q}(t = kT)$, $k = 0, 1, 2, \ldots$, the discretized vehicle model is

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ \theta_{k+1} \end{bmatrix} = \begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} + T \begin{bmatrix} \cos\theta_k & 0 \\ \sin\theta_k & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_k \\ \omega_k \end{bmatrix}. \tag{2}$$

Figure 3 shows the path following errors (lateral ε_L and heading ε_H) as the vehicle travels along a desired path. The desired path is a sequence of evenly and closely spaced discrete waypoints, where each waypoint specifies a desired vehicle pose (x_d, y_d, θ_d). The reference point (x, y) of the robot is at the geometric center of the chassis. We assume the X-axis of the global reference frame is continuously aligned

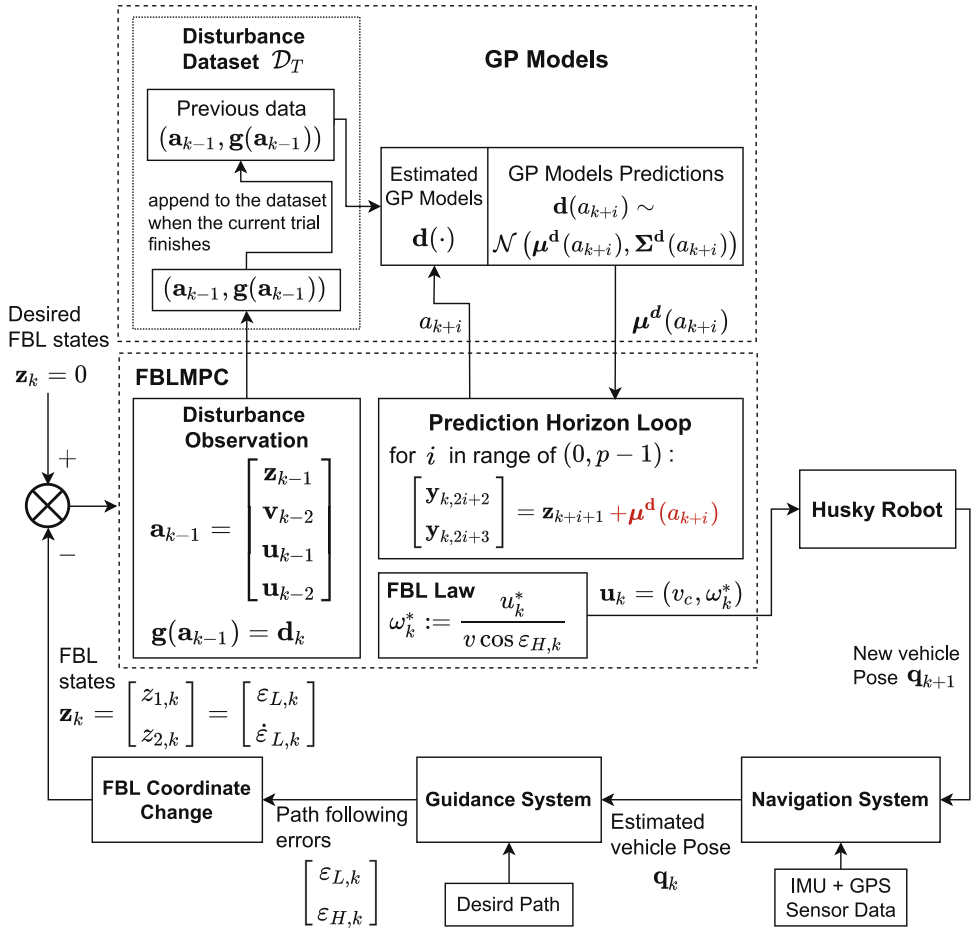


FIGURE 2 Block diagram of the path following control by using the GP-FBLMPC algorithm. GP models learn unmodeled system dynamics from previous driving experience. The estimated GP models ($\boldsymbol{\mu}^{\mathbf{d}}$) are integrated into the system model in the MPC prediction horizon loop.

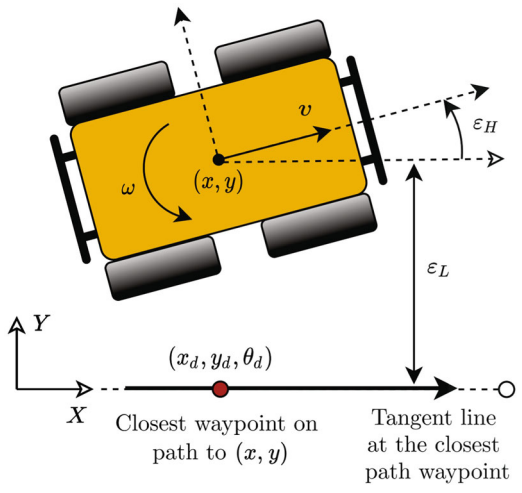


FIGURE 3 Diagram of path following errors defined for Husky robot with respect to a desired path.

with an instantaneous tangent at the closest waypoint to the vehicle so the path following errors are $\varepsilon_L := y$ and $\varepsilon_H := \theta$. Given (1), the instantaneous rate of change of the path following errors are

$$\dot{\varepsilon}_L = \dot{y} = v \sin \varepsilon_H, \tag{3}$$

$$\dot{\varepsilon}_H = \dot{\theta} = \omega. \tag{4}$$

Similar to Dekker et al. (2017), a change of coordinates $z_1 := \varepsilon_L = y$ and $z_2 := \dot{\varepsilon}_L = \dot{y}$ is applied. We refer to these new coordinates $\mathbf{z} = [z_1, z_2]^\top$ as the FBL states and

$$\dot{\mathbf{z}} = \begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u = \mathbf{A}\mathbf{z} + \mathbf{B}u, \tag{5}$$

is the linearized system model where u is a new control input. With constant linear velocity control inputs $v := v_c$, $u = \dot{z}_2 = \omega v_c \cos \varepsilon_H$. Inverting to solve for the corresponding steering rate yields $\omega = u/(v_c \cos \varepsilon_H)$. Clearly, this nonlinear transformation from u to ω works when $v_c \neq 0$ and $\varepsilon_H \in (-\pi/2, \pi/2)$. The FBLMPC formulation

generates a sequence of control inputs $\mathbf{u}_k$ at discrete time indices k such that

$$\mathbf{u}_k := \Delta\mathbf{u}_k + \mathbf{u}_{k-1} = (u_k, u_{k+1}, \ldots, u_{k+p-1}), \tag{6}$$

over a finite time horizon of length p to minimize the predicted path following errors estimated by a model, and $\Delta\mathbf{u}_k = (\Delta u_k, \Delta u_{k+1}, \ldots, \Delta u_{k+p-1})$ is the sequence of linear control updates. Consider a discretized version of (5),

$$\mathbf{z}_{k+1} = \mathbf{F}\mathbf{z}_k + \mathbf{G}u_k = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}\begin{bmatrix} z_{1,k} \\ z_{2,k} \end{bmatrix} + \begin{bmatrix} \frac{T^2}{2} \\ T \end{bmatrix} u_k, \tag{7}$$

for the sample time $0 < T \ll 1$ and $\mathbf{z}_k \equiv \mathbf{z}(t = kT)$, $k = 0, 1, \ldots, m$ with m represents the number of time steps to finish a trial. The predicted FBL state $\mathbf{z}$ over the prediction horizon p is defined as

$$\mathbf{y}_{k+1} := \Delta\mathbf{y}_{k+1} + \mathbf{y}_k = (\mathbf{z}_{k+1}, \mathbf{z}_{k+2}, \ldots, \mathbf{z}_{k+p}), \tag{8}$$

where $\Delta\mathbf{y}_k = (\Delta\mathbf{z}_{k+1}, \Delta\mathbf{z}_{k+2}, \ldots, \Delta\mathbf{z}_{k+p})$ is all of the FBL state updates over the prediction horizon. The discrete updates in control input and FBL states are defined as $\Delta u_k := u_k - u_{k-1}$ and $\Delta\mathbf{z}_k := \mathbf{z}_k - \mathbf{z}_{k-1}$ respectively. Starting at time index k, (7) can be used to describe how the states $\mathbf{z}$ change over the prediction horizon as

$$\mathbf{z}_{k+1} = \mathbf{F}\mathbf{z}_k + \mathbf{G}u_k \tag{9}$$

$$\Delta\mathbf{z}_{k+1} + \mathbf{z}_k = \mathbf{F}(\Delta\mathbf{z}_k + \mathbf{z}_{k-1}) + \mathbf{G}(\Delta u_k + u_{k-1}) \tag{10}$$

$$\Delta\mathbf{z}_{k+1} = \mathbf{F}\Delta\mathbf{z}_k + \mathbf{G}\Delta u_k \tag{11}$$

$$\Delta\mathbf{z}_{k+2} = \mathbf{F}\Delta\mathbf{z}_{k+1} + \mathbf{G}\Delta u_{k+1} \tag{12}$$

$$\begin{aligned} &\vdots \\ \Delta\mathbf{z}_{k+p} &= \mathbf{F}^p\Delta\mathbf{z}_k + \mathbf{F}^{p-1}\mathbf{G}\Delta u_k + \cdots + \mathbf{G}\Delta u_{k+p-1}, \end{aligned} \tag{13}$$

which are rewritten in matrix form as

$$\begin{aligned} \underbrace{\begin{bmatrix} \Delta\mathbf{z}_{k+1} \\ \Delta\mathbf{z}_{k+2} \\ \vdots \\ \Delta\mathbf{z}_{k+p} \end{bmatrix}}_{\Delta\mathbf{y}_{k+1}} &= \underbrace{\begin{bmatrix} \mathbf{F} \\ \mathbf{F}^2 \\ \vdots \\ \mathbf{F}^p \end{bmatrix}}_{\mathbf{L}} \Delta\mathbf{z}_k \\ &+ \underbrace{\begin{bmatrix} \mathbf{G} & 0 & \cdots & 0 & 0 \\ \mathbf{FG} & \mathbf{G} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{F}^{p-1}\mathbf{G} & \mathbf{F}^{p-2}\mathbf{G} & \cdots & \mathbf{FG} & \mathbf{G} \end{bmatrix}}_{\mathbf{M}} \underbrace{\begin{bmatrix} \Delta u_k \\ \Delta u_{k+1} \\ \Delta u_{k+2} \\ \vdots \\ \Delta u_{k+p-1} \end{bmatrix}}_{\Delta\mathbf{u}_k}, \end{aligned} \tag{14}$$

and more compactly in

$$\Delta\mathbf{y}_{k+1} = \mathbf{L}\Delta\mathbf{z}_k + \mathbf{M}\Delta\mathbf{u}_k. \tag{15}$$

The MPC approach seeks to find a sequence of control inputs over a given prediction horizon that minimizes a cost function. In this work, we compare the path following the performance of our GP-FBLMPC algorithm to a GP learning-based nonlinear MPC (GP-NMPC). To conduct a fair comparison, we followed the cost function form defined in the GP-NMPC (Ostafew et al., 2016) as

$$J(\mathbf{u}_k) = \mathbf{y}_{k+1}^\top\mathbf{Q}\mathbf{y}_{k+1} + \mathbf{u}_k^\top\mathbf{R}\mathbf{u}_k, \tag{16}$$

where $\mathbf{Q} \in \mathbb{R}^{2p\times 2p}$ and $\mathbf{R} \in \mathbb{R}^{p\times p}$ are two tunable positive-definite weighting matrices, $\mathbf{y}_{k+1}$ is the predicted FBL state $\mathbf{z}$ over the prediction horizon p, and $\mathbf{u}_k$ is a sequence of the new control inputs at the current time step. This cost function is the quadratic cost form defined in Primbs and Nevistic (1997) that completely converts a nonlinear problem into a linear MPC problem to obtain the most computational efficiency. Since the cost function is a linear optimization problem with no state and input constraints, this optimization is done in a single step with minimal computation expense. Although there are no constraints on the cost function optimization due to the above two reasons, the steering rate command applied to the vehicle (ω_k^*) was constrained after the optimization to ensure reasonable and safe actuation of real vehicles. For the Husky robot, ω_k^* was saturated to ±2 rad/s (shown in Algorithm 1) due to its steering rate limit.

By substituting (6), (8), and (15) to (16), we get

$$\begin{aligned} J(\Delta\mathbf{u}_k) = (\mathbf{L}\Delta\mathbf{z}_k + \mathbf{M}\Delta\mathbf{u}_k + \mathbf{y}_k)^\top\mathbf{Q}(\mathbf{L}\Delta\mathbf{z}_k + \mathbf{M}\Delta\mathbf{u}_k \\ + \mathbf{y}_k) + (\Delta\mathbf{u} + \mathbf{u}_{k-1})^\top\mathbf{R}(\Delta\mathbf{u} + \mathbf{u}_{k-1}). \end{aligned}$$

Because J is quadratic with respect to $\Delta\mathbf{u}_k$ and is convex, the optimal sequence of control updates, denoted $\Delta\mathbf{u}_k^*$, that minimizes J is found by solving $\partial J(\Delta\mathbf{u}_k)/\partial\Delta\mathbf{u}_k = 0$ as

$$\Delta\mathbf{u}_k^* = -(\mathbf{M}^\top\mathbf{Q}\mathbf{M} + \mathbf{R})^{-1}(\mathbf{M}^\top\mathbf{Q}(\mathbf{y}_k + \mathbf{L}\Delta\mathbf{z}_k) + \mathbf{R}\mathbf{u}_{k-1}). \tag{17}$$

By taking the optimal sequence of control updates $\Delta\mathbf{u}_k^*$ produced by (17) and adding it to the previous sequence of control inputs $\mathbf{u}_{k-1}$, we generate a new optimal sequence, $\mathbf{u}_k^* = \Delta\mathbf{u}_k^* + \mathbf{u}_{k-1}$. The first input from $\mathbf{u}_k^*$, defined as u_k^*, is used in real-time on the vehicle by applying the steering rate

$$\omega_k^* = \frac{u_k^*}{v\cos\varepsilon_{H,k}}, \tag{18}$$

to steer the vehicle along the desired path. The linear and angular velocity control inputs at time k are $\mathbf{u}_k = (v_k^{cmd}, \omega_k^{cmd}) = (v_c, \omega_k^*)$. Note that the matrices $(\mathbf{M}^\top\mathbf{Q}\mathbf{M} + \mathbf{R})^{-1}$ and $\mathbf{M}^\top\mathbf{Q}$ are constants at all time steps and thus can be precomputed to reduce real-time computational complexity, which is a key difference between the FBLMPC strategy in this work and traditional nonlinear MPC. So far, an FBLMPC is designed on the unicycle kinematic model, which is very often done as a simplification of the problem when robot dynamics is negotiable (Oriolo et al., 2002). However, for vehicles operating in off-road terrains, and/or massive vehicles, and/or vehicles operating at high speeds, consideration of vehicle dynamics is necessary for realistic control design. We will explain the process to use GP models to learn unmodeled system dynamics from previous driving experience in Section 3.2. And how to integrate the estimated GP models into the system model in the MPC prediction horizon loop in Section 3.3.

3.2 | GP regression

GP regression has become one of the most commonly employed machine learning techniques in learning-based control (Hewing et al., 2020). Consider m input data points $\mathbf{a} = [\mathbf{a}_1, ..., \mathbf{a}_m]^\mathsf{T} \in \mathbb{R}^{n_a \times m}$ and the corresponding measurements $\mathbf{d} = [\mathbf{d}_1, ..., \mathbf{d}_m]^\mathsf{T} \in \mathbb{R}^{n_d \times m}$, related through an unknown function $\mathbf{d}_k = \mathbf{g}(\mathbf{a}_k) + \boldsymbol{\omega}_k$: $\mathbb{R}^{n_a} \to \mathbb{R}^{n_d}$, where $\boldsymbol{\omega}_k$ is i.i.d. Gaussian noise with $\boldsymbol{\omega}_k \sim \mathcal{N}(0, \boldsymbol{\Sigma}^\omega)$ and $\boldsymbol{\Sigma}^\omega = \mathrm{diag}\left(\left[\sigma_1^2, ..., \sigma_{n_d}^2\right]\right)$. The function $\mathbf{g}(\cdot)$ can be identified by the observed input-output data set

$$\mathcal{D}_m = \left\{ \mathbf{a} = [\mathbf{a}_1, ..., \mathbf{a}_m]^\mathsf{T} = \begin{bmatrix} a_{1,1} & \cdots & a_{1,m} \\ \vdots & \ddots & \vdots \\ a_{n_a,1} & \cdots & a_{n_a,m} \end{bmatrix}, \right.$$
$$\left. \mathbf{d} = [\mathbf{d}_1, ..., \mathbf{d}_m]^\mathsf{T} = \begin{bmatrix} d_{1,1} & \cdots & d_{1,m} \\ \vdots & & \vdots \\ d_{M,1} & \ddots & d_{M,m} \\ \vdots & & \vdots \\ d_{n_d,1} & \cdots & d_{n_d,m} \end{bmatrix} \right\}.$$

Assume each output dimension of $\mathbf{d}_k$ is independent of each other given the input $\mathbf{a}_k$. For each dimension $M \in \{1, ..., n_d\}$ of the function output $\mathbf{d}_k$, specifying a GP with zero mean and prior kernel $k^M(\cdot,\cdot)$, the measurement data $[\mathbf{d}_k]_{M,\cdot}$ is normally distributed as $[\mathbf{d}_k]_{M,\cdot} \sim \mathcal{N}(0, K_{\mathbf{aa}}^M + \sigma_M^2)$, where $K_{\mathbf{aa}}^M$ is the Gram matrix of the data points using the kernel function $k^M(\cdot,\cdot)$ on the input locations a, that is, $[K_{\mathbf{aa}}^M]_{ij} = k^M(\mathbf{a}_i, \mathbf{a}_j)$ (Hewing et al., 2018). The choice of kernel functions $k^M(\cdot,\cdot)$ is specified by prior knowledge of the observed data. In this work, we use the squared exponential (SE) kernel like many other GP learning-based robotic control applications (Hewing et al., 2020),

$$K^M(\mathbf{a}, a) = \sigma_{f,M}^2 \exp\left(-\frac{1}{2}(\mathbf{a} - a)^\mathsf{T} L_M^{-1} (\mathbf{a} - a)\right), \tag{19}$$

where a represents new data points where predictions are made, $\sigma_{f,M}^2$ and $L_M \in \mathbb{R}^{n_a \times n_a}$ are hyperparameters that are optimized by the log marginal likelihood function (Rasmussen & Williams, 2006) in (41). The GP models with the optimized hyperparameters are the trained models by using the observed data set.

In the output dimension M, the joint distribution of the observed output $[\mathbf{d}]_{M,\cdot}$ and the prediction output $[\mathbf{d}]_M$ at new data points a is $P([\mathbf{d}]_{M,\cdot}, [\mathbf{d}]_M | \mathbf{a}, a)$, and

$$\begin{bmatrix} [\mathbf{d}]_{M,\cdot} \\ [\mathbf{d}]_M \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K_{\mathbf{aa}}^M + I\sigma_M^2 & K_{a\mathbf{a}}^M \\ K_{\mathbf{a}a}^M & K_{aa}^M \end{bmatrix}\right), \tag{20}$$

where $[K_{\mathbf{a}a}^M]_j = k^M(a_j, a)$, $K_{a\mathbf{a}}^M = (K_{\mathbf{a}a}^M)^\mathsf{T}$, and $K_{aa}^M = k^M(a, a)$. The posterior distribution of $[\mathbf{d}]_M$ conditioned on the observed data can be derived from (20) as $P([\mathbf{d}]_M | [\mathbf{d}]_{M,\cdot}, \mathbf{a}, a) = \mathcal{N}(\mu^M(a), \Sigma^M(a))$ with (Rasmussen & Williams, 2006)

$$\mu^M(a) = K_{a\mathbf{a}}^M \left(K_{\mathbf{aa}}^M + I\sigma_M^2\right)^{-1} [\mathbf{d}]_{M,\cdot}, \tag{21}$$

$$\Sigma^M(a) = K_{aa}^M - K_{a\mathbf{a}}^M \left(K_{\mathbf{aa}}^M + I\sigma_M^2\right)^{-1} K_{\mathbf{a}a}^M. \tag{22}$$

The resulting GP model estimation $\mathbf{d}(\cdot)$ of the unknown function $\mathbf{g}(\cdot)$ is obtained by vertically concatenating the individual output dimension $M \in \{1, ..., n_d\}$ as

$$\mathbf{d}(a) \sim \mathcal{N}(\boldsymbol{\mu}^\mathbf{d}(a), \boldsymbol{\Sigma}^\mathbf{d}(a)), \tag{23}$$

with $\boldsymbol{\mu}^\mathbf{d}(a) = [\mu^1(a), ..., \mu^{n_d}(a)]^\mathsf{T} \in \mathbb{R}^{n_d}$ and $\boldsymbol{\Sigma}^\mathbf{d}(a) = \mathrm{diag}([\Sigma^1(a), ..., \Sigma^{n_d}(a)]^\mathsf{T}) \in \mathbb{R}^{n_d \times n_d}$.

3.3 | Combining GP modeling with MPC

Consider a nonlinear system $\mathbf{x}_{k+1} = \mathbf{f}_{\text{true}}(\mathbf{x}_k, \mathbf{u}_k)$ with observable states $\mathbf{x}_k \in \mathbb{R}^n$ and control input $\mathbf{u}_k \in \mathbb{R}^m$. The true system model can be represented by the sum of a nominal model and a learning-based model as

$$\mathbf{x}_{k+1} = \overbrace{\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k)}^{\text{nominal model}} + \overbrace{\mathbf{g}(\mathbf{x}_k, \mathbf{u}_k)}^{\text{learning-based model}}. \tag{24}$$

The model $\mathbf{f}(\cdot)$ is a nominal process model representing our knowledge of $\mathbf{f}_{\text{true}}(\cdot)$, and $\mathbf{g}(\cdot)$ is a model representing discrepancies between the nominal model and the true system model. We would like to approximate $\mathbf{g}(\cdot)$ as GP models $\mathbf{d}(\cdot)$ that can be estimated by the observation data.

Because GP models output probability distributions, iterative predictions for multiple time steps are analytically intractable and the resulting distribution is no longer Gaussian (Polymenakos et al., 2020). Many approximation methods for multistep-ahead predictions have been proposed (Vinogradska et al., 2018). A common approach is to approximate the distributions of the states at each prediction time step as a Gaussian $x_k \sim \mathcal{N}(\mu_k^x, \Sigma_k^x)$ (Hewing et al., 2018). In this work, we only utilize the mean predictions of the GP models in the controller. The GP mean approximations are explicitly derived below. The variance approximations of the GP model can be found in Hewing et al. (2018).

When we use (24) as the system model, the propagation of the system states from $\mathbf{x}_k$ to $\mathbf{x}_{k+1}$ are expressed in the form of GP means by using the law of iterated expectations and law of conditional variances following (Girard et al., 2003) as

$$\begin{aligned} m(\mathbf{x}_{k+1}) &= \boldsymbol{\mu}_{k+1}^\mathbf{x} \\ &= \mathbb{E}_{\mathbf{x}_k}[\mathbb{E}_\mathbf{d}(\mathbf{x}_{k+1})] \\ &= \mathbb{E}_{\mathbf{x}_k}[\mathbb{E}_\mathbf{d}(\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) + \mathbb{E}_\mathbf{d}(\mathbf{g}(\mathbf{x}_k, \mathbf{u}_k))], \end{aligned} \tag{25}$$

where $\boldsymbol{\mu}_{k+1}^\mathbf{x}$ represents the mean of $\mathbf{x}_{k+1}$, $\mathbb{E}_{\mathbf{x}_k}$ indicates the expectation under $(\mathbf{x}_k, \mathbf{u}_k)$, and $\mathbb{E}_\mathbf{d}$ represents the expectation under the estimated GP model $\mathbf{d}(\mathbf{x}_k, \mathbf{u}_k)$. Substituting (23) into (25), we get

$$\boldsymbol{\mu}_{k+1}^\mathbf{x} = \mathbb{E}_{\mathbf{x}_k}[\mathbb{E}_\mathbf{d}(\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) + \boldsymbol{\mu}^\mathbf{d}(\mathbf{x}_k, \mathbf{u}_k)]. \tag{26}$$

The $\mathbb{E}_\mathbf{d}(\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k))$ is the expected value of $\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k)$ by the estimated GP model $\mathbf{d}(\cdot)$ at $(\mathbf{x}_k, \mathbf{u}_k)$ that equals $\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k)$. Thus,

$$\begin{aligned}\boldsymbol{\mu}_{k+1}^{\mathbf{x}} &= \mathbb{E}_{\mathbf{x}_k}[\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) + \boldsymbol{\mu}^{\mathbf{d}}(\mathbf{x}_k, \mathbf{u}_k)] \\ &= \mathbb{E}_{\mathbf{x}_k}[\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k)] + \mathbb{E}_{\mathbf{x}_k}[\boldsymbol{\mu}^{\mathbf{d}}(\mathbf{x}_k, \mathbf{u}_k)].\end{aligned} \tag{27}$$

The means of the predictive distribution can be approximated by the first-order terms of their Taylor series expansions (Hewing et al., 2018). The functions $\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k)$ and $\boldsymbol{\mu}^{\mathbf{d}}(\mathbf{x}_k, \mathbf{u}_k)$ are approximated by their first order Taylor expansion about $\mathbf{x}_k = \boldsymbol{\mu}_k^{\mathbf{x}}$ and $\mathbf{u}_k = \boldsymbol{\mu}_k^{\mathbf{u}}$, respectively, as

$$\begin{aligned}\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) \approx \mathbf{f}\left(\boldsymbol{\mu}_k^{\mathbf{x}}, \boldsymbol{\mu}_k^{\mathbf{u}}\right) &+ \left.\frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{x}}\right|_{\boldsymbol{\mu}_k^{\mathbf{x}}, \boldsymbol{\mu}_k^{\mathbf{u}}} \left(\mathbf{x}_k - \boldsymbol{\mu}_k^{\mathbf{x}}\right) \\ &+ \left.\frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{u}}\right|_{\boldsymbol{\mu}_k^{\mathbf{x}}, \boldsymbol{\mu}_k^{\mathbf{u}}} \left(\mathbf{u}_k - \boldsymbol{\mu}_k^{\mathbf{u}}\right),\end{aligned} \tag{28}$$

and

$$\begin{aligned}\boldsymbol{\mu}^{\mathbf{d}}(\mathbf{x}_k, \mathbf{u}_k) \approx \boldsymbol{\mu}^{\mathbf{d}}\left(\boldsymbol{\mu}_k^{\mathbf{x}}, \boldsymbol{\mu}_k^{\mathbf{u}}\right) &+ \left.\frac{\partial \boldsymbol{\mu}^{\mathbf{d}}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{x}}\right|_{\boldsymbol{\mu}_k^{\mathbf{x}}, \boldsymbol{\mu}_k^{\mathbf{u}}} \left(\mathbf{x}_k - \boldsymbol{\mu}_k^{\mathbf{x}}\right) \\ &+ \left.\frac{\partial \boldsymbol{\mu}^{\mathbf{d}}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{u}}\right|_{\boldsymbol{\mu}_k^{\mathbf{x}}, \boldsymbol{\mu}_k^{\mathbf{u}}} \left(\mathbf{u}_k - \boldsymbol{\mu}_k^{\mathbf{u}}\right).\end{aligned} \tag{29}$$

By following the predictive mean of Gaussian approximation equations in Girard et al. (2003), we get

$$\mathbb{E}_{\mathbf{x}_k}(\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k)) \approx \mathbf{f}\left(\boldsymbol{\mu}_k^{\mathbf{x}}, \boldsymbol{\mu}_k^{\mathbf{u}}\right), \tag{30}$$

$$\mathbb{E}_{\mathbf{x}_k}(\boldsymbol{\mu}^{\mathbf{d}}(\mathbf{x}_k, \mathbf{u}_k)) \approx \boldsymbol{\mu}^{\mathbf{d}}\left(\boldsymbol{\mu}_k^{\mathbf{x}}, \boldsymbol{\mu}_k^{\mathbf{u}}\right). \tag{31}$$

Substituting (30) and (31) into (27), the mean prediction approximations of the system model represented by (24) is derived as

$$\boldsymbol{\mu}_{k+1}^{\mathbf{x}} \approx \mathbf{f}\left(\boldsymbol{\mu}_k^{\mathbf{x}}, \boldsymbol{\mu}_k^{\mathbf{u}}\right) + \boldsymbol{\mu}^{\mathbf{d}}\left(\boldsymbol{\mu}_k^{\mathbf{x}}, \boldsymbol{\mu}_k^{\mathbf{u}}\right). \tag{32}$$

4 | IMPLEMENTATION

This section illustrates the GP-FBLMPC implementations shown in Figure 2. Section 4.1 explains the *Disturbance Observation* block for disturbance data collection, Section 4.2 describes *GP Models* block for GP models estimation, and Section 4.3 demonstrates *Prediction Horizon Loop* block for disturbance input state a_{k+i} calculation.

4.1 | GP learning-based FBLMPC

The setup is very similar to Ostafew et al. (2016). Let the kinematics and dynamics of our vehicle be given by

$$\text{Kinematics: } \mathbf{x}_{k+1} = \mathbf{f}_{\mathbf{x},\text{true}}(\mathbf{x}_k, \mathbf{v}_k), \tag{33}$$

$$\text{Dynamics: } \mathbf{v}_{k+1} = \mathbf{f}_{\mathbf{v},\text{true}}(\mathbf{v}_k, \mathbf{u}_k), \tag{34}$$

where $\mathbf{x}_k = (x_k, y_k, \theta_k)$ represents the vehicle's pose, $\mathbf{v}_k = (v_k^{act}, \omega_k^{act})$ represents the actual velocity (linear and rotational speed) of the vehicle, and $\mathbf{u}_k = (v_k^{cmd}, \omega_k^{cmd})$ are the control inputs (linear and angular velocity) at time k. By substituting $\mathbf{v}_k = \mathbf{f}_{\mathbf{v},\text{true}}(\mathbf{v}_{k-1}, \mathbf{u}_{k-1})$ into (33), the dynamics can be cascaded into the kinematics as $\mathbf{x}_{k+1} = \mathbf{f}'_{\text{true}}(\mathbf{x}_k, \mathbf{v}_{k-1}, \mathbf{u}_{k-1})$. If we assume the robot dynamics are negligible, then the actual velocity $\mathbf{v}_k$ equals the commanded ones $\mathbf{u}_k$, thus the true process model, $\mathbf{f}'_{\text{true}}(\cdot)$ can be represented by the sum of the nominal and learning-based models as

$$\mathbf{x}_{k+1} = \overbrace{\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k)}^{\text{nominal model}} + \underbrace{\overbrace{\mathbf{g}(\mathbf{x}_k, \mathbf{v}_{k-1}, \mathbf{u}_k, \mathbf{u}_{k-1})}^{\text{learning-based model}}}_{a_k}. \tag{35}$$

Here we follow Ostafew et al. (2016) to define the discrepancies between the nominal model and the true process model that include the unmodeled dynamics of the robot, wheel–terrain interactions, and other disturbances as a disturbance model

$$\mathbf{g}(a_k) = \mathbf{x}_{k+1} - \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k), \tag{36}$$

with the disturbance state, $a_k = (\mathbf{x}_k, \mathbf{v}_{k-1}, \mathbf{u}_k, \mathbf{u}_{k-1})$. In fact, "disturbance" is a common term to define discrepancies between a prior nominal model and the true process model in active disturbance rejection control (Zhao et al., 2020).

In GP-FBLMPC, we use the FBL states $\mathbf{z}_k$ as the system state variables instead of the robot pose states $\mathbf{x}_k$. The right side of (36) is the difference between the robot's actual and predictive pose state $\mathbf{x}_{k+1}$, which are the FBL states, as explained in Section 3.1. The learning-based model of the FBL states is

$$\mathbf{z}_{k+1} = \mathbf{g}(a_k), \tag{37}$$

with the disturbance state, $a_k = (\mathbf{z}_k, \mathbf{v}_{k-1}, \mathbf{u}_k, \mathbf{u}_{k-1})$. If we represent the learning-based model $\mathbf{g}(\cdot)$ as GP models $\mathbf{d}(\cdot)$, by (32), the mean prediction of (37) is

$$\mathbf{z}_{k+1} \approx \boldsymbol{\mu}^{\mathbf{d}}(a_k). \tag{38}$$

Similar to Ostafew et al. (2016), Hewing et al. (2019), and Kabzan et al. (2019), we estimate a separate GP model for each dimension of $\mathbf{g}(\cdot) \in \mathbb{R}^{n_d}$ to model disturbances assuming that they are not correlated. In the FBLMPC, there are two FBL states, thus the output dimension $n_d = 2$. By applying (23), the predictions of the estimated GP models $\mathbf{d}(\cdot)$ at arbitrary new disturbance states a are obtained as

$$\mathbf{d}(a) = \begin{bmatrix}\mathbf{d}^1(a) \\ \mathbf{d}^2(a)\end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix}\mu_1^{\mathbf{d}}(a) \\ \mu_2^{\mathbf{d}}(a)\end{bmatrix}, \begin{bmatrix}\Sigma^1(a) & 0 \\ 0 & \Sigma^2(a)\end{bmatrix}\right), \tag{39}$$

with

$$\begin{aligned}\mu_1^{\mathbf{d}}(a) &= K_{a\mathbf{a}}^1\left(K_{\mathbf{aa}}^1 + I\sigma_1^2\right)^{-1}[\mathbf{d}]_{\cdot,1}, \\ \mu_2^{\mathbf{d}}(a) &= K_{a\mathbf{a}}^2\left(K_{\mathbf{aa}}^2 + I\sigma_2^2\right)^{-1}[\mathbf{d}]_{\cdot,2}, \\ \Sigma^1(a) &= K_{aa}^1 - K_{a\mathbf{a}}^1\left(K_{\mathbf{aa}}^1 + I\sigma_M^2\right)^{-1}K_{\mathbf{a}a}^1, \\ \Sigma^2(a) &= K_{aa}^2 - K_{a\mathbf{a}}^2\left(K_{\mathbf{aa}}^2 + I\sigma_M^2\right)^{-1}K_{\mathbf{a}a}^2.\end{aligned}$$

In which $K^1(\cdot)$ and $K^2(\cdot)$ represent the squared exponential kernel in (19), $\mathbf{a}$ and a represent disturbance states of the previous observed data set used to estimate the learning-based model and new

disturbance states where predictions are made respectively, σ_1 and σ_2 are i.i.d. Gaussian noise, and $[\mathbf{d}]_{:,1}$ and $[\mathbf{d}]_{:,2}$ represent the output data of the observed disturbance data set $\mathcal{D}_m^1$ and $\mathcal{D}_m^2$, respectively, as

$$\begin{aligned}\mathcal{D}_m^1 &= \{\mathbf{a} = [\mathbf{a}_1, ..., \mathbf{a}_{k-1}, ..., \mathbf{a}_m]^\mathsf{T},\\ &\quad [\mathbf{d}]_{:,1} = [d_{1,1}, ..., d_{1,k}, ..., d_{1,m}]^\mathsf{T}\},\\ \mathcal{D}_m^2 &= \{\mathbf{a} = [\mathbf{a}_1, ..., \mathbf{a}_{k-1}, ..., \mathbf{a}_m]^\mathsf{T},\\ &\quad [\mathbf{d}]_{:,2} = [d_{2,1}, ..., d_{2,k}, ..., d_{2,m}]^\mathsf{T}\}.\end{aligned}$$

In each trial of the path following experiment, the observed input–output disturbance data at all time steps are collected as disturbance data $\mathcal{D}_t = \left\{\mathcal{D}_m^1 = \{\mathbf{a}, [\mathbf{d}]_{:,1}\}, \mathcal{D}_m^2 = \{\mathbf{a}, [\mathbf{d}]_{:,2}\}\right\}$, with m being the total time steps to finish a trial. At every time step k, the output data $\mathbf{d}_k$ is calculated by substituting $k-1$ to (37) as

$$\mathbf{d}_k = \mathbf{g}(\mathbf{a}_{k-1}) = \mathbf{z}_k, \tag{40}$$

which equals to the FBL states $\mathbf{z}_k$. The disturbance state $\mathbf{a}_{k-1} = (\mathbf{z}_{k-1}, \mathbf{v}_{k-2}, \mathbf{u}_{k-1}, \mathbf{u}_{k-2})$ are the input data $\mathbf{a}$ of the observation data $\mathcal{D}_t$. When a trial finishes, the data set $\mathcal{D}_T$ is updated by attaching the input–output data $\mathcal{D}_t$ to it as shown in Figure 2. The updated data set $\mathcal{D}_T$ is used to train GP models $\mathbf{d}(\cdot)$ before the next trial starts.

4.2 | GP model estimation

At the beginning of the path following task, in the first trial, the robot was driven autonomously by the FBLMPC without GP models to follow the path and collect disturbance observation data. The disturbance predictions (the means of GP models) in trial 1 are set to zeros. The collected data set $\mathcal{D} = \{\mathcal{D}_m^1, \mathcal{D}_m^2\}$ is used to train GP models offline before the second trial starts. In the second trial and after, the robot uses GP models estimated by the disturbance data collected in previous trials to make disturbance predictions and collects new observation data through the path. When a trial is finished, as shown in Figure 2, the newly collected data is appended to the previous disturbance data set, and the current data set is used to update the GP models to make predictions in the later following trials.

The process to optimize the hyperparameters $\mathbf{H}$ of the kernel functions is the estimation/training of GP models (Wang, 2020). The optimized hyperparameters $\mathbf{H}^*$ are determined by maximizing the log marginal likelihood of collected disturbance observation data using a gradient ascent algorithm (Rasmussen & Williams, 2006) as

$$\mathbf{H}^* = \arg\max_{\mathbf{H}} \log P(\mathbf{d}|\mathbf{a}, \mathbf{H}), \tag{41}$$

where $\mathbf{a}$ and $\mathbf{d}$ are the collected disturbance observed input-output data, and $\mathbf{H}$ represents the hyperparameters of the kernel function. Here, we used the squared exponential kernel defined in (19), and the initial values of the variance and length scale were set to 0.01 and 0.1, respectively, by using the trial and error method. The optimized set of hyperparameters were obtained after the optimization process was restarted 20 times to avoid local maxima.

4.3 | Disturbance state calculations

In the prediction horizon loop, as shown in Figure 2, all of the parameters of $a_{k+i} = (\mathbf{z}_{k+i}, \mathbf{v}_{k-1+i}, \mathbf{u}_{k+i}, \mathbf{u}_{k-1+i})$, $i \in K = \{0, ..., p-1\}$ are required to make disturbance predictions by the estimated GP models $\mathbf{d}(\cdot)$. The third and fourth variables of the control input $\mathbf{u}$ are known because they are the commanded control inputs, the FBL state variables $\mathbf{z}_{k+i} = [z_{1,k+i} \quad z_{2,k+i}]^\mathsf{T}$ and velocity state variables $\mathbf{v}_{k-1+i} = \left[v_{k-1+i}^{act}, \quad \omega_{k-1+i}^{act}\right]^\mathsf{T}$ are required.

At every time step k, shown in Figure 2, based on the estimated vehicle pose $(\hat{x}_k, \hat{y}_k, \hat{\theta}_k)$ by the navigation/localization system, the guidance system finds the closest path waypoint on the desired path to the vehicle and calculates the associated path following errors. In Figure 4, the Euclidean distance between any path waypoint (x_n^d, y_n^d) and the current estimated vehicle position $(\hat{x}_k, \hat{y}_k)$ is calculated as $d_n = \sqrt{(\hat{x}_k - x_n^d)^2 + (\hat{y}_k - y_n^d)^2}$, and the path waypoint with the smallest d_n is the closest desired point (x_d, y_d) to the vehicle. To minimize the guidance system's computational burden at every time step k, the guidance system was designed to calculate d_n for a certain number of path points ahead ($n_a = 20$) and behind ($n_b = 10$) the closest path waypoint from the previous time step $k-1$. Once the closest path waypoint $(x_{d,k}, y_{d,k}, \theta_{d,k})$ has been identified by the guidance system at the current time step k, the current lateral and heading path following errors $(\varepsilon_{L,k}, \varepsilon_{H,k})$ shown in Figure 4 are calculated as

$$\varepsilon_{L,k} = -(\hat{x}_k - x_{d,k})\sin(\theta_{d,k}) + (\hat{y}_k - y_{d,k})\cos(\theta_{d,k}), \tag{42}$$

$$\varepsilon_{H,k} = \hat{\theta}_k - \theta_{d,k}. \tag{43}$$

The Husky A200 mobile robot is modeled as a unicycle vehicle in (2), in the prediction horizon loop $i \in K = \{0, ..., p-1\}$, the model employed in the FBLMPC is

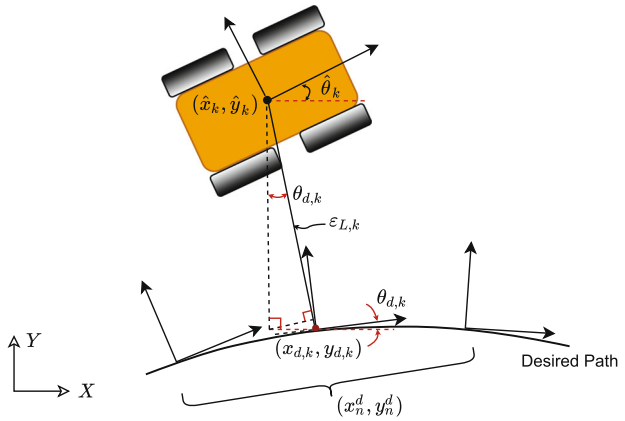


FIGURE 4 The calculation of the lateral and heading path following errors. This path following error calculation is performed differently from Section 3.1 because the global reference frame $[X, Y]$ is world fixed and not continuously aligned with the desired path.

$$\begin{bmatrix} x_{k+i} \\ y_{k+i} \\ \theta_{k+i} \end{bmatrix} = \begin{bmatrix} x_{k-1+i} \\ y_{k-1+i} \\ \theta_{k-1+i} \end{bmatrix} + T \begin{bmatrix} \cos\theta_{k-1+i} & 0 \\ \sin\theta_{k-1+i} & 0 \\ 0 & 1 \end{bmatrix} \underbrace{\begin{bmatrix} v_{k-1+i} \\ \omega_{k-1+i} \end{bmatrix}}_{\mathbf{u}_{k-1+i}}. \tag{44}$$

The predictive lateral and heading errors ($\varepsilon_{L,k+i}$, $\varepsilon_{H,k+i}$) of the prediction horizon loop are calculated as

$$\begin{aligned} \varepsilon_{L,k+i} = &-(x_{k+i} - x_{d,k+i})\sin(\theta_{d,k+i}) \\ &+(y_{k+i} - y_{d,k+i})\cos(\theta_{d,k+i}), \end{aligned} \tag{45}$$

$$\varepsilon_{H,k+i} = \theta_{k+i} - \theta_{d,k+i}, \tag{46}$$

where $(x_{d,k+i}, y_{d,k+i}, \theta_{d,k+i})$ is the closest desired path waypoint of $k+i$ determined by the guidance system. The FBL states $\mathbf{z}_{k+i}$ are calculated as

$$\mathbf{z}_{k+i} = \begin{bmatrix} z_{1,k+i} \\ z_{2,k+i} \end{bmatrix} = \begin{bmatrix} \varepsilon_{L,k+i} \\ v\sin\varepsilon_{H,k+i} \end{bmatrix}, \tag{47}$$

by substituting (45) and (46) to (47). The velocity state variables $\mathbf{v}$ represent the actual velocities of the vehicle, which are different from the commanded input variables $\mathbf{u}$ in (44). The velocity state variables $\mathbf{v}_{k-1+i} = \begin{bmatrix} v_{k-1+i}^{act} & \omega_{k-1+i}^{act} \end{bmatrix}^{\mathrm{T}}$ are calculated as

$$v_{k-1+i}^{act} = \frac{\sqrt{(x_{k+i} - x_{k-1+i})^2 + (y_{k+i} - y_{k-1+i})^2}}{T}, \tag{48}$$

$$\omega_{k-1+i}^{act} = \frac{(\theta_{k+i} - \theta_{k-1+i})}{T}. \tag{49}$$

with the estimated poses $q_{k-1} = (x_{k-1}, y_{k-1}, \theta_{k-1})$ from the navigation system in Figure 2.

We summarize the implementation of GP-FBLMPC as Algorithm 1 in a way that exploits the feedback linearized coordinates for MPC optimization and a prior kinematic nominal process model together with GP learning-based disturbance models for the prediction of future path-following errors. In this algorithm, the current vehicle pose $\mathbf{q}_k$ and path following errors ($\varepsilon_{L,k}$, $\varepsilon_{H,k}$) are used in the first step when $i = 0$.

Algorithm 1 GP-FBLMPC at time step k.

Input: $i = 0$, $\mathbf{q}_k = \mathbf{q}_{k+i}$, $\varepsilon_{L,k}$, $\varepsilon_{H,k}$, $\mathbf{u}_0 = \mathbf{0}$, $\mathbf{z}_0 = \mathbf{0}$, $\mathbf{v}_0 = \mathbf{0}$, $\mathbf{a}_0 = \mathbf{0}$

Output: ω_k^*

$\mathbf{z}_k := \begin{bmatrix} \mathbf{y}_{k,0} \\ \mathbf{y}_{k,1} \end{bmatrix} := \begin{bmatrix} \varepsilon_{L,k} \\ v\sin\varepsilon_{H,k} \end{bmatrix} = \begin{bmatrix} \mathbf{d}_{k,1} \\ \mathbf{d}_{k,2} \end{bmatrix}$

$\mathbf{a}_{k-1} := (\mathbf{z}_{k-1}, \mathbf{v}_{k-2}, \mathbf{u}_{k-1}, \mathbf{u}_{k-2}), \mathbf{g}(\mathbf{a}_{k-1}) := \begin{bmatrix} \mathbf{d}_{k,1} \\ \mathbf{d}_{k,2} \end{bmatrix}$

Save the input-output data pair $[\mathbf{a}_{k-1}, \mathbf{g}(\mathbf{a}_{k-1})]$ to the disturbance dataset $\mathcal{D}_T$

for $i = 0$ **to** $p - 1$ **do**

$\omega_{k+i} = \dfrac{u_{k+i}}{v\cos\varepsilon_{H,k+i}}$

$\mathbf{q}_{k+i+1} = \mathbf{q}_{k+i} + T \begin{bmatrix} \cos\theta_{k+i} & 0 \\ \sin\theta_{k+i} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega_{k+i} \end{bmatrix}$

Calculate current $\varepsilon_{L,k+i+1}$ and $\varepsilon_{H,k+i+1}$,
$\mathbf{z}_{k+i+1} = \begin{bmatrix} \varepsilon_{L,k+i+1} \\ v\sin\varepsilon_{H,k+i+1} \end{bmatrix}$

Make predictions by the GP models $\boldsymbol{d}_1(\cdot)$ and $\boldsymbol{d}_2(\cdot)$
at $a_{k+i} := (\mathbf{z}_{k+i}, \mathbf{v}_{k-1+i}, \mathbf{u}_{k+i}, \mathbf{u}_{k-1+i})$ as
$m(\mathbf{d}_1(a_{k+i})) = \mu_1^{\mathbf{d}}(a_{k+i})$,
$m(\mathbf{d}_2(a_{k+i})) = \mu_2^{\mathbf{d}}(a_{k+i})$

$\begin{bmatrix} \mathbf{y}_{k,2i+2} \\ \mathbf{y}_{k,2i+3} \end{bmatrix} = \mathbf{z}_{k+i+1} + \begin{bmatrix} \mu_1^{\mathbf{d}}(a_{k+i}) \\ \mu_2^{\mathbf{d}}(a_{k+i}) \end{bmatrix}$

end for

Calculate $\Delta\mathbf{u}_k^*$ using (17)

$\mathbf{u}_k^* = \Delta\mathbf{u}_k^* + \mathbf{u}_{k-1}$

$\omega_k^* = \dfrac{u_k^*}{v\cos\varepsilon_{H,k}}$

Constrain ω_k^* to ± 2 rad/s for maximum turning rate

return ω_k^*

Two GP models $\mathbf{d}_1$ and $\mathbf{d}_2$ are estimated by the disturbance dataset $\mathcal{D}_T$ when a trial is finished.

5 | FIELD TESTING

The proposed GP-FBLMPC algorithm was tested in the field via five experiments with three different paths (Figure 5) in sand and grass terrains. Experiment 1 showed the GP-FBLMPC was able to reduce the lateral and heading path-following errors by 85.31% and 59.2%, respectively, than the FBLMPC. Experiment 2 demonstrated that GP models trained by a small number of data (collected in one experiment trial) worked comparably well as GP models estimated with data from multiple trials. Experiment 3 showed the GP-FBLMPC performed equally well as a comparable GP-NMPC (Ostafew et al., 2016) with respect to the path-following errors and is more computationally efficient. Experiment 4 highlighted the GP-FBLMPC algorithm was generalizable to reduce path-following errors for different paths because of combing GP and FBL. Experiment 5 further showed the generalizability of the GP-FBLMPC algorithm to reduce path-following errors for different paths on a complex path, and how simple it is to apply the GP-FBLMPC algorithm in a new environment.

Through extensive field tests in this section, we conclude that by combining FBL with MPC, the GP-FBLMPC was more computationally efficient and simpler to implement than a comparable GP-NMPC. By combining GP and FBL with MPC, the GP-FBLMPC was generalizable to reduce path-following errors on different paths of the same terrain. This is a significant improvement to the GP-NMPC considering the GP-NMPC is only able to reduce path-following errors on the same path. The generalizability feature also makes the GP-FBLMPC easier to apply in a new environment.

5.1 | Overview

The robot used in all the field experiments was a skid-steer Clearpath Husky A200 mobile robot (Figure 1). The computer used to operate the vehicle was a Dell XPS 15 laptop with an eight-core 2.80 GHz Intel Core i7-7700HQ processor and 12 GB of RAM that was running ROS Kinetic on an Ubuntu 16.04 operating system. The outdoor navigation was accomplished using a LORD microstrain 3DM-GX5-25 IMU (gyroscope) and a Swift Navigation Duro GPS with a real-time kinematic (RTK) base station with a Novatel FlexPak6 GPS receiver shown in Figure 1. The vehicle guidance and control systems were written in Python 2.7, the outdoor navigation used the open-source ROS package *robot_localization* to fuse GPS and IMU data for state estimation. The navigation provided

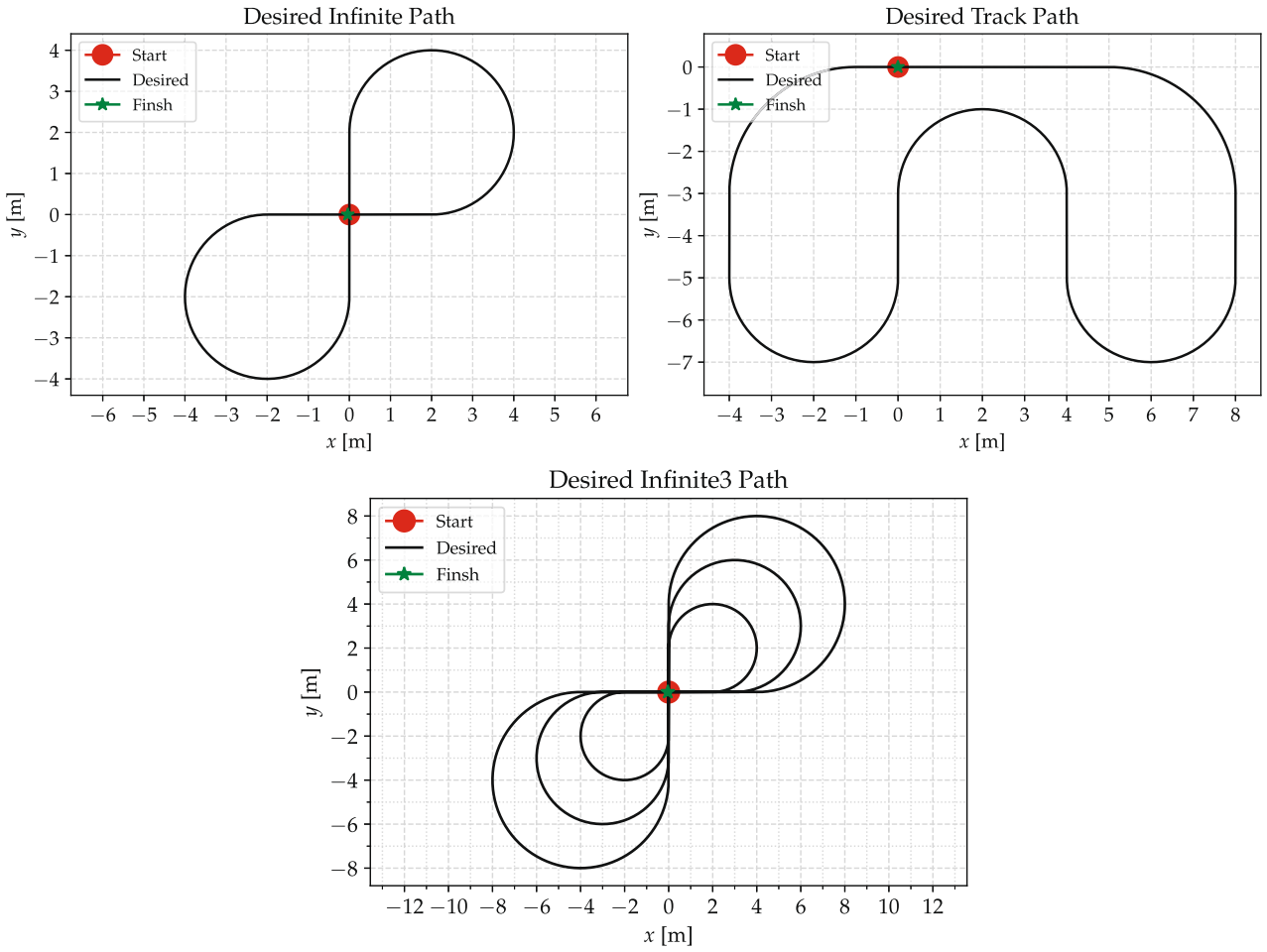


FIGURE 5 Three paths, *infinite*, *track*, and *infinite3*, were designed and used in the field experiments for path-following experiments.

position accuracy of approximately ±1 cm. The *robot_localization* extended Kalman filter (EKF) module operated at 30 Hz, while all path following control operated at 10 Hz.

Three paths, called *infinite*, *track*, and *infinite3* were designed as desired paths for the field experiments shown in Figure 5. Each path consisted of a sequence of evenly and closely spaced (0.05 m apart) discrete waypoints that specify the desired vehicle pose. In each path, starting at the red circle and orienting to the right, the Husky A200 robot drove autonomously to follow the path and finished at the green star.

FIGURE 6 The Husky A200 robot is driven autonomously to follow the infinite path in sand terrain. The desired infinite path (black-dot line) was blended to the field test scene image taken by a DJI Mini.

In all of the field experiments, the Husky robot was commanded to drive at a constant desired forward speed at 0.9 m/s. The speed was chosen mainly because that path following tasks at 0.9 m/s was more challenging for the FBLMPC algorithm compared to low-speed (0.5 m/s) applications shown in Fader (2020). However, in all experiment tests, the robot did not achieve the commanded forward speed most of the time due to this speed being near its mechanical limit. The actual average forward speed of the robot was 0.70 and 0.75 m/s using FBLMPC and GP-FBLMPC respectively. More discussions about this are provided in Figure 10 at Section 5.2.

The FBLMPC was tuned to give a desirable path following behavior before integrating GP models. Good results were achieved with the prediction horizon $p = 10$, and weighting matrix $\mathbf{Q} = 5.0 \times \mathbf{I}_{2p\times 2p}$, $\mathbf{R} = \mathbf{I}_{p\times p}$ in (16). When running NMPC, the number of optimization iterations was limited to allow the controller to complete its calculations within 0.1 s. It was found that the NMPC performed best with $p = 20$ and six iterations and the optimal tuning

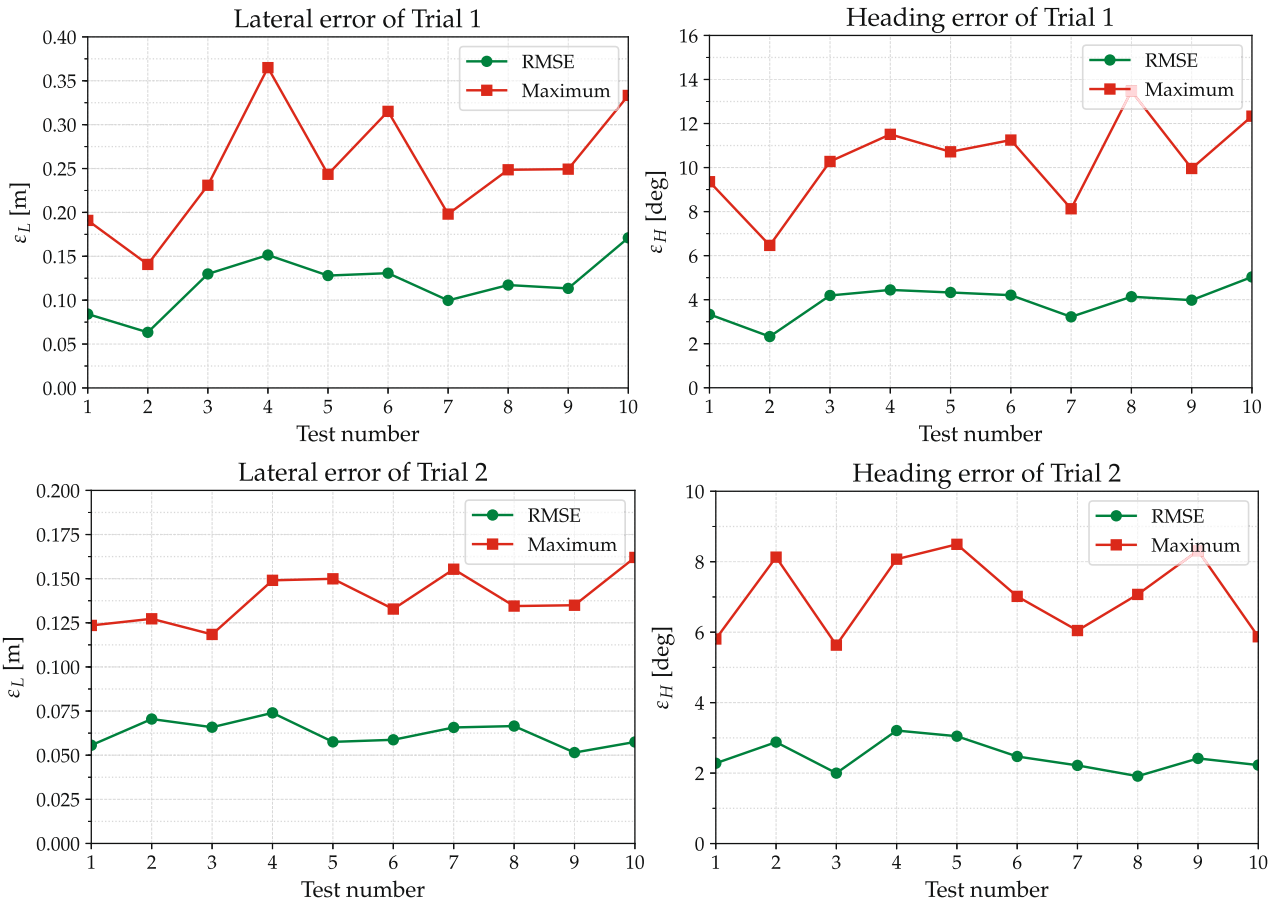


FIGURE 7 The GP-FBLMPC algorithm was tested on the infinite path in sand terrains. The maximum and RMSE of the lateral and heading errors versus the test number in trial 1 and trial 2 were plotted. GP, Gaussian processes; FBL, feedback linearization; MPC, model predictive control.

was $\mathbf{Q} = \mathbf{I}_{3p\times 3p}$ and $\mathbf{R} = 4.0 \times \mathbf{I}_{2p\times 2p}$. The complete controller parameters tuning process can be found in (Fader, 2020, sec. 4.4).

5.2 | Experiment 1: GP-FBLMPC compared with FBLMPC

In the first experiment, we compared the path following performance of the proposed GP-FBLMPC algorithm and the FBLMPC algorithm. The robot was commanded to follow the infinite path in a sand terrain shown in Figure 6 at a constant forward speed at 0.9 m/s. In trial 1, the GP-FBLMPC was the same as FBLMPC. In trial 2, FBLMPC was integrated with GP models trained by the observation data collected in trial 1. In trials 1 and 2, we repeated 10 tests, respectively. To quantify the path following performance for each test in each trial, the root mean square error (RMSE) was calculated for each lateral and heading path following error using

$$\mathrm{RMSE}_{\varepsilon^*} = \sqrt{\sum_{i=1}^{n} \frac{(\varepsilon_d^* - \varepsilon^*)^2}{n}}, \tag{50}$$

where ε^* represents either the lateral error ε_L or heading error ε_H over n samples during a test. Note that ε_d^* represents the desired path following errors which are $\varepsilon_{L_d} = \varepsilon_{H_d} = 0$ at all times. The RMSE calculates the mean value of the lateral and heading errors over a path-following test, thus lower RMSE values indicate that the vehicle follows more closely to the desired path over the length of the test. Another performance metric is the absolute value of the maximum

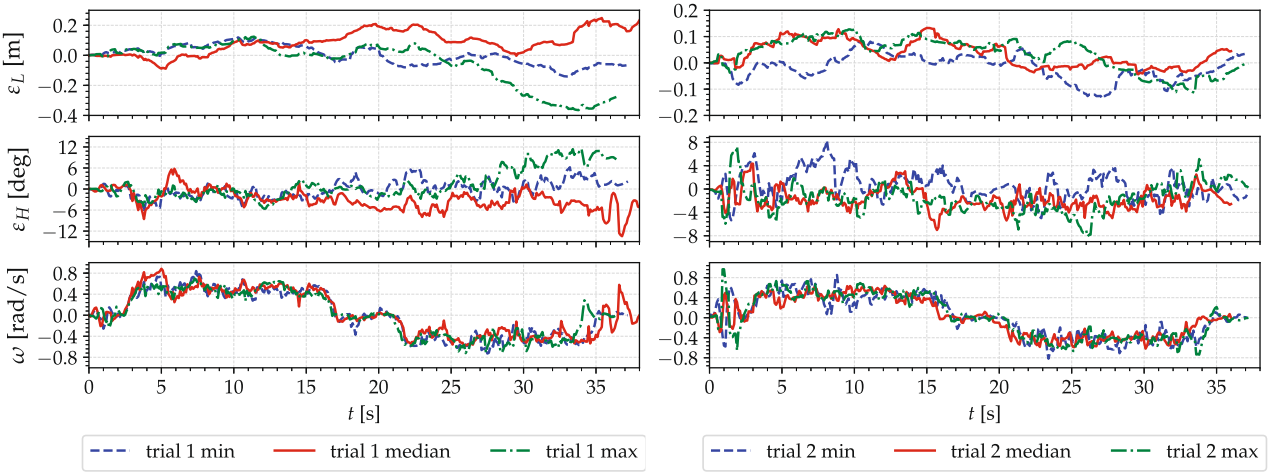


FIGURE 8 Path following errors and commanded steering inputs versus travel time of the minimum, median, and maximum of the nine tests in trials 1 and 2, respectively, for GP-FBLMPC on the infinite path. GP, Gaussian processes; FBL, feedback linearization; MPC, model predictive control.

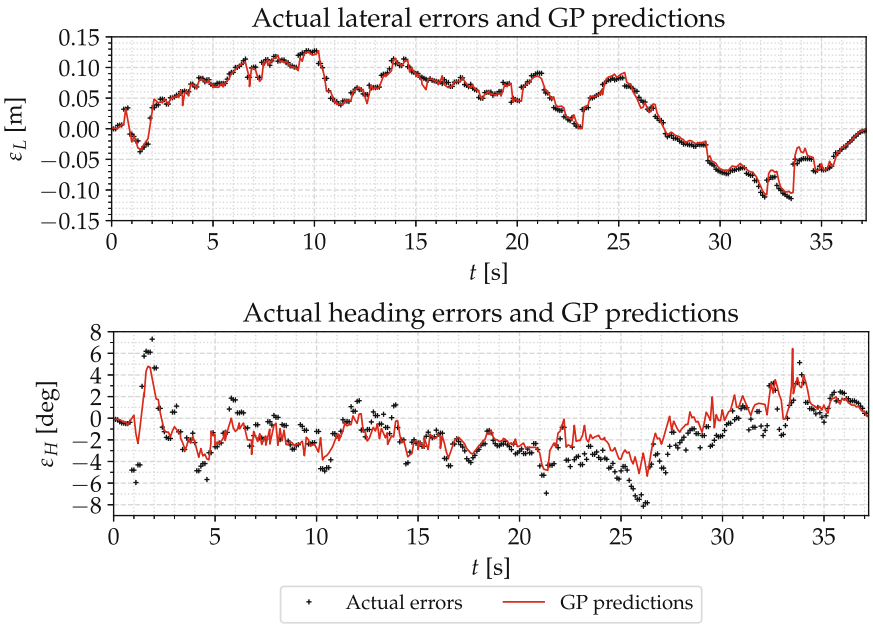


FIGURE 9 The lateral and heading errors of the GP model predictions and the actual errors versus travel time in trial 2 of the GP-FBLMPC on the infinite path. GP, Gaussian processes; FBL, feedback linearization; MPC, model predictive control.

lateral and heading error. This metric is less indicative of the overall path-following performance but it can highlight the largest path-following errors in a given test. Similar to the RMSE values, the lower the maximum errors, the better the performance.

Figure 7 shows RMSE and a maximum of the lateral and heading errors, ε_L and ε_H versus the test number in trials 1 and 2, respectively. Note how the lateral and heading errors decreased over the course of the 10 tests in trial 2. We calculated the mean values of the RMSE (the sum of all RMSE values in each trial divided by the total test number) to reflect the overall performance in each trial. There was an 85.31% lower lateral error mean and 59.2% lower heading error mean in trial 2 compared with trial 1. We sorted the 10 tests by the RMSE of the lateral errors in trials 1 and 2, and treated the test with the highest RMSE lateral error as an outlier in each trial. The reason for doing this is to obtain an odd number of tests such that we can identify the test number with the median result. As shown in Figure 7, we did not delete the one with the highest RMSE lateral error and plotted all 10 test results.

Figure 8 plotted the path following errors (lateral and heading) and the commanded steering inputs versus travel time of the minimum, median, and maximum ones of the remaining nine tests in trials 1 and 2, respectively. It can also be seen that the overall magnitude of the lateral and heading errors ε_L and ε_H were smaller in trial 2.

Figure 9 shows the GP predictions and the actual lateral and heading errors of trial 2 (the test with maximum RMSE lateral error among the nine tests) versus travel time. Note the correspondence between the GP model predictions of the lateral errors and the actual lateral errors. Even when there were more deviations between the predictive and actual heading errors, the estimated GP model of the heading error continued to provide helpful mean predictions of the actual heading errors. Together with the significantly lower path following error results in trial 2 (Figure 7), we can conclude that the estimated GP models usefully learned unmodeled dynamics and were capable of providing predictions that are practically close to the actual path following errors.

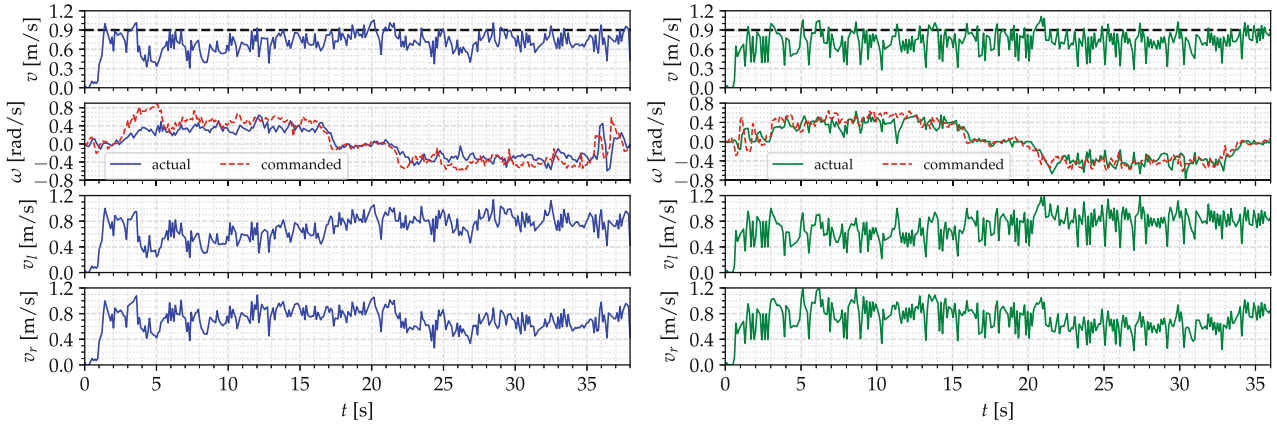


FIGURE 10 Actual linear and angular velocity of the robot together with the linear velocity of the left and right wheels of trial 1 (left) and trial 2 (right) versus travel time were plotted.

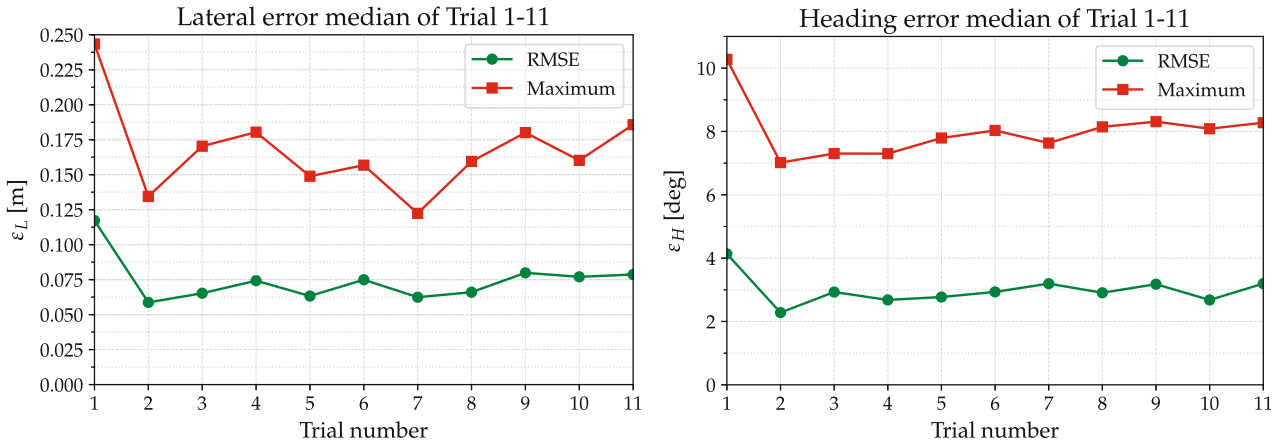


FIGURE 11 The maximum and RMSE path following errors versus trial number of the GP-FBLMPC algorithm tested on the infinite path were plotted. In each trial, the plotted dot was the one with the median RMSE lateral error among nine tests to demonstrate the "typical" performance. GP, Gaussian processes; FBL, feedback linearization; MPC, model predictive control; RMSE, root mean square error.

In Figure 10, we plotted the actual linear and angular velocity of the robot (v and ω, respectively) together with the linear velocity of the left and right wheels (v_l and v_r, respectively) of trials 1 and 2 (the test with the median RMSE of lateral error among the nine tests). The commanded steering inputs shown in Figure 8 were also plotted together with the actual angular velocity. In trial 1 (left plots) and trial 2 (right plots), it can be seen that the robot was only traveling near the commanded linear velocity (v = 0.9 m/s) when it is not turning (approximately 1–3, 17–22, and 36–38 s). This limitation is because the top speed of the Husky robot is (approximate) 1 m/s and it is challenging to maintain a high forward speed when near this mechanical limit, even though we can see that the left and right wheels were able to move at more than 1 m/s occasionally. The average forward speed of the robot was 0.70 and 0.75 m/s in trials 1 and 2, respectively. In trial 1 using the FBLMPC without GP models, we can also see that the actual angular velocity of the robot did not achieve the commanded angular velocity well. This is most obvious when high vehicle yaw rates were required (approximately 3–5 s). In trial 2 using the GP-FBLMPC, it is clear that the actual angular velocity of the robot followed the commanded angular velocity better than the FBLMPC in Trial 1. Together with Figures 7 and 9, this indicates that the estimated GP models were capable to learn unmodeled dynamics effectively including those due to the systems near their mechanical limit.

5.3 | Experiment 2: GP model training analysis

In the second experiment, we continued to test the GP-FBLMPC algorithm on the infinite path (Figure 6) over nine more field trials. We observed that the GP models trained by one trial data (trial 2) worked equally well to the GP models estimated with multiple trials data (trial 3–11). In trial 3–11, like what we did in trials 1 and 2 in Section 5.2, 10 tests were repeated in each trial. Similar to the data processing in trials 1 and 2, we selected the test with median RMSE lateral error in each trial to demonstrate their "typical" performance. Figure 11 plotted the maximum and RMSE path following errors (lateral and heading) of the median test in each trial versus trial number. Similar to the mean of path-following errors (Figure 7), the median of path-following errors also went down in trial 2. The lateral and heading errors were reduced by roughly 80% and 60%, respectively, starting in trial 2 and over the course of the subsequent trials.

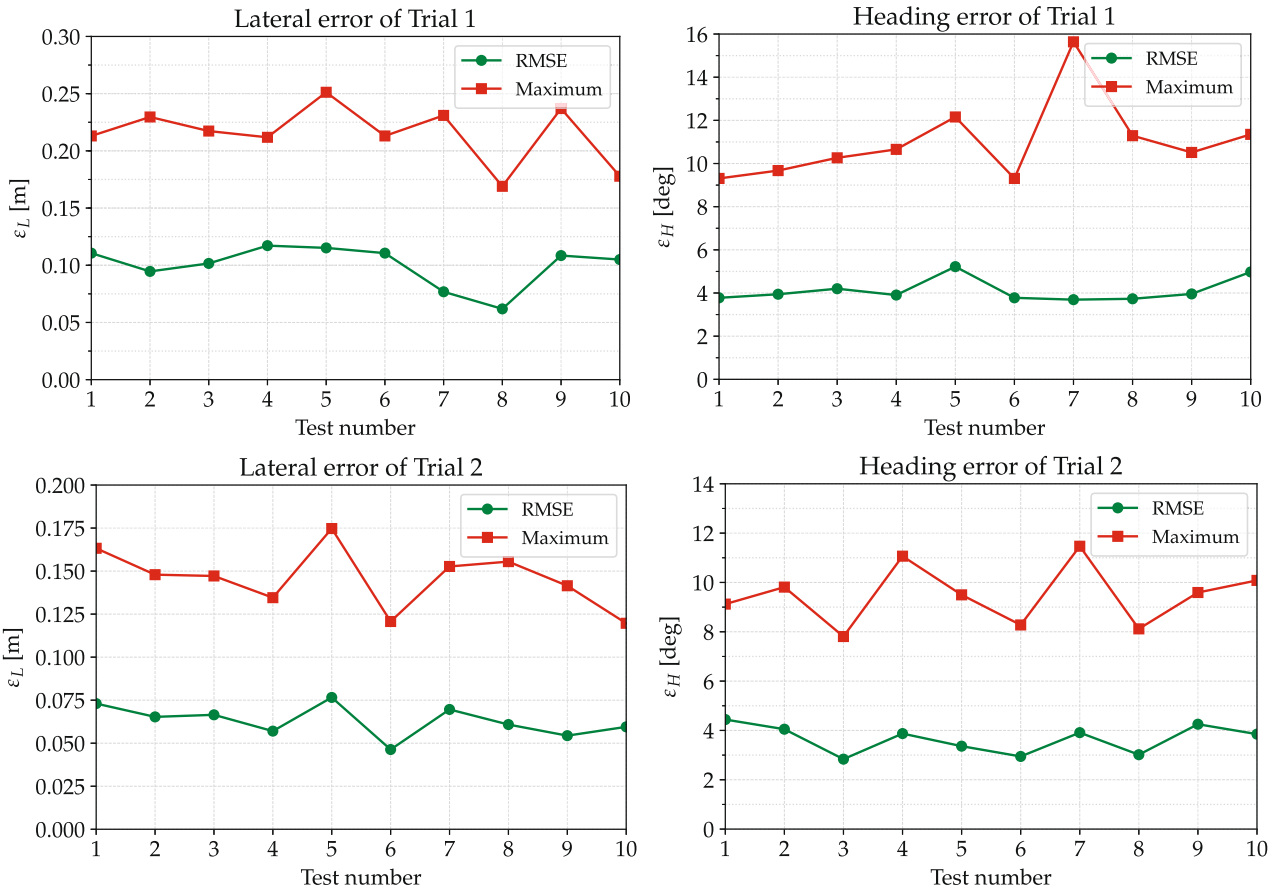


FIGURE 12 The GP-NMPC algorithm was tested on the infinite path in sand terrain. The maximum and RMSE of the lateral and heading errors versus the test number in trials 1 and 2 were plotted. This result is compared to the path following the performance of the GP-NMPC to the GP-FBLMPC in Figure 7. GP, Gaussian processes; FBL, feedback linearization; MPC, model predictive control; NMPC, nonlinear MPC; RMSE, root mean square error.

There were no discernible improvements in the RMSE path following errors in trials 3–11 compared to trial 2. In fact, it can be seen that the maximum and RMSE path following errors varied over the course of trials 2–11, a phenomenon that was similarly observed in Ostafew et al. (2016).

FIGURE 13 In Experiment 4, the Husky A200 robot drove autonomously to follow the track path in sand terrain. The desired track path (black-dot line) was blended to the field test scene image taken by a DJI Mini.

5.4 | Experiment 3: GP-FBLMPC and GP-NMPC comparisons

In this experiment, GP-FBLMPC and GP-NMPC performance were compared with respect to path-following errors and computational time. For the GP-NMPC, the robot followed the infinite path in sand terrain shown in Figure 6 for 10 tests in trials 1 and 2. Figure 12 plotted the maximum and RMSE of the lateral and heading errors versus the test number in trials 1 and 2, respectively. In trial 1, compared to the path following errors of the FBLMPC shown in Figure 7, the NMPC behaved slightly better. The mean of the RMSE lateral errors of the NMPC is 15.1% lower than the FBLMPC, while the FBLMPC had a 5.3% lower heading error mean than the NMPC. After integrating estimated GP models in trial 2, as shown in Figures 7 and 12, the GP-FBLMPC performed similarly to the GP-NMPC. The GP-FBLMPC had a 0.6% lower lateral error mean and 49.6% lower heading error means than the GP-NMPC.

The average time of GP-FBLMPC and GP-NMPC to calculate a single control input is approximately the sum of the MPC computation time and GP models prediction time at each time step. The FBLMPC algorithm is more computationally efficient than the NMPC because it does not require iterative optimization (Fader, 2020). In

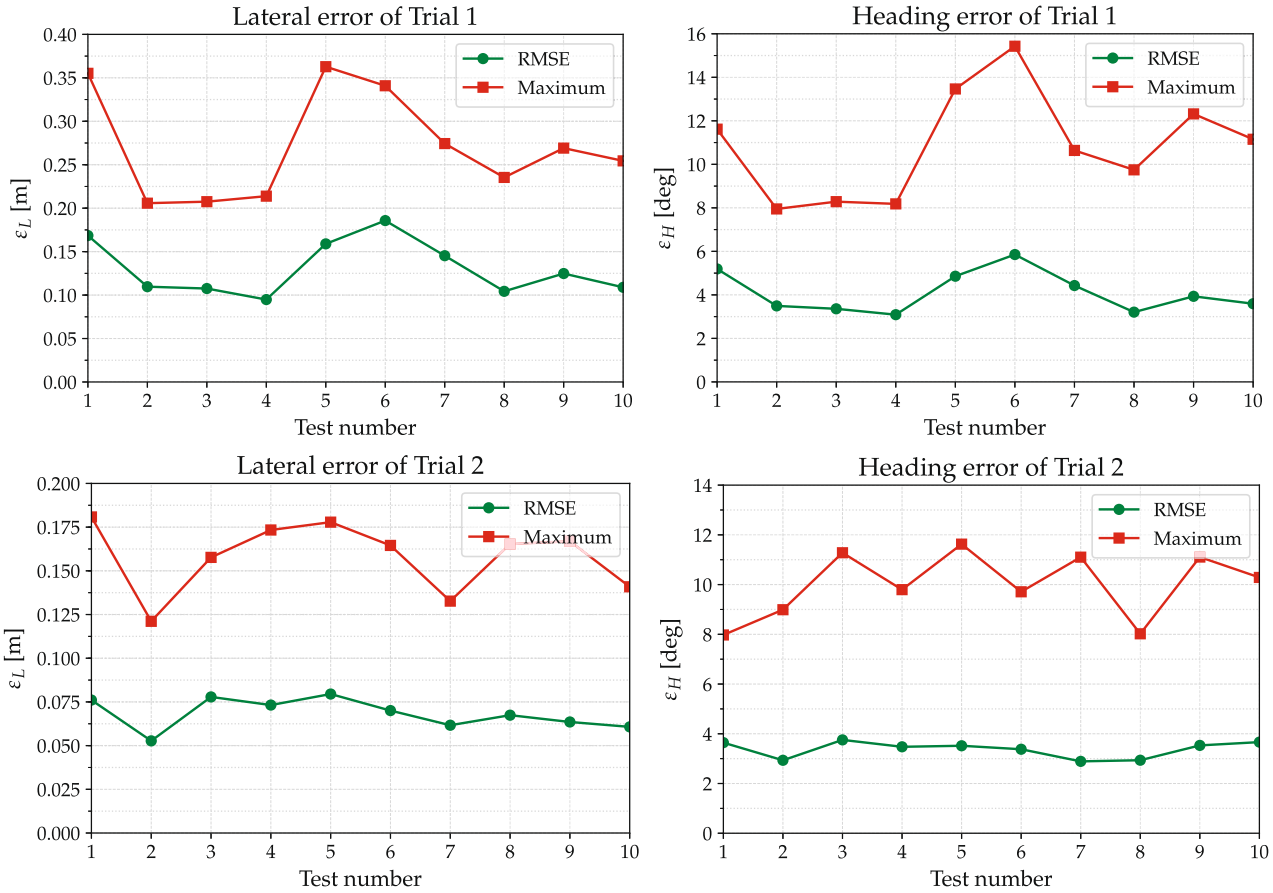


FIGURE 14 The GP-FBLMPC algorithm was tested on the track path in sand terrains. The maximum and RMSE of the lateral and heading errors versus the test number in trial 1 and trial 2 were plotted. GP, Gaussian processes; FBL, feedback linearization; MPC, model predictive control; RMSE, root mean square error.

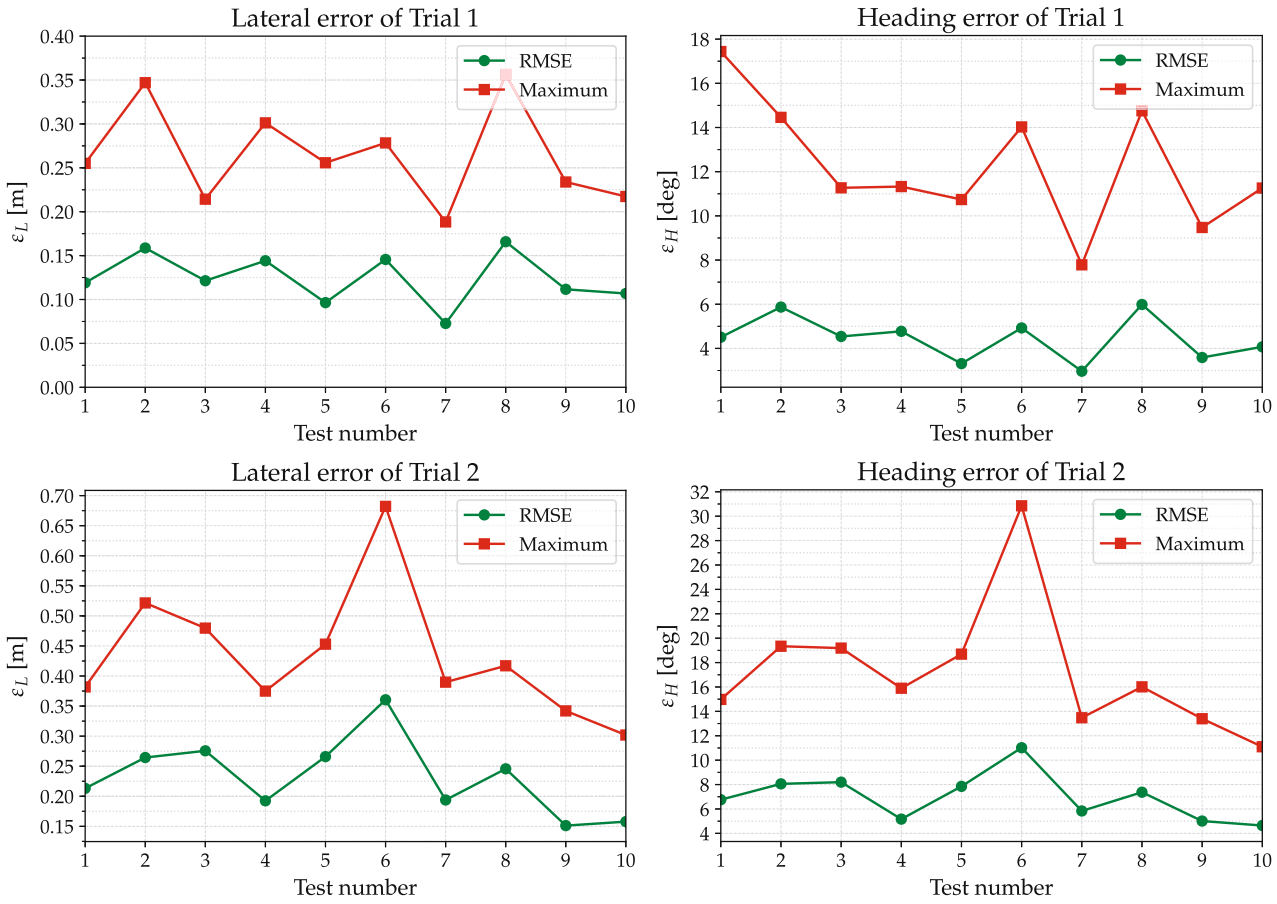
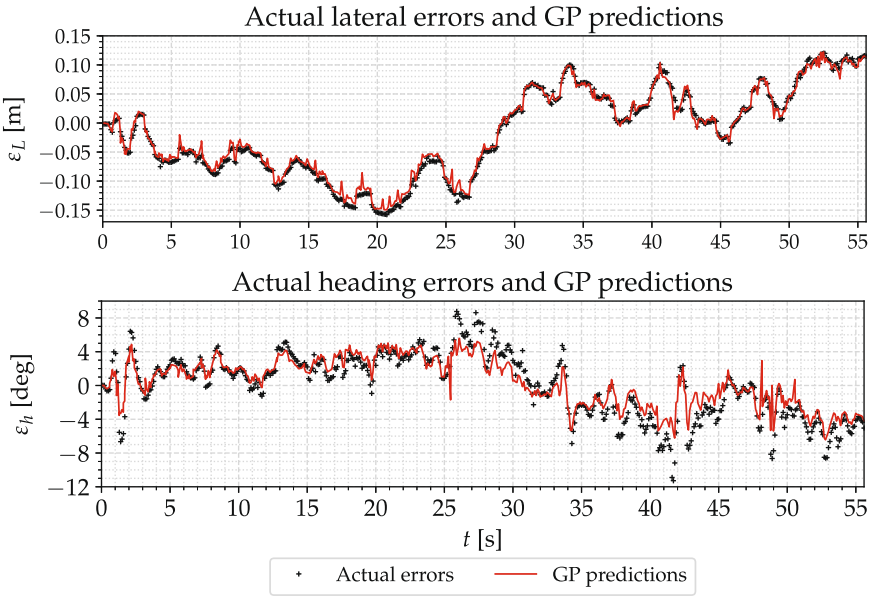


FIGURE 15 The GP-NMPC algorithm was tested on the track path in sand terrains. The maximum and RMSE of the lateral and heading errors versus the test number in trial 1 and trial 2 were plotted. GP, Gaussian processes; NMPC, nonlinear model predictive control; RMSE, root mean square error.

FIGURE 16 The lateral and heading errors of the GP model predictions and the actual errors versus travel time in trial 2 for the GP-FBLMPC on the track path. GP, Gaussian processes; FBL, feedback linearization; MPC, model predictive control.



trial 2, the GP model prediction time of GP-FBLMPC and GP-NMPC was 0.0016 and 0.0022 s, respectively. In the subsequent trials, the GP prediction time grew as more observations of disturbance data were used for the GP model estimation. In trial 7, for example, the average GP model prediction time of the GP-NMPC was 0.0346 s, while the averaged prediction time of the GP-FBLMPC algorithm was 0.0233 s. The averaged GP models prediction time in GP-FBLMPC was approximately 2/3 of the time in GP-NMPC for each trial. This is because at each time step, there were two GP models to make predictions in the GP-FBLMPC, but three for GP-NMPC.

In summary, the GP-FBLMPC performed similarly to the GP-NMPC algorithm with respect to the path following errors. However, the GP-FBLMPC is more computationally efficient than the GP-NMPC because it does not require iterative optimization and requires fewer GP models to make predictions.

FIGURE 17 The Husky A200 robot drove autonomously to follow the infinite3 path in a grass terrain. The desired infinite3 path (black-dot line) was blended to the field test scene image taken by a DJI Mini.

5.5 | Experiment 4: GP-FBLMPC generalizability

The Husky A200 robot was tested on the track path in a sand terrain (Figure 13) to test the generalizability of the GP-FBLMPC, checking whether GP models of the GP-FBLMPC estimated on one path are able to reduce the path following errors for other paths. Instead of estimating new GP models with observation data of the track path in trial 1, we integrated the GP models trained by the infinite path data. In trial 1, the maximum and RMSE lateral and heading errors of GP-FBLMPC and GP-NMPC versus test number are plotted in Figures 14 and 15 respectively. The mean of the GP-NMPC lateral errors was 4.31% lower than the GP-FBLMPC, while the GP-FBLMPC had a 9.7% lower heading error mean than the GP-NMPC.

In trial 2, the GP-FBLMPC integrated GP models are estimated in Section 5.2. Figure 14 shows the GP-FBLMPC gained 86.14% lower lateral error mean and 17.29% lower heading error mean compared to trial 1. On the contrary, after integrating GP models estimated in Section 5.4, the GP-NMPC had a worse performance than trial 1, as shown in Figure 15. Figure 16 shows GP predictions and actual lateral and heading errors of the GP-FBLMPC in trial 2. As explained in Section 4, GP models of the GP-FBLMPC and GP-NMPC were estimated by the observation data with the FBL states $\mathbf{z}$ and robot poses (x, y, θ) as inputs, respectively. Thus, we observed that the GP-FBLMPC algorithm was generally capable of reducing path-following errors for different paths.

5.6 | Experiment 5: GP-FBLMPC follows a complex path in a new environment

We further demonstrated the generalizability of the GP-FBLMPC algorithm on a more complex infinite3 path (Figure 5) in a grass

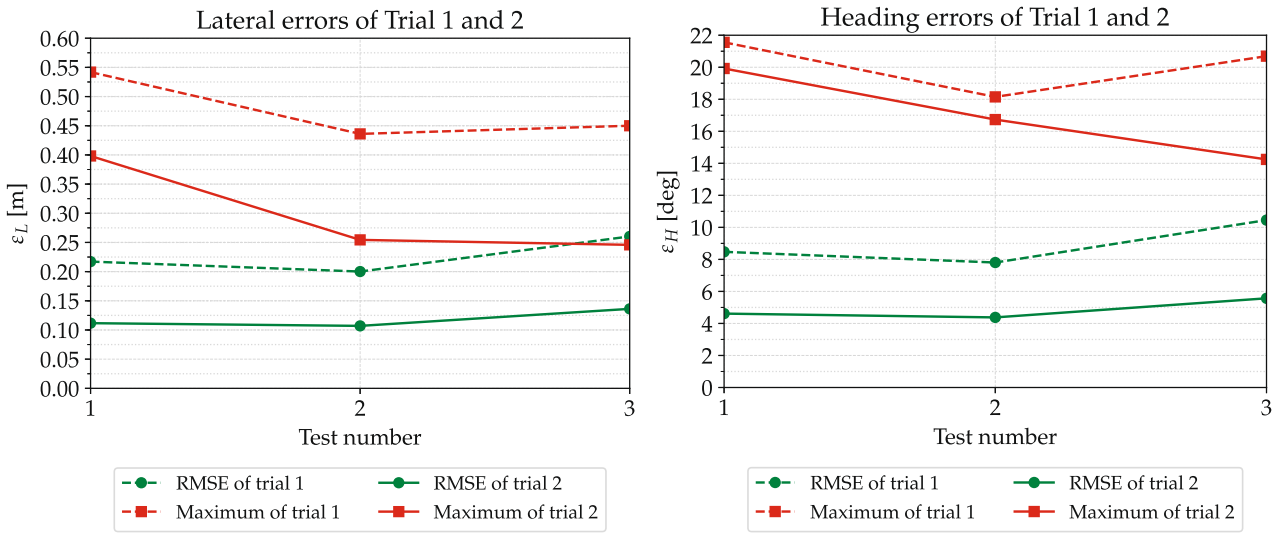


FIGURE 18 The GP-FBLMPC algorithm was tested on the infinite3 path in grass terrains. The maximum and RMSE of the path following errors versus the test number in trial 1 and trial 2 were plotted. GP, Gaussian processes; FBL, feedback linearization; MPC, model predictive control; RMSE, root mean square error.

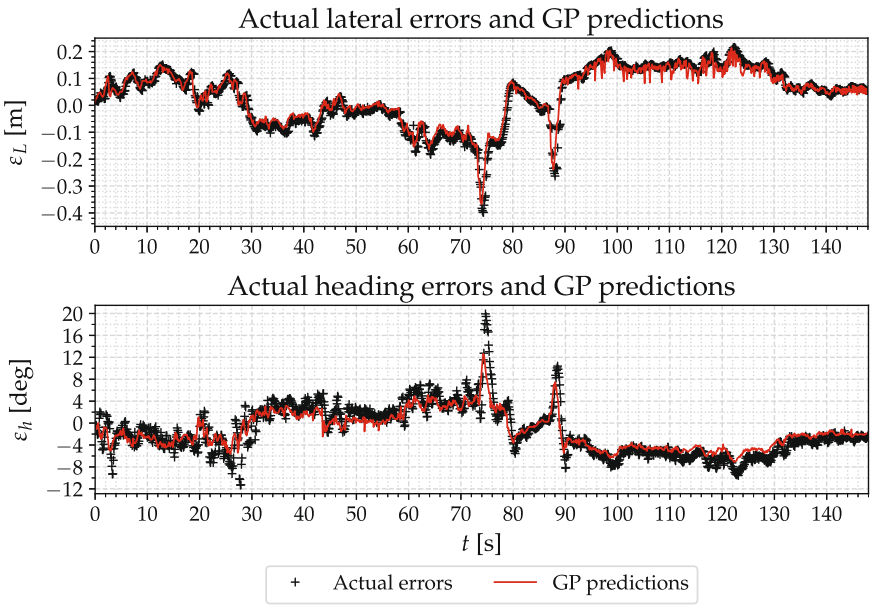


FIGURE 19 GP-FBLMPC lateral and heading errors of the GP model predictions and actual errors versus travel time in trial 2 on the infinite3 path. GP, Gaussian processes; FBL, feedback linearization; MPC, model predictive control.

terrain shown in Figure 17. To apply the GP-FBLMPC algorithm in a new environment, we first need to collect disturbance observation data for training GP models. As discussed in Section 5.3, GP models can be easily estimated with the data collected in one experiment trial by driving the robot on a simple path (the infinite path). The trained GP models can be used in GP-FBLMPC to reduce path-following errors for different more complex paths.

In trial 1, the robot was commanded to follow the infinite3 path by the FBLMPC. The estimated GP models were integrated into the GP-FBLMPC in trial 2. Three tests were repeated in trials 1 and 2, the maximum and RMSE lateral and heading errors of trials 1 and 2 versus test number were plotted shown in Figure 18. In trial 2, there was a 90.94% lower lateral error mean and 39.63% lower heading error mean than in trial 1. Figure 19 showed the GP model predictions together with the actual lateral and heading errors of the one with the maximum RMSE lateral error mean among the three tests. It can be seen that the predictions of GP disturbance models were close to the actual path following errors. The generalizability of the GP-FBLMPC algorithm to reduce path-following errors was validated on the challenging infinite3 path.

6 | CONCLUSIONS AND FUTURE WORK

6.1 | Summary

This paper presents a GP-FBLMPC algorithm for path-following control of ground mobile robots operating in off-road terrains. The GP-FBLMPC algorithm uses a fixed unicycle kinematic model and GP models to learn the unmodeled dynamics from data collected in the field. Extensive outdoor tests of five experiments on the Husky A200 mobile robot demonstrate the GP-FBLMPC algorithm is able to reduce the path following errors effectively compared with controllers that do not use GP modeling. When compared to a similar GP-NMPC algorithm (Ostafew et al., 2016), the GP-FBLMPC algorithm behaves equally well with respect to path-following performance, but with a much higher computational efficiency and is generalizable to reduce path-following errors for different paths.

6.2 | Future directions

Even from an application viewpoint, combining GP and FBL with MPC achieves high-performance path-following behaviors with appealing features, there are several important issues that deserve further research. In the current work, the requirement of offline GP model estimation due to the usage of a full disturbance data set can limit the application of the proposed approach in new unknown environments. Recent studies tackle the challenge by using a small number of inducing points to train GP models online (Kabzan et al., 2019; Torrente et al., 2021). Moreover, the current implementation only used the predicted mean values of the GP models, and we did not consider uncertainty propagation in the GP model approximations. A GP model could become over-confident in the multiple-step ahead predictions, which might result in over-optimistic MPC solutions (Nghiem et al., 2019).

Another important issue is we focused on the field application of the GP-FBLMPC algorithm to mobile robots without considering guarantees on desirable closed-loop properties. Even GP learning-based MPC has received increasing attraction for the past decades (Brunke et al., 2021; Hewing et al., 2020), it remains an open issue to proving desirable closed-loop properties (Berberich et al., 2020). In fact, no learning-based or adaptive MPC framework has been presented in the literature that ensures recursive feasibility and stability for systems in state space under probabilistic constraints (Bujarbaruah et al., 2020). Providing theoretical analysis of

guarantees on desirable closed-loop properties can be significant to safety-critical system applications.

ACKNOWLEDGMENTS

This research was funded by the Natural Sciences and Engineering Research Council of Canada (NSERC) through the NSERC Canadian Robotics Network (NCRN) under grant NETGP 508451-17.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are openly available in GP-FBLMPC (to-be-available) at http://jiewang.name/publications/GP-FBLMPC2022/.

ORCID

Jie Wang http://orcid.org/0000-0001-9970-1808
Michael T. H. Fader http://orcid.org/0000-0001-9217-3096
Joshua A. Marshall http://orcid.org/0000-0002-7736-7981

REFERENCES

Amer, N.H., Zamzuri, H., Hudha, K. & Kadir, Z.A. (2016) Modelling and control strategies in path tracking control for autonomous ground vehicles: a review of state of the art and challenges. *Journal of Intelligent & Robotic Systems*, 86, 225–254.

Berberich, J., Köhler, J., Müller, M.A. & Allgöwer, F. (2020) Data-driven model predictive control with stability and robustness guarantees. *IEEE Transactions on Automatic Control*, 66(4), 1702–1717.

Brunke, L., Greeff, M., Hall, A.W., Yuan, Z., Zhou, S., Panerati, J. & Schoellig, A.P. (2021) Safe learning in robotics: from learning-based control to safe reinforcement learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 5, 411–444.

Bujarbaruah, M., Zhang, X., Tanaskovic, M. & Borrelli, F. (2020) Adaptive stochastic MPC under time-varying uncertainty. *IEEE Transactions on Automatic Control*, 66(6), 2840–2845.

Caldwell, J. & Marshall, J.A. (2021) Towards efficient learning-based model predictive control via feedback linearization and gaussian process regression. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 4306–4311.

Dekker, L.G., Marshall, J.A. & Larsson, J. (2017) Industrial-scale autonomous wheeled-vehicle path following by combining iterative learning control with feedback linearization. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 2643–2648.

Fader, M.T. (2020) Autonomous ground vehicle path following by combining feedback linearization with model predictive control. Master's thesis, Queen's University.

Girard, A., Rasmussen, C.E., Quiñonero-Candela, J. & Murray-Smith, R. (2003) Gaussian process priors with uncertain inputs–application to multiple-step ahead time series forecasting. *Advances in Neural Information Processing Systems*, 15, 529–536.

Greeff, M. & Schoellig, A.P. (2020) Exploiting differential flatness for robust learning-based tracking control using Gaussian processes. *IEEE Control Systems Letters*, 5(4), 1121–1126.

Greeff, M., Zhou, S. & Schoellig, A. P. (2022) Fly out the window: exploiting discrete-time flatness for fast vision-based multirotor flight. *IEEE Robotics and Automation Letters*, 7(2), 5023–5030.

Helwa, M.K., Heins, A. & Schoellig, A.P. (2019) Provably robust learning-based approach for high-accuracy tracking control of lagrangian systems. *IEEE Robotics and Automation Letters*, 4(2), 1587–1594.

Hewing, L., Kabzan, J. & Zeilinger, M.N. (2019) Cautious model predictive control using Gaussian process regression. *IEEE Transactions on Control Systems Technology*, 28(6), 2736–2743.

Hewing, L., Liniger, A. & Zeilinger, M.N. (2018) Cautious NMPC with Gaussian process dynamics for autonomous miniature race cars. In: *Proceedings of the European Control Conference*. IEEE, pp. 1341–1348.

Hewing, L., Wabersich, K.P., Menner, M. & Zeilinger, M.N. (2020) Learning-based model predictive control: toward safe learning in control. *Annual Review of Control, Robotics, and Autonomous Systems*, 3, 269–296.

Iagnemma, K., Kang, S., Shibly, H. & Dubowsky, S. (2004) Online terrain parameter estimation for wheeled mobile robots with application to planetary rovers. *IEEE Transactions on Robotics*, 20(5), 921–927.

Kabzan, J., Hewing, L., Liniger, A. & Zeilinger, M.N. (2019) Learning-based model predictive control for autonomous racing. *IEEE Robotics and Automation Letters*, 4(4), 3363–3370.

Nghiem, T.X., Nguyen, T.-D. & Le, V.-A. (2019) Fast Gaussian process based model predictive control with uncertainty propagation. In: *Proceedings of the Annual Allerton Conference on Communication, Control, and Computing*. IEEE, pp. 1052–1059.

Oriolo, G., De Luca, A. & Vendittelli, M. (2002) WMR control via dynamic feedback linearization: design, implementation, and experimental validation. *IEEE Transactions on Control Systems Technology*, 10(6), 835–852.

Ostafew, C.J., Schoellig, A.P., Barfoot, T.D. & Collier, J. (2016) Learning-based nonlinear model predictive control to improve vision-based mobile robot path tracking. *Journal of Field Robotics*, 33(1), 133–152.

Polymenakos, K., Laurenti, L., Patane, A., Calliess, J.-P., Cardelli, L., Kwiatkowska, M., Abate, A. & Roberts, S. (2020) Safety guarantees for iterative predictions with Gaussian processes. In: *Proceedings of the IEEE Conference on Decision and Control*. IEEE, pp. 3187–3193.

Prado, Á.J., Torres-Torriti, M., Yuz, J. & Cheein, F.A. (2020) Tube-based nonlinear model predictive control for autonomous skid-steer mobile robots with tire-terrain interactions. *Control Engineering Practice*, 101, 104451.

Primbs, J.A. & Nevistic, V. (1997) MPC extensions to feedback linearizable systems. In: *Proceedings of the 1997 American Control Conference (Cat. No. 97CH36041)*. IEEE, vol. 3, pp. 2073–2077.

Rasmussen, C.E. & Williams, C.K.I. (2006) *Gaussian processes in machine learning*. MIT Press.

Slotine, J.-J. & Li, W. (1991) *Applied nonlinear control*. Prentice Hall Englewood Cliffs, NJ.

Spitzer, A. & Michael, N. (2021) Feedback linearization for quadrotors with a learned acceleration error model. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. IEEE, pp. 1050–4729.

Tang, L., Yan, F., Zou, B., Wang, K. & Lv, C. (2020) An improved kinematic model predictive control for high-speed path tracking of autonomous vehicles. *IEEE Access*, 8, 51400–51413.

Torrente, G., Kaufmann, E., Föhn, P. & Scaramuzza, D. (2021) Data-driven MPC for quadrotors. *IEEE Robotics and Automation Letters*, 6(2), 3769–3776.

Umlauft, J., Beckers, T., Kimmel, M. & Hirche, S. (2017) Feedback linearization using Gaussian processes. In: *Proceedings of the Annual Conference on Decision and Control*. IEEE, pp. 5249–5255.

Vinogradska, J., Bischoff, B., Achterhold, J., Koller, T. & Peters, J. (2018) Numerical quadrature for probabilistic policy search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(1), 164–175.

Wang, J. (2020) An intuitive tutorial to Gaussian processes regression. *arXiv preprint arXiv:2009.10862*.

Westenbroek, T., Fridovich-Keil, D., Mazumdar, E., Arora, S., Prabhu, V., Sastry, S.S. & Tomlin, C.J. (2020) Feedback linearization for uncertain systems via reinforcement learning. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. IEEE, pp. 1364–1371.

Wu, Z., Yang, R., Zheng, L. & Cheng, H. (2022) Safe learning-based feedback linearization tracking control for nonlinear system with

event-triggered model update. *IEEE Robotics and Automation Letters*, 7(2), 3286–3293.

Yu, S., Hirche, M., Huang, Y., Chen, H. & Allgöwer, F. (2021) Model predictive control for autonomous ground vehicles: a review. *Autonomous Intelligent Systems*, 1(1), 1–17.

Zhao, P., Snyder, S., Hovakimyana, N. & Cao, C. (2020) Robust adaptive control of linear parameter-varying systems with unmatched uncertainties. arXiv preprint arXiv:2010.04600.

How to cite this article: Wang, J., Fader, M.T.H. & Marshall, J.A. (2023) Learning-based model predictive control for improved mobile robot path following using Gaussian processes and feedback linearization. *Journal of Field Robotics*, 40, 1014–1033. https://doi.org/10.1002/rob.22165