Gaussian Process based Model Predictive Control for Overtaking Scenarios at Highway Curves

Wenjun Liu[1], Yulin Zhai[1], Guang Chen[2,1,*] and Alois Knoll[1]

Abstract— **Model predictive control (MPC) is a commonly applied vehicle control technique, but its performance depends highly on how accurate the model captures the vehicle dynamics. It is disreputable hard to get a precise vehicle model in complex situations. The unmodeled dynamic will cause the uncertainty of the prediction which brings the risk while overtaking. To address this issue, Gaussian process (GP) regression is employed to acquire the unexplored discrepancy between the nominal vehicle model and the real vehicle dynamics which can result in a more accurate model. To achieve safe overtaking at highway curves, the constraint conditions are carefully designed. The implementation of GP-based MPC including approximate uncertainty propagation and safety constraints ensures that the ego vehicle overtakes the obstacles without collision. Simulation results show that GP-based MPC has a strong adaptability to different scenarios and outperforms MPC in overtaking control.**

I. INTRODUCTION

Vehicle control especially overtaking control is an important field of autonomous driving research. As a commonly used vehicle control strategy, model predictive control (MPC) utilises an explicit model of the dynamic procedure to get future plant response [1]. However, the capability of the MPC is highly dependent on how accurate the model reflects the dynamic process, which indicates a small disturbance will cause a huge impact on the performance. In addition, the complexity of the model may also lead to a large increase in calculation affecting the overall efficiency [2]. Therefore, a learning-based MPC controller, which combines a fixed uncomplicated nominal model and a learning based disturbance model, is considered a promising approach to capture unmodeled dynamics and hence enhance the accuracy of the vehicle controller.

Gaussian process (GP) is a recently widely used probabilistic machine learning approach which can represent the uncertainty of estimation based on prior process knowledge and show excellent performance when combined with MPC [3]. According to this feature, GP can be used as a learning-based model in MPC to predict periodic time-varying effects [4]. An application of constrained tracking MPC combined with GP provided by Ostafew et al. [5], which reduces the path-tracking errors of robots, shows that GP can achieve a high-performance dynamic modelling from measured data. Hewing et al. designed designed a GP-based MPC (GPMPC) structure to enhance conventional MPC control effect for a race car [6]. Arany applied GPMPC to obtain vehicle safe and stable control under changing friction road conditions [7]. GPMPC is also implemented to overtaking problems in autonomous driving [8], but it's only applicable to overtaking scenarios on straights.

Overtaking is one of the most complex manoeuvres for road automation which comprises three consecutive steps [9]. The vehicle which plans to overtake is called the ego vehicle and the one which will be overtaken is named the lead vehicle. In general, an autonomous overtaking system consists of two important functions, which were designated as trajectory planning and trajectory tracking [10]. The method based on these functions was typically applied in [11]. However, only a elementary point mass model is considered and the method is not applicable at high speed. Recently, many researchers deal with trajectory planning and trajectory tracking at the same time in a controller. This can be typically seen in [12]. On the other hand, only the static obstacles rather than moving obstacle vehicles are considered which leads to the increased complication when there exist moving obstacles. In [13], a clever overtaking prototype is proposed based on adaptive MPC considering both stationary obstacles and moving obstacle vehicles. The problem of this method is that only the kinematics model is considered. The lateral control on the subject of the tire model is also simplified which results in generating infeasible trajectories. Besides, the algorithm also only considers the application of overtaking scenarios on the straight road. In [14], the authors achieve overtaking by using a high-level decision maker based on finite state machine and a trajectory planner based on chance constrained MPC, however, the algorithm is also only applicable to overtaking scenarios on straights.

As mentioned above, most of the previous studies are based on straight road scenarios and suffer from unmodeled dynamics. If the unmodeled uncertainty is not considered, the overtaking controller can easily drive off the current road or crash the overtaken vehicle, as illustrated in Fig. 1 (a), (b) and (c). In the current study, GPMPC is used to solve the overtaking problems with specific constraints and a model allowing to acquire unmodeled dynamics is proposed. The trajectory generated by GPMPC can be more accurate in curve road scenarios. Its performance is shown in Fig. 1 (d), (e) and (f).

The major intention of this paper is to investigate the application of GPMPC vehicle controller in overtaking scenarios, especially at highway curves. In brief, the contributions will be outlined as follows:

- We introduce a double circle parametrisation of the

[1]The authors are with the Department of Informatics, Technical University of Munich, Munich, Germany {wenjun.liu, knoll}@in.tum.de, zhaiyl926@163.com

[2]The author is with the School of Automotive Studies, Tongji University, Shanghai, China guangchen@tongji.edu.cn

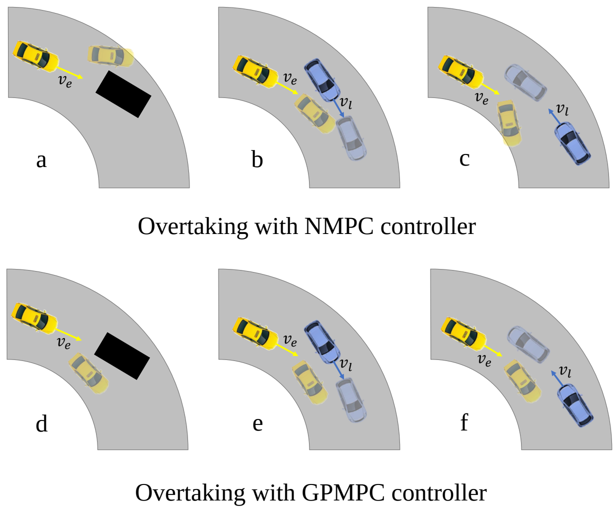

v_e
a
v_e
v_l
b
v_e
v_l
c
Overtaking with NMPC controller
v_e
d
v_e
v_l
e
v_e
v_l
f
Overtaking with GPMPC controller

Fig. 1. Three major overtaking scenarios will be introduced, namely avoiding a still obstacle (black rectangle) shown in (a) and (d), the ego vehicle (yellow car) and the lead vehicles (blue car) driving in the same direction shown in (b) and (e) and opposite direction shown in (c) and (f), respectively. The transparency vehicles represent the predicted position by the controller. Due to the disturbance, driving with NMPC may lead to the collision or being out of boundary while overtaking as shown in (a), (b) and (c). Using GPMPC under the same conditions can ensure the safety and provide a better trajectory to overtake as shown in (d), (e) and (f).

shape of the vehicle. We then achieve safe overtaking on both straight and curved roads by carefully designing the safety constraints based on the double circle parametrisation, which also eliminates the need for path planning.

- We utilise a comparatively uncomplicated vehicle model, but attain better control effects through GP learning.
- We convert the vehicle state constraints into a soft constraint consisting in a relaxed barrier function related to vehicle states. We then integrate this item into a MPC stage cost function, which improves the efficiency of the optimiser.

II. Basic Theory

A. Gaussian Process Regression

The GP regression aims to construct a function that can predict the output at an unrecognised input location, given a certain training data set with input data and corresponding output value pairs. The notation of the training data set can be described as follows:

$$\mathscr{D}=\{\mathbf{Z}=[z_1,\ldots,z_N]\in\mathbb{R}^{n_z\times N}$$
$$\mathbf{Y}=[y_1,\ldots,y_N]\in\mathbb{R}^{1\times N}\}$$

where $z_k\in\mathbb{R}^{n_z}, k=1,...,N$ stands for the input vector and $y_k\in\mathbb{R}, i=1,...,N$ denotes the output vector which is given as below:

$$y_k=d(z_k)+\varepsilon_k \tag{1}$$

where $d:\mathbb{R}^{n_z}\rightarrow\mathbb{R}$ represents an unknown function and $\varepsilon_k\sim\mathscr{N}\left(0,\sigma_\varepsilon^2\right)$ represents the measurement noise of Gaussian process. The posterior distribution of the unknown function at a certain test data z_* is also a Gaussian distribution which could be represented by a mean function $\mu^d(z_*)$ and a variance function $\Sigma^d(z_*)$ [15]:

$$\mu^d(z_*)=\mathbf{K}_*^\top\left[\mathbf{K}+\sigma_n^2\mathbf{I}\right]^{-1}\mathbf{Y} \tag{2}$$

$$\Sigma^d(z_*)=\mathbf{K}_{*,*}-\mathbf{K}_*^\top\left[\mathbf{K}+\sigma_n^2\mathbf{I}\right]^{-1}\mathbf{K}_* \tag{3}$$

where d stands for the d-th dimension of the output. $\mathbf{K}$, $\mathbf{K}_*$ and $\mathbf{K}_{*,*}$ are abbreviated to $\mathbf{K}(\mathbf{Z},\mathbf{Z})$, $\mathbf{K}(\mathbf{Z},z_*)$ and $\mathbf{K}(z_*,z_*)$, separately. Where $[\mathbf{K}(\mathbf{Z},\mathbf{Z})]_{ij}=k(z_i,z_j)$, $[\mathbf{K}(\mathbf{Z},z_*)]_j=k(z_j,z_*)$ and $\mathbf{K}(z_*,z_*)=k(z_*,z_*)$. In this study, a squared exponential kernel with noise variance σ_n^2 is considered to be the choice of the kernel function.

$$k(z,z')=\sigma_f^2\exp\left(-\frac{1}{2}(z-z')^{\mathrm{T}}\mathbf{M}^{-1}(z-z')\right)+\sigma_n^2 \tag{4}$$

where $\mathbf{M}=\operatorname{diag}\left(\left[\ell_1,\ldots,\ell_{n_z}\right]\right)$ stands for the diagonal positive semi-definite matrix used to parameterise the length scale of the covariance and σ_f denotes the signal variance. The free parameters σ_n, σ_f and $\mathbf{M}$ of the kernel were combined into the following hyperparameter vector $\theta=[\ell_1,...,\ell_{n_z},\sigma_f^2,\sigma_n^2]$. To achieve good prediction, the hyperparameters are optimised by maximum likelihood estimate with the log marginal likelihood as evidence [16].

The result of multivariate GP approximation at an unobserved test point z_* combining the output with n_d dimensions is specified as:

$$d(z_*)\sim\mathscr{N}\left(\mu^d(z_*),\Sigma^d(z_*)\right) \tag{5}$$

where $\mu^d(z_*)=\left[\mu^1(z_*),\ldots,\mu^{n_d}(z_*)\right]^\top$, $\Sigma^d(z_*)=\operatorname{diag}\left(\left[\Sigma^1(z_*);\ldots;\Sigma^{n_d}(z_*)\right]\right)$

B. Model Predictive Control

This study mainly concentrates on nonlinear model predictive control (NMPC). The form of the non-linear prediction model is given by:

$$\begin{aligned}x_{k+1}&=f(x_k,u_k)\\ y_k&=h(x_k,u_k)\end{aligned} \tag{6}$$

where $x_k\in\mathbb{R}^n$ represents the system model state and $u_k\in\mathbb{R}^m$ denotes the control inputs sequence. The function f refers to the discrete-time dynamics of the system model and h is the output function related to system states and control inputs.

The purpose of MPC is to minimise the cost function based on the system state and control input constraints. According to Rawlings et al. [17], the cost function $J(k)$ will be written into two parts which are stage cost f_0 and final cost $\phi(x_N)$ for simplification. For now, the constraints are only defined as box constraints with state and input permissible values according to $\mathscr{X}=\{x\in\mathbb{R}^n\mid x_{\min}\leq x\leq x_{max}\}$ and $\mathscr{U}=\{u\in\mathbb{R}^m\mid u_{\min}\leq u\leq u_{\max}\}$.

So far with the predefined cost function and constraints, the MPC optimal control problem will be stated as:

$$\begin{aligned}\min_{x,u} \; & J(k)=\sum_{k=0}^{N-1} f_o\left(x_k, u_k\right)+\phi\left(x_N\right) \\ \text{s.t. } & x_{k+1}=f\left(x_k, u_k\right) \\ & x_k \in \mathscr{X}, k=0, \ldots, N \\ & u_k \in \mathscr{U}, k=0, \ldots, N-1,\end{aligned} \tag{7}$$

When using the above MPC formulation several steps should be followed. First, measurement or estimation of the current state x_0 should be obtained. The second step is to solve the open-loop optimal control formulation of (7). Then the the optimal control input sequence will be obtained. Finally, the first component of it should be applied on the system.

C. Vehicle Dynamic Model

A non-linear single-track approximate model is utilised to imitate the movement of the autonomous vehicle. The morphological parameters of the vehicle are vehicle mass M, the moment of inertia I, steering angle δ, and the distance from the front and rear wheel to the center of gravity of the vehicle L_f and L_r. The longitudinal and lateral direction forces on the front and rear wheels are represented as $F_{f,x}$, $F_{f,y}$, $F_{r,x}$ and $F_{r,y}$, respectively. Lastly, given the global position coordinates (X,Y), the yaw angle φ, v_x and v_y which refers to the longitudinal and lateral velocities of the vehicle and the yaw rate ω, the vehicle dynamic model will be stated:

$$\dot{x}(x,u)=\begin{bmatrix} v_x\cos(\varphi)-v_y\sin(\varphi) \\ v_x\sin(\varphi)+v_y\cos(\varphi) \\ \omega \\ \frac{1}{M}\left(F_{r,x}+F_{f,x}\cos(\delta)-F_{f,y}\sin(\delta)+M\omega v_y\right) \\ \frac{1}{M}\left(F_{r,y}+F_{f,x}\sin(\delta)+F_{f,y}\cos(\delta)-M\omega v_x\right) \\ \frac{1}{I}\left(F_{f,y}L_f\cos(\delta)+F_{f,x}L_f\sin(\delta)-F_{r,y}L_r\right)\end{bmatrix} \tag{8}$$

where the single-track model state is $x=[X,Y,\varphi,v_x,v_y,\omega]^\top$ and $u=[\delta,T]^\top$ represnts the control input. T ($T\in[-1,1]$) denotes the acceleration or deceleration pedal.

The longitudinal forces $F_{f,x}$ and $F_{r,x}$ can be simply calculated with T and the torque distribution ζ, the specific details can be found in our previous work [8]. For the lateral forces $F_{f,y}$ and $F_{r,y}$, Pacejka [18] uses the Magic formula to define these forces in order to approximate the non-linear lateral wheel dynamics:

$$\begin{aligned} F_{f,y}&=D_f\sin\left[C_f\arctan\left(B_f\alpha_f-E_f\left(B_f\alpha_f-\arctan\left(B_f\alpha_f\right)\right)\right)\right] \\ F_{r,y}&=D_r\sin\left[C_r\arctan\left(B_r\alpha_r-E_r\left(B_r\alpha_r-\arctan\left(B_r\alpha_r\right)\right)\right)\right]\end{aligned} \tag{9}$$

where B, D are construed respectively as stiffness and peak factor. C and E are factors both related to shape. α_f and α_r represent the front wheel slip angle and rear wheel slip angle, separately. The model using this Magic formula serves as the real dynamic vehicle model in this study. To explore the potential of the learning-based MPC controller, a simpler nominal vehicle model with linear lateral wheel dynamics is here considered [19]:

$$\begin{aligned} F_{f,y}&=C_{l,f}\alpha_f \\ F_{r,y}&=C_{l,r}\alpha_r\end{aligned} \tag{10}$$

where $C_{l,f}$ and $C_{l,r}$ denote the cornering stiffness from the front and rear sides. The front and rear wheel slip angles α_f and α_r are expressed as:

$$\begin{aligned} \alpha_f&=\arctan\left(\frac{v_y+L_f\omega}{v_x}\right)-\delta \\ \alpha_r&=\arctan\left(\frac{v_y-L_r\omega}{v_x}\right)\end{aligned} \tag{11}$$

The deviation between the true and nominal model will be seen as the unmodeled dynamics and will be captured by GP to learn the vehicle movement process and provide to the MPC controller.

III. GP-based MPC Implementation

A. Model Learning and Data Acquisition

Adding a GP leaning module into the NMPC controller which can capture the unmodeled dynamics offers many benefits [6]. According to the vehicle dynamic model stated in (8), a discrete-time model can be established using the fourth order Runge-Kutta method which can be modified as:

$$x_{k+1}=f_{nom}\left(x_k,u_k\right) \tag{12}$$

where $f_{nom}(x_k,u_k)$ represents the nominal function also known as the discretization of the vehicle dynamic model. Thus, the resulting learning-based prediction which combines the nominal model and a disturbance model is formulated as:

$$x_{k+1}=f_{nom}\left(x_k,u_k\right)+\mathbf{B}_d\left(d\left(z_k\right)+\omega_k\right) \tag{13}$$

where $z_k=[x_k;u_k]$ constitute a pair of state and input data points representing regression features and d refers to the GP for learning unknown dynamics. The matrix $\mathbf{B}_d$ is used to select the states of x_{k+1} which are influenced by the model error. In this study, only the velocity v_x and v_y and the yaw rate ω will be considered to be influenced by the disturbance, which means the matrix in this case will be set as $\mathbf{B}_d=[0\ \mathbf{I}_3]^\top$. The process noise ω_k is independent and identically distributed, hence $\omega_k\sim\mathscr{N}(0,\Sigma^\omega)$, where $\Sigma^\omega=\operatorname{diag}\left[\sigma_{v_x}^2,\sigma_{v_y}^2,\sigma_\omega^2\right]$. The measurement data which is used for the GP model to learn the unmodeled dynamic is calculated by the prediction error at every time step. Therefore, the training output of the input data z_k is given by:

$$y_k=d\left(x_k,u_k\right)+\omega_k=\mathbf{B}_d^\dagger\left(x_{k+1}-f_{nom}\left(x_k,u_k\right)\right) \tag{14}$$

Here $\mathbf{B}_d^\dagger$ denotes the Moore-Penrose pseudo-inverse. The input vector and output data pair (z_k,y_k) is the training data of the GP model. The three affected states are independent of each output dimension which results in the situation that the training for the GP model of these three states is separated [20]. There also exists a problem of training capacity. A large number of data will increase the calculated complexity. To keep the training data size within a balanced range, we limited the capacity of the data dictionary to N_{max}. When the maximum size N_{max} of the training data is reached, some old data should be replaced by new data [8]. The replaced data selection mechanism is according to a distance measure Θ_*.

This method can refer to [2]. The distance measure of a specified data location point z_* is a posterior variance based on all the other data in the training data set $\mathbf{Z}_{\setminus *}$, which can be expressed as follows:

$$\Theta_* = \mathbf{K}_{z_*,z_*} - \mathbf{K}_{z_*,\mathbf{Z}_{\setminus *}}\left(\mathbf{K}_{\mathbf{Z}_{\setminus *},\mathbf{Z}_{\setminus *}} + \sigma \mathbf{I}\right)^{-1} \mathbf{K}_{\mathbf{Z}_{\setminus *},z_*} \tag{15}$$

σ represents a parameter to be tuned. At every selection process, the old data with the lowest Θ_* should be replaced by new data.

B. Approximate Uncertainty Propagation

When the prediction needs several steps forward, the stochastic output needs to be used as input in the next prediction [21]. For a test input z_* with a Gaussian distribution $z_* \sim \mathcal{N}\left(\mu_{z_*}, \Sigma_{z_*}\right)$, the exact predictive distribution can be described by:

$$p\left(d\left(z_*\right) \mid \mu_{z_*}, \Sigma_{z_*}\right) = \int p\left(d\left(z_*\right) \mid z_*\right) p\left(z_k \mid \mu_{z_*}, \Sigma_{z_*}\right) \mathrm{d}z_* \tag{16}$$

Under normal conditions, the Gaussian distribution mapped from a non-linear function results in a non-Gaussian distribution. The predictions then will become random variables and the evaluation of (16) is analytically intractable [22]. One solution to make the prediction tractable is to utilise the extended Kalman filter (EKF) approach to linearise the random variables stated in (13). At every time step the states and non-linear disturbances are jointly Gaussian distributed, which can be written as:

$$\begin{aligned} &\left[\begin{array}{cc} x_k^{\mathrm{T}} & \left(d_k + w_k\right)^{\mathrm{T}} \end{array}\right]^{\mathrm{T}} \sim \mathcal{N}\left(\mu_k, \Sigma_k\right) \\ &= \mathcal{N}\left(\left[\begin{array}{c} \mu_k^x \\ \mu_k^d \end{array}\right], \left[\begin{array}{cc} \Sigma_k^x & \left(\Sigma_k^{dx}\right)^{\mathrm{T}} \\ \Sigma_k^{dx} & \Sigma_k^d + \Sigma^w \end{array}\right]\right) \end{aligned} \tag{17}$$

where d_k is the GP disturbance. Σ_k^{dx} stands for the covariances between states and GP. The predicted states distribution can be depicted by updating the state mean and variance:

$$\mu_{k+1}^x = f_{nom}\left(\mu_k^x, u_k\right) + \mathbf{B}_d \mu_k^d \tag{18}$$

$$\Sigma_{k+1}^x = \left[\nabla_x f_{nom}\left(\mu_k^x, u_k\right) \quad \mathbf{B}_d\right] \Sigma_k \left[\nabla_x f_{nom}\left(\mu_k^x, u_k\right) \quad \mathbf{B}_d\right]^{\top} \tag{19}$$

For now, the result of the linearised transformation is Gaussian distributed. Several terms in the mean and variance from the equations (18) and (19) can be approximated using a first order Taylor approximation based on simple and cheap computation which results in:

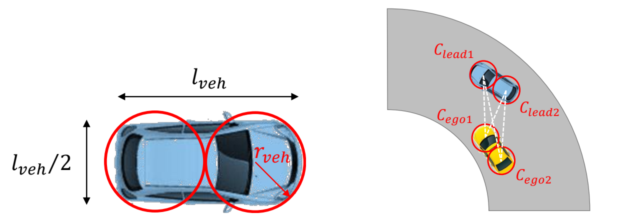
$$\mu_k^d = \mu^d\left(\mu_k^z\right), \quad \Sigma_k^d = \Sigma^d\left(\mu_k^z\right), \quad \Sigma_k^{dx} = \nabla_x \mu^d\left(\mu_k^z\right) \Sigma_k^x \tag{20}$$

C. Safe Overtaking Constraints in Curves

Proper constraints can ensure safety while overtaking for both the ego vehicles and lead vehicles. Constraints can be divided into two types: internal and external constraints [23]. Internal constraints refer to the limitation of the vehicle kinematics and dynamic model. Therefore, the input constraints $\mathscr{U}$ will be subject to the following restrictions:

$$\left[\begin{array}{c} \delta_{\min} \\ T_{\min} \end{array}\right] \leq \left[\begin{array}{c} \delta \\ T \end{array}\right] \leq \left[\begin{array}{c} \delta_{\max} \\ T_{\max} \end{array}\right] \tag{21}$$

On the contrary, external constraints focus more on the driving corridor and obstacles. The vehicle used in this



(a) Vehicle shape (b) Overtaking maneuver at curve

Fig. 2. Circle decomposition of the vehicle

simulation is approximated as a rectangle with the length l_{veh} and the width $l_{veh}/2$. Fig. 2 (a) illustrates the decomposition of the vehicle shape into two circles of radius r_{veh}. As shown in Fig. 2 (b), C_{ego1}, C_{ego1}, C_{lead1} and C_{lead2} refer to the circles decomposed by the ego vehicle and the lead vehicle, respectively. The distance between these circles for the different vehicles should satisfy the threshold and the calculation is based on the current yaw angle φ and the global position coordinates (X,Y) from the state vector x. This leads to four calculation pairs (C_{ego1}, C_{lead1}), (C_{ego1}, C_{lead2}), (C_{ego2}, C_{lead1}) and (C_{ego2}, C_{lead2}) as follows:

$$\begin{aligned} &((X_e + \frac{l_{veh}}{2}\cos\varphi_e) - (X_l + \frac{l_{veh}}{2}\cos\varphi_l))^2 + \\ &((Y_e + \frac{l_{veh}}{2}\sin\varphi_e) - (Y_l + \frac{l_{veh}}{2}\sin\varphi_l))^2 \geq (2r_{veh})^2 \\ &((X_e + \frac{l_{veh}}{2}\cos\varphi_e) - (X_l - \frac{l_{veh}}{2}\cos\varphi_l))^2 + \\ &((Y_e + \frac{l_{veh}}{2}\sin\varphi_e) - (Y_l - \frac{l_{veh}}{2}\sin\varphi_l))^2 \geq (2r_{veh})^2 \\ &((X_e - \frac{l_{veh}}{2}\cos\varphi_e) - (X_l + \frac{l_{veh}}{2}\cos\varphi_l))^2 + \\ &((Y_e - \frac{l_{veh}}{2}\sin\varphi_e) - (Y_l + \frac{l_{veh}}{2}\sin\varphi_l))^2 \geq (2r_{veh})^2 \\ &((X_e - \frac{l_{veh}}{2}\cos\varphi_e) - (X_l - \frac{l_{veh}}{2}\cos\varphi_l))^2 + \\ &((Y_e - \frac{l_{veh}}{2}\sin\varphi_e) - (Y_l - \frac{l_{veh}}{2}\sin\varphi_l))^2 \geq (2r_{veh})^2 \end{aligned} \tag{22}$$

where X_e, Y_e, φ_e and X_l, Y_l, φ_l represent the global position coordinates and current yaw angle of the ego vehicle and the lead vehicle severally.

D. Cost Function and GPMPC Formulation

A strategy named model predictive contouring controller (MPCC) which aims to maximise the progress of vehicle's movement on a specified reference path will be used to establish the cost function [12]. In our study, the reference tracking path is specified as the centre line of a certain lane, but here it is only a measure of progress. The arc length $\xi \in [0, \xi_{max}]$ of the track of the center line is chosen to parameterise the track by using third order spline polynomials, which represents the traveled length along the reference path in the current lap. The corresponding center line position given an ξ is written as $[X_c(\xi), Y_c(\xi)]$, as well as orientation $\Phi_c(\xi)$ and the track radius $R_c(\xi)$ can be obtained using third order spline polynomials interpolation. The contour error e_c, lag error e_l, offset error e_{off} and the orientation error e_o are used

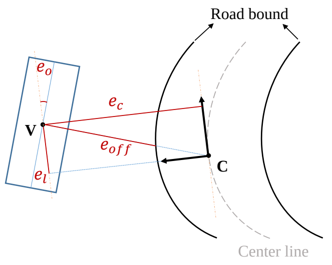


Fig. 3. Lag-, contour-, orientation- and offset error with actual vehicle position V and the the vehicle position C on the centre line projected by the parameter ξ

to penalise the discrepancy from the centre line and ensure the prediction of the vehicle movement close to it. These penalties which are given by the link of the real vehicle position and the position on the centre line shown in Fig. 3 are defined as:

$$\begin{aligned}
e_l\left(u_k^x, \xi_k\right) =& -\cos\left(\Phi\left(\xi_k\right)\right)\left(X_c\left(\xi_k\right)-X_k\right) \\
& +\sin\left(\Phi\left(\xi_k\right)\right)\left(Y_c\left(\xi_k\right)-Y_k\right) \\
e_c\left(u_k^x, \xi_k\right) =& -\sin\left(\Phi\left(\xi_k\right)\right)\left(X_c\left(\xi_k\right)-X_k\right) \\
& -\cos\left(\Phi\left(\xi_k\right)\right)\left(Y_c\left(\xi_k\right)-Y_k\right) \\
e_o\left(u_k^x, \xi_k\right) =& 1-\left|\cos\left(\Phi\left(\xi_k\right)\right)\cos(\varphi)+\sin\left(\Phi\left(\xi_k\right)\right)\sin(\varphi)\right| \\
e_{off}\left(u_k^x, \xi_k\right) =& \frac{1}{R_c\left(\xi_k\right)}\sqrt{e_l\left(u_k^x, \xi_k\right)^2+e_c\left(u_k^x, \xi_k\right)^2}-1
\end{aligned} \tag{23}$$

where $[X_k; Y_k]$ represents the current mean position of the vehicle. If there are no constraints, the MPC formulation can work more efficient. In order to guarantee that the vehicle does not leave the boundary of the road, there are inevitably vehicle state constraints. To improve the computing performance of MPC, we convert the vehicle state constraints into a soft constraint which is a relaxed barrier function related to vehicle states. Then we integrate this item into the stage cost function of MPC. The relaxed barrier function $\mathscr{R}_b(e_{off})$ is designed as:

$$\mathscr{R}_b(e_{off}) = \beta\left(\sqrt{\frac{\left(c+\gamma(\lambda-e_{off})^2\right)}{\gamma}}-(\lambda-e_{off})\right) \tag{24}$$

Here β, λ, γ and c represent the constant parameters. Including all the terms introduced above, the stage cost function can be expressed:

$$\begin{aligned}
l\left(u_k^x, \xi_k\right) =& \left\|e_c\left(u_k^x, \xi_k\right)\right\|_{q_c}^2+\left\|e_l\left(u_k^x, \xi_k\right)\right\|_{q_l}^2 \\
& +\left\|e_o\left(u_k^x, \xi_k\right)\right\|_{q_o}^2+\left\|\mathscr{R}_b(e_{off}\left(u_k^x, \xi_k\right))\right\|_{q_{off}}^2
\end{aligned} \tag{25}$$

where q_l, q_c, q_o and q_{off} are the weights for the corresponding penalty factor. The end cost $\phi(\mu_N^x)$ which is only affected by the last prediction step's state is simply defined as twice of the stage cost function without the influence by input factor:

$$\phi(\mu_N^x) = 2 \cdot f_0(\mu_N^x, 0) \tag{26}$$

Based on all the prerequisites of overtaking optimization problem, the GP-based MPC formulation can be written as:

$$\begin{aligned}
\min_u \; & J(\mu_k^x, \xi_k) = \sum_{k=0}^{N-1} l(\mu_k^x, \xi_k) + \phi(\mu_N^x) \\
\text{s.t. } & u_{k+1}^x = f_{nom}\left(u_k^x, u_k\right) + \mathbf{B}_d\left(d\left(u_k^x, u_k\right)+w_k\right) \\
& (22) \\
& u_k \in \mathscr{U}, k = 0, \ldots, N-1 \\
& \mu_0^x = x(k), \Sigma_0^x = 0, \xi_0 = \xi(k)
\end{aligned} \tag{27}$$

IV. Simulation and Result

In this paper, the basic framework of GPMPC is the same as that in [3] and [6]. But many details are different. For example, simplified state chance constraints are used in their paper, but we instead convert the vehicle state constraints into a soft constraint which is a relaxed barrier function related to the vehicle states. We then integrate this item into the cost function of MPC. What's more, the focus is different. In [3] and [6], they focus on autonomous racing. Their aim is to drive along a defined race track as quickly as possible without leaving the current track. In our study, we focus on improving vehicle overtaking. The safety constraints based on the geometric relationship between the ego vehicle and other obstacle vehicles are added. Then we can not only avoid driving out of the track, but also avoid dynamic and static obstacles in the track. Then the overtaking task can be achieved. But in [3] and [6], they did not consider the avoidance of dynamic and static obstacles. The focus is different, so the comparison is meaningless. Besides, it can often be seen that NMPC is used to avoid dynamic and static obstacles in recent studies [24], [25]. So, we compare the proposed GPMPC with NMPC algorithms in our study.

A. Simulation framework

In our study, three major overtaking scenarios will be introduced as shown in Fig. 1. In the first scenario, an obstacle stays still in the curve and the ego vehicle will try to avoid collision with it and pass the curve. In the scenario of the same direction, the lead vehicle will drive at a constant speed in the curve and the ego vehicle will move faster with a variable speed. The decision of overtaking that occurs from left or right depends on the calculation of the GPMPC controller. In the last opposite direction scenario, the lead vehicle is also given by an invariable speed and the ego vehicle can change the speed according to the different situations. In the simulation, the second and third scenarios will also extend to have multiple lead vehicles.

A circular runway with four ninety-degree angles and a width of 6 meters will be used in the simulation process. All the vehicles and obstacles will be abstracted into small blocks which are 2 meters long and 1 meter wide. The physical parameters of the vehicles and tire Magic formula will be given properly for preparing the simulation.

B. GPMPC controller setup

During the simulation, there will be a comparison between NMPC and GPMPC controller. For the basic MPC, the length of the prediction horizon is set to $N_p = 7$ and the maximum

TABLE I

INPUT CONSTRAINTS

$\delta_{min}(^o)$	T_{min}	$v_{track_{min}}(m/s)$	$\delta_{max}(^o)$	T_{max}	$v_{track_{max}}(m/s)$
-20	-1	5	20	1	25

number of MPC iterations for each time step is limited to 15. The input constraints from (21) are shown in Table I within the controllable range. The defined weights for the corresponding penalty factor of the cost function (25) as well as the smooth barrier function (24) will also be given within the feasible range.

The steps of the simulation with the GPMPC controller are as follows. First, at the initial point the ego vehicle starts only with the NMPC controller which means all the GP variables are kept to zero. After collecting the data into the dictionary from step one, the GP model will be trained to learn the disturbance. Then, by maximizing the log marginal likelihood the optimal hyperparameters will be obtained. Using the loaded data from the dictionary the GPMPC will be activated to predict the trajectory. Please note that during the process the new coming data will be added to the dataset and some old data will be deleted according the data selection method mentioned in (15) when the dataset size reaches $N_{max} = 300$.

C. Result and Analysis

For easier observing, lead vehicles and obstacles will be shown as black-filled blocks and the ego vehicle is a hollow block. The sampling time is set to $T_s = 20s$ with the time step size $d_t = 0.1s$. The blue to yellow colored line represents the route of ego vehicle already traveled and the dots ahead is the trajectory prediction within the predicted horizon. The grey dots refer to the approximate vehicle position projected on the center line. The red arrows show the driving orientation of lead vehicle. We limit the speed of the ego vehicle from 5 m/s to 25 m/s considering the reality. The lead vehicles will move with a invariable speed of 4 m/s and 1 m/s in the same and opposite direction scenarios, respectively.

The left column of Fig. 4 presents the trajectory prediction with NMPC in different scenarios and the right column is with GPMPC. Images in the same row represent the same scenario. Fig. 4 (a) and 4 (b) are the scenario of avoiding a still obstacle. Fig. 4 (c) to 4 (h) are the scenarios of driving in the same direction with a different number of lead vehicles. Fig. 4 (i) to 4 (n) represent the opposite direction scenarios. As demonstrated in Fig. 4 (a) and 4 (i), the ego vehicle with NMPC will face the problem of being out of road bound while overtaking. As a comparison, GPMPC performs better in the same situation and can keep the vehicle within the boundary. Fig. 4 (c) and 4 (m) also reveal that NMPC increases the risk of crashing onto moving lead vehicles. GPMPC can resolve the crisis as shown in Fig. 4 (d) and 4 (n) and accomplish the overtaking mission without collision. Meanwhile, GPMPC can maximise the progress along the centre line and keep an advisable speed while ensuring safety.

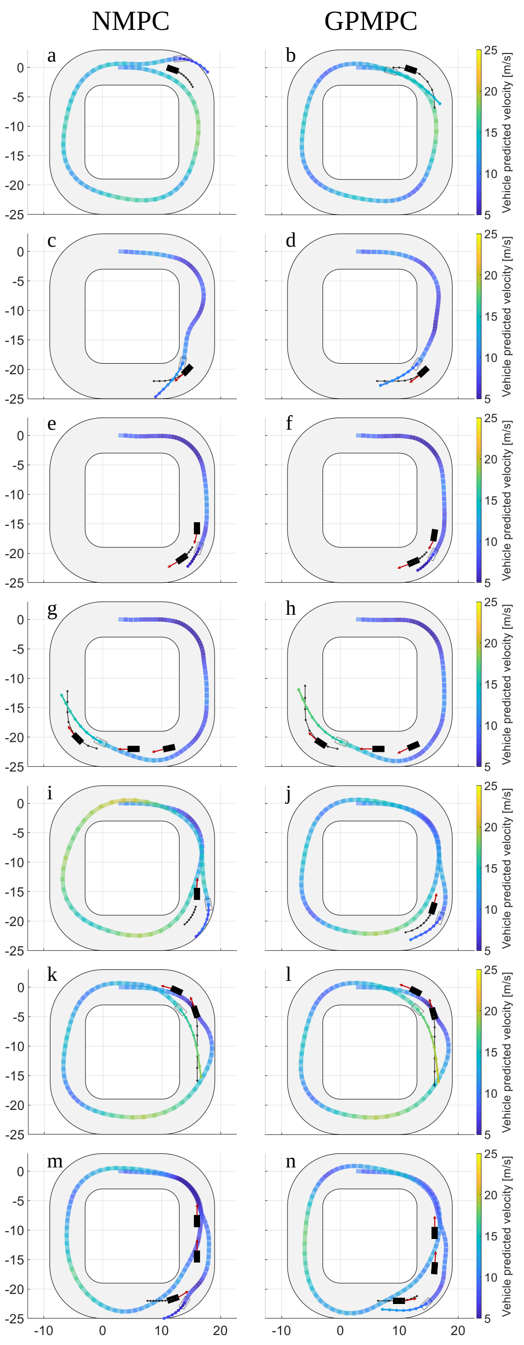


Fig. 4. Comparison of overtaking scenarios between NMPC and GPMPC

Regardless of the different scenes, GPMPC can always give a proper control input which shows the strong adaptability of the updated vehicle controller to different situations.

To quantify how GPMPC improves the prediction performance compared to NMPC owing to the GP learning, the remaining model error in the dimension of v_x, v_y and ω will be determined by the nominal model with NMPC and learning model with GPMPC, separately. The prediction error is the result of subtracting the real state and the predicted state of each controller as defined in (28) and (29):

$$\|\mathbf{e}_{NMPC}\| = \|x_{k+1} - f(x_k, u_k)\| \tag{28}$$

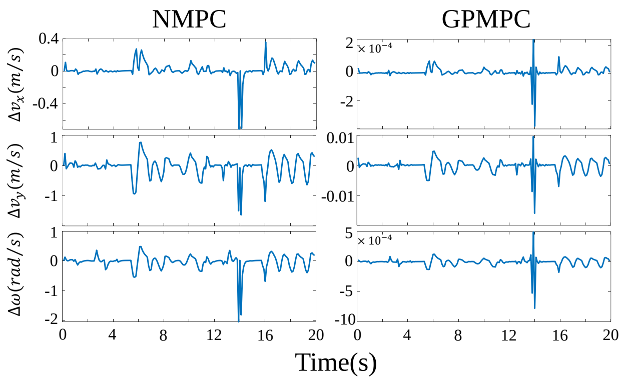


Fig. 5. Prediction error of same direction scenario with one lead vehicle

TABLE II
MSE OF THE PREDICTION ERROR

Model	$\lVert \mathbf{e}_{v_x} \rVert$	$\lVert \mathbf{e}_{v_y} \rVert$	$\lVert \mathbf{e}_{\omega} \rVert$	$\lVert \mathbf{e} \rVert$
NMPC	9.101e-3	1.053e-1	7.056e-2	1.849e-1
GPMPC	2.012e-9	4.961e-6	8.403e-9	1.655e-6

$$\lVert \mathbf{e}_{GPMPC} \rVert = \left\lVert x_{k+1} - \left(f\left(x_k, u_k\right) + \mathbf{B}_d \mu^d\left(z_k\right) \right) \right\rVert \tag{29}$$

The comparison of the prediction error for the second scenario with one lead vehicle is shown in Fig. 5. From the diagrams, it is obvious to see that GPMPC executes much better than NMPC after using the optimal hyperparameters. GPMPC can significantly reduce the prediction error by several orders of magnitude for every dynamic state. For better understanding the ability of the GP, the mean squared error (MSE) of the prediction error of each dynamic state including the average value are shown in Table II. GPMPC has five orders of magnitude higher accuracy than NMPC which shows that GPMPC can bring a strong improvement to the vehicle controller performance.

V. Conclusion

This study has presented an MPC vehicle controller combined with a GP regression model that allows for autonomous driving of different overtaking scenarios on curved roads. The GP model can learn unknown dynamics to reduce the prediction uncertainty, which can considerably enhance the performance of the nominal MPC controller. The specific cost function and constraints are also established to ensure safety while overtaking vehicles in curves. The simulation result shows that the GPMPC controller can accomplish overtaking tasks no matter on straights or corners without collision and reduce the prediction error. The application in the different scenarios with multiple obstacles also shows the strong adaptability of GPMPC.

References

[1] J. Kocijan, R. Murray-Smith, C. E. Rasmussen, and A. Girard, "Gaussian process model based predictive control," in *Proceedings of the 2004 American control conference*, vol. 3. IEEE, 2004, pp. 2214–2219.
[2] J. Kabzan, L. Hewing, A. Liniger, and M. N. Zeilinger, "Learning-based model predictive control for autonomous racing," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3363–3370, 2019.
[3] 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, pp. 2736–2743, 2019.
[4] E. D. Klenske, M. N. Zeilinger, B. Schölkopf, and P. Hennig, "Gaussian process-based predictive control for periodic error correction," *IEEE Transactions on Control Systems Technology*, vol. 24, no. 1, pp. 110–121, 2015.
[5] C. J. Ostafew, A. P. Schoellig, T. D. Barfoot, and J. Collier, "Learning-based nonlinear model predictive control to improve vision-based mobile robot path tracking," *Journal of Field Robotics*, vol. 33, no. 1, pp. 133–152, 2016.
[6] 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)*. IEEE, 2018, pp. 1341–1348.
[7] R. Rezvani Arany, "Gaussian process model predictive control for autonomous driving in safety-critical scenarios," 2019.
[8] W. Liu, C. Liu, G. Chen, and A. Knoll, "Gaussian process based model predictive control for overtaking in autonomous driving," *Frontiers in Neurorobotics*, vol. 15, 2021.
[9] P. Petrov and F. Nashashibi, "Modeling and nonlinear adaptive control for autonomous vehicle overtaking," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 4, pp. 1643–1656, 2014.
[10] S. Dixit, S. Fallah, U. Montanaro, M. Dianati, A. Stevens, F. Mccullough, and A. Mouzakitis, "Trajectory planning and tracking for autonomous overtaking: State-of-the-art and future prospects," *Annual Reviews in Control*, vol. 45, pp. 76–86, 2018.
[11] Y. Gao, T. Lin, F. Borrelli, E. Tseng, and D. Hrovat, "Predictive control of autonomous ground vehicles with obstacle avoidance on slippery roads," in *Dynamic systems and control conference*, vol. 44175, 2010, pp. 265–272.
[12] A. Liniger, A. Domahidi, and M. Morari, "Optimization-based autonomous racing of 1: 43 scale rc cars," *Optimal Control Applications and Methods*, vol. 36, no. 5, pp. 628–647, 2015.
[13] A. Franco and V. Santos, "Short-term path planning with multiple moving obstacle avoidance based on adaptive mpc," in *2019 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. IEEE, 2019, pp. 1–7.
[14] S. Jeon, K. Lee, and D. Kum, "Overtaking decision and trajectory planning in highway via hierarchical architecture of conditional state machine and chance constrained model predictive control," *Robotics and Autonomous Systems*, p. 104014, 2022.
[15] C. K. Williams and C. E. Rasmussen, "Gaussian processes for machine learning, volume 2 (3)," 2006.
[16] H. Bijl, "Gaussian process regression techniques with applications to wind turbines," *Delft University of Technology, Doctoral degree*, 2016.
[17] J. B. Rawlings, D. Q. Mayne, and M. Diehl, *Model predictive control: theory, computation, and design*. Nob Hill Publishing Madison, WI, 2017, vol. 2.
[18] H. Pacejka, *Tire and vehicle dynamics*. Elsevier, 2005.
[19] M. Elbanhawi, M. Simic, and R. Jazar, "Receding horizon lateral vehicle control for pure pursuit path tracking," *Journal of Vibration and Control*, vol. 24, no. 3, pp. 619–642, 2018.
[20] C. J. Ostafew, A. P. Schoellig, and T. D. Barfoot, "Robust constrained learning-based nmpc enabling reliable mobile robot path tracking," *The International Journal of Robotics Research*, vol. 35, no. 13, pp. 1547–1563, 2016.
[21] H.-A. Langåker, "Cautious mpc-based control with machine learning," Master's thesis, NTNU, 2018.
[22] J. Quinonero-Candela, A. Girard, and C. E. Rasmussen, "Prediction at an uncertain input for gaussian processes and relevance vector machines-application to multiple-step ahead time-series forecasting," 2003.
[23] J. Ziegler, P. Bender, T. Dang, and C. Stiller, "Trajectory planning for bertha—a local, continuous method," in *2014 IEEE intelligent vehicles symposium proceedings*. IEEE, 2014, pp. 450–457.
[24] I. Sánchez, A. D'Jorge, G. V. Raffo, A. H. González, and A. Ferramosca, "Nonlinear model predictive path following controller with obstacle avoidance," *Journal of Intelligent & Robotic Systems*, vol. 102, no. 1, pp. 1–18, 2021.
[25] Z. Elmi and S. Elmi, "Autonomous vehicle path planning using mpc and apf," in *Motion Planning*. IntechOpen, 2022.