Lane departure prediction based on closed-loop vehicle dynamics

Daofei Li, Siyuan Lin and Guanming Liu

Abstract
An automated driving system should have the ability to supervise its own performance and to request human driver to take over when necessary. In the lane keeping scenario, the prediction of vehicle future trajectory is the key to realize safe and trustworthy driving automation. Previous studies on vehicle trajectory prediction mainly fall into two categories, that is, physics-based and manoeuvre-based methods. Using a physics-based methodology, this article proposes a lane departure prediction algorithm based on closed-loop vehicle dynamics model. We use extended Kalman filter to estimate the current vehicle states based on sensing module outputs. Then, a Kalman Predictor with actual lane keeping control law is used to predict steering actions and vehicle states in the future. A lane departure assessment module evaluates the probabilistic distribution of vehicle corner positions and decides whether to initiate a human takeover request. The prediction algorithm is capable to describe the stochastic characteristics of future vehicle pose, which is preliminarily proved in simulated tests. Finally, the on-road tests at speeds of 15–50 km/h further show that the proposed method can accurately predict vehicle future trajectory. It may work as a promising solution to lane departure risk assessment for automated lane keeping functions.

Keywords
Trajectory prediction, Kalman predictor, lane keeping assist, automated driving, risk assessment

Introduction
Driving in the expected lane in a safe and comfortable fashion is one of the core tasks of Advanced Driving Assistant Systems such as LKA (Lane Keeping Assist) and LCA (Lane Centering Assist). When such systems are not capable to handle the dynamic driving tasks, a risk assessment module is needed to provide timely warning to the human driver and to initiate necessary human takeover requests. For highly automated driving systems, for example, level 3 conditional driving automation per SAE J3016 standard, such risk assessment module becomes extremely crucial both for safety and customer experience. Inaccurate risk assessment algorithms, aggressive or conservative, will discourage the use of such automated driving functions. For example, if a risk assessment algorithm cannot precisely judge whether an LKA system can handle lane keeping tasks or not, but initiates takeover requests too frequently, the user experience is significantly compromised. Therefore, there are urgent needs for an accurate algorithm of risk assessment, which as a watchdog algorithm can execute online supervision over the closed-loop performances of automated driving.

A successful risk assessment algorithm should be based on accurate perception or prediction of surrounding road environment, including that of other traffic users and lane boundaries. Then, the ego vehicle motion prediction algorithm is to predict the future positions and poses of vehicle body, with which a risk level is obtained for the final assessment of a human takeover request.

As the core of risk assessment and also the focus topic of this article, the ego vehicle motion prediction algorithm is comprised of three parts, that is, vehicle state estimation via sensor fusion, motion prediction and takeover request assessment. As for state estimation, there have been extensive studies using the Kalman filter (KF) or extended Kalman filter (EKF) to compensate sensor...
errors, or unscented Kalman filter (UKF) for highly nonlinear systems.

As for vehicle motion prediction, previous studies can be mainly divided into the physics-based and the manoeuvre-based approaches. The physics-based motion models, as the name indicates, are based on understanding of vehicle physics, in kinematic or dynamic ways. Popular kinematic models include the Constant Velocity (CV) model, the Constant Acceleration (CA) model, the Constant Turn Rate and Velocity (CTRV) model and the Constant Turn Rate and Acceleration (CTRA) model, while the dynamic models include the 2- or 3-degree-of-freedom (2-DOF or 3-DOF) ‘bicycle’ model. Although with great computing efficiency, existing algorithms with physics-based approach cannot consider the control operation from driver or automation agent, so the long-term prediction accuracy cannot be guaranteed.

The second category of vehicle motion prediction, the manoeuvre-based approach, treats driving as a series of manoeuvres and requires more machine learning from obtained data sets. Several machine learning approaches have been used as classifiers to identify the driving intention of the target vehicle, for example, HMM (Hidden Markov Model), DBN (Dynamic Bayesian Network), and SVM (Support Vector Machine). For end-to-end prediction of vehicle trajectory, neural networks have been often adopted, for example, LSTM (Long Short-Term Memory) and FFNN (Feed-Forward Neural Network), DFNN (Deep Fourier Neural Network), and GMM (Gaussian Mixture Model)-based MDN (Mixture Density Network). Given sufficient amount and scenario coverage of training data, the manoeuvre-based models can identify and predict the driver’s intended operation, and thus ensure the long-term prediction accuracy. However, this excellent performance in the theoretical sense calls for a great deal of thorough validation, particularly to meet SOTIF (Safety Of The Intended Functionality) requirements.

By comparing these two approaches, we find that for ego vehicle motion prediction in automated driving, it is better to extend the physics-based approach, especially when considering the better availability of vehicle states and the convenience of SOTIF validation. Furthermore, in an automated vehicle, its control actions are generated according to tracking control law, so basically it can be easier to predict the future manoeuvres as long as the control law is known. This can be viewed as an improved version of physics-based approach, which takes advantage of the closed-loop vehicle dynamics.

In this article, we focus on lane keeping scenarios of automated driving vehicles as shown in Figure 1. The lane centreline is chosen as the target trajectory, while the left and right lane lines hereinafter can be the physical lane marker lines of road or the imaginary lane lines according to different lane keeping preferences. The

rectangle, which is slightly larger than the actual vehicle, is used to indicate the vehicle planar contour. The task is to get the predicted vehicle pose (in dashed green) based on the current vehicle pose (in solid black), and to quantify the risk of lane departure at future time steps. For the front left corner, the light green area schemes its possible position at the current step, while the uncertainties are due to the fusion errors of observers and sensors. The dark green area schemes its possible position at the predicted future time step, while its marginal distance from the left lane line.

To fulfil this, a lane departure prediction method based on closed-loop vehicle dynamics model is proposed, with the framework shown in Figure 2. Unlike the existing approaches of trajectory prediction, here we do not use only sensing and planning modules, but also tracking control module in designing the algorithm. As a part of the entire automated driving system, the necessary laws or strategies for control prediction can be directly migrated from the control module. If the module does not perform as a white box to the prediction algorithm developer, the control laws can be learned via controller-in-the-loop simulation or on-road tests.

The main parts include EKF, Kalman Predictor with Control (KPC) and Lane Departure Assessment...
Vehicle dynamics modelling

A single-track model with 2 degrees of freedom (2-DOFs), that is, the lateral and yaw motion, is adopted.23 Here, we assume the vehicle runs on the flat road surface with a constant longitudinal speed, and ignore both the vehicle pitch and roll motion. Figure 3 shows the simplified model in the moving coordinate system XOY, and also the inertia coordinate system XcYc. To identify the system parameters, we carry out double-lane-change tests at two speeds of 30 and 50 km/h. The identified model is simulated with the exact steering inputs from road tests. Figure 4 presents the steering angle \( \delta_f \) and vehicle responses of both simulation (‘Sim’) and test measurements (‘Mea’), including lateral velocity \( v_y \), vehicle trajectories (North-East), yaw rate \( \omega_r \), heading angle \( \psi \) and lateral acceleration \( a_y \). The results show that the identified model can accurately describe the vehicle lateral and yaw motion in the considered speed range. In real applications, to account for the variation of vehicle parameters and the effects of unknown disturbances from

\[
\begin{align*}
    \dot{v}_y &= \frac{\left( F_{yf} \cos \delta_f + F_{yr} \right)}{m} - v_x \omega_r \\
    \dot{\omega}_r &= \frac{\left( a F_{yf} \cos \delta_f - b F_{yr} \right)}{I_z}
\end{align*}
\]  

where \( v_x \) denotes the constant longitudinal velocity of vehicle centre of gravity (CG); \( v_y \) denotes the CG lateral velocity; \( \omega_r \) denotes the vehicle yaw rate; \( F_{yf} \) and \( F_{yr} \) denote the lateral forces of front and rear tyres, respectively; \( \alpha_f \) and \( \alpha_r \) denote the slip angles of front and rear tyres, respectively; \( \delta_f \) denotes the front wheel steering angle; \( m \) and \( I_z \) denote the mass and yaw moment of inertia of the vehicle, respectively; \( a \) and \( b \) denote the distances from the CG to the front and rear axles, respectively. All symbols and vehicle parameters are listed in Appendix 1.

Assuming that in the lane keeping process, the vehicle lateral acceleration is less than 0.4g, so the tyre lateral force characteristics are in the linear range, and the influence of load transfer can also be ignored. Then, we have

\[
\begin{align*}
    F_{yf} &= -C_f \alpha_f = -C_f \left( \frac{v_x + \alpha_f}{v_x} - \delta_f \right) \\
    F_{yr} &= -C_r \alpha_r = -C_r \left( v_y - a \right) \frac{v_y}{v_x}
\end{align*}
\]

where \( C_f \) and \( C_r \) are the total cornering stiffnesses of the front and rear tyres, respectively.

In the inertia coordinate system X0Y, the vehicle motion equations can be expressed as

\[
\begin{align*}
    \dot{X}_c &= v_x \cos \psi - v_y \sin \psi \\
    \dot{Y}_c &= v_x \sin \psi + v_y \cos \psi \\
    \dot{\psi} &= \omega_r
\end{align*}
\]

where \( X_c \) and \( Y_c \) denote the abscissa and ordinate of the vehicle’s CG in the inertial coordinate system; \( \psi \) denotes the heading angle of vehicle.

The system states are selected as \([v_x \omega_r X_c Y_c \psi]^T\), and the control input is front wheel steering angle \( \delta_f \).

Then, by combing equations (1)–(3), the vehicle motion in lane keeping process can be described as

\[
\begin{align*}
    \dot{v}_y &= -\frac{C_f + C_r}{mv_x} v_y - \frac{a C_f - b C_r + v_x}{mv_x} \omega_r + \frac{C_f}{m} \delta_f \\
    \dot{\omega}_r &= -\frac{a C_f - b C_r}{L_x v_x} v_y - \frac{a^2 C_f + b^2 C_r}{L_x v_x} \omega_r + \frac{a C_f}{L_x} \delta_f \\
    \dot{X}_c &= v_x \cos \psi - v_y \sin \psi \\
    \dot{Y}_c &= v_x \sin \psi + v_y \cos \psi \\
    \dot{\psi} &= \omega_r
\end{align*}
\]
environment, an online estimator can be designed to guarantee the accuracy of vehicle model.

Vehicle trajectory prediction

Path tracking control

To follow the target trajectory, the vehicle is controlled using both the feedforward and feedback laws.24 The feedforward part is obtained from the curvature information of the desired trajectory, and the feedback control is realized by linear quadratic regulator (LQR).

The tracking error dynamics model can be derived from the single-track vehicle model, that is,

\[ \dot{x}_{\text{error}} = Ax_{\text{error}} + B_1 \delta_f + B_2 \psi_{\text{des}} \] (5)

where \( x_{\text{error}} = [e_1, \dot{e}_1, \ e_2, \dot{e}_2]^T \), \( e_1 \) and \( e_2 \) denote the lateral error and heading error between the vehicle and the target path, respectively; \( \dot{e}_1 \) and \( \dot{e}_2 \) are the derivatives of \( e_1 \) and \( e_2 \) with respect to time, respectively; \( \psi_{\text{des}} \) denotes the expected yaw rate according to the path curvature \( \kappa \) and the longitudinal speed \( v_x \), that is, \( \psi_{\text{des}} = v_x \kappa \). The matrices of \( A, B_1 \) and \( B_2 \) are as follows:

\[
A = \begin{bmatrix}
0 & -m v_x & 0 & 0 \\
0 & -\frac{C_f + C_r}{m v_x} & \frac{C_f + C_r}{m v_x} & -a C_r + b C_r \\
0 & 1 & 0 & 0 \\
0 & \frac{a C_f - b C_r}{I_z v_x} & \frac{a C_f - b C_r}{I_z v_x} & a^2 C_f + b^2 C_r - \frac{a^2 C_r + b^2 C_r}{I_z}
\end{bmatrix}
\]

\[
B_1 = \begin{bmatrix}
\frac{1}{m v_x} \\
\frac{a C_f - b C_r}{I_z v_x} \\
0 \\
\frac{a C_f - b C_r}{I_z v_x}
\end{bmatrix}, \quad B_2 = \begin{bmatrix}
a C_f - b C_r & 0 & 0 & 0 \\
0 & a C_f - b C_r & 0 & 0 \\
a^2 C_f + b^2 C_r & 0 & a^2 C_f + b^2 C_r & 0
\end{bmatrix}
\]

The final front wheel steering control input \( \delta_f \) is composed of feedforward and feedback parts, that is, \( \delta_f = \delta_{f}^{\text{ff}} + \delta_{f}^{\text{fb}} \). By solving the equilibrium of equation (5), \( \dot{x}_{\text{error}}^{eq} = 0 = A x_{\text{error}}^{eq} + B_1 \delta_f^{\text{ff}} + B_2 \psi_{\text{des}} \), and setting \( e_1^{eq} = 0 \), we have the feedforward part as \( \delta_{f}^{\text{ff}} = \delta_{f}^{eq} \).

For LQR feedback control, the cost function is set as

\[ J = \frac{1}{2} \int_{t_0}^{\infty} \left[ x_{\text{error}}^T W_1 x_{\text{error}} + \delta_{f}^{\text{ff}} W_2 \delta_{f}^{\text{ff}} \right] dt \] (6)

where \( W_1 \) denotes the positive semi-definite state weight matrix; \( W_2 \) denotes the positive definite control weight coefficient. By solving the minimization problem of cost \( J \), we have the feedback part of steering control \( \delta_{f}^{\text{fb}} = -W_2^{-1} B_1^T P_x \dot{x}_{\text{error}} \), where \( P_x \) satisfies the algebraic Riccati equation

\[ -P_x A - A^T P_x + P_x B_1 W_2^{-1} B_1^T P_x - W_1 = 0 \] (7)

Then, the total control law of the front wheel steering can be finally obtained as

\[ \delta_f = \delta_{f}^{eq} - K_{fb} x_{\text{error}} \] (8)

where the feedback gain \( K_{fb} = W_2^{-1} B_1^T P_x \). Note that for different longitudinal speeds \( v_x \), the tracking error model varies according to vehicle dynamics equation (4). So, for different \( v_x \), the corresponding gains \( K_{fb} \) are obtained and stored in a look-up table for calculating the feedback steering angle.

EKF for current vehicle states

As shown in Figure 2, the on-board sensor data are first filtered with EKF to get the optimal estimation of the vehicle motion and pose information. And then, a
Kalman predictor is designed to make multiple predictions, so as to predict the future vehicle trajectory.

The classical KF is a recursive method, which adopts the idea of prediction and correction. The main process is as follows:

1. Prediction. The estimate of the current state and error variance are calculated to provide a prior estimate for the state at the next moment.
2. Correction. A posterior estimate is obtained by combining the prior estimate with the new measurement.

Since the model equation (4) is nonlinear, EKF is used to estimate the vehicle states. The main process is divided into the following three steps:

Step 1: to organize the state and measurement equations of the nonlinear system to a standard form

\[
\begin{align*}
\dot{x}(t) &= f(x(t), u(t)) + w(t) \\
y(t) &= h(x(t)) + v(t)
\end{align*}
\]

where \( x(t) = [v_x, \omega_x, x_c, y_c, \phi]^T \) denotes the state; \( u(t) \) denotes the control input, \( u(t) = \delta_r \); \( y(t) \) denotes the measured data from sensors; \( w(t) \) and \( v(t) \) denote the process noise and measurement noise, respectively.

Step 2: to linearize the nonlinear system model equation (9).

Equation (9) is expanded by the Taylor expansion, and the higher-order components are abandoned. Take the partial derivatives of \( f \) and \( h \) with respect to the states to get the Jacobian matrices \( F \) and \( H \), respectively, that is, \( F = \frac{\partial f}{\partial x} \) and \( H = \frac{\partial h}{\partial x} \). The continuous differential equations are then discretized and the state transition matrix is obtained as \( \Phi = I + F_t \), where \( t_s \) is the sampling time of the algorithm.

Step 3: to initialize the state and error covariance matrix and then to start the filtering iterations.

The initial state and covariance matrix are

\[
\begin{align*}
\bar{x}^{0|0} &= \mu_0, \\
P^{0|0} &= P_0
\end{align*}
\]

Using \( k \) to denote the discrete time step of the system, the following iterative calculation is carried out when \( k \geq 1 \):

\[
\begin{align*}
\bar{x}^{k|k-1} &= \bar{x}^{k-1|k-1} + t_s f(\bar{x}^{k-1|k-1}, u^{k-1}) \\
P^{k|k-1} &= \Phi P^{k-1|k-1} \Phi^T + Q \\
K^k &= P^{k|k-1} H^T (H P^{k-1|k-1} H^T + R)^{-1} \\
\hat{x}^{k|k} &= \bar{x}^{k|k-1} + K^k (y - h \bar{x}^{k|k-1}) \\
P^{k|k} &= [I - K^k H] P^{k|k-1}
\end{align*}
\]

where \( Q \) and \( R \) denote the covariance matrices of process noise \( w \) and observation noise \( v \), respectively. \( \omega \) and \( \nu \) are the assumed Gaussian white noises, and satisfy the distributions of \( N(0, Q) \) and \( N(0, R) \), respectively.

**KPC for future states**

We first introduce a basic form of Kalman predictor using unchanging control inputs.\(^{25} \) In order to obtain the possible vehicle trajectory in the future, the real-time estimation results of the state \( \hat{x}^{k|k} \) and the error covariance matrix \( P^{k|k} \) obtained in equation (11) are substituted into the multi-step Kalman predictor as follows

\[
\begin{align*}
\hat{x}^{k+i|k} &= \hat{x}^{k+i-1|k} + t_s f(\hat{x}^{k+i-1|k}, u^{k+i-1}) \\
P^{k+i|k} &= \Phi P^{k+i-1|k} \Phi^T + Q
\end{align*}
\]

where \( \hat{x}^{k+i|k} \) denotes the predicted states at the \( i \)th step to predict, and \( P^{k+i|k} \) is the prediction error covariance matrix at the \( i \)th step.

In each step of the above predictor, the control input in the state transition equation (12) uses \( u^{k+i} \) at the beginning of the prediction moment \( k \). It may not be a problem for a system with little change in control input using this method. For the vehicle motion system in the lane keeping situation, the front wheel steering control \( \delta_r \) may vary greatly, so using an unchanging \( u^{k+i} \) in the prediction process may result in a large error of prediction result.

Fortunately, in automated driving mode, the tracking algorithm to control the front wheel steering angle can be migrated or learnt. Based on the Kalman predictor in equations (12) and (13), here we further add a 'control prediction' procedure to predict the control input in future steps, which finally constructs KPC, as shown in Figure 5. Using the predicted states \( \hat{x}^{k+i|k} \), the predicted control input at the \( i \)th step, that is, \( \hat{u}^{k+i} \), can be obtained in Control Predictor procedure via known LQR control law. Then, in State Prediction procedure, the vehicle states of the \( i \)th step, that is, \( \hat{x}^{k+i|k} \), can be calculated.

For KPC, equation (12) is changed to

![Figure 5. State prediction and control prediction in KPC.](image-url)
\[ \dot{x}^k + i_k = \dot{x}^k + i-1_k + t_f \left( \dot{x}^k + i-1_k, u^k + i-1 \right) \]  

(14)

With the target trajectory information, the tracking error \( x_{error} \) can be expressed as \( x_{error} = E(x) \), that is

\[
\begin{align*}
\epsilon_1 &= E_1(X_c, Y_c) \\
\epsilon_2 &= E_2(X_c, Y_c, \psi) \\
\dot{\epsilon}_1 &= v_\psi + v_\psi \epsilon_2 \\
\dot{\epsilon}_2 &= \omega_\psi - v_\psi \kappa
\end{align*}
\]

(15)

Then, the predicted input \( u^k + i \) can be obtained using the control law in equation (8)

\[
\dot{u}^k + i = \delta^q = K_\beta E \left( \dot{x}^k + i_k \right) \triangleq G \left( \dot{x}^k + i_k \right)
\]

(16)

Therefore, equation (14) is revised as

\[
\dot{x}^k + i_k = \dot{x}^k + i-1_k + t_f \left( \dot{x}^k + i-1_k \right)
\]

(17)

where \( f(x^k + i-1_k) \triangleq f(\dot{x}^k + i-1_k, G(x^k + i-1_k)) \).

Corresponding to section “EKF for current vehicle states,” the state transition matrix \( \Phi \) is changed to its closed-loop form \( \Phi_{cl} = I + t_f \delta_{f_r} / \partial x \). Then equation (13) of error covariance matrix in the prediction process should also be changed to

\[
P^k + i_k = \Phi_{cl} P^k + i-1_k \Phi_{cl}^T + Q
\]

(18)

Before the Control Predictor procedure, as shown in Figure 5, we have not considered the state estimation error, which usually exists in actual process. To simulate such characteristics, the predicted tracking error \( x_{error} \) is also added with a small amount of noise. A simulated noise part \( v_{sim} \) is added to the predicted vehicle state \( \dot{x}^k + i_k \), which is sampled from a stochastic process with the same statistical characteristics as the measurement noise \( \nu(t) \) in equation (9). Then, the EKF algorithm is used to filter the added noise \( v_{sim} \), and we obtain the simulated estimation of \( \dot{x}^k + i_k \), denoted as \( \dot{x}_{sim}^k + i_k \).

During prediction, we assume this estimation process reaches steady state, and the KF gain \( K_{\beta} \) at time \( k \) is used. Finally, \( \dot{x}^k + i_k \) in equation (16) is substituted by its simulated estimation \( \dot{x}_{sim}^k + i_k \), and we have the final predicted control as

\[
\dot{u}^k + i = \delta^q - K_{\beta} E \left( \dot{x}_{sim}^k + i_k \right)
\]

(19)

When considering both the closed-loop system dynamics and the simulated estimation process, the error covariance matrix in equation (18) should be further improved. With the same assumption that the estimation process being in steady state, \( P_{cl} \) approximates the covariance matrix of the error between \( x_{sim}^k + i_k \) and \( x^k + i_k \), so the final form of \( P^k + i_k \) is updated as follows

\[
P^k + i_k = \Phi_{cl} P^k + i-1_k \Phi_{cl}^T + Q
\]

\[+ \frac{\partial E}{\partial x} \frac{\partial E}{\partial x}^T K_{\beta} \left( \frac{\partial f_i}{\partial u} \right)^T \]

(20)

With above KPC procedures, we can obtain the predicted vehicle states \( x^k + i_k \) and the covariance matrix of prediction error \( P^k + i_k \), which include the predicted results of vehicle CG position and body heading angle.

**LDA**

According to the predicted vehicle states from time \( k \) and the lane boundary information, we can evaluate the risk of lane departure in the future. The evaluation of risk is mainly based on the marginal distances from the corners of the vehicle planar contour, that is, all four rectangle corners, to the adjacent lane lines. The geometry relationship shown in Figure 1 is used to obtain the distributions of corner positions, and more details of the derivation process can be found in Appendix 2.

Then, we have the Normal distribution of vehicle corner marginal distance before lane departure, as shown in Figure 6. We can design the following two rules to quantify the risk of lane departure:

1. A distance threshold \( \Delta \), as shown in Figure 6, is set to indicate when the marginal distance from corner point \( j \) to its adjacent lane line, \( d_j^k + i \), is not less than this threshold, that is, \( d_j^k + i \geq \Delta \), then the risk of lane departure is tolerable. Here, we choose \( \Delta = 0 \).

2. A probability threshold \( \Pi \) is set to make sure that the predicted marginal distance \( d_j^k + i \) falls in the range of \([d_j^k + i - \Sigma, d_j^k + i + \Sigma]\) with a probability of \( \Pi \). Here, we choose \( \Pi = 99.73 \% \), and the actual marginal distance satisfies

\[
\Pr \left( d_j^k + i - \Sigma \leq d_j^k + i \leq d_j^k + i + \Sigma \right) = \Pi
\]

(21)
With these two rules, if \( d_{ij}^{k} + i/C_0 \geq \Delta \), the vehicle is considered still within the allowable lateral deviation from the target trajectory at the \( i \)th future prediction step from time \( k \), that is, the lateral departure flag is 0. Otherwise, if \( d_{ij}^{k} + i/C_0 < \Delta \), the lateral deviation is too large and the lateral departure flag is assigned value 1, meaning that the driver should be requested to take over the driving tasks.

**Validation via simulation**

**Simulation scenario**

The joint simulation platform of CarSim and MATLAB/Simulink is adopted for preliminary validation of the proposed algorithm. As shown in Figure 7, the driving scenario is designed as follows:

1. Road: the equation of the target trajectory in the inertia coordinate system is \( Y = 2 \), and the lane width is 4 m.
2. Vehicle speed: the vehicle longitudinal speed is maintained near the target speed, that is, \( v_{des} = 30 \) km/h.
3. Sensor noises: to simulate the real sensor characteristics, the zero-mean random noises with Normal distribution are injected into the vehicle state information, with the noise variances given in Table 1.
4. Lane keeping incident: initially, the vehicle drives at the centre of the lane and its heading is parallel to the lane line. To simulate the unknown disturbances acting on the vehicle, it deliberately deviates towards to the left in Stage 1, and then in Stage 2, the lane keeping controller is activated to prevent the vehicle from leaving the lane.

In the considered scenario, the vehicle is likely to deviate to the left, so the marginal distance \( d \) of the vehicle front left corner, as shown in Figure 1, is used to evaluate the lane departure risk.

For the benchmark algorithm, data-driven models are not considered due to their difficulties of safety validation and also fair comparison. To check the potential advantage of using a closed-loop model in prediction over open-loop models, the CTRV (Constant Turning Rate and Velocity) algorithm is selected as the benchmark algorithm. In the simulated scenarios, the longitudinal speed is assumed constant; the CTRV algorithm has satisfactory tracking and prediction performances among open-loop physics-based models.

**Simulation results**

The simulation process is repeated 500 times of successful lane keeping activations, with the sensor noises sampled according to Table 1. Then, for both KPC and CTRV algorithms, we have trajectory prediction results and lane departure risk assessments for statistical comparisons.

Figure 8 presents an example simulation case, where the CTRV vehicle (in red) is wrongly predicted to leave the lane after 1 s, but the KPC algorithm can precisely predict the vehicle trajectory.

The overall simulation results are shown in Figure 9, where the blue circle represents the predicted position of the four vehicle corners by the KPC algorithm, the red star represents that by the CTRV algorithm, and the black solid line represents the actual trajectory of CG in the next 2 s.

Figure 10 shows the position prediction results of the front left corner starting from the beginning of Stage 2 for every 0.1 s. For each prediction step on KPC or

| Signal | Units | Variance |
|--------|-------|----------|
| \( v_y \) | m/s | \( 10^{-6} \) |
| \( \omega_r \) | rad/s | \( 10^{-6} \) |
| \( X_c \) | m | 1 |
| \( Y_c \) | m | 1 |
| \( \psi \) | rad | \( 10^{-2} \) |

Table 1. Sensor noise variances in simulation.
CTRV, the points represent the results of 500 simulation cases. The orange line represents the left line of the target lane. It can be seen that the prediction results of KPC are much closer to the actual position of the front left corner than those of CTRV. With the growth of the prediction time, the prediction error of benchmark algorithm becomes larger, with a maximum lateral position error about 1.5 m. For KPC algorithm, it can also be seen that the distribution of the predicted positions is getting more gathered with the growth of the prediction time, which means the variance of the predicted ordinates of the front left corner $\sigma_{Yf}^2$ decreases. This may be due to the existence of lane keeping algorithm, which makes the whole closed-loop vehicle system become convergent.

Figure 10 also shows that, when calculating the covariance matrix $P_{k+1|k}$ in the process of prediction using equation (20), the variance of front left corner position $\sigma_{Yf}^2$ becomes smaller until it converges to a certain value since $\Phi_d I$, where $I$ is the unit matrix. Figure 11 details the calculated variance $\hat{\sigma}_{Yf}^2$ by KPC, the sample variance of all the prediction results $\sigma_{Yf}^2$, and the mean square error (MSE) of the prediction results. They are close to each other during the whole prediction process, which indicates that the calculated $\sigma_{Yf}^2$ of KPC algorithm can describe the distribution of the errors between the predicted and the real lateral positions in a short period of time in the future.

The histograms of the predicted position distribution of the front left corner at 1 and 2 s in the future are shown in Figures 12 and 13, respectively. It can be seen that the predicted position at 1 and 2 s follows the Normal distribution as expected.

The LDA results of both algorithms in all simulation cases are shown in Figure 14. It can be seen that in the considered cases, KPC mainly believes that the vehicle is still in the lane, while CTRV is more conservative and predicts that the vehicle will drive out of the lane. The departure assessment accuracy is compared in Figure 15, which shows that with the knowledge of closed-loop vehicle behaviour, KPC works more accurately in the whole prediction process. It seems that there are some failed cases of KPC in early period of prediction; however, in these cases, the vehicle is just too close to the left lane line and LDA gives wrong judgements according to a conservative threshold in equation (21).
Validation via on-road test

Experiment setups

The test vehicle, shown in Figure 16, supports fully automated control of longitudinal and lateral motion. The on-board sensors provide main information of chassis and powertrain systems, for example, steering wheel angle and the wheel speeds. A high precision integrated navigation system (INS), that is, CHCNAV CGI610, is used to measure the vehicle motion and position states. The measured signals used for algorithms are listed in Table 2.

The proposed KPC algorithm and the benchmark CTRV algorithm run in the high-performance computer at 100 Hz frequency. Based on tracking errors from sensors, the LQR-based tracking controller runs in NI CompactRIO and provides a front wheel steering angle, which is further realized in the lower-level steering-by-wire controller.

The experiment scenario and procedure are similar to that of simulation, as shown in Figure 7. During the whole process, the vehicle’s longitudinal speed keeps around target speed using a proportional–integral–derivative (PID) controller. To induce lane departure incidents, in Stage 1, the vehicle is deliberately steered by the test driver away from the centre of the target lane.

The experiments are divided into low-speed and high-speed groups, and are carried out in Sites A and B, as shown in Figure 17. In the low-speed group, the vehicle speed is 15, 20 and 25 km/h, and the centre line of the target lane is a straight line of 1 degree North by East. The high-speed group includes vehicle speeds of 4 and 50 km/h, and the centre line of the target lane is a straight line of 3 degrees North by West. The experiment details can be found in the video abstract attached as supplemental material.

Results and discussion

The experimental results of 2 and 4 s predictions are shown in Figures 18–23. In the left subfigures, the black dot indicates the vehicle CG position when LQR controller is switched on; the solid black lines denote the vehicle trajectory after the controller is activated; the solid blue lines denote the vehicle trajectory predicted by KPC algorithm; the solid red lines denote the vehicle trajectory predicted by CTRV algorithm; the solid black rectangles denote the actual vehicle poses in the prediction period; the dashed blue rectangles denote the vehicle poses predicted by KPC algorithm; the dashed red rectangles denote the vehicle poses predicted by CTRV algorithm. The right subfigures present the lane departure flags, in which the value 0 indicates that the vehicle is still in the lane; the value 1 indicates that the vehicle is out of the lane.

For all cases with different speeds, KPC outperforms CTRV in both position and heading predictions. For the departure flag in all cases, CTRV predicts that the vehicle will deviate from the lane border within 1.0–2.5 s, while KPC concludes correctly that the vehicle will be kept in lane. Using the additional knowledge of

| Signal name      | Description                                      | Units   |
|------------------|--------------------------------------------------|---------|
| Velocity north   | Velocity of the measured point in the north direction | m/s     |
| Velocity east    | Velocity of the measured point in the east direction | m/s     |
| Yaw rate         | Vehicle yaw rate                                 | rad/s   |
| Longitude        | Longitude of the measured point on vehicle       | deg     |
| Latitude         | Latitude of the measured point on vehicle        | deg     |
| Heading angle    | Vehicle body heading                             | rad     |
| Steering wheel angle | Steering wheel angle                              | rad     |
| Wheel speeds     | Wheel speeds in equivalent form of forward speed | m/s     |
Figure 16. The test vehicle.

Figure 17. Test sites for (a) low- and (b) high-speed groups.

Figure 18. Trajectory and departure prediction at 15 km/h.

Figure 19. Trajectory and departure prediction at 20 km/h.

Figure 20. Trajectory and departure prediction at 25 km/h.

Figure 21. Trajectory and departure prediction at 40 km/h.
LQR-based closed-loop controller, with KPC the human takeover requests are not necessary to initiate too frequently, and user experience of such automated driving function can be improved.

Comparing these figures, we find that as the vehicle longitudinal speed grows from 15 to 50 km/h, it is obvious that KPC performs worse. It is because the lateral acceleration grows accordingly during lane keeping, and the linear 2-DOFs model deviates from the actual vehicle dynamics. The heading angle prediction results are shown in Figure 23. The largest prediction error of heading at 15 km/h is about 0.03 rad (absolute value), while that at 50 km/h is about 0.07 rad (absolute value).

It can also be found that the vehicle position predicted by the two algorithms has a large longitudinal deviation from the actual vehicle position. For example, as shown in Figure 22, the longitudinal position errors in both algorithms are about 1.8 m. This may be due to the fact that the actual vehicle longitudinal speed varies in the process of steering, while the prediction algorithms assume a constant longitudinal speed. However, for vehicle lane departure prediction, the current assumption of constant speed still works. Considering that the prediction performance may get deteriorated when the vehicle longitudinal speed varies greatly, a separate speed prediction can compensate for this disadvantage. For better prediction, a similar KPC methodology can also be adopted to include the longitudinal motion prediction.

**Conclusion**

A lane departure prediction method based on closed-loop vehicle dynamics is proposed, and it is applied in human takeover decision for automated lane keeping scenario. It is an improved model of physics-based prediction, and can provide both high computing efficiency and long-term prediction accuracy. Simulation results show the algorithm is effective in constant speed lane keeping scenarios, and it can describe the probability distribution characteristics of vehicle pose prediction. Experiment results show that the vehicle trajectory prediction algorithm is effective at speeds of 15–50 km/h. It may work as a promising solution to lane departure risk assessment for automated lane keeping functions.

However, there are also some limitations of this article, which can be further extended in future work:

1. The effects of unknown disturbances and vehicle control actuator errors are not considered in the prediction algorithm, which are common in practical vehicle control. For improvements, a specific observer helps to predict sidewinds, and more accurate actuator modelling can improve the prediction, too. Online estimation of system parameters can also contribute to better model accuracy.

2. For demonstration, here the control law in the predictive algorithm is directly migrated from the actual controller, assuming the developer can access the source codes of lane keeping controller. In some cases, the detailed law of the tracking controller may be unknown to the developer, so it is necessary to design an algorithm to learn the law from controller inputs–outputs mapping. In view of this, some data-driven algorithms can be used to learn the actual control law in the future work.

3. This study focuses on the lateral dynamics of the vehicle, while the longitudinal vehicle dynamics is not considered. In future work, more complex scenarios can be considered, where the longitudinal and lateral models of vehicles should be integrated. In addition, the algorithm reliability for human takeover in dangerous scenarios can be further validated using simulated driving or more comprehensive real road tests.

**Acknowledgements**

The authors appreciated Anfei Zha for his help in carrying out the real vehicle experiments, and Linhui Chen for his help in preparing the video abstract.
Declaration of conflicting interests
The author(s) declared no potential conflicts of interest with respect to the research, authorship and/or publication of this article.

Funding
The author(s) disclosed receipt of the following financial support for the research, authorship and/or publication of this article: This work was supported by Department of Science and Technology of Zhejiang (nos 2022C01241 and 2018C01058) and Ningbo Science and Technology Bureau (nos 2018B10063 and 2018B10064).

ORCID iD
Daofei Li https://orcid.org/0000-0002-6909-0169

Supplemental material
A supplementary video abstract is provided, which includes a brief introduction of the proposed approach and an example of validation experiments, https://youtu.be/rRIUUpBDxV8.

References
1. Ammoun S and Nashashibi F. Real time trajectory prediction for collision risk assessment between vehicles. In: Proceedings of the 2009 IEEE 5th international conference on intelligent computer communication and processing, Cluj-Napoca, 27–29 August 2009, pp.417–422. New York: IEEE.
2. Polychronopoulos A, Tsogas M, Amditis AJ, et al. Sensor fusion for predicting vehicles’ path for collision avoidance systems. IEEE T Intell Transp 2007; 8(3): 549–562.
3. Batz T, Watson K and Beyerer J. Recognition of dangerous situations within a cooperative group of vehicles. In: Proceeding of the 2009 IEEE intelligent vehicles symposium, Xi’an, China, 3–5 June 2009, pp.907–912. New York: IEEE.
4. Schubert R, Richter E and Wanenik G. Comparison and evaluation of advanced motion models for vehicle tracking. In: Proceedings of the 2008 11th international conference on information fusion, Cologne, 30 June–3 July 2008, pp.1–6. New York: IEEE.
5. Xie G, Gao H, Qian L, et al. Vehicle trajectory prediction by integrating physics-based and maneuver-based approaches using interactive multiple models. IEEE T Ind Electron 2018; 65(7): 5999–6008.
6. Xiao W, Zhang L and Meng D. Vehicle trajectory prediction based on motion model and maneuver model fusion with interactive multiple models. In: Proceedings of the WCX 2020 SAE world congress experience, Detroit, MI, 21–24 April 2020. Warrendale, PA: SAE International.
7. Lefèvre S, Vasquez D and Laugier C. A survey on motion prediction and risk assessment for intelligent vehicles. ROBOMECH J 2014; 1(1): 1–14.
8. Tan H-S and Huang J. DGPS-based vehicle-to-vehicle cooperative collision warning: engineering feasibility viewpoints. IEEE T Intell Transp 2006; 7(4): 415–428.
9. Brannstrom M, Coelingh E and Sjoberg J. Model-based threat assessment for avoiding arbitrary vehicle collisions. IEEE T Intell Transp 2010; 11(3): 658–669.
10. Eidehall A and Petersson L. Statistical threat assessment for general road scenes using Monte Carlo sampling. IEEE T Intell Transp 2008; 9(1): 137–147.
11. Wang W, Zhao D, Han W, et al. A learning-based approach for lane departure warning systems with a personalized driver model. IEEE T Veh Technol 2018; 67(10): 9145–9157.
12. Yuan W, Li Z and Wang C. Lane-change prediction method for adaptive cruise control system with hidden Markov model. Adv Mech Eng 10(9): 1–9.
13. Gindele T, Brechtel S and Dillmann R. A probabilistic model for estimating driver behaviors and vehicle trajectories in traffic environments. In: Proceedings of the 13th international IEEE conference on intelligent transportation systems, Funchal, 19–22 September 2010, pp.1625–1631. New York: IEEE.
14. Kumar P, Perrollaz M, Lefèvre S, et al. Learning-based approach for online lane change intention prediction. In: Proceedings of the 2013 IEEE intelligent vehicles symposium (IV), Gold Coast, QLD, Australia, 23–26 June 2013, pp.797–802. New York: IEEE.
15. Izquierdo R, Parra I, Muñoz-Bulnes J, et al. Vehicle trajectory and lane change prediction using ANN and SVM classifiers. In: Proceedings of the 2017 IEEE 20th international conference on intelligent transportation systems, Yokohama, Japan, 16–19 October 2017, pp.1–6. New York: IEEE.
16. Kim B, Kang CM, Kim J, et al. Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network. In: Proceedings of the 2017 IEEE 20th international conference on intelligent transportation systems, Yokohama, Japan, 16–19 October 2017, pp.399–404. New York: IEEE.
17. Deo N and Trivedi MM. Convolutional social pooling for vehicle trajectory prediction. In: Proceedings of the 2018 IEEE/CVF conference on computer vision and pattern recognition workshops, Salt Lake City, UT, 18–22 June 2018, pp.1468–1476. New York: IEEE.
18. Djuric N, Radosavliev V, Cui H, et al. Uncertainty-aware short-term motion prediction of traffic actors for autonomous driving. In: Proceedings of the 2020 IEEE winter conference on applications of computer vision, Snowmass, CO, 2–5 March 2020, pp.2084–2093. New York: IEEE.
19. Deo N and Trivedi MM. Multi-modal trajectory prediction of surrounding vehicles with maneuver based LSTMs. In: Proceedings of the 2018 IEEE intelligent vehicles symposium (IV), Changshu, China, 26–30 June 2018, pp.1179–1184. New York: IEEE.
20. Tan D, Chen W and Wang H. On the use of Monte-Carlo simulation and deep Fourier neural network in lane departure warning. IEEE Intel Transp Sy Mag 2017; 9(4): 76–90.
21. Boulton FA, Grigore EC and Wolff EM. Motion prediction using trajectory sets and self-driving domain knowledge, 8 June 2020, https://arxiv.org/abs/2006.04767
Appendix 1

Notations

| Variable | Description |
|----------|-------------|
| $d$ | Distances between the centre of mass and the front/rear axles, 1.13 and 1.55 m |
| $B$ | Half width of vehicle, 0.93 m |
| $C_{t,r}$ | Lateral stiffness of the front and rear axle tyres, 1.65 and 2.65 N/rad |
| $d_{ij}$ | Actual value of the distance between corner point $j$ and its adjacent lane line at time $k+i$ |
| $e_{1i}, e_{2i}$ | Lateral error and heading error between vehicle and target trajectory |
| $F_{f,r}$ | Jacobian matrices of $f$ and $h$ |
| $i$ | Prediction step, $i$th step |
| $l_i$ | Vehicle yaw moment of inertia, 3200 kg m$^2$ |
| $k$ | Discrete time, $i$th step |
| $K_f$ | Feedback gain |
| $l_f$ | Distance between vehicle’s centre of mass and the front end of vehicle contour, 2.11 m |
| $m$ | Vehicle mass, 2030 kg |
| $Q,R$ | Covariance matrices of process noise and observation noise |
| $t_i$ | Sampling time of the algorithm |
| $u(t)$ | Control input |
| $v_c$ | Longitudinal velocity of vehicle CG |
| $v_f$ | Lateral velocity of vehicle CG |
| $v(t), w(t)$ | Measurement and process noises |
| $W_1, W_2$ | State weighting matrix and control weighting |
| $x(t), y(t)$ | System states, Measured outputs |
| $X_c, Y_c$ | East and North coordinates of CG in the inertia coordinate system |
| $X_{c+1}, Y_{c+1}$ | Actual East and North coordinates of vehicle's centre of mass at time $k+i$ |
| $X_{left}, Y_{left}$ | Coordinates of front left corner |
| $X_{right}, Y_{right}$ | Coordinates of front right corner |
| $\alpha_{fr}$, $\alpha_{rr}$ | Slip angles of front and rear tyres |
| $\delta_f$ | Vehicle front wheel steering angle |
| $\kappa$ | Curvature of target path |
| $\Phi$ | State transition matrix |
| $\Phi_d$ | Closed-loop state transition matrix |

(continued)

Appendix 2

Derivation for the distribution of marginal distance $d$

Based on the probabilistic distribution of CG position and vehicle heading, the task is to determine the probabilistic distribution of the marginal distance of all four corners to their corresponding lane lines. Assuming that the vehicle shape is a rectangle when viewed from top, as shown in Figure 1, we use $l_f$ to denote the distance from CG to the front end of the vehicle contour, and $B$ to denote the vehicle half width. Taking the front left corner as an example, its coordinates are

$$
\begin{cases}
X_{fl} = X_c + l_f \cos(\psi + \varphi) \\
Y_{fl} = Y_c + l_f \sin(\psi + \varphi)
\end{cases}
$$

where $l_f = (l_f^2 + B^2)^{1/2}$, and $\varphi = \sin(B/l_f)$. The predicted vehicle heading angle includes two parts, that is, the actual value $\psi$ and the predicted error $\Delta \psi$, $\hat{\psi} = \psi + \Delta \psi$. According to the Taylor expansion, the nonlinear part containing trigonometric functions can be expanded as

$$
\cos(\hat{\psi} \pm \varphi) = \cos(\psi \pm \varphi) - \sin(\psi \pm \varphi) \Delta \psi
\sin(\hat{\psi} \pm \varphi) = \sin(\psi \pm \varphi) + \cos(\psi \pm \varphi) \Delta \psi
$$

According to the prediction algorithm, the prediction result of step $i$ is shown in Table 3.

For simplicity, the covariances between vehicle states are ignored, which is appropriate in this short-term prediction problem. Then, the vehicle coordinates $X_{c+1}^{k+i}$ and $Y_{c+1}^{k+i}$ and the heading angle $\hat{\psi}^{k+i}$ conform to independent Normal distributions as follows

$$
\begin{cases}
X_{c+1}^{k+i} \sim N\left(X_c^{k+i}, \sigma_{X_c}^2\right) \\
Y_{c+1}^{k+i} \sim N\left(Y_c^{k+i}, \sigma_{Y_c}^2\right) \\
\hat{\psi}^{k+i} \sim N\left(\psi^{k+i}, \sigma_{\psi}^2\right)
\end{cases}
$$

where $X_c^{k+i}$ and $Y_c^{k+i}$ denote the actual value of the abscissa and ordinate of the vehicle’s centre of mass at time $k+i$, and $\hat{\psi}^{k+i}$ denotes the actual value of the heading angle of vehicle at time $k+i$. 

---

(continued)
For the prediction process of the $i$th step, the coordinates of the predicted position of the front left corner, $(\hat{X}_{k}^{i}+j, \hat{Y}_{k}^{i}+j)$, also approximately follow the Normal distribution, according to equations (22)–(24)

\[
\begin{align*}
\hat{X}_{k}^{i}+j & \sim N \left( X_{0}^{i}+j + u_{X}^{i} \cos \psi_{k}^{i}+j + \sigma_{X}^{2} \right) \\
\hat{Y}_{k}^{i}+j & \sim N \left( Y_{0}^{i}+j + u_{Y}^{i} \sin \psi_{k}^{i}+j + \sigma_{Y}^{2} \right)
\end{align*}
\]

(25)

where $\psi_{k}^{i}+j = \psi^{i}+j + \varphi$.

Assuming that the actual marginal distance from corner point $j$ to its adjacent lane line can be expressed as a function $d_{j} = D_{j}(x, y)$, the predicted value of $d_{j}$ is $d_{j}^{k} + \overline{i} = D_{j}(\hat{X}_{k}^{i}+j, \hat{Y}_{k}^{i}+j)$, which can be further expanded as follows using the Taylor formula

\[
d_{j}^{k} + \overline{i} = D_{j}(x^{k}, y^{k}) + \frac{\partial D_{j}}{\partial x} x^{k} + \frac{\partial D_{j}}{\partial y} y^{k}
\]

(26)

Then, the marginal distance $d_{j}$ can be described using a Normal distribution

\[
d_{j}^{k} + \overline{i} \sim N \left( d_{j}^{k} + \overline{i}, \left( \frac{\partial D_{j}}{\partial x} \right)^{2} \sigma_{X}^{2} + \left( \frac{\partial D_{j}}{\partial y} \right)^{2} \sigma_{Y}^{2} \right)
\]

(27)

### Table 3. The predicted states and error covariances.

| Predicted vehicle states $\bar{x}^{k+i}$ | Prediction error covariance matrix $P^{k+i}$ |
|-----------------------------------------|------------------------------------------|
| $x_{k}^{i}+j$                             | $\sigma_{x}^{2}$                                |
| $y_{k}^{i}+j$                             | $\text{cov}(y_{k}, \omega_{k})$            |
| $\theta_{k}^{i}+j$                        | $\sigma_{\theta}^{2}$                                |
| $\Delta x_{k}^{i}+j$                      | $\text{cov}(\Delta x_{k}, \Delta y_{k})$          |
| $\Delta y_{k}^{i}+j$                      | $\text{cov}(\Delta y_{k}, \Delta y_{k})$          |
| $\Delta \theta_{k}^{i}+j$                 | $\text{cov}(\Delta \theta_{k}, \Delta \theta_{k})$          |

---

**Table 3.** The predicted states and error covariances.