Online Learning of Interaction Dynamics with Dual Model Predictive Control for Multi-Agent Systems Using Gaussian Processes

T.M.J.T. Baltussen, E. Lefeber, R. Tóth, W.P.M.H. Heemels, A. Katriniok

***Abstract*— The control of a single agent in complex and uncertain multi-agent environments requires careful consideration of the interactions between the agents. In this context, this paper proposes a dual model predictive control (MPC) method using Gaussian process (GP) models for multi-agent systems. While Gaussian process MPC (GP-MPC) has been shown to be effective in predicting the dynamics of other agents, current methods do not consider the influence of the control input on the covariance of the predictions, and hence lack the dual control effect. Therefore, we propose a dual MPC that directly optimizes the actions of the ego agent, and the belief of the other agents by jointly optimizing their state trajectories as well as the associated covariance while considering their interactions through a GP. We demonstrate our GP-MPC method in a simulation study on autonomous driving, showing improved prediction quality compared to a baseline stochastic MPC. The results show that GP-MPC can learn the interactions between the agents online, demonstrating the potential of GPs for dual MPC in uncertain and unseen scenarios.**

***Keywords* — Predictive control for nonlinear systems, Autonomous systems, Statistical learning**

I. Introduction

Many real-world multi-agent systems are complex and unsuitable for centralized control approaches and require distributed or decentralized controllers [1]. Examples include autonomous vehicles (AVs), robotics, aerial vehicles and energy systems, which often have coupled dynamics, constraints or policies, making these systems *interactive*. The control of a single agent requires the awareness and consideration of these interactions with other agents in its environment to achieve safety and performance without relying on communication between the agents. In the case of AVs, the control of such an agent includes maneuvering and avoiding collisions with other agents. As AVs often lack explicit communication, we must deduce and consider these uncertain interactions when planning the AV's motion [2].

Model predictive control (MPC) is an interesting paradigm for this local control as it can directly optimize the agent's trajectories while ensuring safety constraints are satisfied. Particularly, stochastic MPC (SMPC) can address uncertainties by incorporating a probabilistic description of model uncertainty into a stochastic optimal control problem (SOCP). In SMPC, the control inputs not only affect the system state, but also the probability distribution associated with this state, which is referred to as the dual control effect [3], [4]. This dual control effect is two-fold. Firstly, control policies with the dual control effect causally anticipate future observations, and, hence, will be less conservative in regions of low uncertainty. Secondly, dual control policies can feature an inherent probing effect that facilitates *active learning* [3]. Although solving the SOCP via the Bellman equation naturally yields these aspects of the dual control effect, this is often intractable and requires some approximation which may eliminate the dual control effect [4]. In this work, we propose a learning-based MPC method aimed towards advancing dual MPC in complex and dynamic environments.

Learning-based MPC uses data-driven modeling and machine learning methods to improve the system model and the parameterization of the SOCP. In particular, Gaussian processes (GPs) are an often-used modeling approach for non-parametric learning-based MPC, as they can compensate for complex model mismatch and enable direct assessment of the approximate model uncertainty [5]. As GPs enable universal and differentiable regression [6], joint conditioning of states [7], and tractable uncertainty propagation [8], they have strong potential for interaction-aware dual MPC.

A. Related Work

Dual MPC is shown to be effective in simultaneous identification and control of systems with parametric and structural uncertainties [9]. In addition, sampling-based methods have recently demonstrated the potential of interaction-aware control [10], and active learning in multi-agent environments [11]. However, these methods currently rely on parametric models with fixed model structures. By extending these methods to non-parametric models, a larger class of problems can potentially be addressed. Gaussian process MPC has been used to learn and predict the motion of other agents in autonomous racing [12], [13], and autonomous driving [14]. However, previous works fix the predictions of the other agents [12] or the covariance of these predictions [12]–[14] prior to solving the MPC problem, making the controller agnostic to the interactions. Consequently, these methods lack the dual control effect. While [13], [14] rely on offline training data, using online training data can help to reduce uncertainty and to improve the generalizability of learning-based control in unseen scenarios, as can be seen in [12].

B. Contributions

In this paper, we present a novel learning-based dual MPC method to learn the unknown dynamics and/or policy of other agents and bridge the previously identified research gap through the following contributions: (i) We extend GP-MPC for the local control of a single agent in a multi-agent

All authors are with the Eindhoven University of Technology, The Netherlands. Roland Tóth is also with the Systems and Control Laboratory, HUN-REN Institute for Computer Science and Control, Hungary. {t.m.j.t.baltussen, a.a.j.lefeber, r.toth, m.heemels, a.katriniok}@tue.nl

environment by including the dual control effect through a custom GP implementation. Although we do not incentivize active learning, our MPC explicitly considers the joint covariance of the agents in its optimization. The MPC leverages the joint probability of the GP to adapt the state covariance of the other agents through the dual control effect, using Bayesian inference to predict the interactions between the ego agent and the other agents. (ii) Our GP-MPC uses online measurements to update the GP. Through online learning, the GP-MPC can adapt its predictions and handle uncertain and unseen behavior of other agents. (iii) We apply the proposed method in a simulation study on lane merging with an autonomous vehicle. In this study, the GP-MPC learns the closed-loop policy of another agent and adapts the constraints based on its confidence. To the authors' best knowledge, this is the first work that uses GPs to learn and predict the interactions between agents online via dual MPC.

II. Problem Formulation

We consider a multi-agent system of $n_a \in \mathbb{N}_{\geq 2}$ agents and the task of devising a control policy for the ego Agent 0, which is described by a discrete-time dynamical system

$$x_{k+1}^0 = f^0(x_k^0, u_k^0) \tag{1}$$

that is subject to state and input constraints, $x_k^0 \in \mathbb{X}^0 \subseteq \mathbb{R}^{n_x}$, $u_k^0 \in \mathbb{U}^0 \subseteq \mathbb{R}^{n_u}$ at discrete time step $k \in \mathbb{N}$. We assume to have perfect knowledge of the function f^0. The dynamics of Agent $j \in \mathcal{J}$, where $\mathcal{J} = \{1, 2, \dots, n_a - 1\}$, are composed of a known, nominal function f^j, and, an unknown and uncertain, residual function g^j that influences the state via the full column rank matrix $B^j \in \mathbb{R}^{n_x \times n_g}$:

$$x_{k+1}^j = f^j\left(x_k^j\right) + B^j g^j\left(\mathbf{x}_k, \mathbf{u}_k\right) + w_k^j, \tag{2}$$

where $\mathbf{x}_k = \operatorname{vec}\left(x_k^0, x_k^1, \dots, x_k^{n_a-1}\right)$ is the joint state vector, and $\mathbf{u}_k = \operatorname{vec}\left(u_k^0, u_k^1, \dots, u_k^{n_a-1}\right)$ is the joint input vector, and $w_k^j \in \mathbb{R}^{n_x}$ is a Gaussian disturbance process. Note that the residual dynamics g^j depends on $\mathbf{x}_k$ and $\mathbf{u}_k$, making the agents *interactive*. We assume that the control input of Agent $j \in \mathcal{J}$ is composed of a closed-loop state feedback policy:

$$u_k^j = \kappa^j(\mathbf{x}_k). \tag{3}$$

As Agent 0 shares its environment with Agent $j \in \mathcal{J}$, we have to consider their coupled safety constraints that define a safe set $\mathbb{X}_k^j\left(x_k^j\right) = \{x_k^0 \in \mathbb{R}^{n_x} \mid h^j\left(x_k^0, x_k^j\right) \leq 0\}$. We assume that f, g, h and κ are continuously differentiable functions. We consider the problem of finding a controller that safely controls Agent 0 in the sense that $x_k^0 \in \bigcap_{j \in \mathcal{J}} \mathbb{X}_k^j\left(x_k^j\right) \cap \mathbb{X}^0$.

III. Dual Stochastic Model Predictive Control

A. Modeling Interactions Between Agents

In this section, we propose a learning-based dual MPC for the control of Agent 0. Since we do not have full knowledge of the dynamics of Agent $j \in \mathcal{J}$, nor of its policy, i.e., of its intentions, it is essential that we consider the uncertainty of the residual dynamics g^j and their policy κ^j in the formulation of our MPC formulation in order to realize the safety of the controller as these interactions will affect the safe set $\mathbb{X}_k^j(x_k^j)$. As we only have partial information of the other agents, we predict their future states through Bayesian inference using GP regression. Firstly, we collect measurements of the states of all agents and the ego's input $\mathbf{z}_k = [\mathbf{x}_k^\top, u_k^{0\top}]^\top \in \mathbb{R}^{n_z}$, and we collect samples of the residual dynamics $y_k^j \in \mathbb{R}^{n_g}$ as follows:

$$y_k^j = B^{j^\dagger}\left(x_{k+1}^j - f^j(x_k)\right) = g^j\left(\mathbf{x}_k, \kappa^j(\mathbf{x}_k), u_k^0\right) + \tilde{w}_k^j, \tag{4}$$

with $B^{j^\dagger} = \left(B^{j\top} B^j\right)^{-1} B^{j\top}$ and $\tilde{w}_k^j \sim \mathcal{N}(0, \Sigma_w^j)$. Then, we use the collected training data $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_{n_D}] \in \mathbb{R}^{n_z \times n_D}$, $\mathbf{y}^j = [y_1^j, y_2^j, \dots, y_{n_D}^j] \in \mathbb{R}^{n_g \times n_D}$ in order to approximate the residual dynamics by a GP $\mathbf{d}^j : \mathbb{R}^{n_z} \to \mathbb{R}^{n_g}$ such that:

$$g^j\left(\mathbf{x}_k, \kappa^j(\mathbf{x}_k), u_k^0\right) \approx \mathbf{d}^j\left(\mathbf{z}_k\right). \tag{5}$$

In order to predict the future states of the other agents, we condition the posterior of the GP on the training data $\mathcal{D} = \left\{\mathbf{Z}, \left(\mathbf{y}^j\right)_{j=1}^{n_a-1}\right\}$. For the sake of simplicity, we confine ourselves to scalar GPs ($n_g = 1$). Typically, multiple variables are predicted with $n_g \leq n_x$ independent GPs. We refer the interested reader to [8], for details on vectorial GPs.

B. Gaussian Process Regression

Let us focus on predicting the dynamics of a single Agent $j \in \mathcal{J}$ and omit the superscript j for the sake of readability. Firstly, we impose a prior distribution on the GP $d\left(\mathbf{z}\right) \in \mathbb{R}$ through a user-defined kernel function $k\left(\mathbf{z}, \mathbf{z}'\right)$ [6]. As we use the GP for model augmentation, we employ a zero mean prior. We assume that the measured training outputs $\mathbf{y}$, and the prior distribution of the GP $d\left(\mathbf{z}\right)$ are jointly Gaussian:

$$\begin{bmatrix} \mathbf{y} \\ d\left(\mathbf{z}\right) \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K_{\mathbf{Z}\mathbf{Z}} + I\sigma_w^2 & \mathbf{k}_{\mathbf{Z}\mathbf{z}} \\ \mathbf{k}_{\mathbf{z}\mathbf{Z}} & k_{\mathbf{z}\mathbf{z}} s^2 \end{bmatrix}\right), \tag{6}$$

where $k_{\mathbf{z}\mathbf{z}'} = k\left(\mathbf{z}, \mathbf{z}'\right)$, $\mathbf{k}_{\mathbf{Z}\mathbf{z}} \in \mathbb{R}^{n_D}$ is the concatenation of the kernel function evaluated at the test point $\mathbf{z}$ and the training set $\mathbf{Z}$, where $[\mathbf{k}_{\mathbf{Z}\mathbf{z}}]_i = k\left(\mathbf{z}_i, \mathbf{z}\right)$ and $\mathbf{k}_{\mathbf{Z}\mathbf{z}}^\top = \mathbf{k}_{\mathbf{z}\mathbf{Z}}$, and $K_{\mathbf{Z}\mathbf{Z}} \in \mathbb{R}^{n_D \times n_D}$ is a Gram matrix, which satisfies $[K_{\mathbf{Z}\mathbf{Z}'}]_{ij} = k\left(\mathbf{z}_i, \mathbf{z}_j'\right)$. The Gaussian noise on $\mathbf{y} \in \mathbb{R}^{1 \times n_D}$ is typically handled by regularizing the Gram matrix with the noise variance σ_w^2, while the predictions on d are noise free. Marginalizing the jointly Gaussian prior over the training outputs $\mathbf{y}$ yields the GP posterior [6], which is our conditioned belief of $d(\mathbf{z})$ due to the observations $\mathcal{D}$:

$$\Pr\left(d(\mathbf{z}) \mid \mathbf{z}, \mathcal{D}\right) = \mathcal{N}\left(\mu^d\left(\mathbf{z}\right), \Sigma^d\left(\mathbf{z}\right)\right), \tag{7}$$

where the posterior mean and covariance functions read as:

$$\mu^d(\mathbf{z}) = \mathbf{k}_{\mathbf{z}\mathbf{Z}}\left(K_{\mathbf{Z}\mathbf{Z}} + I\sigma_w^2\right)^{-1}\mathbf{y}, \tag{8a}$$

$$\Sigma^d(\mathbf{z}) = k_{\mathbf{z}\mathbf{z}} - \mathbf{k}_{\mathbf{z}\mathbf{Z}}\left(K_{\mathbf{Z}\mathbf{Z}} + I\sigma_w^2\right)^{-1}\mathbf{k}_{\mathbf{Z}\mathbf{z}}. \tag{8b}$$

Subsequently, we query the posterior of the GP $d^j(\mathbf{z})$, at a test point $\mathbf{z} \in \mathbb{R}^{n_z}$, to predict the residual dynamics g^j of Agent j over a time prediction horizon $i = 0, 1, \dots, N-1$:

$$x_{i+1|k}^j = f^j\left(x_{i|k}^j\right) + B^j d^j\left(\mathbf{z}_{i|k}\right), \tag{9a}$$

$$d^j\left(\mathbf{z}\right) \sim \mathcal{N}\left(\mu^{d^j}\left(\mathbf{z}\right), \Sigma^{d^j}\left(\mathbf{z}\right)\right), \tag{9b}$$

where $x_{i|k}$ denotes the i-th step-ahead prediction of x_{k+i} made at time step k given the data $\mathcal{D}_k$, where μ^{d^j} and Σ^{d^j} denote the predicted mean and covariance function of the posterior distribution of the GP at a given test point $\mathbf{z}$.

C. Gaussian Process Prediction Model

We exploit the joint probability of the GP d^j to account for the uncertainty in our safety constraint set $\mathbb{X}^j$ by formulating *chance constraints* that should be satisfied with a probability $p_x \in (0,1]$. To this end, we optimize the control sequence of Agent 0 $U_k = (u^0_{0|k}, \dots, u^0_{N-1|k})$ in the following SOCP:

$$\min_{U_k} \quad J\big(x^0_k, U_k\big) \tag{10a}$$
$$\text{s.t. } x^0_{i+1|k} = f^0\big(x^0_{i|k}, u^0_{i|k}\big), \quad i = 0,1,\dots,N-1, \tag{10b}$$
$$x^j_{i+1|k} = f^j\big(x^j_{i|k}\big) + B^j d^j\big(\mathbf{z}_{i|k}\big) \quad i = 0,1,\dots,N-1, \quad j \in \mathcal{J}, \tag{10c}$$
$$\Pr\Big(x^0_{i|k} \in \mathbb{X}^j_{i|k}\big(x^j_{i|k}\big)\Big) \geq p_x, \quad i = 1,2,\dots,N, \quad j \in \mathcal{J} \tag{10d}$$
$$x^0_{i|k} \in \mathbb{X}^0, \quad i = 1,2,\dots,N, \tag{10e}$$
$$u^0_{i|k} \in \mathbb{U}^0, \quad i = 0,1,\dots,N-1, \tag{10f}$$
$$x^j_{0|k} = x^j_k, \quad j \in \{0 \cup \mathcal{J}\}. \tag{10g}$$

Note that the posterior GP $d^j(\mathbf{z})$ is a function of the control sequence U_k. As SOCP (10) is generally intractable, we subsequently apply tractable approximations for consecutive evaluations of the GP and the chance constraints at the expense of formal safety guarantees. In order to propagate the covariance of the predicted state Σ^{x^j}, we make the following assumptions. We assume that consecutive GP evaluations are independent and the predicted state x^j and posterior GP $d^j(\mathbf{z})$ are assumed to be jointly Gaussian at each prediction step:

$$\begin{bmatrix} x^j_i \\ d^j(\mathbf{z}_i) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu^{x^j}_i \\ \mu^{d^j}(\mathbf{z}_i) \end{bmatrix}, \begin{bmatrix} \Sigma^{x^j}_i & \Sigma^{x^j d^j}(\mathbf{z}_i) \\ \Sigma^{d^j x^j}(\mathbf{z}_i) & \Sigma^{d^j}(\mathbf{z}_i) \end{bmatrix} \right), \tag{11}$$

such that the covariance can be approximated and propagated over the horizon. We obtain $\mu^{d^j}, \Sigma^{x^j d^j}, \Sigma^{d^j}$ through a first-order Taylor approximation of the GP evaluated at its posterior mean [15], which provides a good trade-off between approximation accuracy and computational complexity [8]:

$$\mu^{d^j}_i = \mu^{d^j}\left(\mu^{\mathbf{z}}_i\right), \tag{12a}$$
$$\Sigma^{x^j d^j}_i = \Sigma^{x^j}_i \left(\nabla^\top_{x^j} \mu^{d^j}\left(\mu^{\mathbf{z}}_i\right)\right)^\top, \tag{12b}$$
$$\Sigma^{d^j}_i = \Sigma^{d^j}\left(\mu^{\mathbf{z}}_i\right) + \nabla^\top_{x^j} \mu^{d^j}\left(\mu^{\mathbf{z}}_i\right) \Sigma^{x^j}_i \left(\nabla^\top_{x^j} \mu^{d^j}\left(\mu^{\mathbf{z}}_i\right)\right)^\top, \tag{12c}$$

where $\mu^{\mathbf{z}}_i$ denotes the mean value of the test point $\mathbf{z}_i$ at prediction step i, with $\mu^{d^j}(\mathbf{z})$ and $\Sigma^{d^j}(\mathbf{z})$ as in (8). We propagate the mean and covariance of the posterior GP (8) over the horizon through the following approximation [8]:

$$\mu^{x^j}_{i+1|k} = f^j\left(\mu^{x^j}_{i|k}\right) + B^j \mu^{d^j}\left(\mu^{\mathbf{z}}_{i|k}\right), \tag{13a}$$
$$\Sigma^{x^j}_{i+1|k} = \left[\nabla^\top f^j\left(\mu^{x^j}_{i|k}\right) \; B^j\right] \Sigma^j_{i|k} \left[\nabla^\top f^j\left(\mu^{x^j}_{i|k}\right) \; B^j\right]^\top, \tag{13b}$$

for $i = 0,1,\dots,N-1$, where the initial prediction equals the current state $\mu^{x^j}_{0|k} = x^j_k$ with $\Sigma^{x^j}_{0|k} = \mathbf{0}$, and $\Sigma^j_{i|k}$ denotes the joint covariance matrix (11). We approximate the full GP by a Sparse Pseudo-Input GP (SPGP) [16], which exploits the Nyström projection to reduce the computational complexity of the GP by preconditioning the GP on *inducing points*. For details on SPGPs, we refer the reader to [16]. We select the inducing points by taking equidistant samples from the predicted trajectory at time $k-1$, since the test points can be expected to be close to this trajectory [8].

D. Learning-based Dual Model Predictive Control

In this section, we propose our dual MPC that uses the GP prediction model, presented in Sec. III-C, to jointly condition the interactions of the other agents with the control inputs of Agent 0. The dual control effect was first formalized in [3], and according to [4] is defined as follows:

Definition 3.1: A control input is said to have the dual control effect if it can affect, with non-zero probability, at least one r^{th}-order central moment of a state variable ($r \geq 2$).

We formulate a tractable dual MPC to devise a control policy for Agent 0 and model the interactions g^j by an SPGP:

$$\min_{U_k} \; J\big(x^0_k, U_k\big) \tag{14a}$$
$$\text{s.t. } x^0_{i+1|k} = f^0\big(x^0_{i|k}, u^0_{i|k}\big), \quad i = 0,1,\dots,N-1, \tag{14b}$$
$$\mu^{x^j}_{i+1|k} = f^j\left(\mu^{x^j}_{i|k}\right) + B^j \mu^{d^j}\left(\mu^{\mathbf{z}}_{i|k}\right), \quad i = 0,1,\dots,N-1, \quad j \in \mathcal{J}, \tag{14c}$$
$$\Sigma^{x^j}_{i+1|k} = \left[\nabla^\top f^j\left(\mu^{x^j}_{i|k}\right) \; B^j\right] \Sigma^j_{i|k} \left[\nabla^\top f^j\left(\mu^{x^j}_{i|k}\right) \; B^j\right]^\top, \quad i = 0,1,\dots,N-1, \quad j \in \mathcal{J}, \tag{14d}$$
$$u^0_{i|k} \in \mathbb{U}^0, \quad i = 0,1,\dots,N-1, \tag{14e}$$
$$x^0_{i|k} \in \tilde{\mathbb{X}}^j_{i|k}\left(\mu^{x^j}_{i|k}, \Sigma^{x^j}_{i|k}\right), \quad i = 1,2,\dots,N, \quad j \in \mathcal{J}, \tag{14f}$$
$$x^0_{i|k} \in \mathbb{X}^0, \quad i = 1,2,\dots,N, \tag{14g}$$
$$x^0_{0|k} = x^0_k, \tag{14h}$$
$$\mu^{x^j}_{0|k} = x^j_k, \quad j \in \mathcal{J}, \tag{14i}$$
$$\Sigma^{x^j}_{0|k} = \mathbf{0}, \tag{14j}$$

where the chance constraints from (10d) are approximated by a general tightened constraint set $\tilde{\mathbb{X}}^j\big(\mu^{x^j}, \Sigma^{x^j}\big) \subseteq \mathbb{X}^j\big(\mu^{x^j}\big)$, based on the covariance of the GP [8]. The specific form of constraint tightening may depend on the nature of the problem. An example hereof is provided in Sec. IV. The dual MPC exploits the joint conditioning of the predicted states of all the agents through the regression feature $\mathbf{z}$ of the GP. Consequently, the control sequence U_k explicitly affects the constraint tightening ($\tilde{\mathbb{X}}^j$) through Bayesian inference. The dual control effect provides an inherent degree of caution based on the correlation with past observations [3], which is essential to a truly interaction-aware controller. To this end, we make the following proposition.

Proposition 3.2: The control input sequence of Agent 0 (U^*_k) resulting from the locally optimal solution to the MPC problem (14) affects both the trajectory of Agent 0 and the prediction of Agent $j \in \mathcal{J}$ over the horizon. Moreover, U^*_k affects the predicted covariance of the state of Agent $j \in \mathcal{J}$, and, thus the control input u^{0*}_k has the dual control effect.

Proof: By construction, the control sequence resulting from (14) $U^*_{0:i-1|k}$ determines the predicted future states of Agent 0 $x^0_{i|k}$, and, consequently, the predicted test points $\mathbf{z}_{i|k}$ of the GP. Therefore, the control sequence $U^*_{0:i-1|k}$ affects the posterior mean $\mu^{x^j}_{i|k}$ (14c) and posterior covariance $\Sigma^{x^j}_{i|k}$ (14d) of the predicted states of Agent $j \in \mathcal{J}$ at time k such that

$$\Sigma^{x^j}_{i|k} \triangleq \mathbb{E}\big[\Sigma^{x^j}_i \mid \mathcal{D}_k, U^*_{0:i-1|k}\big] \neq \mathbb{E}\big[\Sigma^{x^j}_i \mid \mathcal{D}_k\big] \tag{15}$$

for $i = 1,2,\dots,N$. Hence, u^{0*}_k has the dual control effect. ■

IV. Interaction-Aware Motion Planning

A. Lane Merging Use Case

In this section, we demonstrate the proposed GP-MPC method for the motion planning of an autonomous vehicle in a forced lane merging use case, and compare it against a stochastic baseline MPC. This simple use case promotes the qualitative interpretability of GP-MPC. Here, the target lane is occupied by a leading and a following vehicle, as seen in Fig. 1 in Sec. V. The Ego vehicle is controlled by the GP-MPC from (14). The Follower's interactive driving policy follows a Merge-Reactive Intelligent Driver Model (MR-IDM) [17]. The Follower tries to close the gap with its Leader, hindering the Ego from merging. We assume the target vehicles remain in their lane center. The Ego, Follower, and Leader are denoted by $j = 0, 1, 2$, respectively. For simplicity, we learn only the residual dynamics of the Follower (g^1), assuming the Leader drives at constant speed ($g^2 = 0$), and assume the disturbance processes w_k^j are zero.

B. Vehicle Modeling

All vehicles are modeled by a deterministic, kinematic bicycle model, which is sufficiently accurate for motion planning. The state of a vehicle is described by the state vector $x = \begin{bmatrix} X & Y & v & \psi & \delta \end{bmatrix}^\top \in \mathbb{R}^5$, where X and Y are the longitudinal and lateral positions of the rear axle, respectively. The longitudinal velocity of the rear axle is denoted by v, ψ is the heading angle of the vehicle, and δ is the steering angle of the front axle. The vehicle's acceleration a and the steering rate r are inputs to the system: $u = \begin{bmatrix} a & r \end{bmatrix}^\top \in \mathbb{R}^2$. The continuous-time dynamics of the kinematic bicycle model $\dot{x}(t) = f_c\left(x(t), u\left(t\right)\right)$ are defined by the state-space equations:

$$\dot{X}(t) = v(t)\cos\left(\psi\left(t\right)\right), \tag{16a}$$
$$\dot{Y}(t) = v(t)\sin\left(\psi\left(t\right)\right), \tag{16b}$$
$$\dot{v}(t) = a(t), \tag{16c}$$
$$\dot{\psi}(t) = \tfrac{v(t)}{l}\tan\left(\delta\left(t\right)\right), \tag{16d}$$
$$\dot{\delta}(t) = r(t), \tag{16e}$$

where $t \in \mathbb{R}$ denotes time and l denotes the vehicle's wheelbase. We discretize (16) with the fourth-order Runge-Kutta method using a sampling time of $T_s = 0.25$ [s].

The Follower's MR-IDM policy [17] is a driver model that considers agents merging into its lane. This closed-loop policy $a^1 = a_{\text{MR-IDM}}\left(\mathbf{x}, a^0, a^2\right)$ couples the dynamics of the Ego and the Follower. In order to utilize the framework outlined in Sec. III, we reformulate this policy as a function of $\mathbf{x}_k$ and u^0 through (3). For simulation purposes, we assume the Follower has access to a^0 and $a^2 = 0$. The MR-IDM maps the relative longitudinal and lateral position to an effective distance gap for an IDM. The vehicle inducing the largest deceleration determines the output of the MR-IDM. For details on the MR-IDM we refer the reader to [17].

C. Motion Planning Problem

We incentivize the Ego to maintain its initial velocity v_0^0 and stay in its lane, until the merge lane closes and the MPC decides that we should merge. We use a smooth function $m\left(X\right) : \mathbb{R} \to \mathbb{R}$ to describe the center of the merge lane as a function of the longitudinal position, which coincides with the center of the target lane after the merge lane has fully closed. Accordingly, the Ego's reference x^r is defined as:

$$x^r\left(X\right) = \begin{bmatrix} 0 & m(X) & v_0^0 & 0 & 0 \end{bmatrix}^\top. \tag{17}$$

We employ the following objective function:

$$\begin{aligned} J(x_k^0, u_{k-1}^0, U_k) = &\sum_{i=1}^{N} \left\| x_{i|k}^0 - x^r\left(X_{i|k}^0\right)\right\|_Q^2 \\ &+ \sum_{i=0}^{N-1} \left\| u_{i|k}^0 \right\|_R^2 + \left\| \Delta u_{i|k}^0 \right\|_S^2, \end{aligned} \tag{18}$$

where $\|z\|_Q^2 = z^\top Q z$ with positive (semi-) definite weighing matrices $Q, S \succeq 0$, $R \succ 0$ and $\Delta u_{i|k} = u_{i|k} - u_{i-1|k}$ with $u_{-1|k} = u_{k-1}$. We limit the absolute acceleration and steering angle rate to 5 [m/s^2] and 5 [deg/s] through $u \in \mathbb{U}^0$. Constant state constraints partially govern road boundaries and limit the Ego's maximum velocity to 36 [m/s], its absolute heading to 15 [deg] and its absolute steering angle to 5 [deg]. The right road boundary is dependent on its position and is governed through a separate constraint:

$$h_r\big(x_{i|k}^0\big) = m\big(X_{i|k}^0\big) - Y_{i|k}^0 + \tfrac{W-w}{2} \leq 0, \tag{19}$$

where the road boundary is parallel to the center of the merge lane, W and w denote the vehicle and lane width. Consequently, we have the following state constraint set:

$$\mathbb{X}^0 := \{x^0 \in \mathbb{R}^5 \mid x_{\min} \leq x^0 \leq x_{\max},\, h_r\big(x^0\big) \leq 0\}. \tag{20}$$

Similar to Zhu *et al.* [13], we account for the uncertainty in the predicted position of the target vehicle by expanding the semi-axes of elliptical collision avoidance constraints. Again, we assume no uncertainty in their lateral position. By expanding the major semi-axis of the ellipse, we obtain a tractable, *tightened* reformulation of the chance constraints:

$$\begin{aligned} \tilde{\mathbb{X}}_{i|k}^j\big(x_{i|k}^j\big) = \big\{&x_{i|k}^0 \in \mathbb{R}^{n_x} \mid h_c\big(x_{i|k}^0, \mu_{i|k}^{x^j}, \Sigma_{i|k}^{X^j}\big) = \\ &- \frac{\left(c_{x,i|k}^j - c_{x,i|k}^0\right)^2}{\left(\mathcal{E}_{c,A} + \sigma\sqrt{\Sigma_{i|k}^{X^j}}\right)^2} - \frac{\left(c_{y,i|k}^j - c_{y,i|k}^0\right)^2}{\mathcal{E}_{c,B}^2} + 1 \leq 0\big\}, \end{aligned} \tag{21}$$

where $c_{x,i|k}^j$ and $c_{y,i|k}^j$ denote the longitudinal and lateral component of the centroid, respectively, and $\Sigma_{i|k}^{X^j}$ denotes the predicted covariance of the longitudinal position of Agent j at time step $k + i$. The major and minor semi-axis of the ellipse are denoted by $\mathcal{E}_{c,A}$ and $\mathcal{E}_{c,B}$, respectively. The constraint tightening can be tuned with $\sigma \in \mathbb{R}_{\geq 0}$, where $\sigma = 2$ is the amount of standard deviation that is accounted for. In absence of formal feasibility guarantees of the MPC, the collision avoidance constraints are softened using an l_1 penalty function to retain feasibility. Since we assume the Leader maintains its initial velocity, we set $\Sigma^{x^2} = \mathbf{0}$.

D. Constant Velocity Model Predictive Control

The stochastic baseline MPC (CV-MPC) uses a constant velocity prediction model for both the Leader and Follower:

$$v_{i|k}^j = v_k^j, \quad \text{for } i = 0, 1, \ldots, N. \tag{22}$$

Hence, the nominal predictions f^j of the target vehicles are:

$$\mu_{i+1|k}^{x^j} = A\mu_{i|k}^{x^j}, \quad \text{for } i = 0, 1, \ldots, N-1, \quad j = 1, 2, \tag{23}$$

$$\text{where } A = \begin{bmatrix} 1 & 0 & T_s & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{24}$$

The CV-MPC accounts for the Follower's change in velocity Δv^1 by a fixed covariance $\sigma_{v^1}^2 = 0.3$ that we propagate as:

$$\Sigma_{i+1|k}^{x^1} = A\Sigma_{i|k}^{x^1} A^\top + B^1 \sigma_{v^1}^2 B^{1\top}, \tag{25}$$

for $i = 0, 1, \ldots, N-1$, where $B^1 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \end{bmatrix}^\top$.

E. Gaussian Process Model Predictive Control

We use past observations $\mathbf{y}$ and predict the incremental velocity by the posterior of the GP through Bayesian inference. Recall that the MPC only has access to the current state of other agents. Therefore, the GP-MPC predicts the change of velocity by observing the difference between the Follower's actual velocity and the constant velocity prediction:

$$g^1(\mathbf{x}_k, \mathbf{u}_k) = y_k = v_{k+1}^1 - v_k^1 = \Delta v_k^1. \tag{26}$$

We 'select' specific states of interest by applying a linear map C to $\mathbf{z}$, such that we infer these predictions using the velocities and relative positions of the different vehicles:

$$C\,\mathbf{z} = \left[v^0, v^1, v^2, \left(X^1 - X^0\right), \left(X^1 - X^2\right), \left(Y^1 - Y^0\right)\right]^\top. \tag{27}$$

The relative lateral position of the Follower and Leader is assumed to be zero and is omitted from training point $\mathbf{z}$.

We employ the frequently used squared exponential kernel and map the regression features through C, such that:

$$k\left(\mathbf{z}, \mathbf{z}'\right) = \sigma_d^2 \exp\left(-\tfrac{1}{2}\left(C\mathbf{z} - C\mathbf{z}'\right)^\top L_d^{-2}\left(C\mathbf{z} - C\mathbf{z}'\right)\right), \tag{28}$$

where σ_d^2 is the prior covariance and L_d is the length-scale matrix. If sufficient training data is available, these hyper-parameters can be inferred using log-likelihood optimization [6]. Here, we use fixed hyper-parameters for the kernel with $L_d = \text{diag}\,(3, 3, 3, 17, 17, 5)$. The prior covariance is equal to the covariance of the baseline prediction model $\sigma_d^2 = 0.3$, making the CV-MPC equivalent to the GP-MPC prior to inference. Since we do not consider any noise, we set $\sigma_w^2 = 0$.

V. Results

We compare our proposed GP learning-based dual MPC (GP-MPC) with a stochastic baseline MPC that uses a constant velocity prediction (CV-MPC) for the Follower, and evaluate their performance on the lane merging use case detailed in Sec. IV. We test the GP-MPC with and without pre-training. Here, we refer to pre-training as the availability of past measurement data at the start of the experiment.

A. Numerical Case Study

We perform 51 Monte Carlo simulations to assess the prediction capabilities of the GP-MPC and CV-MPC. As initial conditions, we take $X_0^1 = -75$ [m], $v_0^1 = 31$ [m/s] for the Follower, and $X_0^2 = 0$ [m], $v_0^2 = 25$ [m/s] for the Leader to ensure a sufficiently challenging scenario. The initial position of the Ego is equidistantly sampled from $X_0^0 \in [-100, -75]$ with $v_0^0 = 31$ [m/s]. Computations are performed on an Intel i7 with 32 GB RAM, in MATLAB using CasADi [18], with the IPOPT [19] and MA57 [20] solvers. We use a custom GP implementation in CasADi to compute the joint covariance. We take a horizon of $N = 12$, $M = 4$ inducing points for the SPGP and a sampling time of $T_s = 0.25$ [s]. At each time $k+1$, we append the training data: $\mathcal{D}_{k+1} = \mathcal{D}_k \cup (\mathbf{z}_k, \mathbf{y}_k)$. Although we currently do not discard any data points, selection algorithms can help reduce the size of the training set [21]. To assess the generalizability to unseen behavior without pre-training, we take $\mathcal{D}_0 = \emptyset$. With pre-training, we initialize the GP with $n_D = 80$ observations from a single closed-loop experiment with $X_0^0 = -85$ [m].

TABLE I
Results of the Monte Carlo simulations.

Algorithm	Success	$\overline{\|e\|}$ [m/s]	$\overline{T}_c$ [ms]	$T_c < T_s$ [%]
CV-MPC	20/51	0.679	50	100
GP-MPC	33/51	0.645	128	95.6
GP-MPC with pre-training	35/51	0.440	206	69.0

Table I summarizes the results of the Monte Carlo trials. In this study, we consider a merge successful if the Ego merges between the Follower and Leader. The prediction error $\|e(k)\| := \frac{1}{N}\|\{v_{i|k} - v_{k+i}\}_{i=1}^N\|_1$ is the normalized norm of the difference between the predicted trajectory at time k and the realized trajectory from time k to $k+N$. The average prediction error over all time steps and all trials $\overline{\|e\|}$ quantifies the prediction quality of the CV-MPC and GP-MPC. The average solve time $\bar{T}_c$ and the percentage of iterations within sampling time $T_c < T_s$ show that the algorithm can be run (near-)realtime with these amounts of training data.

B. Discussion

The GP-MPC jointly optimizes the predictions and the associated uncertainty through the dual control effect, as seen in Fig. 1, which shows a successful merge by the GP-MPC without pre-training. The approximate covariance of the GP is exploited to account for uncertain interactions providing inherent caution through dual MPC. As the Ego is agnostic to the Follower's response to the merging action, it takes extra caution when merging by further tightening the constraints.

Both the CV-MPC and the GP-MPC without pre-training do not capture the high frequent dynamics during the merge, as seen in Fig. 2. During the merge, approximately after $t = 10$ [s], the GP-MPC without pre-training has an increased prediction error as it has not observed these high frequent dynamics. Still, the GP-MPC has superior prediction quality prior to merging and is able to identify a safe gap to merge in between. The GP-MPC with pre-training exploits past observations from a similar scenario, leveraging Bayesian inference and the dual control effect to exercise less caution and shows increased prediction quality. When compared to CV-MPC, the online learning GP-MPC sees a 25% to 29% increase in successful merges over all experiments without and with pre-training, respectively, cf. Table I. While GPs are attractive for Bayesian inference, the computational complexity increases with the number of training data. This stresses the importance of online learning

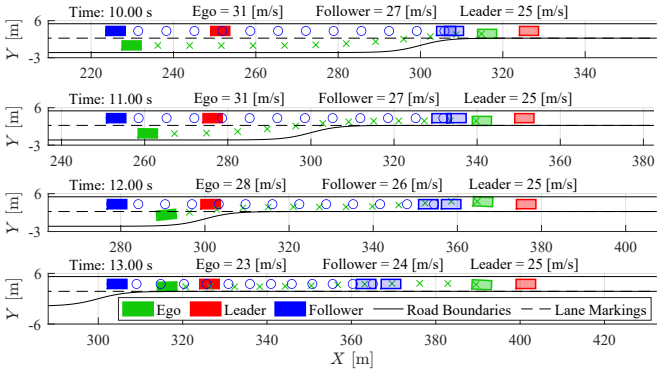

Fig. 1. Time-lapse of a lane merge by the GP-MPC. The transparent rectangles show the final predicted positions of vehicles, while for the Follower two rectangles indicate 2σ-bounds on its final predicted position.

in dynamic environments, such that the MPC does not have to rely on large sets of training data, but rather adapt online. Simulation results show that our GP-MPC can successfully predict the Follower's motion from various initial conditions by only relying on online training data, demonstrating its generalizability beyond a priori available training data.

VI. Conclusions

We presented a Gaussian process-based online learning MPC that uses the dual control effect to learn and predict the dynamics of other agents while accounting for their joint uncertainty. We validated our GP-MPC in a numerical case study on automated lane merging, by leveraging dual MPC to jointly optimize the motion of the ego vehicle and the predictions of a target vehicle through an interactive GP model. By explicitly considering the effect of control inputs on the covariance of predictions of other agents, our GP-MPC can leverage this effect in the tightening of constraints. Through online learning, the GP-MPC can adapt to uncertain and unseen behavior of other agents, showing strong potential for scenarios that extend beyond a priori available training data sets.

The dual control effect is an important notion that can be leveraged by the MPC. While this paper focuses on a proof of concept of the proposed method, the next research steps for learning-based GP-MPC include: (i) incorporating the probing effect of dual control, i.e., active learning, (ii) predicting the residual dynamics of multiple agents, and (iii) providing performance and recursive feasibility guarantees of the MPC. In the context of autonomous driving, planned future work includes validation of the prediction model, the learning of various driving behaviors, and experimental validation.

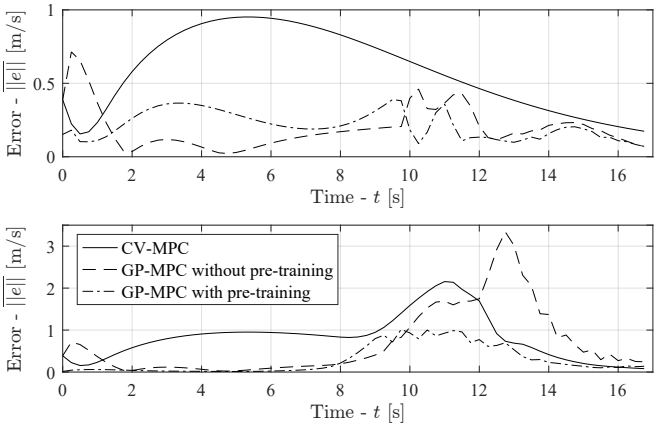

Fig. 2. Average prediction error of Follower's velocity over the prediction horizon in scenarios in which all planners merge behind (average over 16 experiments, shown in the top figure) and merge in between (average over 20 experiments, shown in the bottom figure).

References

[1] D. Q. Mayne, "Model predictive control: Recent developments and future promise," *Automatica*, vol. 50, no. 12, 2014.

[2] W. Schwarting, J. Alonso-Mora, and D. Rus, "Planning and Decision-Making for Autonomous Vehicles," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, no. 1, 2018.

[3] Y. Bar-Shalom and E. Tse, "Dual effect, certainty equivalence, and separation in stochastic control," *IEEE Transactions on Automatic Control*, vol. 19, no. 5, 1974.

[4] A. Mesbah, "Stochastic model predictive control with active uncertainty learning: A Survey on dual control," *Annual Reviews in Control*, vol. 45, 2018.

[5] L. Hewing, K. P. Wabersich, M. Menner, and M. N. Zeilinger, "Learning-Based Model Predictive Control: Toward Safe Learning in Control," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, no. 1, 2020.

[6] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning*. MIT Press, 2006.

[7] P. Trautman and A. Krause, "Unfreezing the robot: Navigation in dense, interacting crowds," in *IEEE/RSJ Int. Conf. on Intell. Robots and Syst. (IROS)*, 2010.

[8] L. Hewing, J. Kabzan, and M. N. Zeilinger, "Cautious Model Predictive Control Using Gaussian Process Regression," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 6, 2020.

[9] E. Arcari, L. Hewing, M. Schlichting, and M. N. Zeilinger, "Dual Stochastic MPC for Systems with Parametric and Structural Uncertainty," in *Proceedings of the 2nd Conference on Learning for Dynamics and Control*, vol. 120. PMLR, Jun. 2020.

[10] R. Wang, M. Schuurmans, and P. Patrinos, "Interaction-aware model predictive control for autonomous driving," in *European Control Conference (ECC)*, 2023.

[11] J. Knaup, J. D'sa, B. Chalaki, T. Naes, H. N. Mahjoub, E. Moradi-Pari, and P. Tsiotras, "Active learning with dual model predictive path-integral control for interaction-aware autonomous highway on-ramp merging," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2024.

[12] T. Brüdigam, A. Capone, S. Hirche, D. Wollherr, and M. Leibold, "Gaussian Process-based Stochastic Model Predictive Control for Overtaking in Autonomous Racing," *arXiv:2105.12236*, 2021.

[13] E. L. Zhu, F. L. Busch, J. Johnson, and F. Borrelli, "A Gaussian Process Model for Opponent Prediction in Autonomous Racing," in *IEEE/RSJ Int. Conf. on Intell. Robots and Syst. (IROS)*, 2023.

[14] J. Bethge, M. Pfefferkorn, A. Rose, J. Peters, and R. Findeisen, "Model Predictive Control with Gaussian-Process-Supported Dynamical Constraints for Autonomous Vehicles," in *22nd IFAC World Congress*. IFAC-PapersOnLine, 2023.

[15] J. Kocijan, *Modelling and Control of Dynamic Systems Using Gaussian Process Models*. Springer International Publishing, 2016.

[16] E. Snelson and Z. Ghahramani, "Sparse Gaussian Processes using Pseudo-inputs," *Adv. in Neural Inf. Processing Syst.*, vol. 18, 2005.

[17] D. Holley, J. D'sa, H. N. Mahjoub, G. Ali, B. Chalaki, and E. Moradi-Pari, "MR-IDM - Merge Reactive Intelligent Driver Model: Towards Enhancing Laterally Aware Car-following Models," in *IEEE Int. Conf. on Intell. Transp. Syst. (ITSC)*, 2023.

[18] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi: a software framework for nonlinear optimization and optimal control," *Math. Progam. Comput.*, vol. 11, no. 1, 2019.

[19] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, no. 1, 2006.

[20] HSL. (2023) Coin-HSL. [Online]. Available: https://licences.stfc.ac.uk/product/coin-hsl

[21] 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, 2019.