Kernel-Based Metrics Learning for Uncertain Opponent Vehicle Trajectory Prediction in Autonomous Racing

Hojin Lee, *Graduate Student Member, IEEE*, Youngim Nam, *Graduate Student Member, IEEE*, Sanghun Lee, and Cheolhyeon Kwon, *Member, IEEE*

***Abstract*—Autonomous racing confronts significant challenges in safely overtaking Opponent Vehicles (OVs) that exhibit uncertain trajectories, stemming from unknown driving policies. To address these challenges, this study proposes heterogeneous kernel metrics for Deep Kernel Learning (DKL), designed to robustly capture the diverse driving policies of OVs, and carry out precise trajectory predictions along with the associated uncertainties. A key virtue of the proposed kernel metrics lies in their ability to align similar driving policies and disjoin dissimilar ones in an unsupervised manner, given the observed interactions between the Ego Vehicle (EV) and OVs. The efficacy of the proposed method is substantiated through experimental studies on a 1/10th scale race-car platform, demonstrating improved prediction accuracy and thereby safely overtaking against OVs. Furthermore, our method is computationally efficient for onboard computing units, affirming its viability in fast-paced racing environments.**

***Index Terms*—Planning under uncertainty, integrated planning and learning, machine learning for robot control.**

I. Introduction

AUTONOMOUS racing has emerged as a significant subfield of autonomous driving, attracting considerable interest and fostering competitions such as Roborace, Indy Autonomous Challenge, and F1TENTH [1]. One of the key challenges in autonomous racing lies in safely running against OVs and executing overtaking maneuvers. Various strategies have been developed to address these challenges [2], [3], among which a commonly adopted approach involves predicting the future trajectory of OV and then planning the overtaking maneuver accordingly [1].

Received 30 May 2024; accepted 16 October 2024. Date of publication 24 October 2024; date of current version 31 October 2024. This article was recommended for publication by Associate Editor Rudolf Lioutikov and Editor Jens Kober upon evaluation of the reviewers' comments. This work was supported by the National Research Foundation of Korea (NRF) funded by the Korea Government (MSIT) under Grant RS-2024-00342930 and Grant 2020R1A5A8018822. *(Corresponding author: Cheolhyeon Kwon.)*

The authors are with the Department of Mechanical Engineering, Ulsan National Institute of Science and Technology, Ulsan 44919, South Korea (e-mail: hojinlee@unist.ac.kr; nyi0944@unist.ac.kr; sanghun17@unist.ac.kr; kwonc@unist.ac.kr).

The video and source code can be found at https://github.com/HMCL-UNIST/OpponentPredictionWithKMDKL.git.

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2024.3486178, provided by the authors.

Digital Object Identifier 10.1109/LRA.2024.3486178

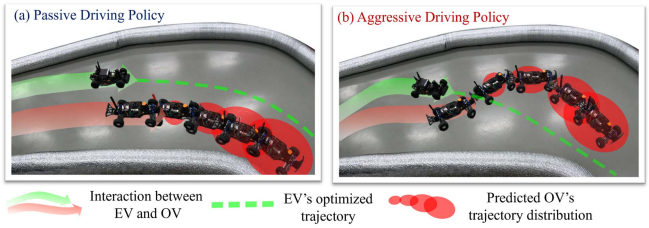

Fig. 1. Overtaking scenarios where the EV maximizes its progress while taking into account the predicted trajectory distribution of the OV, which is based on the driving policy inferred from the observed EV-OV interaction. (a) OV exhibits non-blocking behavior, allowing the EV to navigate without interference; (b) OV actively tries to block the EV's progress.

Although previous studies have made strides in trajectory prediction for autonomous racing, they still face challenges in addressing diverse driving policies of OVs [4]. On the one hand, physics-based prediction methods are grounded in first-principles dynamics models. They are capable of anticipating momentary behavior but struggle to capture the long-term interactions between EV and OVs [4]. On the other hand, planning-based prediction methods infer the OVs' actions as solutions to optimization problems that account for the strategic objectives of the race and tactical interactions with the EV [2]. However, these methods often demand intensive computation, which is critical for fast-paced racing environments. Moreover, the prediction performance heavily relies on the fidelity of the optimization problem, which necessitates encoding the OV's real objective [4]. Lastly, recent advancements in learning-based methods, which empirically examine driving data patterns, have shown promising results in trajectory prediction [5], [6]. Nonetheless, data scarcity often hinders the implementation of learning-based methods within the autonomous racing regime. Moreover, when the encountered driving policy of the OV deviates from the training data, the resulting predictions fall short in planning safe maneuvers of the EV against OV [1].

This paper presents a learning-based method that can address the diverse driving policies of OVs, thereby accurately predicting the OV's trajectory and further assessing the uncertainty of prediction results (See Fig. 1). First, the one-step OV's state prediction model is trained from the observed interaction between the EV and OV through DKL. In detail, a multi-scale Convolutional Neural Network (CNN) is employed to learn the

interactions between the EV and OV, transforming the interaction patterns into latent representations of the OV's driving policy in an unsupervised manner. The represented OV's driving policy is then processed through a Gaussian Process (GP) model to predict the OV's state at the next time step. Building upon this process, we propose novel heterogeneous kernel-based metrics whereby DKL can more adeptly ascertain driving policy from complex interaction patterns. The proposed metrics are particularly devised to assess the similarities across different spaces, i.e., the latent space representing driving policies and the output space depicting the OV's state prediction. Then, these metrics are optimized to render the latent space of DKL such that similar driving policies are clustered closely, while dissimilar ones are simultaneously spread apart. Such a strategic alignment improves the GP model's ability to capture the correlation between the driving policy and the state prediction. As a result, the DKL model not only enhances prediction performance but also addresses calibrated uncertainty concerning both trained and untrained driving policies.

Next, the one-step OV state prediction model is recursively executed to generate multi-step trajectory samples. These samples are utilized to construct the probabilistic OV trajectory distribution, predicting the state at each time step by its mean and variance. Subsequently, the predicted trajectory distribution information is integrated into the EV's Model Predictive Control (MPC) framework as dynamic obstacle constraints. This enables the EV to execute safe and agile overtaking maneuvers while considering the OV's uncertain predicted state distributions. To the best of the authors' knowledge, the proposed method for the first time attempts to apply kernel-based metrics learning to the realm of autonomous racing. Our method marks a notable advancement in OV trajectory prediction and surpasses the existing work in racing against OVs with diverse driving policies. The contributions of this paper are highlighted as follows:

- A novel kernel-based metrics learning framework for DKL is proposed, aimed at representing the diverse driving policies of OVs that are aligned with respect to their similarity.
- A probabilistic OV trajectory prediction model is developed, improving prediction accuracy and uncertainty calibration based on the inferred OV's driving policy.
- An MPC-based planning for EV maneuver is seamlessly integrated with the trajectory prediction results, empowering the EV to safely overtake the OV.
- Extensive real autonomous racing experiments with a 1/10th scale racecar demonstrate the effectiveness of the proposed algorithm, both quantitatively and qualitatively, compared to the existing algorithms.

II. Related Work

A. Physics and Planning-Based Trajectory Predictions

Physics-based trajectory prediction methods exploit the kinematics and dynamics of OVs, tailored by statistical filtering techniques [4]. However, these methods often face challenges in long-term forecasting due to their limited understanding of the interactions between the EV and OVs, implying a lack of consideration for the underlying driving policies of OVs. To account for the interactions, planning-based methods presume that the OVs logically decide their actions through optimization [2]. These methods consider OVs as rational entities seeking to maximize their internal objectives. However, the accuracy of predictions hinges on the coherence between the presumed objective and the true objective of the OV. Furthermore, planning-based methods often demand heavy computations [7], presenting a formidable burden in fast-paced racing circumstances.

B. Learning-Based Trajectory Prediction

Recent trajectory prediction research has primarily shifted towards learning-based methods that account for the complex interactions involving human social behavior [5] and multiple vehicles in urban settings [6]. Nevertheless, their applicability degenerates when situated in autonomous racing, mostly due to deficient datasets in both quantity (limited number of racing scenes) and quality (lack of semantic cues) [8]. Moreover, the diversity in driving policies across different OVs further complicates learning the prediction model, requiring pertinent representations of driving policies. Such a complication is further exacerbated by imbalanced datasets (e.g., overfitting on the dominant driving policies in the training data), making it difficult to generalize the prediction model. In response, methods such as meta-learning and representation learning have been explored to address the generalization of the prediction model against diverse driving policies [9]. These methods employ distance metrics to capture complex data patterns and establish robust representations for similar/dissimilar patterns, improving prediction performance while reducing the risk of overfitting. While the choice of distance metrics, such as Euclidean, cosine, and kernel-based metrics, greatly impacts the representation model, determining the optimal metrics for measuring similarities remains an unresolved task to date [10]. Lastly, the learning-based prediction model can be susceptible to out-of-training data distribution, meaning it is prone to yield unreliable prediction results when the encountered OV's driving policy deviates from the training data [11].

C. Trajectory Prediction With Uncertainty

Extensive investigation has been carried out to address uncertainty in trajectory prediction [6], among which GP has gained substantial attention due to its quantitative measure of uncertainty. Taking these advantages, GP has proven its effectiveness in cut-in behavior prediction [12], and multi-step trajectory prediction of OV in autonomous racing [3]. Typical GP describes the unknown function using stationary kernels, assuming that the properties of the function are constant across all input values. However, this often results in poor uncertainty calibration for non-stationary stochastic processes that exhibit intricate data patterns in practice [13]. To handle the non-stationary aspect within GP, different techniques have been explored, such as input-dependent parametric kernels, Deep GP, and DKL [13]. In particular, DKL leverages the non-linear mapping capability of neural networks to transform complex non-stationary input space into the latent feature space that can be effectively processed with stationary kernels in GP. However, training these

complex kernels remains a significant challenge, as highlighted by recent studies [14]. Addressing this issue, this paper introduces a kernel-based metrics learning framework for DKL. This framework presents heterogeneous kernels to fine-tune the alignment of latent features transformed by the neural network, thereby adequately embedding the non-stationary properties into DKL.

III. Problem Formulation

A. Vehicle Dynamics

The racing vehicles are modeled using a dynamic bicycle model [15], where the state is represented by the vector $X = [p_x, p_y, \psi_z, v_{lon}, v_{lat}, w_z]^T$. This vector consists of the vehicle's pose in Cartesian frame, $[p_x, p_y]^T$, the heading angle, ψ_z, the longitudinal and lateral velocities, $[v_{lon}, v_{lat}]^T$, and the yaw rate, w_z. The input vector $u = [F_{xx}, \delta]^T$ comprises the rear longitudinal tire force F_{xx} and the front steering wheel angle δ. To discretize the vehicle's nonlinear dynamics, we apply the 4^{th} order Runge-Kutta method with a sampling time step size T_s. The resulting discrete-time vehicle dynamics is compactly described as $X_{k+1} = f(X_k, u_k)$, where the subscription k indicates the time index. To examine the racing vehicle's motion, we interpret the vehicle's pose as Frenet-Serret formulas with respect to the racing track's centerline [15]. Let $\mathcal{C} : \mathbb{R}^4 \mapsto \mathbb{R}^4$ denotes the invertible operator which projects the vehicle's pose in Cartesian frame to the curvilinear frame. Accordingly, the pose in curvilinear frame is denoted as $c = \mathcal{C}(p)$ where $p = [p_x, p_y, \psi_z, v_{lon}]^T$ and $c = [s, e_{lat}, e_\psi, v_{lon}]^T$. Here, s is the distance progress along the track's center line, e_{lat} is the lateral offset from the center line, and e_ψ is the angular deviation from the center line's tangent. The track's signed curvature at any distance $s \in [0, \tau]$ is represented as $\eta(s)$, where τ is the track length. To distinguish the state variables of EV and OV, we use superscripts in notations like X^{ev}, c^{ev}, X^{ov}, and c^{ov}.

B. EV Trajectory Planning Problem for Racing

The EV trajectory planning problem is formulated as Model Predictive Contouring Control (MPCC), which allows the EV to plan its trajectory over N steps ahead at time step k, defined as $\hat{\mathbf{X}}^{ev}_{k:k+N} = \{\hat{X}^{ev}_k, \ldots, \hat{X}^{ev}_{k+N}\}$, with respect to the projected distance progress $\bar{s}_t$ on the centerline [15]. Further, the constraints of MPCC related to dynamic obstacles are augmented to incorporate the predicted OV trajectory. Given the predicted trajectory of the OV over N steps ahead at time step k, i.e., $\{\hat{X}^{ov}_k, \ldots, \hat{X}^{ov}_{k+N}\}$, the MPCC problem for the EV is formulated as follows:

$$\min_{\mathbf{u},\mathbf{v}} \sum_{t=0}^{N-1} ||e_c(\hat{X}^{ev}_t, \bar{s}_t)||^2_{q_c} + ||u_t||^2_{R_u} + ||u_t - u_{t-1}||^2_{R_d} - q_s \bar{v}_N \tag{1a}$$

$$\text{s.t. } \hat{X}^{ev}_0 = X^{ev}_k,\ \bar{s}_0 = s^{ev}_k,\ u_{-1} = u^{ev}_{k-1}, \tag{1b}$$

$$\hat{X}^{ev}_{t+1} = f(\hat{X}^{ev}_t, u_t), \qquad t = 0, \ldots., N-1 \tag{1c}$$

$$\bar{s}_{t+1} = \bar{s}_t + T_s \bar{v}_t, \qquad t = 0, \ldots, N-1 \tag{1d}$$

$$\hat{X}^{ev}_t \in X_{\text{track}},\ u_t \in \mathbf{U}, \qquad t = 0, \ldots, N \tag{1e}$$

$$h(\hat{X}^{ev}_t, \hat{X}^{ov}_{k+t}) \le 0, \qquad t = 0, \ldots, N \tag{1f}$$

where $\mathbf{u} = \{u_0, \ldots, u_{N-1}\}$, $\mathbf{v} = \{\bar{v}_0, \ldots, \bar{v}_{N-1}\}$. $\bar{v}_t$ represents the progression rate, and e_c measures the approximated projection errors between $\hat{X}^{ev}_t$ and the reference centerline at $\bar{s}_t$ [15]. The objective cost (1a) conceives of maximizing the race progress while penalizing the centerline deviation according to weighting parameters, $q_c,\ q_s > 0$. To ensure a smooth control profile, cost minimizes the magnitude and rate of the input with the weights R_u and $R_d \succ 0$, respectively. The dynamics of the EV is governed by (1c) and (1d). In (1e), $X_{track} = \{X | -W_{track}/2 \le e_{lat} \le W_{track}/2\}$ and $\mathbf{U}$ represent the constraints imposed by the track boundary and the control input limits, where W_{track} is the track width. The constraint in (1f) ensures collision avoidance, given that the predicted OV state forms a probability distribution, such as Gaussian distribution $\hat{X}^{ov}_{k+t} \sim \mathcal{N}(\mu(\hat{X}^{ov}_{k+t}), \sigma^2(\hat{X}^{ov}_{k+t}))$. In constructing (1f), the EV's safety margin is drawn as a collection of four circles, while an ellipse represents the OV's margin. To accommodate the uncertainty encoded in the predicted state distribution $\hat{X}^{ov}_{k+t}$, the size of the OV's ellipse is dynamically adjusted in proportion to its variance, $\sigma^2(\hat{X}^{ov}_{k+t})$. Detailed derivation for the ellipse size can be found in Section IV of [3], which is omitted here for brevity. Notably, the ellipsoidal safety margin is widely adopted due to its effectiveness in capturing Gaussian-distributed uncertainties in the predicted OV state and its computational efficiency in solving nonlinear MPC problems [16].

C. GP and DKL for OV Trajectory Prediction

To predict the OV trajectory which is implicated in the patterns of both EV and OV states, we employ a GP Regression (GPR) model to be trained using the EV-OV interaction dataset. GPR is a non-parametric Bayesian regression that embeds a GP, characterized by a mean function $m(\cdot)$ and a covariance function $\kappa(\cdot, \cdot)$, commonly referred to as a kernel. Typically assuming a zero-mean GP prior for simplicity, the posterior distribution of GPR is given by $\hat{f}(x) \sim \mathcal{N}(\bar{m}(x), \bar{\sigma}(x))$ where $\bar{m}(x)$ and $\bar{\sigma}(x)$ represent the posterior mean and covariance functions calculated from the training data, respectively.

In this paper, we adopt the Mat $\acute{e}$ rn kernel, $\kappa(x, x')$, which has proven to be effective in the autonomous driving domain [3], [12]. Mat $\acute{e}$ rn kernel, however, is a stationary kernel that may be insufficient to capture the complex patterns in the trajectories of different OVs with diverse driving policies. To address this, DKL applies a non-linear transformation to the input space and utilizes the same stationary kernels for the transformed space, namely feature space. Accordingly, the kernel in DKL can be reformulated as:

$$\bar{\kappa}(x, x') = \kappa(\mathbf{\Phi}(x), \mathbf{\Phi}(x'))$$

where $\mathbf{\Phi} : \mathbb{R}^n \to \mathbb{R}^{n'}$ denotes the transformation function designed as a neural network, mapping inputs to feature space. However, learning the optimal design of $\mathbf{\Phi}$ in an unsupervised manner is challenging due to complex EV-OV interactions that yield diverse OV driving policies.

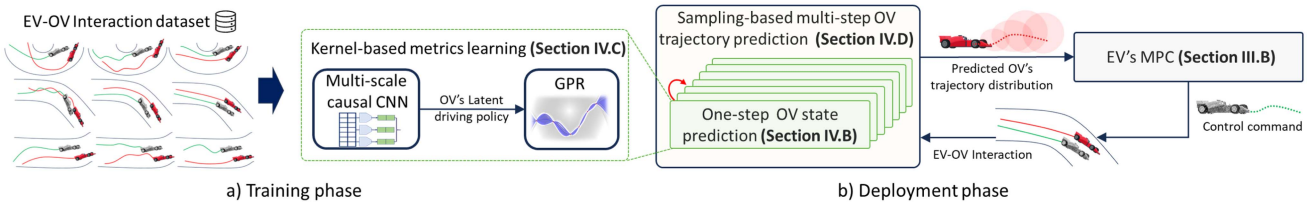

Fig. 2. Architecture of the proposed algorithm: (a) The training process of the one-step state prediction model, which utilizes the proposed kernel-based metrics learning for DKL; and (b) an autonomous racing algorithm for EV, which plans its maneuver based on the predicted trajectory distribution of OV.

IV. Algorithm Development

A. Algorithm Overview

In this section, we introduce the main algorithm to predict the trajectory distributions of OVs. First, the one-step OV state prediction model is developed using DKL, which is trained on the EV-OV interaction dataset. Then, we introduce a DKL training framework where novel kernel metrics are employed to address the diversity of OV driving policies. Based on the trained DKL, a sampling-based method is applied to the one-step prediction model to yield a multi-step predicted trajectory distribution. The overall training and deployment phases of the proposed OV trajectory prediction are outlined in Fig. 2.

B. One-Step OV State Prediction Using DKL

To model the one-step state prediction of the OV, we employ DKL, whose input is the past trajectories of both EV and OV, signifying their interaction patterns over a high-dimensional space. The input for the DKL at time step k is formulated as a sequence $\mathbf{z}_{k-M:k} = [\mathbf{z}_{k-M}^T, \dots, \mathbf{z}_k^T]^T \in \mathbb{R}^{10\times M}$, where M denotes the sequence length that records the past interactions, including the current states of EV and OV. Specifically, the vector $\mathbf{z}_k$ includes the state information of both vehicles along with track geometry, defined as:

$$\mathbf{z}_k = \left[(s_k^{ov} - s_k^{ev}), e_{lat,k}^{ov}, e_{\psi,k}^{ov}, v_{lon,k}^{ov}, e_{lat,k}^{ev}, e_{\psi,k}^{ev}, v_{lon,k}^{ev}, \tilde{\eta}_k^{ov}\right]^T$$

where $\tilde{\eta}_k^{ov} = [\eta(s_k^{ov}), \eta(s_k^{ov}+\zeta), \eta(s_k^{ov}+2\zeta)]^T \in \mathbb{R}^3$ represents the vector of track curvatures at look-ahead distances $s_k^{ov} + i\zeta$, $i = 0, 1, 2$ with some offset $\zeta > 0$. Notably, the vector $\mathbf{z}_k$ is crafted to capture the interaction between the EV and OV, specifically contextualized within the racing track geometry. This facilitates a more accurate description of interaction patterns relative to the track layout, rather than relying solely on the absolute global pose.

The DKL transforms the input into the feature space, representing the OV's latent driving policy as follows:

$$\hat{\phi}_k^{ov} := \mathbf{\Phi}(\mathbf{z}_{k-M:k}) \tag{2}$$

where $\mathbf{\Phi} : \mathbb{R}^{10\times M} \to \mathbb{R}^L$ is a nonlinear mapping function that encodes the interaction history between EV and OV over the past M steps. By this definition, $\mathbf{\Phi}$ is designed to map different EV-OV interaction patterns to OV driving policies. In this paper, we implement $\mathbf{\Phi}$ as a neural network model trained on racing data against various OVs. Consequently, the resulting $\hat{\phi}_k^{ov}$ exhibits "diverse driving policies" that arise from the different interaction patterns of the OVs. In pursuit of extracting an informative policy representation from the interaction data, we adopt a multi-scale encoder architecture as detailed in [10]. This architecture employs multi-filters with varying kernel sizes to capture both global and local interaction patterns, utilizing 1D causal convolution layers to preserve only the causal relationships in the data sequence. Subsequent to the multi-filtering, the final layers average the outputs of different kernel sizes, resulting in a comprehensive representation of the interaction patterns.

Subsequently, the driving policy $\hat{\phi}_k^{ov}$ inputs into GPR to predict the uncertain state propagation of the OV at the next time step, expressed by the following output vector:

$$y_k = \left[s_{k+1}^{ov} - s_k^{ov}, e_{lat,k+1}^{ov} - e_{lat,k}^{ov}, \times e_{\psi,k+1}^{ov} - e_{\psi,k}^{ov}, v_{lon,k+1}^{ov} - v_{lon,k}^{ov}\right]^T$$

This one-step state difference $y_k \in \mathbb{R}^4$ is modeled as unknown stochastic processes g as follows:

$$y_k = [y_k^{(1)}, y_k^{(2)}, y_k^{(3)}, y_k^{(4)}]^T = g(\mathbf{\Phi}(\mathbf{z}_{k-M:k}))$$

Without loss of generality, we assume the individual output elements $y^{(1)}, y^{(2)}, y^{(3)}$, and $y^{(4)}$ are independent of each other. Then, the GPR produces a probability distribution for each output element $y^{(n)}$ as follows:

$$y^{(n)} = g^{(n)}(\mathbf{\Phi}(\mathbf{z}_{k-M:k})) \sim \mathcal{N}\left(\mu^{(n)}(\mathbf{\Phi}(\mathbf{z}_{k-M:k})), (\sigma^{(n)}(\mathbf{\Phi}(\mathbf{z}_{k-M:k})))^2\right)$$

where $\mu^{(n)}$ and $(\sigma^{(n)})^2$, $n = 1, \dots, 4$, represent the posterior means and variances, respectively. It is noted that the obtained variances of the GPR capture the epistemic uncertainty, primarily emerging from the diversity of OVs' driving policies, as well as aleatoric uncertainty stemming from measurement noise.

C. Kernel-Based Metrics Learning for DKL

In this section, a novel training framework for DKL is introduced. The DKL is typically optimized in regard to the negative log marginal likelihood loss, $\mathcal{L}_{mll}$. However, solely optimizing $\mathcal{L}_{mll}$ is prone to overfitting and often yields suboptimal performance during testing [14]. To address this challenge, we incorporate additional regularization to better align the latent driving policies, $\hat{\phi}_k^{ov}$, computed by the neural network transformation, $\mathbf{\Phi}$. Central to our approach is the strategic clustering of similar latent driving policies while distancing dissimilar ones, based on the similarities in the output space, i.e., OV's state differences.

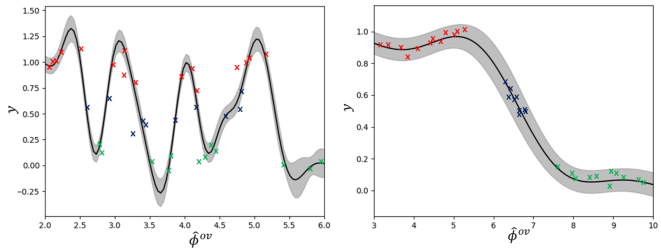

Fig. 3. Impact of latent space alignment on DKL prediction model: (Left) Without latent space alignment, (Right) With latent space alignment achieved by clustering similar features and dispersing dissimilar ones.

Fig. 3 illustrates the impact of alignment in latent driving policies on the performance of GPR given the same stochastic unknown processes. In the left subfigure, the input and output arrangement is disorganized, leading to increased susceptibility to overfitting and difficulties in training a GPR. Such a poor alignment eventually degrades the prediction model. In contrast, the right subfigure demonstrates a strategically aligned arrangement of latent driving policies, mirroring similarities observed in the output space. This, in turn, ensures that the driving policy ϕ_k^{ov} is robustly represented, enabling the trained OV state propagation model, g, to accurately predict the output y_k from ϕ_k^{ov}, even when ϕ_k^{ov} is influenced by variations or uncertainties in the input $\mathbf{z}_{k-M:k}$. As a result, this alignment reduces the risk of overfitting and improves robustness by minimizing sensitivity to input variations, thereby enhancing overall prediction performance.

To align the latent driving policies, we introduce a distance-matching loss term denoted as $\mathcal{L}_{dist}$ to minimize the dissimilarity between the distances in the pair of latent features and the distances in the pair of outputs. This loss for a mini-batch $\mathcal{D} = \{\hat{\phi}_i^{ov}, y_i\}_{i\in[1:D]}$ can be written as:

$$\mathcal{L}_{dist} = \frac{1}{D}\sum_{i\neq j}^{D} ||\kappa_{\mathbf{\Phi}}\left(\hat{\phi}_i^{ov}, \hat{\phi}_j^{ov}\right) - \kappa_y(y_i, y_j)||_2 \qquad (3)$$

where D is the size of the mini-batch. $\kappa_{\mathbf{\Phi}}$ and κ_y are Matérn kernels, the same type of kernel used in GPR. The prediction model regularized by (3) enforces the data to be aligned in the relevant distance.

On the other hand, the extent of alignment is strongly subject to the hyperparameters of $\kappa_{\mathbf{\Phi}}$ and κ_y, i.e., lengthscales $l_{\mathbf{\Phi}}$ and l_y, respectively. These hyperparameters directly influence the representation of data, thereby affecting the alignment of latent features. For instance, a smaller lengthscale in the kernel increases its sensitivity to small differences between data, causing them to be treated as significantly dissimilar. Conversely, a larger lengthscale reduces this sensitivity, allowing for a broader margin to identify similarities among distanced data. To find the best lengthscales for enhanced clustering of similar data and dispersion of dissimilar ones in latent space, we adjust the kernel sensitivity: decreasing it in the latent space, $\kappa_{\mathbf{\Phi}}$, to widen the distribution of dissimilar data, while increasing it in the output space, κ_y, for precise capture of data differences. This dual regularization is facilitated by introducing a sensitivity loss as follows:

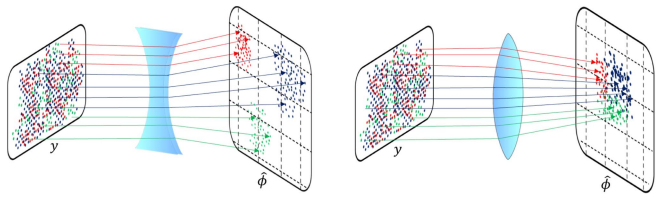

Fig. 4. Alignment in latent space with varying kernel lengthscales: (Left) $l_y < l_{\mathbf{\Phi}}$, (Right) $l_y > l_{\mathbf{\Phi}}$.

$$\mathcal{L}_{sense} = \ln\left(\frac{l_y}{l_{\mathbf{\Phi}}}\right) + \max(0, w_1(\sigma_{sense}(\hat{\phi}^{ov}) - \alpha)) \qquad (4)$$

where w_1 and α denote the predefined weight and the threshold, respectively. The second term in (4) prevents excessive divergence in the latent space, where the standard deviation of the latent features of the batch data, i.e., $\{\hat{\phi}_i^{ov} | i \in 1, \ldots, D\}$, is denoted as $\sigma_{sense}(\hat{\phi}^{ov})$. Fig. 4 graphically visualizes the effectiveness of the proposed regularization by (4). Here, the left-hand side illustrates the preferred scenario in which the different latent driving policies are more widely dispersed across various data.

Incorporating all the aforementioned loss terms, DKL is trained to optimize both the neural network and the GP's hyperparameters by minimizing the following composite loss:

$$\mathcal{L}_{DKL} = \mathcal{L}_{mll} + w_2\mathcal{L}_{dist} + w_3\mathcal{L}_{sense} \qquad (5)$$

where w_2 and w_3 are scaling constants. In a nutshell, our distinct novelty lies in the inclusion of both $\mathcal{L}_{dist}$ and $\mathcal{L}_{sense}$ in DKL. While $\mathcal{L}_{mll}$ addresses the nominal DKL model for prediction, $\mathcal{L}_{dist}$ is essential for appropriately aligning the latent driving policies and $\mathcal{L}_{sense}$ is for fine-tuning of kernel sensitivity. Without $\mathcal{L}_{sense}$, kernel-based metrics for both output and latent space may exhibit reduced sensitivities, undermining the clustering efficacy intended by $\mathcal{L}_{dist}$. It is to be noted that $\mathcal{L}_{dist}$ and $\mathcal{L}_{sense}$ are mutually complementing each other to achieve the latent feature alignment. Therefore, their interdependency obviates the need for an ablation study of each loss term individually.

D. Sampling-Based Multi-Step OV Trajectory Prediction

The one-step OV state prediction model, trained by the proposed DKL framework, is used to predict the OV trajectory over the N step horizon. The basic idea of this multi-step prediction is consecutively executing one-step predictions over future time steps, considering not only the past interaction history but also future interaction. This requires the predicted EV trajectory, $\hat{\mathbf{X}}_{k:k+N-1}^{ev}$, which can be obtained from the open-loop solution to (1) for the EV motion planning at $k-1$ time step.

Analytically computing the solution to the multi-step prediction model is intractable in general [3]. To address this, we utilize a sampling-based approach with Q samples to estimate the multi-step state propagation of the OV. Subsequently, we calculate the statistics of these samples to derive the nominal trajectory prediction along with its variances. Specifically, for the i^{th} sample, starting at time step k, we sample OV states at

Algorithm 1: Multi-Step Trajectory Prediction With DKL.

Set: Prediction horizon length N and the number of samples Q.
Input : Current EV and OV states, c_k^{ev} and c_k^{ov}; EV's predicted states, $\hat{\mathbf{X}}_{k:k+N-1}^{ev}$; interaction history between EV and OV, $\mathbf{z}_{k-M:k}$.

$c_k^{ov,i} = c_k^{ov}$, $\mathbf{z}_{k-M:k}^i = \mathbf{z}_{k-M:k}, \forall i = 1, \ldots, Q$
for $i = 1, \ldots, Q$ **do**
  for $t = 0, \ldots, N-1$ **do**
    Compute $\hat{\phi}_{k+t}^{ov,i}$ in (2) given $\mathbf{z}_{k-M+t:k+t}^i$
    Sample $y_{k+t}^i \sim \mathcal{N}(\mu(\hat{\phi}_{k+t}^{ov,i}), \sigma^2(\hat{\phi}_{k+t}^{ov,i}))$ with GP
    $c_{k+t+1}^{ov,i} = c_{k+t}^{ov,i} + y_{k+t}^i$
    if $t < N-1$ **then**
      Update $\mathbf{z}_{k-M+t+1:k+t+1}^i$ given $\hat{X}_{k+t+1}^{ev}$, c_{k+t+1}^{ov}
  end for
end for
for $t = 1, \ldots, N$ **do**
  Compute c_{k+t}^{ov} and Σ_{k+t}^{ov} by (6)
end for
Output : $c_{k+t}^{ov}, \Sigma_{k+t}^{ov} \ \forall t = 1, .., N$

time step $k+1$ as $c_{k+1}^{ov,i}$ using one-step prediction. Given the predicted states of the EV and OV, we construct $\mathbf{z}_{k+1}^i$ along with the track information. This vector is used to update the input to the one-step prediction model at the next time step, $\mathbf{z}_{k-M+1:k+1}^i$. We then continue to roll out the consecutive steps for multi-step prediction, repeating over N steps, resulting in the set of multi-step samples, $\{c_{k+t}^{ov,i} | \ i = 1, \ldots, Q, t = 1, \ldots, N\}$. Correspondingly, the predicted trajectory distribution at each future time step, $k+t$, is defined by its mean, c_{k+t}^{ov}, and covariance, Σ_{k+t}^{ov}, calculated as follows:

$$c_{k+t}^{ov} = \frac{1}{Q}\Sigma_{i=1}^{Q} c_{k+t}^{ov,i}$$

$$\Sigma_{k+t}^{ov} = \frac{1}{Q-1}\Sigma_{i=1}^{Q}\left(c_{k+t}^{ov,i} - c_{k+t}^{ov}\right)\left(c_{k+t}^{ov,i} - c_{k+t}^{ov}\right)^T \quad (6)$$

Lastly, the prediction outcomes are utilized to define collision avoidance constraints, as formulated in (1f), ensuring safe and agile racing maneuvers of EV. The detailed procedure is described in Algorithm 1.

V. Hardware Experiments

A. Experiment Setup

The EV and OV are built on the 1/10th scale racecar shown in Fig. 5. To assure reliable real-time execution, the EV utilizes a Jetson AGX Orin, while the OV operates efficiently with a Jetson Orin NX. In terms of software configuration, both vehicles operate with ROS Noetic. They utilize the Cartographer [17] to perform local pose estimation within a shared map common to both the EV and OV. The state estimation involves a customized estimator based on GTSAM [18]. Instead of detecting and tracking the states of other vehicles using sensor measurements, the EV and OV exchange their own state estimates via ROS

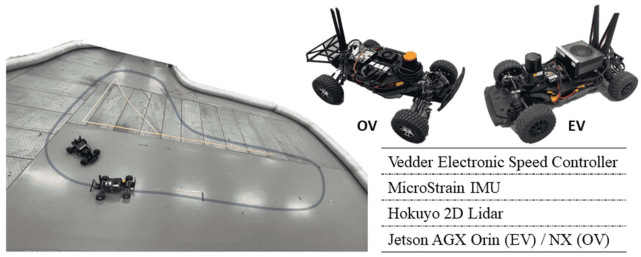

Fig. 5. 1/10th scale racing experiments setup. L-shape racing track and the vehicles' specifications.

topics. Both the EV and OV solve optimal control problems using the FORCESPRO Nonlinear Interior-Point solver [19] through code generation. Additionally, DKL is implemented within the PyTorch framework, incorporating variational GPR from GPyTorch [20]. Control operations are executed at 20Hz, while multi-step trajectory predictions are performed at 10Hz.

B. Diverse Driving Policies of Opponent Vehicles

The baseline driving policy for both the EV and OV is established by the MPCC formulation in (1) with a planning horizon of $N = 12$ and the sample time of $T_s = 0.1s$. To facilitate overtaking maneuvers and ensure close interaction between vehicles, longitudinal speed constraints of 1.9m/s for the EV and 1.6m/s for the OV are imposed, respectively.

OV driving policies are designed to interact with the EV in two different manners: through direct blocking maneuvers (i.e., aggressive driving policy) or by focusing solely on minimal lap time without blocking (i.e., passive driving policy). The implementation of the OV's aggressive driving policy, as outlined in [3], actively obstructs the EV by minimizing the lateral distance between the OV and EV. This is achieved by including the blocking weights in the cost function (1a). Conversely, the passive driving policy is implemented by omitting the collision avoidance constraints in (1f) and not incorporating the blocking weights, thus disregarding any interactive maneuvers with the EV.

To train the prediction model, two datasets, $\mathcal{D}_{aggr}$ and $\mathcal{D}_{pass}$, are collected to capture the interaction with OVs under these two different driving policies: aggressive and passive, respectively. For our one-step OV state prediction model, we set the dimension of the latent driving policy to $L = 11$ and the observed interaction sequence length to $M = 10$. The weights in (4) and (5) are tuned as $w_1 = 2$, $w_2 = 1$, and $w_3 = 0.05$. The training, validation, and test datasets are constructed using samples that are randomly pooled from $\mathcal{D}_{aggr}$ and $\mathcal{D}_{pass}$ without any annotations. Finally, for the multi-step prediction model, we set the number of samples in generating the predicted trajectory distribution to $Q = 25$.

C. Racing Results and Discussion

Extensive experiments are carried out to assess the prediction performance of the proposed method and further validate its effectiveness in head-to-head racing, particularly in terms of

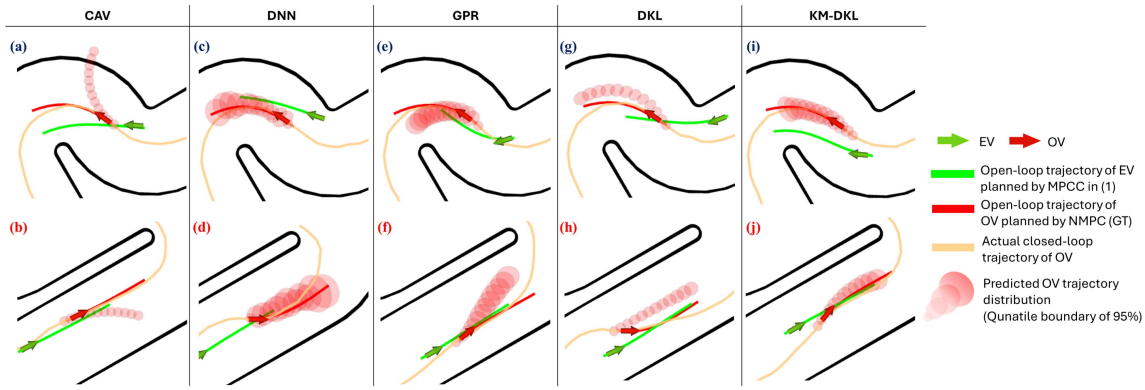

Fig. 6. Snapshots of EV racing maneuvers against OV with passive (a,c,e,g,i) and aggressive (b,d,f,h,j) driving policies. Black solid lines represent the track boundaries. A comprehensive video demonstration of the hardware experiments is accessible at https://github.com/HMCL-UNIST/OpponentPredictionWithKMDKL.git.

overtaking maneuvers. For comparative analysis, we evaluate our multi-step trajectory prediction method versus five distinct baseline trajectory predictors. The first baseline, referred to as NMPC (a.k.a Ground Truth, GT), utilizes a planning-based nonlinear MPC predictor. This method employs the open-loop solution to the OV's MPCC problem, serving as the ground truth prediction from the OV's perspective. The second baseline, denoted as Constant Angular Velocity (CAV), is a physics-based prediction method that propagates the OV's current state under constant linear and angular velocity assumptions. The third baseline, DNN, adapted from [6], is a deep learning-based method originally developed for urban environments. The input is modified to include EV and OV state histories and track information, i.e., $\mathbf{z}_{k-M:k}$. Additionally, the output provides the mean and variances of the multi-step predicted state propagation, which are matched with the output of Algorithm 1 for a fair comparison with other baselines. The fourth baseline, denoted as GPR, adapted from [3], is a learning-based method that employs GPR with a stationary Mat $\acute{e}$ rn kernel to predict the multi-step trajectory together with its associated uncertainties. This method utilizes the current driving scene information only, i.e., z_k. Consequently, it faces limitations in capturing the interaction patterns between the EV and OV that are influenced by OV's driving policy. The fifth baseline denoted as DKL, represents an ablation version of our proposed method. It utilizes the same architecture as the one-step prediction model but omits the proposed kernel-based metrics learning. This is more or less nominal DKL that solely aims at maximizing the log marginal likelihood of observations, i.e., $\mathcal{L}_{mll}$. Lastly, our proposed method, incorporating a novel kernel-based metrics learning as in (5), is referred to as KM-DKL. Unlike the other methods that provide prediction results as trajectory distributions, CAV and NMPC (GT) predictors provide deterministic trajectories, lacking a probabilistic measure of prediction confidence. For the sake of fair comparison concerning uncertainty quantification ability, we set a fixed circular boundary to their prediction results.

The experiment involves twenty races for each baseline method. The OV is placed in various starting positions, which remain the same for all predictors. Each race concludes once the EV completes three laps or if a major collision occurs that drastically changes the course of both cars, making it impossible to continue the race. For each race, we set the OV's driving policy to either passive or aggressive, resulting in ten races for each policy. The prediction performance is then evaluated by the mean square errors of longitudinal and lateral poses between predicted and actual trajectories at the last prediction step when the EV is in the proximity to the OV, i.e., $0 \leq s^{ov} - s^{ev} \leq 2$.

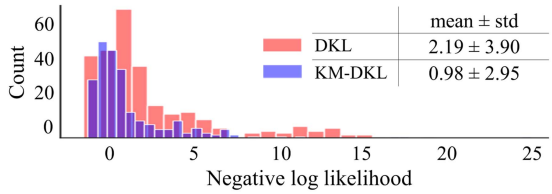

Fig. 7. Histograms of the negative log-likelihood for prediction results in races against OVs exhibiting unseen driving policies.

Notably, given the result statistics detailed in Table I and the prediction snapshots in Fig. 6, our method surpasses all the baselines in lateral prediction accuracy, with NMPC(GT) achieving the lowest longitudinal mean error. Even though NMPC(GT) anticipates the ground truth open-loop solution from OV's perspective, it fails to capture the multi-step ahead closed-loop interactions between EV and OV in real racing. GPR's prediction surpasses the physics-based CAV method, yet struggles to adapt to different OV driving policies, leading to a higher risk of collisions due to misapprehension of the driving policy, e.g., blocking as a non-blocking policy and vice versa. Similarly, DKL and DNN falls short of capturing the interactions between EV and OV, resulting in a higher collision rate compared to KM-DKL. In contrast, our KM-DKL, with its prediction ability against different OV driving policies, facilitates agile and safe overtaking maneuvers without major collisions. Furthermore, computational complexity is evaluated using the average computation time in milliseconds. Although our method requires more computation time compared to other baselines, it meets the 10 Hz real-time requirement, making it feasible for high-speed autonomous racing.

D. Ablation Study for Unseen Driving Policy

To further investigate the virtue of our method, we examine the uncertainty calibration performance of the proposed KM-DKL

TABLE I
RACING RESULT STATISTICS AGAINST OVS WITH DIVERSE DRIVING POLICIES

	NMPC(GT)	CAV	DNN [6]	GPR [3]	DKL	KM-DKL(proposed)
Longitudinal MSE (mean ± std)	**0.178** ± 0.11	0.746 ± 0.307	0.839 ± 0.12	0.290 ± 0.15	0.256 ± 0.193	0.244 ± **0.09**
Lateral MSE (mean ± std)	0.217 ± 0.14	0.584 ± 0.35	0.377 ± 0.15	0.219 ± 0.14	0.261 ± 0.17	**0.139** ± **0.08**
Overtaking rate per race	0.85	0.55	0.65	0.35	0.65	**0.95**
Collision rate per race (minor)	**0.30**	0.45	0.85	0.40	0.50	**0.30**
Collision rate per race (major)	0.10	0.30	0.25	0.55	0.30	**0.00**
Average computation time (ms)	12	**6**	39	84	97	98

model compared to the nominal DKL, the ablated version of our model. This involves evaluating the prediction model against out-of-training-distribution scenarios, specifically when the encountered OV's driving pattern is not included in the training dataset, i.e., unseen driving policies. To create the unseen driving policy, we introduce a cooperative driving policy designed to yield the pathway to the EV. We realize this policy by modifying the aggressive driving policy's cost with the reversed sign of the blocking weight. This encourages the OV to move away from the EV's anticipated path during the race.

We carry out the experiment in a similar fashion to the aforementioned racing setup but with the unseen OV driving policy. The prediction models, DKL and KM-DKL, are evaluated in terms of the Negative Log-Likelihood (NLL) concerning the predicted OV pose at the last prediction step. NLL quantifies the likelihood of the predicted state given the observed data, with lower values indicating more accurate uncertainty calibration to unseen data [21]. Fig. 7 indicates that KM-DKL achieves lower NLL values than the DKL predictor. This demonstrates our method's improved uncertainty calibration, highlighting the advantages of incorporating kernel metrics learning into DKL.

VI. CONCLUSION

This study has developed a learning-based OV trajectory prediction model and downstream EV planning framework for autonomous racing against OVs with diverse driving policies. The key idea is to introduce novel heterogeneous kernel metrics and embed them into the learning pipeline of DKL. The prediction model is then learned in an unsupervised manner based on the EV-OV interaction dataset, adeptly differentiating the diverse driving policies. Compared to existing work, the proposed method enhances prediction accuracy and improves uncertainty calibration, which is instrumental for the safe and agile trajectory planning of EV. This has been validated on a 1/10th scale racecar platform, where the experimental results have shown better-calibrated prediction errors, lower collision rates, and higher overtaking success rates. Future work will focus on: i) online learning and adaptation of the algorithm to streaming data from EV-OV interactions; and ii) scaling the method to full-scale vehicles, addressing perception errors and multi-vehicle interactions in real-world racing.

REFERENCES

[1] J. Betz et al., "Autonomous vehicles on the edge: A survey on autonomous vehicle racing," *IEEE Open J. Intell. Transp. Syst.*, vol. 3, pp. 458–488, 2022.

[2] M. Wang, Z. Wang, J. Talbot, J. C. Gerdes, and M. Schwager, "Game-theoretic planning for self-driving cars in multivehicle competitive scenarios," *IEEE Trans. Robot.*, vol. 37, no. 4, pp. 1313–1325, Aug. 2021.

[3] E. L. Zhu, F. L. Busch, J. Johnson, and F. Borrelli, "A Gaussian process model for opponent prediction in autonomous racing," in *Proc. 2023 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2023, pp. 8186–8191.

[4] M. Gulzar, Y. Muhammad, and N. Muhammad, "A survey on motion prediction of pedestrians and vehicles for autonomous driving," *IEEE Access*, vol. 9, pp. 137957–137969, 2021.

[5] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human trajectory prediction in crowded spaces," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 961–971.

[6] B. Varadarajan et al., "MultiPath: Efficient information fusion and trajectory aggregation for behavior prediction," in *Proc. Int. Conf. Robot. Automat.*, 2022, pp. 7814–7821.

[7] Y. Chen, U. Rosolia, W. Ubellacker, N. Csomay-Shanklin, and A. D. Ames, "Interactive multi-modal motion planning with branch model predictive control," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 5365–5372, Apr. 2022.

[8] L. Feng, M. Bahari, K. M. B. Amor, É. Zablocki, M. Cord, and A. Alahi, "Unitraj: A unified framework for scalable vehicle trajectory prediction," 2024, *arXiv:2403.15098*.

[9] J. Liang, L. Jiang, and A. Hauptmann, "*SimAug*: Learning robust representations from simulation for trajectory prediction," in *Proc. 16th Eur. Conf. Comput. Vis.*, Glasgow, U.K., 2020, pp. 275–292.

[10] X. Zheng, X. Chen, M. Schürch, A. Mollaysa, A. Allam, and M. Krauthammer, "SimTS: Rethinking contrastive representation learning for time series forecasting," 2023, *arXiv:2303.18205*.

[11] T. Gilles, S. Sabatini, D. Tsishkou, B. Stanciulescu, and F. Moutarde, "Uncertainty estimation for cross-dataset performance in trajectory prediction," 2022, *arXiv:2205.07310*.

[12] Y. Yoon, C. Kim, J. Lee, and K. Yi, "Interaction-aware probabilistic trajectory prediction of cut-in vehicles using gaussian process for proactive control of autonomous vehicles," *IEEE Access*, vol. 9, pp. 63440–63455, 2021.

[13] A. G. Wilson, Z. Hu, R. Salakhutdinov, and E. P. Xing, "Deep Kernel learning," in *Proc. Artif. Intell. Statist.*, 2016, pp. 370–378.

[14] S. W. Ober, C. E. Rasmussen, and M. van der Wilk, "The promises and pitfalls of deep Kernel learning," in *Proc. Uncertainty Artif. Intell.*, 2021, pp. 1206–1216.

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

[16] R. Reiter, A. Nurkanović, J. Frey, and M. Diehl, "Frenet-Cartesian model representations for automotive obstacle avoidance within nonlinear MPC," *Eur. J. Control*, vol. 74, 2023, Art. no. 100847.

[17] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2D LiDAR SLAM," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2016, pp. 1271–1278.

[18] F. Dellaert and GTSAM Contributors, "borglab/gtsam," Georgia Tech Borg Lab, May 2022. [Online]. Available: https://github.com/borglab/gtsam

[19] A. Zanelli, A. Domahidi, J. Jerez, and M. Morari, "FORCES NLP: An efficient implementation of interior-point methods for multistage nonlinear nonconvex programs," *Int. J. Control*, vol. 93, pp. 13–29, 2017.

[20] J. Gardner, G. Pleiss, Q. Kilian, D. W. Bindel, and A. G. Wilson, "GPyTorch: Blackbox matrix-matrix Gaussian process inference with GPU acceleration," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 7587–7597.

[21] A. Marco, E. Morley, and C. J. Tomlin, "Out of distribution detection via domain-informed Gaussian process state space models," in *Proc. 62nd IEEE Conf. Decis. Control*, 2023, pp. 5487–5493.