Gaussian Process-based Stochastic Model Predictive Control for Overtaking in Autonomous Racing

T. Brüdigam[1], A. Capone[2], S. Hirche[2], D. Wollherr[1], and M. Leibold[1]

Abstract— A fundamental aspect of racing is overtaking other race cars. Whereas previous research on autonomous racing has majorly focused on lap-time optimization, here, we propose a method to plan overtaking maneuvers in autonomous racing. A Gaussian process is used to learn the behavior of the leading vehicle. Based on the outputs of the Gaussian process, a stochastic Model Predictive Control algorithm plans optimistic trajectories, such that the controlled autonomous race car is able to overtake the leading vehicle. The proposed method is tested in a simple simulation scenario.

I. INTRODUCTION

Whereas research on automated road vehicles has dominated the past decades, autonomous racing is a field that has only emerged recently. Roborace and the 2021 Indy Autonomous Challenge provide real-world opportunities to apply theoretic results to the racetrack.

Previous work on autonomous racing mainly focuses on lap-time optimization. In [1] a model-free learning method is proposed, where the maximum accelerations in longitudinal and lateral direction are adapted. This proposed approach may be incorporated into existing planners and feasibility was demonstrated on a full-size race car used for Roborace. In [2], trajectories for racing are planned with a multi-layered graph-based planner, verified again on a Roborace vehicle.

Furthermore, learning-based MPC methods have shown promising results for lap-time optimization. In [3] an iterative learning MPC framework was proposed that uses information from previous laps, evaluated in simulations. Advances were made in [4], where the proposed approach was successfully implemented on cars the size of remote control vehicles. A Gaussian process-based learning MPC algorithm is presented in [5], improving lap-times while considering safety. The proposed algorithm was applied to a full-size vehicle.

Previous work on autonomous racing majorly focuses on improving lap times; however, a fundamental part of racing is neglected: overtaking other race cars. While Roborace included overtaking maneuvers, these maneuvers where passive ones. Once the trailing vehicle was close enough to the leading vehicle in a specific part of the race track, the leading vehicle had to allow the trailing vehicle to overtake. Active overtaking will be inevitable for participants of the Indy Autonomous Challenge, where active overtaking is required.

In this work, we propose a combined Gaussian Process (GP) and stochastic Model Predictive Control (MPC) method, which allows to actively overtake other race cars.

Gaussian process regression is a powerful non-parametric tool used to infer values of an unknown function given previously collected measurements. In addition to exhibiting very good generalization properties, a major advantage of GPs is that they come equipped with a measure of model uncertainty, making them particularly beneficial for safety-critical applications. These characteristics have made GPs particularly attractive for developing control algorithms [6]– [9]. In the context of autonomous driving, GPs have also seen a rise in interest. In [5], [10], GP regression is used to improve the model of the autonomous vehicle using collected data, which in turn leads to an improvement in control performance. GPs have also been employed to predict the behavior of cut-in maneuvers of surrounding vehicles and obtain safe autonomous vehicle control [11].

Stochastic MPC (SMPC) has mostly been studied for road vehicles. SMPC allows to treat constraints in a probabilistic way, enabling less conservative solutions [12], [13]. In [14] an SMPC trajectory planner for automated vehicles is presented, considering the most likely future maneuver of surrounding vehicles and Gaussian prediction uncertainty. The Gaussian uncertainty allows to analytically reformulate the probabilistic constraint into a deterministic formulation that is tractable by a solver. In [15] a sampling-based SMPC approach for automated driving is proposed, based on scenario MPC [16]. The approach of [15] is extended in [17] to account for more complex vehicle behavior. Whereas standard SMPC approaches allow a small probability of collision, a safe SMPC framework is developed in [18].

This paper outlines a combined GP and SMPC approach for autonomous overtaking maneuvers in racing. The major challenge is to plan trajectories for a controlled race car such that a leading race car may be passed. Based on previous behavior of the leading vehicle, given the interaction between both vehicles, a GP is trained. The GP predictions for the leading vehicle are then used in an SMPC algorithm to plan efficient overtaking maneuvers. Whereas we only show preliminary results here, the proposed method has the potential to be a powerful method in autonomous racing. Ideally, the GP identifies weaknesses in the driving behavior of the leading vehicle while the controlled trailing vehicle is trying to overtake. The SMPC planner allows to efficiently consider the GP output and to plan optimistic vehicle trajectories, which are fundamental for racing. Given an increased sample set of data, the controlled race car increases its chances of

[1]The authors are with the Chair of Automatic Control Engineering at the Technical University of Munich, Arcisstrasse 21, 80333 Munich, Germany. {tim.bruedigam; dw; marion.leibold}@tum.de
[2]The authors are with the Chair of Information-oriented Control at the Technical University of Munich, Barer Strasse 21, 80333 Munich, Germany. {alexandre.capone; hirche}@tum.de

finding the right spot on a race track and a suitable driving approach to successfully overtake.

This paper is structured as follows. Section II introduces the vehicle model considered for the prediction. In Section III the proposed GP-SMPC method for race overtaking maneuvers is described. Simulation results are given in Section IV, while a conclusion and outlook follows in Section V.

II. VEHICLE MODEL

We consider two vehicles. The controlled race car is denoted as the ego vehicle (EV), whereas the race car to be overtaken is a target vehicle (TV).

MPC requires prediction models for both vehicles. The TV prediction, used by the EV, is based on the GP described in Section III-B, whereas the actual TV behavior is described in Section IV-A of the simulation part. For the EV, a kinematic bicycle model is used, given by the continuous-time system

$$\dot{s}= v \cos(\phi + \alpha), \tag{1a}$$

$$\dot{d}= v \sin(\phi + \alpha), \tag{1b}$$

$$\dot{\phi}= \frac{v}{l_{\mathrm{r}}} \sin \alpha, \tag{1c}$$

$$\dot{v}= a, \tag{1d}$$

$$\alpha= \arctan \left(\frac{l_{\mathrm{r}}}{l_{\mathrm{r}} + l_{\mathrm{f}}} \tan \delta \right), \tag{1e}$$

where l_{r} and l_{f} represent the distances from the vehicle center of gravity to the rear and front axles, respectively. The state and input vectors are $\boldsymbol{\xi} = [s, d, \phi, v]^{\top}$ and $\boldsymbol{u} = [a, \delta]^{\top}$, respectively, with vehicle velocity v, acceleration a, and steering angle δ. The longitudinal position along the road is s, the lateral vehicle deviation from the race track centerline is d, and the orientation of the vehicle with respect to the road is ϕ. We summarize the nonlinear vehicle model (1) as $\dot{\boldsymbol{\xi}} = \boldsymbol{f}^{\mathrm{c}}(\boldsymbol{\xi}, \boldsymbol{u})$.

Efficient MPC requires a discrete-time prediction model, which is obtained by first linearizing (1) at the current EV state $\boldsymbol{\xi}^* = \boldsymbol{\xi}_0$ and the EV input $\boldsymbol{u}^* = [0,0]^{\top}$ and then discretizing with sampling time T. These steps yield the time-discrete system

$$\boldsymbol{\xi}_{k+1}= \boldsymbol{\xi}_0 + T\boldsymbol{f}^{\mathrm{c}}(\boldsymbol{\xi}_0, \boldsymbol{0}) + \boldsymbol{A}_{\mathrm{d}}(\boldsymbol{\xi}_k - \boldsymbol{\xi}_0) + \boldsymbol{B}_{\mathrm{d}}\boldsymbol{u}_k \tag{2a}$$

$$= \boldsymbol{f}^{\mathrm{d}}(\boldsymbol{\xi}_0, \boldsymbol{\xi}_k, \boldsymbol{u}_k) \tag{2b}$$

with the discretized system matrices $\boldsymbol{A}_{\mathrm{d}}, \boldsymbol{B}_{\mathrm{d}}$ and a nonlinear term $\boldsymbol{f}^{\mathrm{c}}(\boldsymbol{\xi}_0, \boldsymbol{0})$. Details on the linearization and discretization are provided in [18], [19].

The EV is subject to constraints. We consider input constraints

$$\boldsymbol{u}_{\min} \leq \boldsymbol{u}_k \leq \boldsymbol{u}_{\max} \tag{3a}$$

$$\Delta\boldsymbol{u}_{\min} \leq \Delta\boldsymbol{u}_k \leq \Delta\boldsymbol{u}_{\max} \tag{3b}$$

limiting the absolute value and rate of change of the acceleration and steering angle, where $\Delta\boldsymbol{u}_{k+1} = \boldsymbol{u}_{k+1} - \boldsymbol{u}_k$. In addition, the road constraint is given by $d_{\min} \leq d_k \leq d_{\max}$ and we require a non-negative velocity $v_k \geq 0$. In the following, the input and state constraints are summarized by the set of admissible inputs $\mathcal{U}$ and the set of admissible states $\boldsymbol{\Xi}$. Additional collision avoidance constraints will be designed in the following section.

III. GP-BASED SMPC FOR AUTONOMOUS RACING

Autonomous racing requires overtaking maneuvers. In order to plan successful overtaking maneuvers, the EV needs a precise prediction of the future behavior of the TV. Here, we propose a combined GP and SMPC framework, where GP is used to predict the future TV behavior and SMPC plans optimistic EV trajectories, facilitating an overtaking maneuver to pass the TV.

In the following, we first present details on the GP design. Then, the generation of safety constraints is briefly addressed and, depending on the GP output, these constraints are tightened. Eventually, the tightened safety constraints are included into an SMPC optimal control problem to avoid collisions.

A. Gaussian Processes

Gaussian processes are used to infer the values of an unknown function given measurement data $\mathcal{D} = \{\boldsymbol{x}_n, \boldsymbol{y}_n\}_{n=1}^{N}$, where the training inputs

$$\boldsymbol{x}_n := \left(\boldsymbol{\xi}_n^{\top}, (\boldsymbol{\xi}_n^{\mathrm{TV}})^{\top} \right)^{\top} \tag{4}$$

correspond to the concatenation of the EV and TV states, and the training outputs

$$\boldsymbol{y}_n := \left(\boldsymbol{\xi}_{n+1}^{\mathrm{TV}} - \boldsymbol{\xi}_n^{\mathrm{TV}} \right)^{\top} \tag{5}$$

are the difference between the TV states for two time steps.

A Gaussian process is formally defined as a collection of random variables, any subset of which is jointly normally distributed [20]. It is fully specified by a prior mean, which we set to zero without loss of generality, and a kernel function $\kappa : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$. The kernel $\kappa(\cdot, \cdot)$ encodes function properties and any prior assumptions, e.g., Lipschitz continuity, periodicity and magnitude. In the following, we employ a squared-exponential kernel

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \sigma^2 \exp \left(-\frac{(\boldsymbol{x} - \boldsymbol{x}')^{\top} \boldsymbol{L}^{-2} (\boldsymbol{x} - \boldsymbol{x}')}{2} \right), \tag{6}$$

which can approximate continuous functions in compact spaces arbitrarily accurately [21].

By modeling the state transition dynamics $\boldsymbol{\xi}_{n+1}^{\mathrm{TV}} - \boldsymbol{\xi}_n^{\mathrm{TV}}$ of the TV with a GP, we implicitly assume that any set of evaluations is jointly normally distributed. By conditioning the GP on the training data $\mathcal{D}$, we obtain the posterior mean and variance for the d-th entry of the transition dynamics at an arbitrary point $\boldsymbol{x}^*$,

$$\mu_d(\boldsymbol{x}^*|\mathcal{D}) = \boldsymbol{\kappa}^{\top} \boldsymbol{K}^{-1} \boldsymbol{\gamma}_d$$
$$\sigma_d(\boldsymbol{x}|\mathcal{D}) = \kappa^* - \boldsymbol{\kappa}^{\top} \boldsymbol{K}^{-1} \boldsymbol{\kappa}, \tag{7}$$

where $\kappa^* = \kappa(\boldsymbol{x}^*, \boldsymbol{x}^*)$, $\boldsymbol{\kappa} = (\kappa(\boldsymbol{x}_1, \boldsymbol{x}^*), \ldots, \kappa(\boldsymbol{x}_n, \boldsymbol{x}^*))$, the entries of the matrix $\boldsymbol{K}$ are given by $K_{ij} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$, and $\boldsymbol{\gamma}_d = (y_{1,d}, \ldots, y_{n,d})$ concatenates the training outputs corresponding to the d-th entry.

B. Generating Sample TV Trajectories

We employ the GP model described in Section III-A to generate M sample TV trajectories

$$\boldsymbol{\xi}_k^{\text{TV},(m)}, \qquad k \in \{1,\dots,N\}, \qquad m \in \{1,\dots,M\}. \tag{8}$$

To this end, we sequentially draw a sample from the posterior GP distribution (7), apply the sampled dynamics to the TV prediction, and then condition the GP on the sampled point, similarly to [8].

The GP sample trajectories correspond to a computational complexity of order $\mathcal{O}(MN^3)$, which can become cumbersome for long horizons. However, the scalability of GPs can be improved considerably by employing several different approximations, e.g., by employing a set of inducing points or approximating the squared-exponential kernel by a finite-dimensional feature map [22].

From the sample trajectories, we deduce the mean and variance of the TV states at the prediction time steps

$$\bar{\boldsymbol{\xi}}_k^{\text{TV}} = \frac{1}{M} \sum_{m=1}^{M} \boldsymbol{\xi}_k^{\text{TV},(m)}, \qquad k \in \{1,\dots,N\},$$

$$\boldsymbol{\Sigma}_k^2 = \frac{1}{M-1} \sum_{m=1}^{M} \left(\boldsymbol{\xi}_k^{\text{TV},(m)} - \bar{\boldsymbol{\xi}}_k^{\text{TV}}\right)^{\top} \left(\boldsymbol{\xi}_k^{\text{TV},(m)} - \bar{\boldsymbol{\xi}}_k^{\text{TV}}\right), \tag{9}$$

which are later used for the SMPC part. The diagonal elements of $\boldsymbol{\Sigma}_k^2$ are $[\Sigma_{k,x}^2, \Sigma_{k,v_x}^2, \Sigma_{k,y}^2, \Sigma_{k,v_y}^2]$.

C. Constraint Generation

We generate constraints similar to [18]. A safety rectangle with length

$$a_{\text{r}} = l_{\text{veh}} + \tilde{a}_{\text{r}}\left(\boldsymbol{\xi},\boldsymbol{\xi}^{\text{TV}}\right). \tag{10}$$

and width

$$b_{\text{r}} = w_{\text{veh}} + \varepsilon_{\text{safe}} \tag{11}$$

is designed that surrounds the TV, where l_{veh} and w_{veh} denote the vehicle length and width, respectively. The size of the safety rectangle length depends on the velocity difference between the EV and TV, summarized by the term $\tilde{a}_{\text{r}}\left(\boldsymbol{\xi},\boldsymbol{\xi}^{\text{TV}}\right)$. A lateral safety distance parameter $\varepsilon_{\text{safe}}$ is used. Details are given in [18], [19].

Depending on the positioning of the EV with respect to the TV, different cases are considered for the constraint generation. The cases are summarized in Table I and illustrated in Fig 1. Inclined constraints are limited to a horizontal level, as shown in case C(2) in Fig. 1. Once the EV gets too close to the road boundary, i.e., overtaking is impossible, the constraint is chosen such that the EV plans to overtake on the other side (cases D and E).

Based on this case differentiation, linear safety constraints follow, given by

$$0 \geq q_y\left(\boldsymbol{\xi}_0,\boldsymbol{\xi}_k^{\text{TV}}\right) d_k + q_x\left(\boldsymbol{\xi}_0,\boldsymbol{\xi}_k^{\text{TV}}\right) s_k + q_t\left(\boldsymbol{\xi}_0,\boldsymbol{\xi}_k^{\text{TV}}\right) \tag{12}$$

with the coefficients q_y and q_x for the EV states d_k and s_k, and the intercept q_t. The coefficients q_y, q_x, and q_t

TABLE I: Constraint Generation Cases

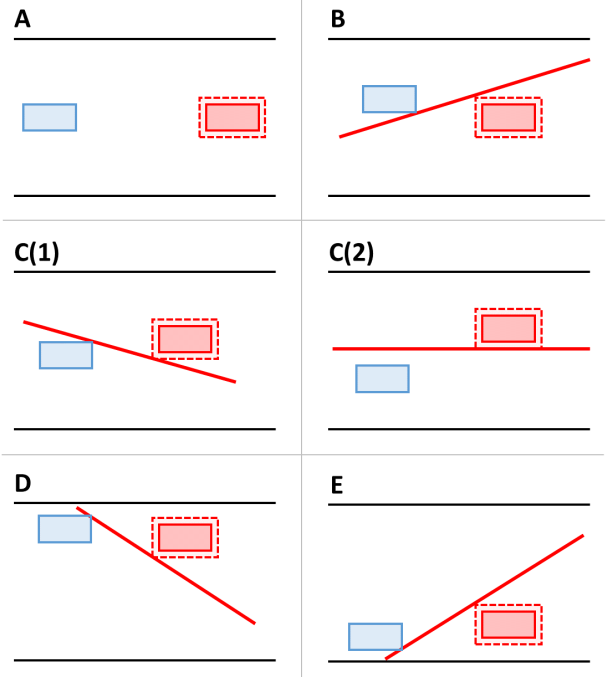| case	EV setting (w.r.t. TV)	constraint
A	large distance	no constraint
B	left of TV	inclined (pos.) constraint
C	right of TV	inclined (neg.) constraint
D	left of TV (close to road limit)	inclined (neg.) constraint
E	right of TV (close to road limit)	inclined (pos.) constraint

Fig. 1: Constraint cases. Driving direction is from left to right. The ego vehicle (EV) and target vehicle (TV) are shown in blue and red, respectively. The dashed red line represents the safety area around the TV.

depend on the current EV state $\boldsymbol{\xi}_0$ and the predicted TV states $\boldsymbol{\xi}_k^{\text{TV}}$. Whereas future TV state predictions are used, only the current EV state is considered in order to generate linear constraints.

D. SMPC Details

In the previous computation of the safety rectangle, no TV prediction uncertainty was considered. In the following, we extend the safety rectangle to account for TV prediction uncertainty, yielding the updated safety rectangle length and width values

$$b_{\text{r},k} = w_{\text{veh}} + e_{x,k,\eta} \tag{13a}$$

$$a_{\text{r},k} = l_{\text{veh}} + \tilde{a}_{\text{r}}\left(\boldsymbol{\xi}_0,\boldsymbol{\xi}_k^{\text{TV}}\right) + e_{y,k,\eta} \tag{13b}$$

with constraint tightening according to $e_{x,k,\eta}$ and $e_{y,k,\eta}$ as discussed next.

SMPC considers probabilistic constraints, i.e., chance constraints, of the form

$$\Pr\left(\boldsymbol{\xi}_k \in \boldsymbol{\Xi}_{k,\text{safe}}\right) \geq \beta, \tag{14}$$

which must be fulfilled with a probability larger than the risk parameter β. The safe set $\boldsymbol{\Xi}_{k,\text{safe}}$ depends on the TV safety rectangle. It is not possible to directly solve (14); hence, a reformulation is necessary.

Based on the approximated error covariance matrices Σ_k^2, considering only diagonal elements, the TV safety rectangle adaptations $e_{x,k,\eta}$ and $e_{y,k,\eta}$ are obtained by

$$e_{x,k,\eta} = \Sigma_{k,x}\sqrt{\eta} \tag{15a}$$
$$e_{y,k,\eta} = \Sigma_{k,y}\sqrt{\eta}. \tag{15b}$$

with the *chi-squared distribution*

$$\eta = \chi_2^2(1-\beta) \tag{16}$$

as derived in detail in [18].

The generated safety constraints based on the adapted safety rectangles are now included into an SMPC optimal control problem.

E. Optimal Control Problem

Given the constraint generation and constraint tightening, we formulate the deterministic representation of the SMPC optimal control problem

$$V^* = \min_{\boldsymbol{U}} \sum_{k=1}^{N} \|\Delta\boldsymbol{\xi}_k\|_{\boldsymbol{Q}} + \|\boldsymbol{u}_{k-1}\|_{\boldsymbol{R}} + \|\Delta\boldsymbol{u}_{k-1}\|_{\boldsymbol{S}} \tag{17a}$$

s.t. $\boldsymbol{\xi}_{k+1} = \boldsymbol{f}^{\text{d}}\left(\boldsymbol{\xi}_0, \boldsymbol{\xi}_k, \boldsymbol{u}_k\right)$ (17b)

(17c)

$\boldsymbol{\xi}_k \in \boldsymbol{\Xi} \quad \forall k \in \{1,\dots,N\},$ (17d)

$\boldsymbol{u}_k \in \mathcal{U} \quad \forall k \in \{0,\dots,N-1\},$ (17e)

$0 \geq q_y\left(\boldsymbol{\xi}_0, \boldsymbol{\xi}_k^{\text{TV}}\right) y_k + q_x\left(\boldsymbol{\xi}_0, \boldsymbol{\xi}_k^{\text{TV}}\right) x_k + q_t\left(\boldsymbol{\xi}_0, \boldsymbol{\xi}_k^{\text{TV}}\right)$
$\quad \forall k \in \{0,\dots,N\}$ (17f)

where $\|\boldsymbol{z}\|_{\boldsymbol{Z}} = \boldsymbol{z}^\top \boldsymbol{Z} \boldsymbol{z}$ and $\Delta\boldsymbol{\xi}_k = \boldsymbol{\xi}_k - \boldsymbol{\xi}_{k,\text{ref}}$ with the EV reference state $\boldsymbol{\xi}_{k,\text{ref}}$. The weighting matrices are given by $\boldsymbol{Q}$, $\boldsymbol{S}$, and $\boldsymbol{R}$.

The resulting optimal control problem is a quadratic program and can be solved efficiently. The constraint tightening steps are performed before the optimal control problem is solved.

IV. SIMULATION RESULTS

In the following, we briefly analyze the proposed SMPC approach in a simulation scenario.

A. Simulation Setup

We consider a simple race scenario, where the EV intends to overtake the TV on a straight road, as illustrated in Fig. 2.

All units are given as SI units. The simulations are carried out in Matlab. The MPC algorithm is based on the NMPC toolbox [23], using the *fmincon* solver.
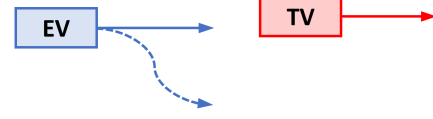

Fig. 2: Racing scenario with two possible EV trajectories.

TABLE II: General Simulation Parameters

scalars	vectors	matrices
$w_{\text{lane}} = 12$	$\boldsymbol{u}_{\max} = [10, 0.2]^\top$	$\boldsymbol{Q} = \text{diag}(0, 0.25, 0.2, 10)$
$l_{\text{veh}} = 5$	$\boldsymbol{u}_{\min} = [\text{-}15, \text{-}0.2]^\top$	$\boldsymbol{R} = \text{diag}(0.33, 5)$
$w_{\text{veh}} = 2$	$\boldsymbol{u}_{\max}^{\text{TV}} = [10, 0.4]^\top$	$\boldsymbol{S} = \text{diag}(0.33, 15)$
$l_{\text{f}} = l_{\text{r}} = 2$	$\boldsymbol{u}_{\min}^{\text{TV}} = [\text{-}15, \text{-}0.4]^\top$	
$\varepsilon_{\text{safe}} = 0.5$		

The MPC algorithm uses a sampling time $T = 0.2$ and a horizon $N = 10$. The linearized, discretized EV model (2) is used for the prediction. The optimized MPC inputs are then applied to the nonlinear, time-continuous EV model (1) in the simulation. Important simulation parameters are summarized in Table II. In case of an infeasible SMPC optimal control problem, the previously optimized solution is applied.

A simple TV motion planner was implemented, depending on previous and current vehicle states. The TV states are $\boldsymbol{\xi}^{\text{TV}} = [x^{\text{TV}}, v_x^{\text{TV}}, y^{\text{TV}}, v_y^{\text{TV}}]^\top$ with longitudinal and lateral inputs u_x^{TV} and u_y^{TV}. The TV state update is computed by

$$\boldsymbol{\xi}_{k+1}^{\text{TV}} = \boldsymbol{A}\boldsymbol{\xi}_k^{\text{TV}} + \boldsymbol{B}\boldsymbol{u}_k^{\text{TV}} \tag{18}$$

with

$$\boldsymbol{A} = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \boldsymbol{B} = \begin{bmatrix} 0.5T^2 & 0 \\ T & 0 \\ 0 & 0.5T^2 \\ 0 & T \end{bmatrix} \tag{19}$$

and

$$\boldsymbol{u}_k^{\text{TV}} = \boldsymbol{K}^{\text{TV}}\left(\boldsymbol{\xi}_k^{\text{TV}} - \boldsymbol{\xi}_{\text{ref},k}^{\text{TV}}\right), \tag{20a}$$

$$\boldsymbol{K}^{\text{TV}} = \begin{bmatrix} 0 & -0.55 & 0 & 0 \\ 0 & 0 & -0.63 & -1.15 \end{bmatrix} \tag{20b}$$

based on a TV reference state $\boldsymbol{\xi}_{\text{ref},k}^{\text{TV}}$ and a stabilizing feedback matrix $\boldsymbol{K}^{\text{TV}}$. The TV inputs are limited by $\boldsymbol{u}_{\max}^{\text{TV}}$ and $\boldsymbol{u}_{\min}^{\text{TV}}$.

The TV reference state is then adapted depending on the EV and TV setting. Once the EV has passed the TV, the TV aims to drive straight. As long as the EV is located behind the TV, the TV tries to block the EV from overtaking. This is done by setting the lateral TV position reference to the current EV lateral position, while maintaining a constant longitudinal velocity. Similar to many real-world races, the TV is only allowed to choose a maneuver once, i.e., the TV
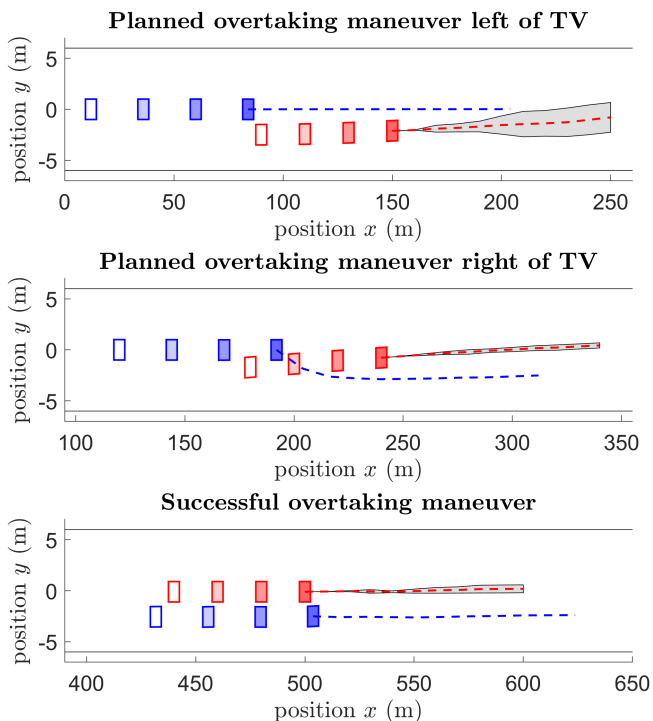
Fig. 3: Shots of the overtaking maneuver. Fading boxes show past vehicle states. The EV is shown in blue, the TV in red. Predicted EV and TV trajectories are indicated by dashed lines. The GP 2σ confidence region for the lateral TV motion is visualized by the gray area.

Fig. 4: EV states and inputs. Gray areas are displayed in Fig. 3.

cannot continuously drive left and right to block the EV but must only move into one direction or drive straight.

B. Overtaking Maneuver

We analyze the proposed method in a simple race scenario. The initial EV state is $\boldsymbol{\xi}_0 = [0, 0, 0, 60]^\top$ and the initial TV state is $\boldsymbol{\xi}_0^{\mathrm{TV}} = [80, 50, -2.5, 0]^\top$. The EV aims at maintaining its initial velocity, its center road position, and a straight orientation with respect to the road, i.e., $v_{\mathrm{ref}} = 60$, $d_{\mathrm{ref}} = 0$, and $\phi_{\mathrm{ref}} = 0$, respectively.

Shots of the vehicle configuration are shown in Fig. 3. The EV states and inputs are displayed in Fig. 4.

Initially, the EV attempts to overtake the TV on the left. In the beginning, the TV moves towards the left to block the EV overtaking maneuver on the left. At step 17, the GP prediction indicates that the TV will continue its move to the left. As more data is available, the GP variances decrease. From this point on, it is more beneficial for the EV to switch its strategy and attempt to overtake the TV on the right. As the TV is only allowed to change its lateral direction once, the EV successfully overtakes the TV eventually. Throughout the simulation, the EV maintains its

V. Conclusion

In this work, we proposed a method where the results of Gaussian process learning are used by stochastic MPC to plan over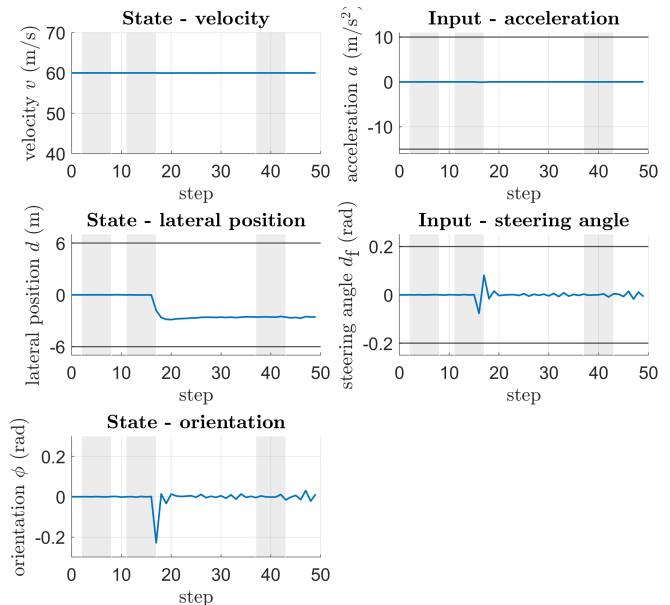taking maneuvers in autonomous racing. The presented work presents a starting point regarding GP and SMPC based research for overtaking in autonomous racing.

Obvious extensions include using a dynamic vehicle model for the MPC prediction and a more sophisticated Gaussian process learning approach. The next step would then be to extend the method such that a full lap and race with multiple surrounding vehicles is possible.

Eventually, the aim is to develop a method for competitive racing: First, the controlled autonomous race car learns weak spots of the leading vehicle while driving behind the leading vehicle. Then, the stochastic MPC approach allows to optimistically overtake the leading vehicle at the right part of the race track, based on the results of the Gaussian process.

References

[1] A. Wischnewski, J. Betz, and B. Lohmann. A model-free algorithm to safely approach the handling limit of an autonomous racecar. In *2019 IEEE International Conference on Connected Vehicles and Expo (ICCVE)*, pages 1–6, 2019.

[2] T. Stahl, A. Wischnewski, J. Betz, and M. Lienkamp. Multilayer graph-based trajectory planning for race vehicles in dynamic scenarios. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 3149–3154, 2019.

[3] U. Rosolia, A. Carvalho, and F. Borrelli. Autonomous racing using learning model predictive control. In *2017 American Control Conference (ACC)*, pages 5115–5120, 2017.

[4] U. Rosolia and F. Borrelli. Learning how to autonomously race a car: A predictive control approach. *IEEE Transactions on Control Systems Technology*, 28(6):2713–2719, 2020.

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

[6] A. Capone and S. Hirche. Backstepping for partially unknown nonlinear systems using gaussian processes. *IEEE Control Systems Letters*, 3(2):416–421, 2019.

[7] P. Jagtap, George J. Pappas, and M. Zamani. Control barrier functions for unknown nonlinear systems using gaussian processes*. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 3699–3704, 2020.

[8] A. Capone and S. Hirche. Anticipating the long-term effect of online learning in control. In *2020 American Control Conference (ACC)*, pages 3865–3872, 2020.

[9] J.F. Fisac, A.K. Akametalu, M.N. Zeilinger, S. Kaynama, J. Gillula, and C.J. Tomlin. A general safety framework for learning-based control in uncertain robotic systems. *IEEE Transactions on Automatic Control*, 64(7):2737–2752, 2019.

[10] L. Hewing, A. Liniger, and M.N. Zeilinger. Cautious nmpc with gaussian process dynamics for autonomous miniature race cars. In *2018 European Control Conference (ECC)*, pages 1341–1348, 2018.

[11] 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*, pages 1–1, 2021.

[12] A. Mesbah. Stochastic model predictive control: An overview and perspectives for future research. *IEEE Control Systems*, 36(6):30–44, Dec 2016.

[13] M. Farina, L. Giulioni, and R. Scattolini. Stochastic linear model predictive control with chance constraints – a review. *Journal of Process Control*, 44(Supplement C):53 – 67, 2016.

[14] A. Carvalho, Y. Gao, S. Lefevre, and F. Borrelli. Stochastic predictive control of autonomous vehicles in uncertain environments. In *12th International Symposium on Advanced Vehicle Control*, Tokyo, Japan, 2014.

[15] G. Schildbach and F. Borrelli. Scenario model predictive control for lane change assistance on highways. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, pages 611–616, Seoul, South Korea, June 2015.

[16] G. Schildbach, L. Fagiano, C. Frei, and M. Morari. The scenario approach for stochastic model predictive control with bounds on closed-loop constraint violations. *Automatica*, 50(12):3009 – 3018, 2014.

[17] G. Cesari, G. Schildbach, A. Carvalho, and F. Borrelli. Scenario model predictive control for lane change assistance and autonomous driving on highways. *IEEE Intelligent Transportation Systems Magazine*, 9(3):23–35, Fall 2017.

[18] T. Brüdigam, M. Olbrich, D. Wollherr, and M. Leibold. Stochastic model predictive control with a safety guarantee for automated driving. *IEEE Transactions on Intelligent Vehicles*, pages 1–1, 2021.

[19] T. Brüdigam, M. Olbrich, D. Wollherr, and M. Leibold. Stochastic model predictive control with a safety guarantee for automated driving: Extended version, 2020. arXiv: 2009.09381.

[20] C.E. Rasmussen. Gaussian processes in machine learning. In *Summer school on machine learning*, pages 63–71. Springer, 2003.

[21] C.A. Micchelli, Y. Xu, and H. Zhang. Universal kernels. *Journal of Machine Learning Research*, 7(95):2651–2667, 2006.

[22] J. Wilson, V. Borovitskiy, A. Terenin, P. Mostowsky, and M. Deisenroth. Efficiently sampling functions from Gaussian process posteriors. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 10292–10302. PMLR, 13–18 Jul 2020.

[23] L. Grüne and J. Pannek. *Nonlinear Model Predictive Control*. Springer-Verlag, London, 2017.