2024 European Control Conference (ECC)
June 25-28, 2024. Stockholm, Sweden

Gaussian-Process-Based Adaptive Trajectory Tracking Control for Autonomous Ground Vehicles

Kristóf Floch[1], Tamás Péni[1] and Roland Tóth[1,2]

***Abstract*— This paper proposes an adaptive trajectory tracking control algorithm for autonomous ground vehicles. The nonlinear vehicle dynamics are decoupled into two subsystems corresponding to the longitudinal and lateral motions. Each subsystem is augmented with a Gaussian Process to compensate for modeling errors and external disturbances. Based on the augmented subsystems, adaptive control algorithms are synthesized. To give a mathematically correct performance measure, the induced $\mathcal{L}_2$-gain of the nonlinear closed-loop system is computed. The efficiency of the learning-based control method is demonstrated on a high-fidelity physical simulator using a digital twin model of the 1/10 scale F1TENTH vehicle platform.**

I. INTRODUCTION

Nowadays, autonomous mobile robots, such as small-scale car-like ground vehicles are starting to appear in various industrial applications, hence, further improvement of autonomous maneuvering capabilities that can exploit the motion dynamics of these robots is a subject of scientific research. An essential prerequisite for reaching general utilization of these vehicles is the development of algorithms that can cope with unknown variations in their dynamics and to come up with simple control structures that can ensure high-performance maneuvering.

The state-of-the-art trajectory tracking algorithms for car-like ground vehicles are usually studied in the context of autonomous racing and include model-based approaches such as *sliding mode control* (SMC), *nonlinear model predictive control* (NMPC), dynamic inversion, and also learning based control solutions [1]. For model-based techniques, either first principle modeling [2] or system identification [3] is required to come up with a dynamic model of the vehicle. The main advantage of first principle modeling is that the model states and parameters have clear physical interpretations. However, these physically derived vehicle models are highly nonlinear, often cannot capture vehicle-specific effects like friction, and are difficult to tune for the control design. On the other hand, system identification techniques usually result in a black-box representation and we lose the physical interpretation of the states and parameters of the models.

To handle modeling uncertainties, adaptive control approaches have also been studied. These algorithms usually either tune the underlying model parameters [4] or the parameters of the control algorithms [5]. Other than that, model augmentation has also been investigated with *artificial neural network* (ANN) methods [6], and *Gaussian process* (GP) based approaches [7], as these techniques are also capable of handling variation in the model structure. The GP augmentation has proven to be beneficial for a wide range of mobile robotic applications [8] [9], due to its high approximation capability and the uncertainty characterization of the estimates. However, most of the introduced papers rely on MPC algorithms for trajectory tracking, instead of which, the less computationally demanding feedback-based solutions are often favored [10].

Furthermore, previous papers mainly focused on racing scenarios, while in mobile robotics, the importance of precision and robustness may supersede speed. However, stability and robustness studies typically do not take into account the full nonlinear vehicle dynamics; instead, they rely on simplified, often linear decoupled models.

To summarize, the main expectations of control algorithms for such vehicles are (a) adaptability to partly known dynamics and changing environmental conditions; (b) computationally efficient implementation; (c) guaranteed stability and performance. The current adaptive approaches either have (a) and (b), but lack (c), or they have (a) and (c), but lack (b). To cope with the presented challenges, the main contributions of this paper are as follows:

- C1 We propose a computationally efficient learning-based trajectory tracking solution for car-like mobile platforms, capable of handling large model mismatch. The controller relies on a first principle dynamic vehicle model which is augmented with GP-based learning components to estimate model uncertainties.
- C2 We provide a computable performance metric for the complete nonlinear closed-loop system by calculating an upper bound for the induced $\mathcal{L}_2$ gain using an iterative algorithm inspired by [11].
- C3 We evaluate the efficiency of the developed algorithm on the digital twin model of an F1TENTH vehicle [12], implemented in a high-fidelity simulation environment.

The remainder of the paper is organized as follows. Section II provides an overview of GP regression, followed by the utilized vehicle model and the problem formulation in

*This project has received funding from the European Defence Fund programme under grant agreement number No 101103386 and has also been supported by the Air Force Office of Scientific Research under award number FA8655-23-1-7061 and also has been supported by the ÚNKP-23-2-I-BME-171 New National Excellence Program of the Ministry for Culture and Innovation from the source of the National Research, Development and Innovation Fund. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the granting authority can be held responsible for them.

[1]Systems and Control Lab, HUN-REN Inst. for Computer Science and Control, Budapest, Hungary (e-mails: {kristof.floch,tamas.peni}@sztaki.hu)

[2]Control Systems Group, Eindhoven University of Technology, Eindhoven, The Netherlands (e-mail: r.toth@tue.nl)

Copyright ©2024 EUCA

Section III. Section IV presents the trajectory tracking algorithm, with the performance analysis outlined in Section V. Finally, the simulation study is presented in Section VI.

II. GAUSSIAN PROCESS REGRESSION

Gaussian processes (GPs) have been a popular choice for model augmentation, see [7], [8], as their nonparametric structure offers flexible utilization [13]. Furthermore, unlike other learning approaches such as ANNs, GPs provide information on the uncertainty of the approximation, which can be employed for designing robust control algorithms [14]. Next, we briefly introduce GPs and *sparse* GPs as relevant to our proposed control approach.

A. Gaussian processes

Let $\mathcal{D}_N = \{X_N, Y_N\}$ be a dataset, where $X_N = [x_1^\top \ \cdots \ x_N^\top] \in \mathbb{R}^{N\times n_x}$ collects the inputs, $Y_N = [y_1 \ \cdots \ y_N]^\top \in \mathbb{R}^N$ collects the scalar outputs generated by the following model:

$$y_i = f(x_i) + \epsilon_i, \quad i \in \mathbb{I}_1^N, \tag{1}$$

where $f : \mathbb{R}^{n_x} \to \mathbb{R}$ is an unknown function, $\epsilon_i \sim \mathcal{N}(0, \sigma_\epsilon^2)$ is an i.i.d. Gaussian noise and $\mathbb{I}_{\tau_1}^{\tau_2} = \{i \in \mathbb{Z} \mid \tau_1 \le i \le \tau_2\}$ is the corresponding index set. The core idea of GP-based estimation of f is to consider that candidate estimates g belong to a GP, seen as a prior distribution. Then, using $\mathcal{D}_N$ and this prior, a predictive GP distribution of g is computed that provides an estimate of f in terms of its mean and describes the uncertainty of this estimate by its variance.

In terms of definition, a scalar-valued *Gaussian Process* $\mathcal{GP} : \mathbb{R}^{n_x} \to \mathbb{R}$ assigns to every point $x \in \mathbb{R}^{n_x}$ a random variable $\mathcal{GP}(x)$, such that, for any finite set $\{x_i\}_{i=1}^Z \in \mathbb{R}^{n_x}$, the joint probability distribution of $\mathcal{GP}(x_1), \cdots, \mathcal{GP}(x_Z)$ is Gaussian. Due to this property, $g \sim \mathcal{GP}(m, \kappa)$ is fully determined by its mean m and covariance κ expressed as

$$m(x) = \mathbb{E}\{g(x)\}, \tag{2a}$$
$$\kappa(x, \tilde{x}) = \mathbb{E}\{(g(x) - m(x))(g(\tilde{x}) - m(\tilde{x}))^\top\}, \tag{2b}$$

where $x, \tilde{x} \in \mathbb{R}^{n_x}$. This distribution describes our prior belief of the function space in which the estimate of the unknown function f is searched for. We assume that the prior mean is zero ($m(x) = 0$) and covariance of the distribution can be well described by a *squared exponential* (SE) kernel, a common choice for estimation of smooth functions:

$$\kappa_{\mathrm{SE}}(x, \tilde{x}) = \sigma_{\mathrm{f}}^2 \exp\left(-\frac{1}{2}(x - \tilde{x})^\top \Lambda^{-1}(x - \tilde{x})\right), \tag{3}$$

where $\sigma_{\mathrm{f}} \in \mathbb{R}$ is a scaling factor and $\Lambda \in \mathbb{R}^{n_x \times n_x}$ is a positive definite and symmetric matrix that determines the smoothness of the candidate functions. Together, σ_{f}, Λ and σ_ϵ in (1) are the so-called hyperparameters of the GP that can be tuned by maximizing the marginal likelihood computed from observed data [13].

Based on $\mathcal{D}_N$ and the prior $g \sim \mathcal{GP}(m, \kappa)$, the predictive distribution for $g(x_\star)$ at a test point $x_\star$ is the posterior $p(g(x_\star)|\mathcal{D}_N, x_\star) = \mathcal{N}(\mu(x_\star), \Sigma(x_\star))$ characterised by

$$\mu(x_\star) = K_N^\top(x_\star)(K_{NN} + \sigma_\epsilon^2 I)^{-1} Y, \tag{4a}$$
$$\Sigma(x_\star) = \kappa(x_\star, x_\star) - K_N^\top(x_\star)(K_{NN} + \sigma_\epsilon^2 I)^{-1} K_N(x_\star) \tag{4b}$$

where $[K_N(x_\star)]_i = \kappa(x_i, x_\star),\ i \in \mathbb{I}_1^N$ and $[K_{NN}]_{i,j} = \kappa(x_i, x_j),\ i \in \mathbb{I}_1^N,\ j \in \mathbb{I}_1^N$. Eq. (4a) describes the mean as the estimate of function f and (4b) is the variance, which gives the uncertainty of the approximation. Although the computation of (4) only requires elementary matrix multiplications, every training point is required for making predictions. Therefore in case of large training datasets, the real-time implementation of GPs can be challenging.

B. Sparse Gaussian processes

One way to tackle the computational limitation of the traditional GP regression is to use *sparse Gaussian processes* (SGPs), where the number of data points used in the evaluation is limited to a fixed number. The goal of the SGP is to find a set of pseudo inputs and outputs defined as $\mathcal{D}_M = \{X_M, Y_M\}$, where $X_M = [\hat{x}_1^\top \cdots \hat{x}_M^\top]$ and $Y_M = [\hat{y}_1 \cdots \hat{y}_M]^\top$ with $M \ll N$ such that $\mathcal{D}_M$ sufficiently describes $\mathcal{D}_N$. These points are also referred to as inducing points in the literature. The core philosophy of the SGP is to select the set $\mathcal{D}_M$ in a manner that minimizes the KL divergence between the GP trained on it and the GP trained on the initial D_N set. To find the proper $\mathcal{D}_M$ set, [15] proposes the *variational free energy* (VFE) concept, which jointly selects the inducing inputs and the hyperparameters of the GP by maximizing a lower bound to the marginalized likelihood of the prediction. As a result, the pseudo outputs are fully determined by the hyperparameters, hence they are not part of the optimization. See [15] for the full derivation.

The resulting posterior distribution after the training at an arbitrary test point $x_\star$ is $\mathcal{N}(\mu_{\mathrm{p}}(x_\star), \Sigma_{\mathrm{p}}(x_\star))$, where

$$\mu_{\mathrm{p}}(x_\star) = K_M^\top(x_\star) K_{MM}^{-1} m_{\mathrm{p}}, \tag{5a}$$
$$\begin{aligned}\Sigma_{\mathrm{p}}(x_\star) = {} & \kappa(x_\star, x_\star) \\ & + K_M^\top(x_\star)(K_M^{-1} - K_{MM}^{-1}\mathcal{K}_{\mathrm{p}} K_{MM}^{-1}) K_M(x_\star),\end{aligned} \tag{5b}$$

corresponding to a Nyström projection of the original GP to the pseudo inputs X_M, and where $[K_{MN}]_{i,j} = \kappa(\hat{x}_i, x_j),\ i \in \mathbb{I}_1^M,\ j \in \mathbb{I}_1^N$ is the covariance matrix between the pseudo inputs and all training inputs, $[K_{MM}]_{i,j},\ i, j \in \mathbb{I}_1^M$ is the covariance of the pseudo inputs and $[K_M(x_\star)]_i = \kappa(x_i, x_\star)$. Furthermore, m_{p} and $\mathcal{K}_{\mathrm{p}}$ are the mean and covariance of the approximation of the true posterior distribution, calculated as

$$m_{\mathrm{p}} = \sigma_\epsilon^{-2} \mathcal{K}_{\mathrm{p}} K_{MM}^{-1} K_{MN} Y, \tag{6a}$$
$$\mathcal{K}_{\mathrm{p}} = K_{MM}(K_{MM} + \sigma_\epsilon^{-2} K_{MN} K_{NM})^{-1} K_{MM}. \tag{6b}$$

This formulation reduces the $\mathcal{O}(N^3)$ computational complexity of the training of the traditional GP to $\mathcal{O}(NM^2)$. Furthermore, it can also be recursively trained with high efficiency [9].

In the following, SGPs will be utilized as the learning component of the proposed trajectory tracking approach, which provides efficient adaptation to modeling uncertainties and external disturbances.

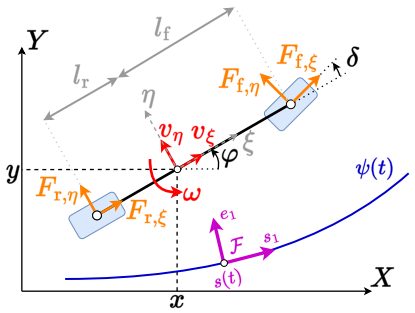

Fig. 1: Single-track vehicle model and reference trajectory.

III. VEHICLE MODEL & TRAJECTORY TRACKING PROBLEM

A. First principle model

The baseline vehicle model relies on a dynamic single-track representation, which has been commonly used for describing the behavior of small-scale car-like vehicles, see [16], [17], as it can capture the primary characteristics of the motion. The modelling concept is depicted in Fig. 1 and the resulting model is described as

$$\dot{x} = v_\xi \cos(\varphi) - v_\eta \sin(\varphi), \tag{7a}$$
$$\dot{y} = v_\xi \sin(\varphi) + v_\eta \cos(\varphi), \tag{7b}$$
$$\dot{\varphi} = \omega, \tag{7c}$$
$$\dot{v}_\xi = \frac{1}{m}\left(F_\xi + F_\xi \cos(\delta) - F_{\mathrm{f},\eta}\sin(\delta) + m v_\eta \omega\right), \tag{7d}$$
$$\dot{v}_\eta = \frac{1}{m}\left(F_{\mathrm{r},\eta} + F_\xi \sin(\delta) + F_{\mathrm{f},\eta}\cos(\delta) - m v_\xi \omega\right), \tag{7e}$$
$$\dot{\omega} = \frac{1}{I_z}\left(F_{\mathrm{f},\eta} l_\mathrm{f}\cos(\delta) + F_{\mathrm{f},\xi} l_\mathrm{f}\sin(\delta) - F_{\mathrm{r},\eta} l_\mathrm{r}\right), \tag{7f}$$

where (x, y) is the position and φ is the orientation of the vehicle in the global coordinate frame. The states v_ξ and v_η denote the longitudinal and lateral velocity of the vehicle in a body fixed frame and ω is the yaw rate. The parameters of the model are the distance of the front and rear axis from the center of mass, denoted as l_f and l_r, the mass of the vehicle m and the inertia along the vertical axis I_z.

The longitudinal tire force F_ξ is determined by a drivetrain model, which assumes a first-order connection between the motor input and the velocity of the vehicle. This modeling technique has been successfully utilized previously in [16] and [17] for electric vehicles, hence, we adopted the following variant:

$$F_\xi = C_{\mathrm{m}1} d - C_{\mathrm{m}2} v_\xi - C_{\mathrm{m}3}, \tag{8}$$

where $C_{\mathrm{m}1}$, $C_{\mathrm{m}2}$, $C_{\mathrm{m}3}$ are lumped drivetrain parameters and $d \in [0, 1]$ is the motor input. Lastly, the lateral tire forces are often calculated using a simplified linear tire model as

$$F_{\mathrm{r},\eta} = C_\mathrm{r} \arctan\left(\frac{-v_\eta + l_\mathrm{r}\omega}{v_\xi}\right), \tag{9a}$$
$$F_{\mathrm{f},\eta} = C_\mathrm{f} \arctan\left(\delta - \frac{v_\eta + l_\mathrm{f}\omega}{v_\xi}\right), \tag{9b}$$

where C_f and C_r are cornering stiffness values of the front and rear tire respectively. Finally, the control inputs of the vehicle are the steering angle δ and the motor input d, which a controller can directly actuate.

B. Trajectory tracking problem

To outline the trajectory tracking problem, we first need to define the reference trajectories for the vehicle. The path of the desired trajectory is expressed as a two-dimensional spline curve $\psi(s^\mathrm{ref}(t))$, defined by coordinate functions $(x(s^\mathrm{ref}(t)), y(s^\mathrm{ref}(t))$, where s^ref is a time domain signal defined as $s^\mathrm{ref} : \mathbb{R} \to [0, L]$. The arc length of the full path is denoted as L, hence $s^\mathrm{ref}(t)$ describes the desired vehicle position along the path at t. Both $x(s^\mathrm{ref}(t))$, and $y(s^\mathrm{ref}(t))$ are monotonic in $s^\mathrm{ref}(t)$, furthermore $(x(0), y(0))$ and $(x(L), y(L))$ assign the endpoints of the curve. The speed reference $v^\mathrm{ref}(s^\mathrm{ref}(t)) = v^\mathrm{ref}(t)$ along the trajectory is also given. These types of reference motion trajectories can be obtained by regular path planning algorithms.

With the trajectory known, we can transform (7) into a curvilinear coordinate frame (depicted in Fig. 1 as $\mathcal{F}$) that is parameterized by the position along the reference path [16]:

$$\dot{s} = (v_\xi \cos(\theta_\mathrm{e}) - v_\eta \sin(\theta_\mathrm{e}))/(1 - c(s)e_s), \tag{10a}$$
$$\dot{e}_s = v_\xi \sin(\theta_\mathrm{e}) + v_\eta \cos(\theta_\mathrm{e}), \tag{10b}$$
$$\dot{\theta}_\mathrm{e} = \omega - c(s)\dot{s}, \tag{10c}$$

where the newly introduced states are the position along the path s, the lateral deviation e_s and the heading error θ_e, while the lateral (v_η) longitudinal (v_ξ) and angular velocities (ω) are the same as in (7). Furthermore, $c(s)$ describes the curvature of the reference path at s.

The main advantage of this model is that the tracking errors explicitly appear in (10). Furthermore, due to the physics-inspired model description, all the states can be easily determined from measurements, which allows the design of a full state-feedback controller for the vehicle.

IV. GP-BASED ADAPTIVE CONTROL

A. Control architecture

Based on the trajectory tracking model (10), we propose a computationally efficient feedforward-feedback control algorithm. As (10) is a complex nonlinear system for which efficient control laws are hard to derive, we decouple the nonlinear vehicle dynamics into two subsystems, which correspond to the longitudinal and lateral motion of the vehicle. Then, to adapt to modeling uncertainties and external disturbances, we augment each subsystem with a GP-based learning component to capture the model mismatch and compensate for it via feedforward. Finally, based on the remaining nominal model structure, we synthesize LQ state-feedback controllers to track the given reference. The overall architecture is depicted in Fig. 2.

B. Decoupling

We decouple (10) into a lateral and longitudinal subsystem for individual control design. The longitudinal controller is responsible for tracking the reference velocity and position along the path and the lateral controller is used for path tracking.

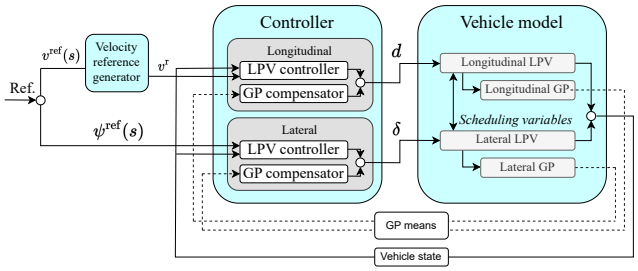

Fig. 2: Proposed control architecture for trajectory tracking.

From (10), the **longitudinal model** becomes

$$\dot{s} = \frac{v_\xi \cos(\theta_\mathrm{e}) - v_\eta \sin(\theta_\mathrm{e})}{1 - c(s)e_s}, \tag{11a}$$

$$\dot{v}_\xi = \frac{1}{m}\big((1+\cos(\delta))(C_{\mathrm{m}1}d - C_{\mathrm{m}2}v_\xi - C_{\mathrm{m}3}) - F_{\mathrm{f},\eta}\sin(\delta) + mv_\eta\omega\big), \tag{11b}$$

where the state vector $x_\mathrm{lo} = [s\ v_\xi]^\top$ describes the position along the path and the longitudinal velocity, while d is the actuated control input. Furthermore, the heading error θ_e, lateral deviation e_s, lateral velocity v_η, steering angle δ and $F_{\mathrm{f},\eta}$ are considered as external varying parameters depending on the lateral subsystem. The reference input of the system is $x_\mathrm{lo}^\mathrm{ref} = [s^\mathrm{ref}\ v^\mathrm{ref}]^\top$ defined by the trajectory. To simplify the control design and analysis, we separate the position and velocity states by introducing a virtual velocity generator that provides a position-adjusted virtual reference velocity as $v^\mathrm{r} = v^\mathrm{ref} - k_v(s - s^\mathrm{ref})$. This results in the modified control objective $v_\xi \to v^\mathrm{r}$, and the system dimensions are reduced, as only the dynamics of v_ξ are needed to be considered. Therefore, the system can be expressed as:

$$\underbrace{\dot{v}_\xi}_{\dot{\chi}_\mathrm{lo}} = \underbrace{\frac{-C_{\mathrm{m}2}(1+\cos(\delta))}{m}}_{A_\mathrm{lo}(\delta)}\underbrace{v_\xi}_{\chi_\mathrm{lo}} + \underbrace{\frac{C_{\mathrm{m}1}(1+\cos(\delta))}{m}}_{B_\mathrm{lo}(\delta)}\underbrace{d}_{u_\mathrm{lo}} + \underbrace{\frac{-C_{\mathrm{m}3}(1+\cos(\delta))}{m}}_{B_0(\delta)}\underbrace{\mathrm{sign}(v_\xi)}_{w_0} + \underbrace{\frac{-F_{\mathrm{f},\eta}\sin(\delta)}{m} + v_\eta\omega}_{w_1} \tag{12}$$

where $A_\mathrm{lo}(\delta)$ and $B_\mathrm{lo}(\delta)$ can be considered as parameter-varying state-transition and input matrices with scheduling variable δ, resulting in a *linear parameter-varying* (LPV) representation [18]. Furthermore, w_0 is a nonlinearity introduced by the dry friction of the drivetrain and w_1 lumps together the effects of the lateral subsystem.

Next, we consider the **lateral model**. The lateral behavior of the vehicle is described as

$$\dot{e}_s = v_\xi \sin(\theta_\mathrm{e}) + v_\eta \cos(\theta_\mathrm{e}), \tag{13a}$$

$$\dot{\theta}_\mathrm{e} = \omega - c(s)\dot{s}, \tag{13b}$$

$$\dot{v}_\eta = \frac{1}{m}\left(F_{\mathrm{r},\eta} + F_\xi \sin(\delta) + F_{\mathrm{f},\eta}\cos(\delta) - mv_\xi\omega\right), \tag{13c}$$

$$\dot{\omega} = \frac{1}{I_z}\left(F_{\mathrm{f},\eta}l_\mathrm{f}\cos(\delta) + F_{\mathrm{f},\xi}l_\mathrm{f}\sin(\delta) - F_{\mathrm{r},\eta}l_\mathrm{r}\right), \tag{13d}$$

where the state vector $x_\mathrm{la} = [e_s\ \theta_\mathrm{e}\ v_\eta\ \omega]^\top$ consists of the lateral error e_s, heading error θ_e, lateral velocity v_η and yaw rate ω, while v_ξ is considered a scheduling variable with the steering angle δ as the actuated input. To simplify the model, we first substitute the lateral tire models into (13), use small angle approximations ($\sin(\alpha) \approx \alpha$, $\cos(\alpha) \approx 1$), neglect the longitudinal tire force ($F_\xi \approx 0$) and approximate the velocity along the path as $\dot{s} \approx v_\xi$ which leads to the model [19]:

$$\begin{bmatrix}\dot{e}_s\\ \dot{v}_\eta\\ \dot{\theta}_\mathrm{e}\\ \dot{\omega}\end{bmatrix} = \begin{bmatrix}0 & 1 & v_\xi & 0\\ 0 & -\frac{C_\mathrm{f}+C_\mathrm{r}}{mv_\xi} & 0 & -v_\xi - \frac{l_\mathrm{f}C_\mathrm{f}-l_\mathrm{r}C_\mathrm{r}}{mv_\xi}\\ 0 & 0 & 0 & 1\\ 0 & \frac{l_\mathrm{r}C_\mathrm{r}-l_\mathrm{f}C_\mathrm{f}}{I_z v_\xi} & 0 & -\frac{l_\mathrm{f}^2C_\mathrm{f}+l_\mathrm{r}^2C_\mathrm{r}}{I_z v_\xi}\end{bmatrix}\begin{bmatrix}e_s\\ v_\eta\\ \theta_\mathrm{e}\\ \omega\end{bmatrix} + \begin{bmatrix}0\\ \frac{C_\mathrm{f}}{m}\\ 0\\ \frac{l_\mathrm{f}C_\mathrm{f}}{I_z}\end{bmatrix}\delta + \begin{bmatrix}0\\ 0\\ v_\xi\\ 0\end{bmatrix}c(s), \tag{14}$$

where the path curvature is regarded as an external disturbance. Further simplification can be achieved by expressing the lateral model only in terms of the error variables and their derivatives. Let the second derivative of the (already linearized) lateral error be expressed as $\ddot{e}_s = \dot{v}_\eta + v_\xi\dot{\theta}_\mathrm{e}$. Then, as proposed in [20], the error dynamics have the form:

$$\begin{bmatrix}\dot{e}_s\\ \ddot{e}_s\\ \dot{\theta}_\mathrm{e}\\ \ddot{\theta}_e\end{bmatrix} = \begin{bmatrix}0 & 1 & 0 & 0\\ 0 & -\frac{C_\mathrm{f}+C_\mathrm{r}}{mv_\xi} & \frac{C_\mathrm{f}+C_\mathrm{r}}{m} & -\frac{l_\mathrm{f}C_\mathrm{f}+l_\mathrm{r}C_\mathrm{r}}{mv_\xi}\\ 0 & 0 & 0 & 1\\ 0 & \frac{l_\mathrm{r}C_\mathrm{r}-l_\mathrm{f}C_\mathrm{f}}{I_z v_\xi} & \frac{l_\mathrm{f}C_\mathrm{f}-l_\mathrm{r}C_\mathrm{r}}{I_z} & -\frac{l_\mathrm{f}^2C_\mathrm{f}+l_\mathrm{r}^2C_\mathrm{r}}{I_z v_\xi}\end{bmatrix}\begin{bmatrix}e_s\\ \dot{e}_s\\ \theta_\mathrm{e}\\ \dot{\theta}_\mathrm{e}\end{bmatrix} + \begin{bmatrix}0\\ \frac{C_\mathrm{f}}{m}\\ 0\\ \frac{l_\mathrm{f}C_\mathrm{f}}{I_z}\end{bmatrix}\delta + \begin{bmatrix}0\\ \frac{l_\mathrm{r}C_\mathrm{r}-l_\mathrm{f}C_\mathrm{f}}{m} - 1\\ 0\\ -\frac{l_\mathrm{f}^2C_\mathrm{f}+l_\mathrm{r}^2C_\mathrm{r}}{I_z v_\xi}\end{bmatrix}c(s). \tag{15}$$

As discussed in [21], regulating both the heading and the lateral error to the origin results in poor tracking performance, as the two quantities cannot be simultaneously zero along the path if we assume perfect tracking. Therefore, we separate the lateral and the heading dynamics for the control design. We use the lateral error dynamics for feedback control design, and only regulate the heading error with a feedforward, as proposed in the so-called Stanley controller [22]. Furthermore, we also introduce a new state as the integral of the lateral error, i.e. $q = \int_0^t e_s \mathrm{d}t$ to assure asymptotic convergence of e_s without offset. The final control-oriented lateral model can be expressed as

$$\underbrace{\begin{bmatrix}\dot{q}\\ \dot{e}_s\\ \ddot{e}_s\end{bmatrix}}_{\dot{\chi}_\mathrm{la}} = \underbrace{\begin{bmatrix}0 & 1 & 0\\ 0 & 0 & 1\\ 0 & 0 & -\frac{C_\mathrm{f}+C_\mathrm{r}}{mv_\xi}\end{bmatrix}}_{A_\mathrm{la}(v_\xi)}\underbrace{\begin{bmatrix}q\\ e_s\\ \dot{e}_s\end{bmatrix}}_{\chi_\mathrm{la}} + \underbrace{\begin{bmatrix}0\\ 0\\ \frac{C_\mathrm{f}}{m}\end{bmatrix}}_{B_\mathrm{la}}\delta + \underbrace{\begin{bmatrix}0\\ 0\\ \frac{l_\mathrm{r}C_\mathrm{r}-l_\mathrm{f}C_\mathrm{f}}{m} - 1\end{bmatrix}}_{B_c}\underbrace{c(s)}_{w_c} + \underbrace{\begin{bmatrix}0 & 0\\ 0 & 0\\ \frac{C_\mathrm{r}+C_\mathrm{f}}{m} & -\frac{l_\mathrm{f}^2C_\mathrm{f}+l_\mathrm{r}^2C_\mathrm{r}}{I_z v_\xi}\end{bmatrix}}_{B_2(v_\xi)}\underbrace{\begin{bmatrix}\theta_\mathrm{e}\\ \dot{\theta}_\mathrm{e}\end{bmatrix}}_{w_2}, \tag{16}$$

where the state vector $\chi_{\mathrm{la}} = [q\ e_s\ \dot{e}_s]^\top$ now only contains the lateral error, its integral and derivative, while $A_{\mathrm{la}}(v_\xi)$ is the parameter varying state-transition matrix with scheduling signal v_ξ and B_{la} is the input matrix. Moreover, w_c is the path curvature and w_2 is used to lump together the unmodeled path and heading dynamics.

C. GP-based model augmentation

Note that during the derivation of (7d) and (16), we have made simplifications. Furthermore, in case of changing environmental conditions or modeling uncertainties, the model mismatch between the control-oriented model and the true vehicle can reach a point where the tracking performance significantly decreases. Therefore, to capture this model mismatch, we augment the nominal (12) and (16) with GPs:

$$\dot{\chi}_{\mathrm{la}} = A_{\mathrm{la}}(v_\xi)\chi_{\mathrm{la}} + B_{\mathrm{la}}\delta + B_c w_c + B_{\mathcal{GP}}\mathcal{GP}_{\mathrm{la}}(z_{\mathrm{la}}), \quad (17a)$$

$$\dot{\chi}_{\mathrm{lo}} = A_{\mathrm{lo}}(\delta)\chi_{\mathrm{lo}} + B_{\mathrm{lo}}(\delta) + B_0(\delta)w_0 + \mathcal{GP}_{\mathrm{lo}}(z_{\mathrm{lo}}), \quad (17b)$$

where $\mathcal{GP}_{\mathrm{lo}}(z_{\mathrm{lo}})$ and $\mathcal{GP}_{\mathrm{la}}(z_{\mathrm{la}})$ denote the GPs linked to the lateral and longitudinal subsystems, respectively. Based on the structure of the nominal path following model (10), we can observe that the first three equations only capture kinematic relationships. Therefore, we assume that modeling uncertainties only affect the velocity states v_ξ, v_η, ω, as proposed in [8]. Furthermore, we can also note that $B_{\mathcal{GP}} = [0\ 0\ 1]^\top$ as uncertain dynamic effects only affect $\dot{e}_s$.

Observing the original vehicle model (7), we can also note that any change in the environmental conditions affects the dynamics through the acting wheel forces. As these models depend on the velocity states (v_ξ, v_η, ω), we choose these variables to construct the GPs. Furthermore, in the lateral case, the path curvature $c(s)$ is also added as a variable because of its effect on the lateral dynamics. Therefore, the GP inputs are $z_{\mathrm{lo}} = [v_\xi\ v_\eta\ \omega]^\top$ and $z_{\mathrm{la}} = [v_\xi\ v_\eta\ \omega]^\top$.

As we previously assumed that all the vehicle states are available, training inputs can be collected from the logged measurement data of driving experiments with the vehicle. By numerical differentiation, we can obtain the state derivatives and the outputs for the GPs can be expressed from (17) to generate the training dataset.

Because of the large number of training points in the dataset, utilization of SGPs is necessary to reduce the computation complexity of the evaluation. The resulting GP components are taken into consideration during control synthesis as follows. The means μ_{lo} and μ_{la} are cancelled by introducing the following feedforward terms:

$$d_{\mathcal{GP}} = 1/B_{\mathrm{lo}}(\delta)\mu_{\mathrm{lo}}(z_{\mathrm{lo}}), \quad (18a)$$

$$\delta_{\mathcal{GP}} = B_{\mathrm{la}}^{\dagger} B_{\mathcal{GP}}\mu_{\mathrm{la}}(z_{\mathrm{la}}), \quad (18b)$$

while the remaining zero-mean Gaussian random variables characterized by the variances Σ_{lo}, Σ_{la} are considered as external disturbances to reject. Furthermore, we can utilize the variances for fine-tuning the GPs, by collecting additional training data in regions where the uncertainty is high [23].

D. LQ state feedback design

Assuming that the feedforward terms (18) can efficiently eliminate the model mismatch, we stabilize the subsystems by the following control laws:

$$\delta_{\mathrm{nom}} = K_{\mathrm{la}}(v_\xi)\chi_{\mathrm{la}} - \theta_{\mathrm{e}} - B_{\mathrm{la}}^{\dagger} B_c w_c, \quad (19a)$$

$$d_{\mathrm{nom}} = K_{\mathrm{lo}}(\delta)(v_\xi - v^{\mathrm{r}}) + 1/B_{\mathrm{lo}}(\delta)A_{\mathrm{lo}}(\delta)v^{\mathrm{r}} - 1/B_{\mathrm{lo}}(\delta)B_{w_0}(\delta)w_0, \quad (19b)$$

where $K_{\mathrm{la}}(v_\xi)$ and $K_{\mathrm{lo}}(\delta)$ are parameter-dependent feedback matrices and the additional terms are used to achieve reference tracking and to cancel out known disturbances. The feedback matrices are obtained using the nominal subsystems (12) (16) with the LPV-LQR synthesis proposed in [24].

Consider an LPV system in the general form as $\dot{\chi} = A(\rho)\chi + B(\rho)u + B_\Sigma n$, where n is a zero-mean white noise acting on the system, characterized by its variance Σ. The optimal parameter dependent feedback matrix $K(\rho)$ that minimizes the quadratic cost $J_{\mathrm{LQ}} = \int_0^\infty \chi^\top(t)Q\chi(t) + u^\top(t)Ru(t)\ \mathrm{d}t$ with $Q \in \mathbb{R}^{n_x \times n_x}$ and $R \in \mathbb{R}^{n_u \times n_u}$ can be obtained by solving the convex optimization problem:

$$\max_{K,X,Y} \quad \mathrm{trace}(X), \quad (20a)$$

$$\text{s.t.} \quad X \succ 0, \quad (20b)$$

$$M(X, Y, \mathcal{Q}, \mathcal{R}, \rho) \succ 0 \quad \forall \rho \in \mathbb{G}, \quad (20c)$$

where $X \in \mathbb{R}^{n_x \times n_x}$, and the *linear matrix inequality* (LMI) constraint $M(X, Y, \mathcal{Q}, \mathcal{R}, \rho)$ is defined as

$$\begin{bmatrix} -\mathrm{He}(A(\rho)X + B(\rho)Y(\rho)) & (\mathcal{Q}X + \mathcal{R}Y(\rho))^\top \\ (\mathcal{Q}X + \mathcal{R}Y(\rho)) & I \end{bmatrix} \quad (21)$$

where the gain matrices of the quadratic cost are encoded in $\mathcal{Q} = [Q^{\frac{1}{2}}\ 0]^\top$ and $\mathcal{R} = [0\ R^{\frac{1}{2}}]^\top$ and $\mathrm{He}(X) = X^\top + X$. Furthermore, $Y(\rho) \in \mathbb{R}^{n_u \times n_x}$ is parameterized as follows:

$$Y(\rho) = Y_0 + \rho Y_1 + \rho^2 Y_2 + \cdots + \rho^n Y_n \quad (22)$$

and $\mathbb{G} \subset \Gamma$ is a discrete grid, used to relax the infinite number of LMI constraints. After solving (20), the parameter-dependent feedback matrix is obtained as $K(\rho) = Y(\rho)X^{-1}$.

Furthermore, [24] also defines a performance measure for the state-feedback controller, which describes the effect of the Gaussian white noise on the LQ cost:

$$\sigma_\infty \leq \lim_{T\to\infty} \sup_{\rho\in\Gamma} \mathbb{E}\left\{ \frac{1}{T}\int_0^T \chi^\top(t)Q\chi(t) + u^\top(t)Ru(t)\mathrm{d}t \right\} = \mathrm{trace}(B_\Sigma \Sigma B_\Sigma^\top X^{-1}). \quad (23)$$

The calculated upper bound on σ_∞ quantifies the degradation of the control performance that originates from the Gaussian noise, i.e., the remnant uncertainty of the GP.

The parameter-dependent state-feedback gains K_{lo} and K_{la} for the longitudinal and lateral subsystems have been obtained using the outlined LPV-LQR synthesis with aprioi fixed weighting matrices Q_{lo}, Q_{la}, R_{lo}, R_{la}. The performance of the controller can be improved by reducing the uncertainty of the GP approximation. This can be achieved by collecting more training data based on the variance of the

posterior distribution of the GP. This concept is implemented in Section VI, where the covariance of the GP is used to construct specific training trajectories.

V. $\mathcal{L}_2$-GAIN ANALYSIS

To analyze stability and performance of the proposed control approach, we perform an $\mathcal{L}_2$-gain analysis on the nonlinear model (10) with the two independently designed controllers (19) in closed-loop. Since the mean of the GP is canceled by a feedforward term, the nonlinear closed-loop system has 4 external inputs: two zero mean Gaussian noises from the GP, the velocity reference v^{r} and the curvature c of the path. In the following analysis, we focus on how v^{r} and c affect the tracking performance, so we analyze the $\mathcal{L}_2$-gain w.r.t. to these channels.

For analysis, consider the closed loop in the form of

$$\mathcal{S}\begin{cases} \dot{x} = f_{\mathrm{cl}}(x, w) \\ z = h(x, w) \end{cases} \tag{24}$$

where $f_{\mathrm{cl}} : \mathcal{X} \times \mathcal{W} \to \mathcal{X}$ corresponds to (10) with the lateral and longitudinal control laws (19), while $x = [q,\ e_s,\ \theta_{\mathrm{e}},\ \tilde{v}_\xi,\ v_\eta,\ \omega]^\top \in \mathcal{X}$ is the state vector, $w = [\tilde{v}^{\mathrm{r}},\ c] \in \mathcal{W}$ is the generalized disturbance signal and z is the generalized performance signal that contains the tracking errors as $z = h(x, w) = [v_\xi - v^{\mathrm{r}},\ e_s]^\top \in \mathcal{Z}$. Note that both the state and the disturbance input are centered such that $\tilde{v}_\xi = v_\xi - v^{\mathrm{r}}_{\mathrm{cent}}$ and $\tilde{v}^{\mathrm{r}} = v^{\mathrm{r}} - v^{\mathrm{r}}_{\mathrm{cent}}$ in order to achieve $w \in \mathcal{L}_2$ and $0 = f_{\mathrm{cl}}(x, w)$ equilibrium. Furthermore, to characterize that v^{r} is only active in the low-frequency range, we augment the system with a first-order low-pass filter. By determining the $\mathcal{L}_2$-gain of the resulting system from w to z, we can provide a quantitative measure for the reference tracking performance of the proposed algorithm.

The calculation of the $\mathcal{L}_2$-gain relies on the theory of dissipative dynamical systems [25]. A system is said to be dissipative w.r.t a quadratic supply rate in the form of $s(w, z) = \gamma^2 w^\top w - z^\top z$ if there exists a non-negative storage function $V : \mathcal{X} \to \mathbb{R}$ such that, in case V is differentiable, the differential dissipation inequality

$$\dot{V}(x) \leq \gamma^2 w^\top w - z^\top z \tag{25}$$

is satisfied for all (x, z, w) trajectories of $\mathcal{S}$. If w is restricted to squared integrable signals, i.e., $\mathcal{L}_2$, then the induced $\mathcal{L}_2$-gain is the smallest γ for which (25) holds, formally: $\sup_{w\in\mathcal{L}_2}\{\|z\|_2/\|w\|_2\} \leq \gamma$. A finite $\mathcal{L}_2$-gain also proves the stability of the corresponding system.

To estimate γ, we propose an iterative, optimization-based approach, inspired by [11], which consists of two components: a learner and a verifier. First, the learner is responsible for finding a storage function candidate V and corresponding $\mathcal{L}_2$-gain γ by solving a convex optimization problem. To formulate the learner, we restrict ourselves to quadratic supply function candidates in the form of $V(x) = x^\top P(x) x$, where $P(x) = P_0 + P_1 x_1 + \cdots + P_6 x_6 \succ 0$. By substituting the storage function and (24) into (25), we obtain

$$J(x, w, \{P_i\}_{i=1}^6, \gamma^2) = x^\top P(x) f_{\mathrm{cl}}(x, w) + f_{\mathrm{cl}}^\top(x, w) P(x) x + x^\top \frac{\mathrm{d}P(x)}{\mathrm{d}t} x - \gamma^2 w^\top w + h^\top(x, w) h(x, w) \leq 0. \tag{26}$$

Note that if x and w are fixed at constant values, (26) is linear in the unknown variables P_i and γ^2. Therefore, by introducing $\mathbb{X} \subset \mathcal{X}$ and $\mathbb{W} \subset \mathcal{W}$, we propose the following convex optimization as the learner:

$$\min_{P_0,\cdots,P_6,\gamma^2} \ \gamma^2 \tag{27a}$$

$$\text{s.t. } P(x) \succ 0, \tag{27b}$$

$$J(x, w, \{P_i\}_{i=1}^6, \gamma^2) \leq 0, \tag{27c}$$

$$\forall (x, w) \in \mathbb{X} \times \mathbb{W}$$

where $\mathbb{X}$ and $\mathbb{W}$ are discrete grids, constructed by sampling the compact sets such that the sample points sufficiently cover $\mathcal{X}\times\mathcal{W}$. As (27c) is linear in the optimization variables, we can utilize this gridding approach even up to the case of 6 dimensions, as state-of-the-art numerical solvers can efficiently handle even a large number of linear constraints.

Note that the learner only guarantees that the differential dissipation inequality is satisfied at the discrete grid points. Therefore, the verifier is introduced, which essentially tries to find counterexamples where (27c) does not hold in $\mathcal{X}\times\mathcal{W}$ for the previously obtained γ^2 and V. For a fixed $\{P_i\}_{i=1}^6$ and γ^2, the verifier is formulated as the nonlinear optimization:

$$\max_{x,w} \quad J(x, w, \{P_i\}_{i=1}^6, \gamma^2) \tag{28a}$$

$$\text{s.t.} \quad x \in \mathcal{X}, \quad w \in \mathcal{W}. \tag{28b}$$

As (28) is a small dimensional problem, numerical solvers are capable of handling it. If a positive J is obtained by (28), the corresponding x and w variables are added to the discrete sets $\mathbb{X}$, $\mathbb{W}$ and the iteration is repeated until either (27) becomes infeasible which means that the we cannot obtain a bound for the induced $\mathcal{L}_2$-gain with the proposed storage function structure or the optimal value of (28a) remains negative which means that an upper bound for $\mathcal{L}_2$-gain is found with the storage function V, also showing stability.

VI. SIMULATION STUDY

A. Simulation Environment

The proposed GP-based learning control approach is evaluated in a high-fidelity simulation environment, where real-world scenarios can be efficiently emulated. The simulator is based on the open-source MuJoCo physics engine [26] and incorporates a digital twin model of a real 1/10 scale electric F1TENTH car. The parameters of the dynamic vehicle model used for the control design have been identified as described in [16], where the obtained numerical values are reported. To artificially generate a large model mismatch, the parameters of the simulated vehicle model have been heavily modified. The friction coefficients between the wheels and the ground have been reduced, while their radius and the overall inertia of the vehicle have been increased. The original and altered model parameters are displayed in Tab I.

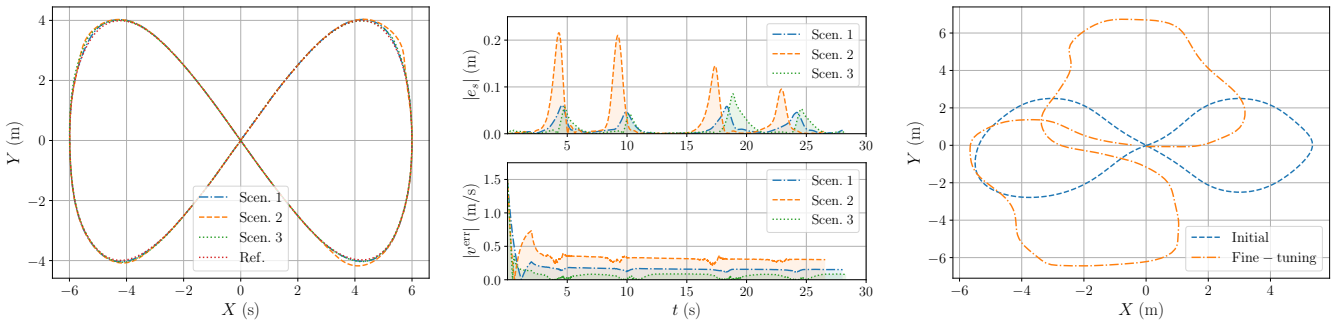

Fig. 3: Simulation results, with the comparison between the realized and the reference trajectories (a), the tracking errors (b) of the three scenarios, and the training trajectories (c)

TABLE I: Initial and altered model parameters

Parameter	Initial	Altered
Friction coefficients	2.5	0.5
Inertia (kgm^2)	0.078	0.090
Wheel radius (m)	0.052	0.072

B. *Trajectory tracking simulations*

Since the LPV-LQR synthesis (20) does not require the GPs to be trained, we can construct a nominal LPV feedback controller without the adaptive terms. The parameter-dependent feedback matrices are obtained by solving (20), which has been implemented in Python with CVXPY [27] and solved with Mosek. The weighting matrices of the LQR have been tuned using numerical simulations with the nominal dynamic model resulting in $Q_{\text{la}} = \text{diag}(1, 80, 0)$, $R_{\text{la}} = 500$ and $Q_{\text{lo}} = 1$, $R_{\text{lo}} = 100$ for the lateral and longitudinal controller, respectively. Furthermore, the gain of the virtual velocity reference generator is $k_v = 0.1$.

To demonstrate adaptability of the proposed control algorithm, we compare three scenarios, where the vehicle is required to track the same predefined lemniscate trajectory.

- In Scenario 1, we show that the nominal controller can track the predefined reference precisely if there is no significant model mismatch.
- Next, in Scenario 2, we present that changing the model parameters can lead to a significant degradation in the tracking performance.
- Finally, Scenario 3 demonstrates the improved tracking accuracy of the learning-based controller.

All three scenarios are depicted in Fig.3.

The training of the learning components has been carried out in a two-step process. First, an initial training round is performed on a manually designed reference trajectory. Then, we evaluate the trained GPs on a grid defined by the training input to obtain the variance for each point. Based on this variance, we design a second training trajectory which includes the state points characterized by the highest uncertainty to provide additional training data for fine tuning. This two-step approach is proven to be beneficial as the maximal variance has successfully been decreased from $\bar{\Sigma}_{\text{lo}}^{(1)} = 0.8497$ and $\bar{\Sigma}_{\text{la}}^{(1)} = 2.0858$ to $\bar{\Sigma}_{\text{lo}}^{(2)} = 0.4178$ and $\bar{\Sigma}_{\text{la}}^{(2)} = 0.5916$. The learning components have been implemented using the SGP regression outlined in Section II-B. For the training, we have utilized the GPyTorch [28] package with $M = 20$ inducing points, while the total size of the training set has been $N = 3500$.

As depicted in Fig. 3 the learning-based adaptive controller has been successful in compensating for the unknown dynamics between the nominal and the altered models. Furthermore, we can also highlight that the GP augmentation has been capable of capturing the model mismatch between the nonlinear and the simplified control-oriented model, so Scenario 3 outperforms even Scenario 1. Overall, the obtained results suggest that the proposed method is suitable for hardware implementation.

C. $\mathcal{L}_2$-*gain performance analysis*

To give a quantifiable measure of performance, we have performed the iterative $\mathcal{L}_2$ gain calculation method of Section V. For the analysis, we have used the following bounds for $\mathcal{X}$ and $\mathcal{W}$:

$$x_1 \in [-0.2, 0.2], \quad x_2 \in [-0.2, 0.2], \quad x_3 \in [-0.5, 0.5],$$
$$x_4 \in [-0.5, 0.5], \quad x_5 \in [-0.75, 0.75], \quad x_6 \in [-3.5, 3.5],$$
$$w_1 \in [-1.44, 1.44], \quad w_2 \in [-0.75, 0.75],$$

which were selected such that they contain all state trajectories obtained from the simulations. Furthermore, 10 Hz has been selected as the cutoff frequency of the low-pass shaping filter, and $v_{\text{cent}}^{\text{r}} = 1.25$ m/s has been chosen as the central velocity for the equilibrium calculation.

The overall algorithm has been implemented in Python. The convex problem of the learner has been formulated with CVXPY [27] and solved with Mosek. The verifier nonlinear program has been implemented and solved using CasADi [29]. $\mathcal{X}$ and $\mathcal{W}$ have been considered with an intial equidistant grid of 5 points along each dimension. After 73 iterations, we have obtained a valid V storage function and the corresponding upper bound for the induced $\mathcal{L}_2$ gain:

$$\gamma = 0.104. \tag{30}$$

Finite $\mathcal{L}_2$-gain implies stability and bounded tracking errors in the operating region $\mathcal{X} \times \mathcal{W}$. Furthermore, given the small magnitude of the gain, the controllers are capable of practically rejecting the external disturbance, which implies adequate tracking performance.

VII. CONCLUSION

In this paper, a learning-based trajectory tracking approach has been proposed to reliably track reference motion trajectories with autonomous car-like mobile robots. By decoupling the nonlinear vehicle dynamics and augmenting the subsystems with GPs, we have been able to efficiently capture and compensate large modeling uncertainties. As a result, compared to the state-of-the-art MPC methods, our feedforward-feedback algorithm offers computational efficiency and easy integration, as it does not require online optimization in the control loop. Furthermore, we provided an estimate for the upper bound of the induced $\mathcal{L}_2$-gain of the nonlinear closed-loop system to give global performance guarantees.

In the future, we aim to further enhance the performance of our controllers with systematically designed learning trajectories based on the uncertainty of the GPs. Furthermore, we plan to increase the adaptability of our algorithm by recursive update of the SGPs.

REFERENCES

[1] J. Betz, H. Zheng, A. Liniger, U. Rosolia, P. Karle, M. Behl, V. Krovi, and R. Mangharam, "Autonomous vehicles on the edge: A survey on autonomous vehicle racing," *IEEE Open Journal of Intelligent Transportation Systems*, vol. 3, pp. 458–488, 2022.

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

[3] B. A. H. Vicente, S. S. James, and S. R. Anderson, "Linear system identification versus physical modeling of lateral–longitudinal vehicle dynamics," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 3, pp. 1380–1387, 2021.

[4] S. Vaskov, R. Quirynen, M. Menner, and K. Berntorp, "Friction-adaptive stochastic predictive control for trajectory tracking of autonomous vehicles," in *Proc. of the American Control Conference*, 2022, pp. 1970–1975.

[5] Y. Kebbati, V. Puig, N. Ait-Oufroukh, V. Vigneron, and D. Ichalal, "Optimized adaptive mpc for lateral control of autonomous vehicles," in *Proc. of 9th International Conference on Control, Mechatronics and Automation*, 2021, pp. 95–103.

[6] X. Ji, X. He, C. Lv, Y. Liu, and J. Wu, "Adaptive-neural-network-based robust lateral motion control for autonomous vehicle at driving limits," *Control Engineering Practice*, vol. 76, pp. 41–53, 2018.

[7] J. Kabzan, L. Hewing, A. Liniger, and M. N. Zeilinger, "Learning-based model predictive control for autonomous racing," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3363–3370, 2019.

[8] L. Hewing, A. Liniger, and M. N. Zeilinger, "Cautious nmpc with gaussian process dynamics for autonomous miniature race cars," in *Proc. of the European Control Conference*, 2018, pp. 1341–1348.

[9] Y. Liu, P. Wang, and R. Tóth, "Learning for predictive control: A dual gaussian process approach," *arXiv preprint arXiv:2211.03699*, 2022.

[10] J. Becker, N. Imholz, L. Schwarzenbach, E. Ghignone, N. Baumann, and M. Magno, "Model- and acceleration-based pursuit controller for high-performance autonomous racing," in *Proc. of IEEE International Conference on Robotics and Automation*, 2023, pp. 5276–5283.

[11] S. Chen, M. Fazlyab, M. Morari, G. J. Pappas, and V. M. Preciado, "Learning lyapunov functions for hybrid systems," in *Proc. of the 24th International Conference on Hybrid Systems: Computation and Control*, New York, NY, USA, 2021.

[12] M. O'Kelly, H. Zheng, D. Karthik, and R. Mangharam, "F1tenth: An open-source evaluation environment for continuous control and reinforcement learning," in *Prof. of the NeurIPS 2019 Competition and Demonstration Track*, ser. Proceedings of Machine Learning Research, vol. 123. PMLR, 08–14 Dec 2020, pp. 77–89.

[13] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. The MIT Press, 11 2005.

[14] P. Antal, T. Péni, and R. Tóth, "Backflipping with miniature quadcopters by gaussian-process-based control and planning," *IEEE Transactions on Control Systems Technology*, pp. 1–12, 2023.

[15] M. Titsias, "Variational learning of inducing variables in sparse gaussian processes," in *Proc. of the 12th International Conference on Artificial Intelligence and Statistics*, vol. 5. PMLR, Apr 2009, pp. 567–574.

[16] K. Floch, "Model-based motion control of the f1tenth autonomous electrical vehicle," Bachelor's thesis, Budapest University of Technology and Economics, 2022. [Online]. Available: https://eprints.sztaki.hu/10544/

[17] A. Liniger, A. Domahidi, and M. Morari, "Optimization-based autonomous racing of 1:43 scale rc cars," *Optimal Control Applications and Methods*, vol. 36, no. 5, pp. 628–647, 2015.

[18] R. Tóth, *Modeling and Identification of Linear Parameter-Varying Systems*. Springer Berlin, Heidelberg, 2010.

[19] A. Gupta, M. Nilsson, P. Falcone, E. Klintberg, and L. J. Mårdh, "A framework for vehicle lateral motion control with guaranteed tracking and performance," in *Proc. of the IEEE Intelligent Transportation Systems Conference*, 2019, pp. 3607–3612.

[20] J. M. Snider, "Automatic steering methods for autonomous automobile path tracking," Robotics Institute, Pittsburgh, PA, Tech. Rep., 2009.

[21] C. Hu, R. Wang, F. Yan, and N. Chen, "Should the desired heading in path following of autonomous vehicles be the tangent direction of the desired path?" *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 6, pp. 3084–3094, 2015.

[22] G. M. Hoffmann, C. J. Tomlin, M. Montemerlo, and S. Thrun, "Autonomous automobile trajectory tracking for off-road driving: Controller design, experimental validation and racing," in *Proc. of the American Control Conference*, 2007, pp. 2296–2301.

[23] D. Gángó, T. Péni, and R. Tóth, "Learning based approximate model predictive control for nonlinear systems," *IFAC-PapersOnLine*, vol. 52, no. 28, pp. 152–157, 2019.

[24] F. Wu, "Control of linear parameter varying systems," Ph.D. dissertation, University of California at Berkeley, 1995.

[25] A. van der Schaft, *L_2-Gain and Passivity Techniques in Nonlinear Control*. Springer Cham, 2017.

[26] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 5026–5033.

[27] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016.

[28] J. R. Gardner, G. Pleiss, D. Bindel, K. Q. Weinberger, and A. G. Wilson, "Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration," in *Advances in Neural Information Processing Systems*, 2018.

[29] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi – A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.