A Reinforcement Learning-Based Adaptive Path Tracking Approach for Autonomous Driving

Yunxiao Shan, Boli Zheng, Longsheng Chen, Long Chen, *Senior Member, IEEE*, and De Chen

***Abstract*—The path tracking system is a crucial component in fully autonomous vehicles. While many methods provide state-of-the-art tracking performance, they tend to emphasize tracking accuracy for safety, at the price of ride quality. Although methods based on a dynamic model and optimization theory guarantee improved tracking performance, a high-fidelity model is unavailable in most autonomous systems, and the complex optimization process may increase the computational burden. Therefore, for practical, real-life systems, it is urgent to develop a tracking method with a simple, effective control framework for real-time implementation. Moreover, the tracking approach should adaptively balance between tracking accuracy and the passenger experience. With that in mind, a simple tracking scheme with adaptive lateral and longitudinal control approaches is proposed. For lateral control, a pure pursuit (PP) geometric controller is used as a basic tracking framework to design the steering method. A detailed analysis shows that PP inefficiently reduces the lateral error. A PID controller is integrated with PP by a customized reinforcement learning model to better deal with tracking error by trading off between PP and PID. Moreover, a rough-to-fine velocity adaptation method is proposed to adjust the speed according to the geometry of the predefined path and the real-time tracking feedback. The proposed controlling approach is tested in different path tracking scenarios. The results show that the approach can adaptively change the weights of PP and PID to maintain a balance between tracking error (within ± 0.5 m) and jerk (within $\pm 0.5\ \mathrm{m/s^3}$), and can adapt to high-speed scenarios (up to 80 km/h). Finally, a field test is carried out to validate the practicability.**

***Index Terms*—Autonomous driving, adaptive path tracking, reinforcement learning, pure pursuit, PID.**

I. Introduction

AUTONOMOUS vehicles are becoming available for public transportation through the efforts of universities, institutes, and commercial companies. People believe that autonomous vehicles can provide a more comfortable and safer driving experience than human drivers. For excellent ride quality, besides the sensing [1], perception [2], decision, and planning modules [3], [4], smooth and accurate tracking control is crucial, as it directly impacts the safety and passenger experience. In this paper, we pay attention to the path tracking control of autonomous vehicles, calculating the steering angle to steer the vehicle and the speed necessary to follow a predefined path, which can be recorded offline or online by planners [5].

It is a challenge for a path tracking system to synchronously provide a safe and comfortable driving experience for the real-time application of an autonomous vehicle. First, the tracking system should deploy an efficient lateral controller to reduce the tracking error and limit the computation load to enable real-time implementation. Second, the effects of speed should be emphasized when tracking a path. Steering performance varies greatly by speed, and it is difficult to ensure lateral accuracy with improper velocity. Third, while many tracking methods emphasize the control of tracking error to guarantee safety, the passenger experience is equally important under most driving conditions. Therefore, a qualified path tracking approach should accurately (safely) and smoothly (comfortably) track the predefined path. However, it is difficult to trade off between them.

Many researchers have studied the tracking requirements of autonomous vehicles. Model-free PID controllers [6], [7] prevail due to their simple formulation. Nevertheless, PID is so sensitive to direct feedback error that it is difficult to guarantee system stability. Pure pursuit (PP) based on geometric models is widely used in real-life autonomous systems. Recently, methods based on optimization theory and dynamic models, such as LQR [8], MPC [9], and NMPC [10], have been widely used for the path tracking control of autonomous vehicles. These methods require the repeated solution of an optimization problem at each control step, which may increase the computational burden and challenge their real-time application. To meet real-time requirements, Rankin *et al.* [11] proposed to integrate simple controllers in a complementary fashion and build a combination model of PP and PID. Experimental results demonstrated improved tracking performance. Chen *et al.* [12] further optimized the combination and concluded that PID can weaken the impacts on PP due to an improper look-ahead distance.

Although several optimization methods consider the speed and optimize the object function to balance the tracking accuracy and smoothness, their efficiency is unsatisfactory, and accurate dynamic models are difficult to obtain. Combination methods demonstrate potential for real-time tracking problems. A combination method can be expressed as a feedback-feedforward

Manuscript received July 4, 2019; revised January 3, 2020, May 15, 2020, June 23, 2020, and July 31, 2020; accepted August 2, 2020. Date of publication August 6, 2020; date of current version October 22, 2020. This work was supported in part by the Research and Development Plan in Key Areas of Guangdong Province under Grants 2019B090919003 and 2018B010108004 and in part by the Fundamental Research Funds for the Central Universities under Grant 19lgpy229. The review of this article was coordinated by Dr. TVT Administrator. *(Corresponding author: Long Chen.)*

The authors are with the School of Data and Computer Science, Sun Yat-sen University, Guangzhou 510006, China (e-mail: shanyx@mail.sysu.edu.cn; zhengbli3@mail2.sysu.edu.cn; chenlsh25@mail2.sysu.edu.cn; chenl46@mail.sysu.edu.cn; de_chen@liyuanheng.com).

Digital Object Identifier 10.1109/TVT.2020.3014628

controller [13], since PP can be seen as a feedforward controller incorporating curvature and look-ahead constraints, and PID will respond directly to lateral tracking error. However, it is challenging to weight PP and PID according to their respective advantages to adapt to various tracking scenarios. A tracking method must behave differently based on the tracking scenario. For example, the weight of the PID should be increased in scenarios requiring error control, and the effects of PP should be increased when the tracking error is acceptable. In addition, current combination models ignore the effects of speed on tracking performance. Excellent tracking performance cannot be achieved if the speed cannot adapt to the scenario.

Given the above challenges, aiming to provide an accurate and comfortable driving experience, an adaptive path tracking scheme is proposed by integrating reinforcement learning and a combination model. The proposed scheme has two parts: an adaptive steering controller (lateral controller) and a speed adaptation approach. In the steering controller, PP is used to smoothly steer the vehicle based on geometric constraints, and PID is integrated with PP to build a steering model, called PP_PID, to better deal with the tracking error. A customized reinforcement learning (RL) model is implanted in PP_PID to obtain RL_PP_PID, which adaptively adjusts the weights of PP and PID to balance their effects according to the tracking error, lateral acceleration, and change range of the steering angle. Besides the steering controller, the reasonable selection of tracking speed is also crucial for tracking performance at high speeds. In our tracking scheme, a rough-to-fine speed adaptation method with fuzzy logic is proposed. The curvature of predefined paths is used in the rough adaptation, and tracking error and lateral acceleration are deployed for online refinement. Experimental results validate the performance of the proposed tracking scheme, whose contributions are as follows:

- An adaptive tracking scheme is proposed that combines two simple tracking methods to enable real-time, accurate, and comfortable driving.
- The combination is analyzed theoretically, and an adaptive RL-based weight-adjustment model is customized to adjust the weight in PP_PID to trade off between the effects of PP and PID. A reward function uses the lateral tracking error and lateral acceleration to balance accuracy and smoothness.
- A speed adaptation method is proposed to roughly adjust the speed offline according to the curvature of the predefined path, and modify the speed online based on the tracking error and lateral acceleration.
- Experiments confirm the performance of the proposed tracking scheme in different driving scenarios, and the adaptive mechanism, high speed performance, impact of noises and the performance of the speed adaptation method are investigated in detail.

The remainder of this paper is organized as follows. In Section II, we investigate related work on path tracking methods. Section III introduces the PP steering controller. In Section IV, we show the proposed adaptive tracking approach in detail, and we conduct simulation and field tests to demonstrate the state-of-the-art performance of the proposed adaptive control scheme in Section V. Finally, We provide our conclusions in Section VI.

II. Related Work

Path tracking controllers can be roughly divided into three groups, consisting of geometric model controllers, model-free controllers, and model-based controllers.

A. Geometric Controllers

The geometric model of a vehicle is established based on the Ackerman steering configuration, by which lines perpendicular to each wheel should intersect at the center point of the vehicle's cornering arc, where the radius of a turn is R [14]. The most popular geometric controller used in autonomous vehicles is PP, which was used by the MIT autonomous vehicle team [15], which won third place in the 2007 DARPA challenge. We [16] proposed an improved PP that replaced the circle fitting method with a clothoid curve to improve fitting accuracy. Moreover, fuzzy logic was deployed to adaptively adjust the look-ahead distance, which greatly impacts the performance of PP. The Stanley method also plays an important role in geometric controllers [17]. Zhu *et al.* [18] proposed an adaptive Stanley method by learning the parameter k in a Stanley controller with neural dynamic programming. While geometric controllers are reportedly simple and can adjust parameters to realize different types of performance, they struggle to achieve a timely response to feedback error because of overemphasis on the satisfaction of geometric constraints.

B. Model-Free Controllers

The classic model-free controller is PID, which can output steering angles based on feedback. Unlike geometric controllers, PID can directly react to feedback. Park *et al.* [19] applied a PID controller to a steering-actuator control to generate the correct steering angle from a steering controller. A compensator was employed to overcome the dead band and improve tracking performance. Amer *et al.* [6] employed PID controllers in both of their controller loops for steering control. Although satisfactory PID performance can be achieved by finely adjusting parameters, the poor link between PID and system models complicates parameter adjustment and causes preset parameters to adapt with difficulty to driving conditions.

C. Model-Based Controllers

Model-based controllers can build a stronger relationship than PID with the model of a vehicle. Ollero *et al.* [20] used a generalized predictive controller to control a CMU autonomous vehicle, and conducted additional application research [21], [22]. The computational burden of optimization limits the wide application of MPC. To reduce the computation cost, Beal *et al.* [23] improved the optimization method and achieved computations within 2 ms by using a custom C-Code [24]. Merabti *et al.* [10] deployed Beal's optimization method to solve the nonlinear problem of a vehicle. Besides MPC, LQR methods are also built on the kinematic model of vehicles and optimal control

theory. The controller gains are determined by a linear quadratic optimization approach. Snider [25] outlined the implementation of an LQR controller in path tracking control with dynamic feedback. However, due to the absence of error predictions, the controller performed poorly. Sharp *et al.* [26] solved the problem of LQR in [25] by sampling several preview values ahead of the predefined path. While the tracking performance of model-based controllers is claimed to be better than that of PID and geometric controllers, they require powerful computing capacity, and the accuracy of a model that is not guaranteed by most systems has a great influence on the performance of controllers. Shladover *et al.* [27] proposed a feedback-feedforward controller which makes use of path curvature and longitudinal force to design the feedforward controller, and lateral deviation, heading deviation, and their derivatives for the feedback controller. The feedforward controller can respond to future predictions, and the feedback controller will correct immediate deviations. Kapania *et al.* [13] further addressed driving safety at the limits of handling with a feedback-feedforward controller, in which the condition of sideslip-tangency is incorporated in the steering feedback to improve lateral path tracking performance at the sacrifice of stability. The feedforward controller with sideslip demonstrates robust and accurate tracking capability.

A new trend is the efficient, intelligent [28], and adaptive [29], [30] controller. Ahmed *et al.* [31] proposed a fuzzy neural PID method to control mobile robots. The intelligent fuzzy neural PID can adaptively compensate for the effects of friction and disturbance. Bakdi *et al.* [32] used a genetic algorithm to plan an optimal path and an adaptive fuzzy-logic controller to track the planned path. Jardine [33] studied a reinforcement learning approach to learn the parameters in MPC for a system whose dynamics are not well known. Kapania *et al.* [34] used an iterative learning method to gradually determine the proper steering input for a transient driving maneuver, and to improve reference tracking performance with information from previous iterations. Although existing adaptive/intelligent controllers may guarantee robust tracking by modeling disturbances or dynamics that are difficult to express analytically, they require massive training data and adequate computational power. Moreover, real-life autonomous vehicles have difficulty collecting ground-truth data of disturbances.

Instead of improving the tracking performance by modeling disturbances or dynamics, it is realistic to focus on the consequences of external disturbances and partially unknown dynamics, including uncertain tracking error and an unsmooth control process. Accordingly, the RL_PP_PID steering approach is proposed in this paper.

III. Background

A. PP Controller

The PP controller is derived from a kinematic vehicle model that is based on Ackerman steering geometry, which builds a geometric model of a vehicle turning (Fig. 1), whose formulation can be represented by Eq. (1). The steering angle of PP, δ_{pp}, is decided by the vehicle location and target point on the predefined path. Therefore, besides location, the performance of PP totally

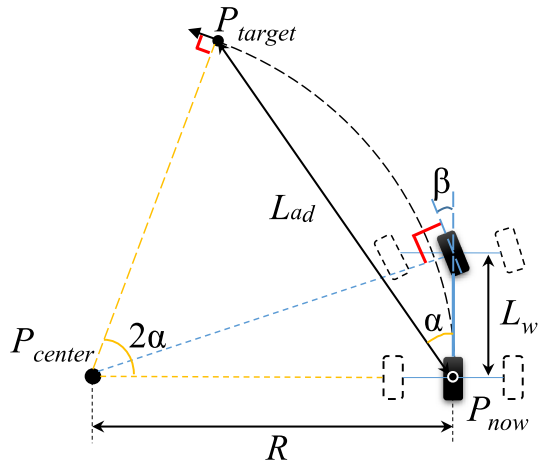

Fig. 1. Geometric principle of PP controller.

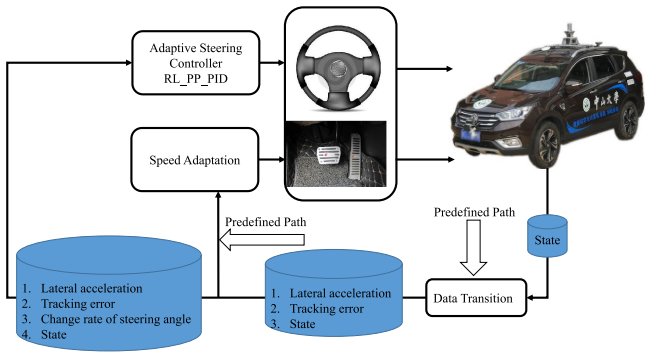

Fig. 2. Proposed tracking scheme.

depends on the look-ahead distance L_{ad}. Satisfactory control can be achieved by selecting a proper L_{ad} because of the relatively robust geometric model. However, a PP controller may not well control the tracking error when turning.

$$\delta_{pp} = tan^{-1}\left(\frac{2L_w sin(\alpha)}{L_{ad}}\right), \tag{1}$$

where L_w is the wheelbase, L_{ad} is the look-ahead distance between the location and the goal point, and α is the intersection angle between the vehicle's heading and the look-ahead direction.

IV. Adaptive Path Tracking Scheme

Fig. 2 illustrates the operational details of the proposed tracking scheme on an autonomous system. The real-time state collected from the vehicle and the predefined path will be transformed to the tracking error, lateral acceleration, and rate of steering angle, which will be regarded as the input to the adaptive steering controller. The speed adaptation controller requires only the tracking error and lateral acceleration as inputs. The output steering angle and speed will be used to control the steering and throttle (or brake).

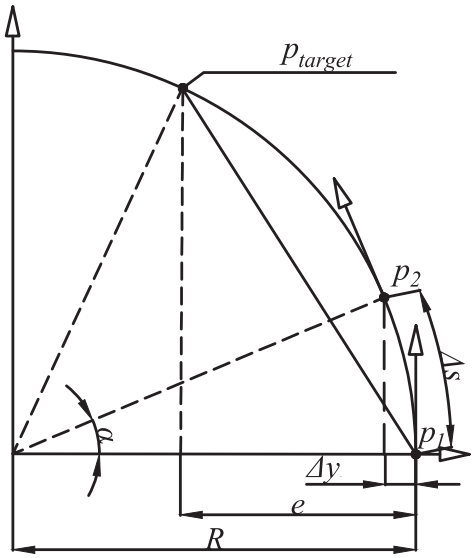

Fig. 3. Interpretation of the change of lateral error.

A. RL_PP_PID

1) Reason for the Combination of PP and PID: Inspired by the feedback-feedforward steering controllers [13], the combination of PP and PID can also be interpreted as a feedback-feedforward controller since PP can be seen as a feedforward controller incorporating curvature and look-ahead constraints, and PID will be response to lateral tracking error directly. As a classical controller, PP is widely used in various robotic applications. However, both corner-cutting and understeer problems are still inevitable. These problems have long been attributed to improper selection of look-ahead distance, and several look-ahead distance adjustment methods are proposed to improve PP [15], [16]. In actual, the nature of these problems is the poor ability of PP in error control. In order to make it clear, without loss of generality, in Fig. 3, we assume the predefined path is straight and the tracking error is represented by e. We can conclude that: The performance of PP depends on the selection of L_{ad}, whose large value can increase stability while decreasing control of tracking error, and conversely, whose small value can intensify the ability to decrease tracking error but may cause shaking. Ollero *et al.* [20] analyzed the stable condition of PP and concluded that L_{ad} should be greater than 1 m. However, there is no clear explanation of how PP reduces tracking error.

To clarify, in Fig. 3, we used the lateral displacement between the target point and current pose to represent tracking error e, which shows the difference between the current pose and future state of vehicles. Compared with the projection distance which is decided by the current pose and predefined path, lateral displacement between the target point and current pose can directly reflect the impacts of L_{ad}. Moreover, it is convenient for the analysis due to that the projection distance may vary greatly between different predefined paths. From Fig. 3, we can conclude that

$$R^2 - (L_{ad}^2 - e^2) = (R - e)^2, \tag{2}$$

where R is the turning radius.

Thus, the relation between R and e is obtained as

$$R = \frac{L_{ad}^2}{2e}. \tag{3}$$

According to Ackerman theory, the vehicle will move along a circle when applying a constant steering angle δ. Therefore, we assume that the steering angle does not change in a sufficiently small time interval, and the vehicle moves from $p1$ to $p2$, as shown in Fig. 3. According to simple circular geometry, we can conclude that

$$\alpha = \frac{\Delta s}{R}, \tag{4}$$

where Δs is the translation of a vehicle driven at a constant speed and a steering angle δ. Substitute Eq. (3) in Eq. (4) to obtain

$$\alpha = \frac{2ev\Delta t}{L_{ad}^2}. \tag{5}$$

According to the geometric relationship shown in Fig. 3, the lateral translation Δy can be expressed as

$$\Delta y = R - R * cos(\Delta\alpha), \tag{6}$$

i.e.,

$$\Delta y = \frac{L_{ad}^2}{2e}\left(1 - cos\left(\frac{2e\Delta s}{L_{ad}^2}\right)\right). \tag{7}$$

To rapidly reduce lateral error, Δy should increase with e. A reasonable assumption is that $L_{ad} \gg e$, and Δs is small enough. The derivative of Eq. (7) is

$$\dot{\Delta y} = 2L_{ad}^2\left(cos\left(2\Delta s \cdot \frac{e}{L_{ad}^2} - 1\right)\right)/e^3 + \left(2\Delta s \cdot \frac{sin\,(2\Delta s \cdot e)}{L_{ad}^2}\right)/e^2. \tag{8}$$

Eq. (8) can be simplified by the above assumptions to

$$\dot{\Delta y} = \frac{4\Delta s^2}{(L_{ad}^2 \cdot e)} > 0. \tag{9}$$

From Eq. (9), $\dot{\Delta y} > 0$ implies that Δy monotonically increases with e, and the increment of Δy will decrease as e increases. Moreover, $\dot{\Delta y}$ is so small that the impact of e on Δy is trivial. When lateral error increases upon turning, PP cannot reduce it as expected. Although the short L_{ad} can reduce the tracking error, frequent adjustments of look-ahead distance may increase the instability of tracking systems. Therefore, a more practical choice for using PP steering method is to intensify the control of tracking error without adjusting the look-ahead distance, which is exactly what the combination can do. With the combination of PID, on the one hand, the feedforward property of PP is not changed. On the other hand, the error control is achieved by PID directly.

2) Combination Steering Controller: PID is deployed to intensify the control of tracking error by combining with PP, i.e., PP_PID. Although a similar method was proposed [11], two important modifications can improve the steering process. First, in the designed PID component, besides the lateral error, the heading error is still considered. The differential constraints make

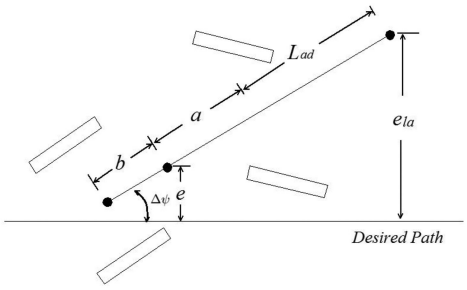

Fig. 4. Definition of lateral error (e), heading error ($\Delta\psi$), and look-ahead error (e_{la}).

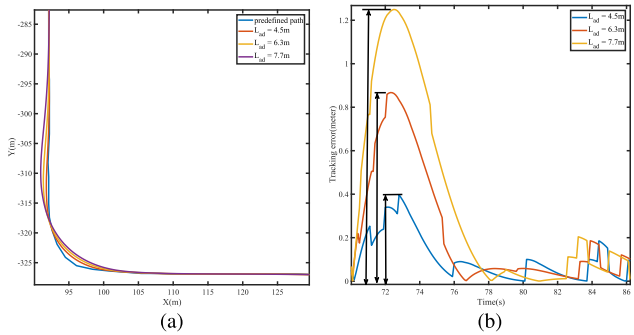

Fig. 5. Effects on PP from different L_{ad}. (a) Tracking performance. (b) Tracking error.

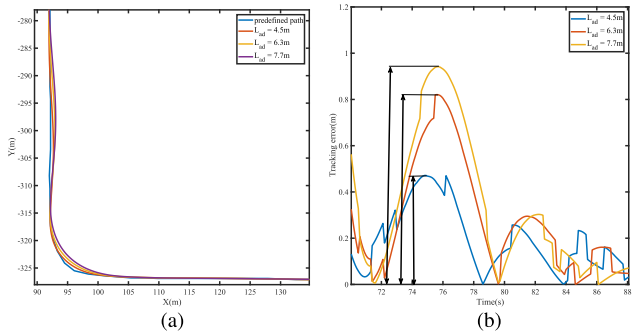

Fig. 6. Effects on PP_PID from different L_{ad}. PP_PID decreases the tracking error and weakens the effects of L_{ad}. (a) Tracking performance. (b) Tracking error.

the vehicle unable to translate transversely without changes of heading. Therefore, PID that depends on lateral error [11] may lead to the deviation of the heading from the path to be tracked. In our specially-designed PID controller, both the lateral error and look-ahead heading error are used to represent the deviation from the path to be tracked.

- PID Component
  The formulation of PID is represented as

$$\delta_{pid} = K_p \cdot e_{la}(t) + K_i \cdot \int_0^t e_{la}(t) + K_d \cdot \frac{\mathrm{d}e_{la}(t)}{\mathrm{d}t}, \tag{10}$$

  where K_p is the proportional control parameter, K_i is the integral parameter, and K_d is the differential parameter. Note that the look-ahead error e_{la} is a combination of both lateral error e and heading error $\Delta\psi$:

$$e_{la} = e + (a + L_{ad})\sin(\Delta\psi), \tag{11}$$

  where e is lateral error, $\Delta\psi$ is heading error, and e_{la} is look-ahead error, as defined in Fig. 4.
  The PID controller can directly respond to the tracking error. Once an improper L_{ad} is selected in PP, and the tracking error is increased, PID will make up for it and make the controller pay more attention to the tracking error.
- Low-pass filter (LPF)
  To smooth the control, a low-pass filter is proposed to smooth the control. LPF can reduce the amplitude of a sudden change in a certain period of time. Actually, the sudden change of an autonomous vehicle may occur in many situations, such as through errors of perception and uncertain location. The filter is shown as

$$\delta = \sum_{0}^{\ell-2} \delta_{previous} * w_{previous} + \delta_{current} * w_{current}, \tag{12}$$

  where $w_{previous}$ is the weight of the previous steering angle in the filter window (Eq. 13), $w_{current}$ is the weight of the current steering angle, ℓ is the length of the filter window,

$$w_{previous} = \frac{1 - w_{current}}{\ell - 1}, \tag{13}$$

  and $\delta_{current}$ is the current steering angle.

PP_PID is summarized as

$$\delta(t) = Lowpass(K_{pp} \cdot \delta_{pp} + K_{pid} \cdot \delta_{pid}), \tag{14}$$

where $\delta(t)$is the final steering angle, and K_{pp} and K_{pid} are the weights of PP and PID, respectively. A turning test was carried out to show the combination performance of PP_PID, as shown in Fig. 5 and Fig. 6. In the test, we show the effects on PP and PP_PID when selecting different look-ahead distances. Fig. 5(b) demonstrates the poor ability of PP to control the tracking error. With the combination of PID, the tracking error is significantly reduced in Fig. 6(b), and the effects of look-ahead distances are weakened. Performance experiences are further discussed in our previous work [12]. While the complementary performance of PP_PID has been proven in our previous work, we were exhausted by the adjustment of K_{pp} and K_{pid} because of the complicated tracking conditions.

3) Adaptive Model Based on Reinforcement Learning: The PP and PID components play different roles in PP_PID. PP can output a basic steering angle and maintain a smooth tracking process with a proper look-ahead distance. When the tracking error is unacceptable, PID will rapidly output an adjustment value for the output of PP to reflect the effects of direct tracking error. A predefined weight set of PP_PID that performs well in a tracking scenario may not operate as well in other scenarios. It is a nontrivial task to determine a proper weight, as concluded in our previous work [12]. Moreover, tracking conditions have different concerns. For example, when the tracking error is in an acceptable range, smoothness should be a priority, and the corresponding weight of PP should be increased. But if the

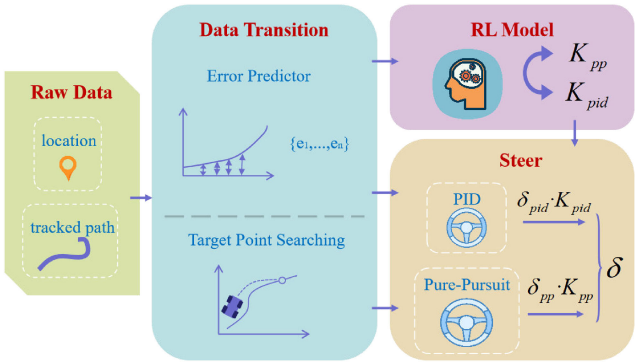

Fig. 7. Learning framework. The raw data are transformed to the required state of the RL model for training the weight-adjustment model, and then the weight set is fed into the steering controller to output the steering angle.

deviation is unacceptable, then the control of tracking error should be emphasized over smoothness, and the weight of PID should be increased. While the working mechanism is clear, it is difficult to decide when and how to adjust the weight set to satisfy the complicated requirements of the whole tracking process.

An efficient RL algorithm, proximal policy optimization (PPO) [35], is used to automatically learn the weight set. Fig. 7 shows the real-time learning framework. The original location and path to be tracked must be transferred for the input of the PPO model and PP_PID in the data transition step. To improve the predictive ability of our tracking approach, a simple tracking-error predictor is customized in our application. For simplicity, we assume that the vehicle will hold the same direction and speed during a fixed time range, and then the tracking error set $\{\vec{e_y}, \vec{e_\theta}\}$ of the next N sampling time steps can be calculated in sequence. Moreover, the data transition step takes responsibility for the input of PP_PID. The trained RL model will adjust the weight set of PP_PID in real time according to $\{\vec{e_y}, \vec{e_\theta}\}$.

The PPO used in this work follows the actor-critic style, and the customized input can be defined as follows:

$$\begin{aligned} s &= (\vec{e_y}, \vec{e_\theta}, K, v, \hbar), \\ \vec{e_y} &= \{e_{y0}, e_{y1}, \ldots, e_{eN}\}, \\ \vec{e_\theta} &= \{e_{\theta 0}, e_{\theta 1}, \ldots, e_{\theta N}\}, \\ \hbar &= \begin{cases} 1, & e_{y0} < 0.3 \\ 0.5, & 0.3 < e_{y0} < 0.6 \\ 0, & \text{others} \end{cases}, \end{aligned} \tag{15}$$

where e_{yi} is the lateral error, $e_{\theta i}$ is the yaw error, K is the curvature of the nearest path point to the location of the host vehicle, v is the forward speed of the vehicle, and $\hbar$ is a flag indicating whether the tracking error is beyond the setting range. For the compatibility and robustness of the trained model, the state for training includes the lateral error, yaw error, curvature of the predefined path, and velocity. The curvature can improve the ability of the proposed RL model to adapt to the geometry of the predefined path. Moreover, the coupling effects of velocity on steering are considered by integrating velocity in the state. For safety reasons, a large tracking error is unacceptable. Therefore, a flag $\hbar$ indicates whether the tracking error is beyond the acceptable range. With the defined input, PPO will output the corresponding actions. Our designed action-space model is composed of separate weight coefficients for PP and PID:

$$a = (K_{pp}, K_{pid}). \tag{16}$$

Besides the input and output, it is crucial for PPO to define a proper reward function to improve the convergence efficiency of the neural network. To balance between the smoothness and tracking error, a customized reward function is developed to reflect various driving conditions. The reward function is as follows:

$$\begin{aligned} r &= -(k_0 \cdot a_x + k_1 \cdot |\dot{\delta}| + k_2 \cdot |\bar{e}_y|) \\ k_0 &= \begin{cases} c_1, & \hbar < 1, |e_{y1}| > e_{matc}, |e_{y2}| > e_{matc} \\ c_2, & \text{others} \end{cases} \\ k_1 &= \begin{cases} c_3, & |e_{y1}| > e_{matc}, |e_{y2}| > e_{matc} \\ c_4, & \text{others} \end{cases} \\ k_2 &= \begin{cases} c_5, & |e_{y1}| > e_{matc}, |e_{y2}| > e_{matc} \\ c_6, & \text{others}, \end{cases} \end{aligned} \tag{17}$$

where a_y is the lateral acceleration, $\dot{\delta}$ is the rate of the steering angle, and $\bar{e}_y$ is the average tracking error of $\vec{e_y}$. Each coefficient k_i is composed of two constant values determined by the concentration of different driving conditions. While there may be many concentrations during the driving process, only the tracking error for safety and control smoothness for ride quality are considered in our reward function. During the tracking process, the concentration will dynamically switch between the control of tracking error and smoothness control. In our customized reward function, a simple and efficient strategy is deployed to show the transition of the concentration. First, we set a key point value of the tracking error, e_{matc}, for the shift of concentration. Second, when the tracking error is in the acceptable range ($\hbar < 1$), the concentration will be decided by comparing the results of the first two lateral tracking errors (e_{y1} and e_{y2}) with e_{matc}. Based on this strategy, c_1, c_2, c_3, c_4, c_5, c_6 will be adjusted according to the concentration.

For policy iteration, PPO maximizes a clipped surrogate objective, as shown in Eq. (18), and penalizes changes to the policy that might decrease the reward signal. In every training step, ten cycles are required to reuse the data for training because of the penalty and clipped loss. To ensure the exploration ability of PPO, the probabilities of the actions to be chosen are decided by a normal distribution whose standard deviation and mean are produced by the fully-connected layer in the actor network, and whose standard deviation starts from a stochastic variable and ultimately converges to an orderly variable.

$$\begin{aligned} L^{CLIP} &= \hat{\mathbb{E}}\left[min(R_t\hat{A}_t, clip(R_t, 1-\epsilon, 1+\epsilon)\hat{A}_t)\right] \\ R_t &= \frac{\pi_\theta}{\pi_{\theta_{old}}}, \end{aligned} \tag{18}$$

where R_t is the probability ratio of the new policy π_θ and old policy $\pi_{\theta_{old}}$, and $\hat{A}_t$ is an advantage estimator computed by the T time steps of the reward signal and the predicted value from the critic model.

The workflow of PPO is shown in Fig. 8. Algorithm 1 interprets the learning process, and the line number corresponds to the number in Fig. 8. We first transform the vehicle's state and

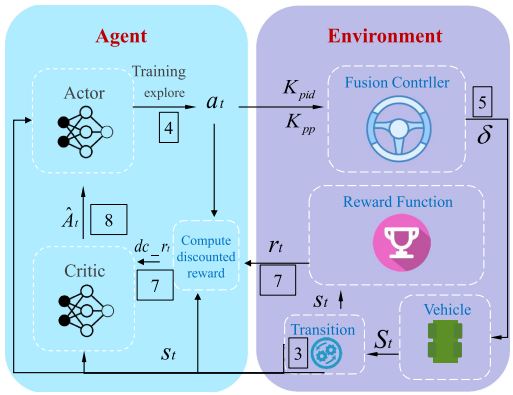

Fig. 8. Workflow of proposed RL model in the fusion steering controller.

Algorithm 1: Learning Process.

Require: $episodes = n, M = 10$

1: **for** $i \in \{1, \ldots, n\}$ **do**
2: **while** Location (x, y) is not close to $(x_{ref}^{end}, y_{ref}^{end})$ **do**
3: $s_t \leftarrow$ transform_location($\vec{S}$, $\vec{S}_{ref}$);
4: $a_t \leftarrow$ select_action(s_t);
5: $\delta \leftarrow$ compute_steering_angle(a_t);
6: $r_t \leftarrow$ get_reward(s_t, a_t);
7: $dc\_r_t \leftarrow$ compute_discounted_reward(s_t, a_t, r_t);
8: Estimate advantages $\hat{A}_t$ and update θ_{old};
9: **for** $j \in \{1, \ldots, M\}$ **do**
10: $l_{actor} = L^{CLIP}$;
11: update_actor();
12: **end for**
13: **for** $j \in \{1, \ldots, M\}$ **do**
14: $l_{critic} = \hat{A}_t$;
15: update_critic();
16: **end for**
17: **end while**
18: **end for**

the set of reference paths to a PPO state space (line 3), and then compute the weights a_t of PP_PID by the policy model. Based on the calculated weights, PP_PID can output the steering angle δ for lateral control. With the state s_t and action a_t, we can get the reward value r_t through the reward function in Eq. (17). The advantage $\hat{A}_t$ is then estimated and $\pi_{\theta_{old}}$ is updated. Finally, the actor model and critic model are updated by the optimizer. There are two policy models, including the new policy π_θ and old policy $\pi_{\theta_{old}}$, to save the parameters of the actor networks. With these networks, the probability ratio of π_θ to $\pi_{\theta_{old}}$ is calculated to minimize the slight changes that might lead the training policy to deviate from the high reward. After n episodes of online training, the actor-critic model will converge, and the weight adjustment model will be successfully trained.

B. Rough-to-Fine Speed Adaptation

The speed adaptation framework is shown in Fig. 9. The speed is first roughly decided by the predefined path, and then a real-time online refinement based on fuzzy logic is deployed to adjust

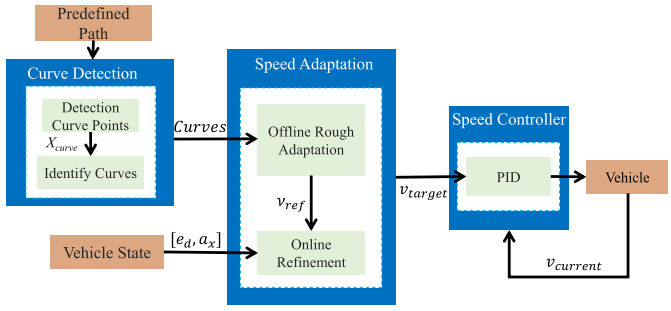

Fig. 9. The framework of speed adaptation. The predefined path is fed into the curve detector to identify the curves in the path to calculate the rough speed v_{ref} attached to the curves offline. Afterward, the real-time state is used to finely modify v_{ref} to obtain the tracking speed v_{target}. Finally, a PID controller is in series to control the throttle (or brake) to reach v_{target}.

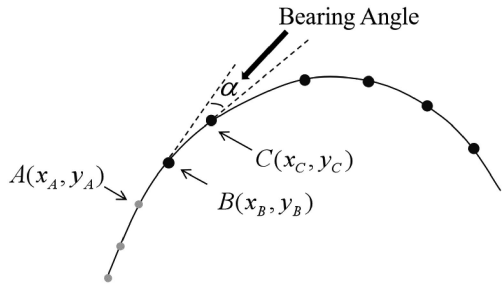

Fig. 10. Definition of bearing angle.

the rough speed v_{ref} according to the real-time tracking state of the vehicle. A curve-detection method is used to identify curve segments, and then the curves detected previously will be used to obtain the rough speed following the method in [36]. With a rough speed, the online refinement module will further adjust it. Afterward, an augment-PID will be used to compute the value for throttle control to make the vehicle reach the speed.

1) Curve Detection: The geometry of the path to be tracked has a great impact on the speed. In normal driving experiences, the speed should change with the tightness of a curve. Therefore, we first need to find all possible curves X_{curve} in the predefined path X by the bearing angle, as shown in Fig. 10.

X_{curve} is composed with the curve point whose bearing angle (α) is greater than a threshold. c_{flag} flags the curve point in X.

$$\alpha = \cos^{-1}\left(\frac{(x_B - x_A)\cdot(x_C - x_B) + (y_B - y_A)\cdot(y_C - y_B)}{\sqrt{(x_B - x_A)^2 + (y_B - y_A)^2}\cdot\sqrt{(x_C - x_B)^2 + (y_C - y_B)^2}}\right)\frac{180}{\pi}$$

$$c_{flag} = \begin{cases} True, & \alpha \geq \gamma \\ False, & \alpha < \gamma \end{cases}$$

$$X_{curve} = \{x_1, x_2 \ldots x_i | x_i(c_{flag} = True), i \in |X|\} \tag{19}$$

To define α requires the sequential selection of three consecutive points in the path (A, B, and C in Fig. 10), and the bearing angle within these three points is determined by Eq. (19). If the

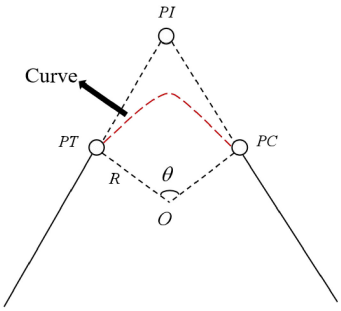

Fig. 11. Parameters of the curve.

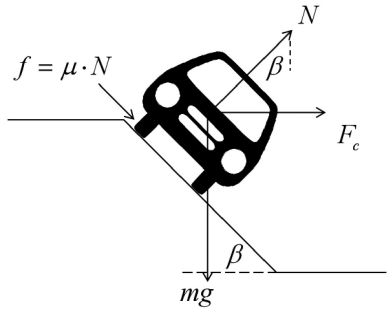

Fig. 12. Force analysis of vehicles. N is normal force, f is friction force, β is the banked angle, μ is the friction coefficient, m is the vehicle mass, g is the gravity, and F_c is the centripetal force.

bearing angle is greater than a threshold γ, then B is considered as a curve point.

$$R = \sqrt{(x_{PC} + x_O)^2 + (y_{PC} + y_O)^2}$$
$$C = \sqrt{(x_{PT} + x_O)^2 + (y_{PT} + y_O)^2}$$
$$\theta = 2 \cdot sin^{-1}\left(\frac{C}{2R}\right) \cdot \frac{180}{\pi} \tag{20}$$

Fig. 11 describes the parameters of a detection curve, which is marked by a start point PT and end point PC. The calculation method for these parameters is shown in Eq. (20).

C. Speed Adaptation Approach

The customized speed adaptation approach includes offline rough speed adaptation and online refinement. With the geometric parameters from the curves, the predefined path can be attached with v_{ref} according to the winding degree offline, a process called rough adaptation. However, the real-time state must be reflected in the speed adaptation for safety. A fine adaptation method is deployed to output the target speed (v_{target}). Finally, an augmented PID (Eq. 21) controls the throttle to reach v_{target}.

$$\Delta v_t = v_{target} - v_{current}$$
$$\Delta o_t = k_p \cdot \Delta v_t + k_d \cdot (\Delta v_t - \Delta v_{t-1}) + k_i \cdot \sum \Delta v$$
$$o_t = o_{t-1} + \Delta o_t, \tag{21}$$

where $v_{current}$ is the current speed of the vehicle, k_p is the proportion coefficient, k_i is the integral coefficient, k_d is the differential coefficient, and o_t is the throttle value.

1) Offline Rough Speed Adaptation: Below is the method to obtain a rough speed [36]; v_{ref} is estimated by the centripetal and centrifugal forces in each curve.

$$F_c = \frac{mv^2}{r}$$
$$Ncos\beta = mg + fsin\beta$$
$$F_c = Nsin\beta + N\mu cos\beta \tag{22}$$

Eq. (22) builds the relationship between the centripetal and centrifugal forces, as shown in Fig. 12. The speed for a specific curve is

$$v = \sqrt{\frac{(e + \mu)g}{\kappa}}, \tag{23}$$

where $e = tan\beta$, and $\kappa = \frac{1}{r}$ is the curvature.

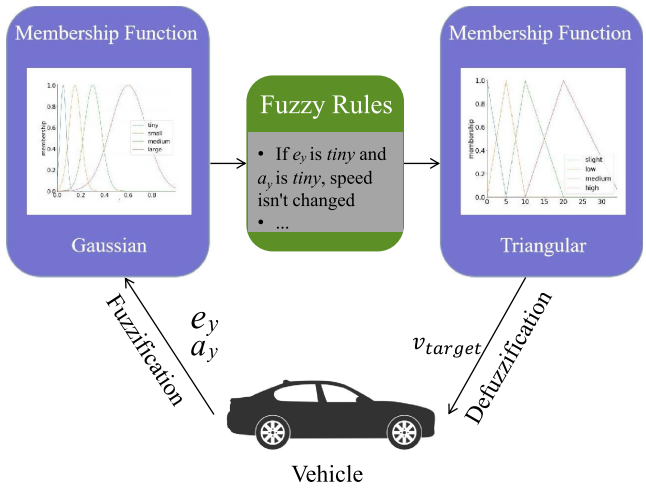

Fig. 13. Online refinement based on fuzzy logic. The lateral tracking error (e_d) and lateral acceleration (a_y) go through fuzzification and then are fed into the rule base for inference. The output v_{target} is implanted in the vehicle.

2) Online Refinement: During real-time tracking, v_{ref} must be modified with the tracking error and the lateral acceleration. According to tests and driving experiments, the speed should be decreased if the tracking error or lateral acceleration increases. However, to build a model to reflect the relationship between the tracking condition and the corresponding speed is nontrivial. We use fuzzy logic to model the relationship, as shown in Fig. 13. Although speed is important to the tracking process, there is no need to overemphasize the accuracy of speed regulation. Therefore, instead of a complicated model, manual regulation and rule setting are deployed in our approach. There are several basic rules:

- If the tracking error and lateral acceleration are small, then the vehicle should not slow down.
- If the tracking error is large and the lateral acceleration is small, then holding the throttle will bring the vehicle back to the center of the path more quickly.

- If the tracking error is small and the lateral acceleration is large, then proper deceleration is needed to achieve the desired ride quality.
- If the tracking error and lateral acceleration are large, then the vehicle should be slowed down to prevent instability and inaccuracy.

These basic rules will be refined to obtain specific fuzzy rules, and the rules and parameters in fuzzy speed adaptation will be iteratively adjusted according to tests.

V. Simulation and Field Tests

We performed several tests to validate our adaptive tracking approach, using the Carla [37] simulator and a real-life autonomous vehicle. Firstly, we introduce the training process of the RL model for weight adjustment. Three challenging scenarios were used to test the adaptivity of RL_PP_PID. The adaptive mechanism was analyzed according to its performance in a typical driving scenario. Secondly, the steering performance is investigated further by comparing with Stanley, LQR and MPC. Thirdly, the performance of the proposed speed adaptation method is demonstrated by comparing RL_PP_PID without speed adaptation, RL_PP_PID with offline speed adaptation, and RL_PP_PID with integrated speed adaptation. Thirdly, a filed test is carried out to show the practicability. Finally, to evaluate the performance, the effectiveness is measured by two metrics, including the tracking error for accuracy and the jerk[1] for smoothness. For the use of the jerk, on the one hand, we learn from planning methods that make use of the lateral jerk to smooth the planned path. On the other hand, in real applications, lateral jerks have been used in the field of path tracking [38], [39].

A. Setup

A complicated S trajectory for training was collected for the host vehicle in a CARLA simulator, as shown in Fig. 15. PP and PID were separately applied to track the trajectory and ensure they could track successfully by manually tuning their parameters. PPO was used to update the actor and critic models and to iteratively collect about 32 frames of transition data $\{s_t, a_t, r_t\}$. The training process stops, and the model converges after about 50 episodes. A laptop with an Intel i7-7700HQ CPU and 8 GB RAM was used, on which the whole learning process ran in about three hours because of the model's quick convergence. Moreover, the steering angle can be output at 30 HZ on average.

B. Test Scenarios

As shown in Fig. 16, the test included three scenarios: T-shape (typical shape) (Fig. 16(a)), U-shape (Fig. 16(c)), O-shape (Fig. 16(b)) and Random-shape (Fig. 16(d)). All the predefined paths were collected manually, and not planned. This ensured that the comparison test was carried out on the same path to be tracked, which is difficult to plan, even with same constraints. The T-shape is the same as the test scenario of PP_PID [12], which can objectively prove the improvement of RL_PP_PID.

[1]For simplicity, jerk means lateral jerk in this paper.

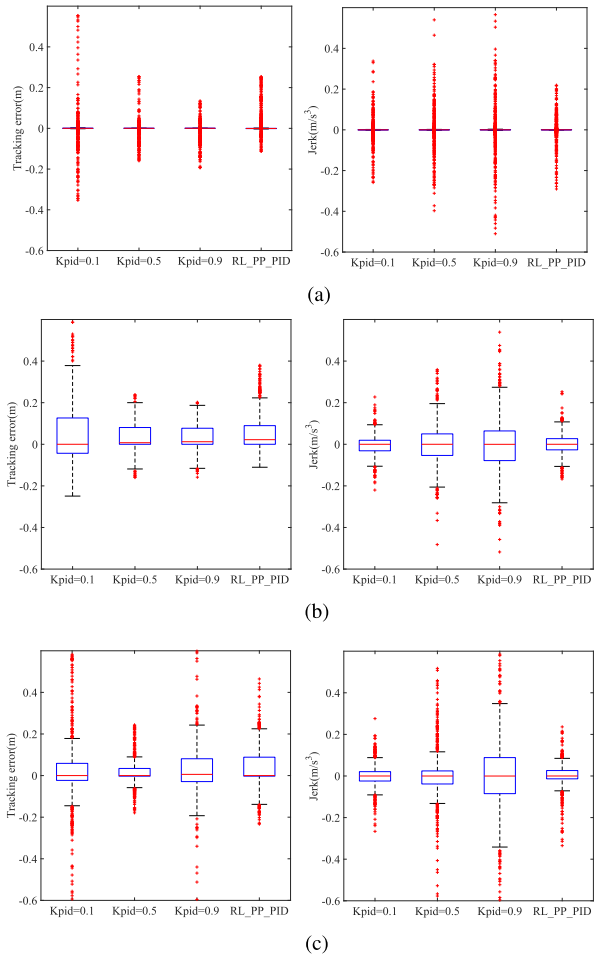

Fig. 14. The comparison results of the three scenarios, including tracking error and lateral jerk. (a) T-shape. (b) O-shape. (c) U-shape.

The U-shape requires a sharp turn, and the O-shape requires the vehicle to keep steering throughout the trajectory. For the case of random shape, it will be used for testing later-on high-speed performance.

C. Tracking Performance

Table I shows the statistical results of tracking error and jerk, and we can observe that RL_PP_PID can achieve average high-level performance both in the training and test scenarios. With the customized reward function and PPO, RL_PP_PID can learn from the training scenario to adaptively deal with the curves of the test scenarios by adjusting the weights of PP and PID according to the features shown by the path to be tracked, along with the vehicle's real-time state. While the average tracking error and jerk of all the scenarios are acceptable, the performance in the training scenario is worse than in the test scenarios. A reasonable interpretation is that the training scenario is specially designed and more complicated than the test scenarios, which makes the vehicle difficult to follow. In this test, the velocity was set to 35 km/h, and the high-speed application will be demonstrated later.

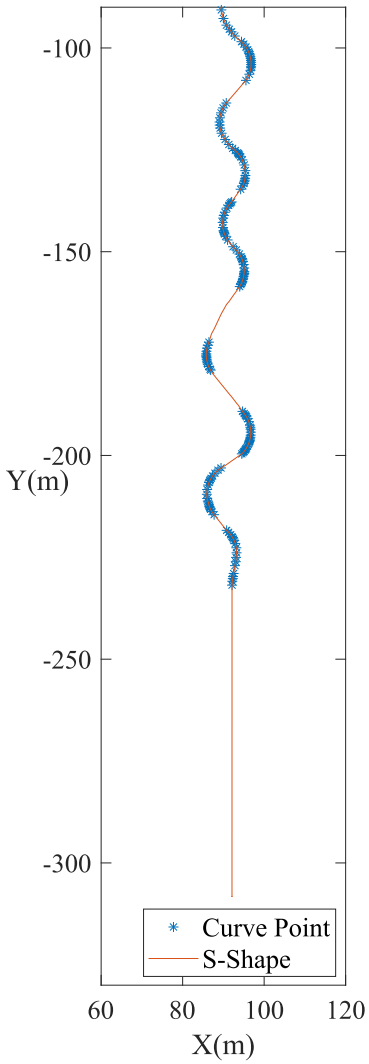

Fig. 15. S shape trajectory collected by the host vehicle in CARLA. The host vehicle is driven by human beings, and the trajectory collected needs to contain as many as curve scenarios to cover more possible driving scenarios.

D. Adaptive Mechanism

PP_PID [12] is used as a comparison to show the improvements of RL_PP_PID over PP_PID on the above three test scenarios. The same constant velocity (35 km/h) was used in the tracking process so that comparisons are independent of velocity. Two sets of weights were selected to show the weight-adjustment procedure of PP_PID. The results of Fig. 14 indicate the difficulties in matching proper weights to PP and PID. Based on careful observation of performance on tracking error, the tracking accuracy can be improved dramatically by increasing the weight of PID (Kpid $= 0.5$, Kpid $= 0.9$), as shown in the tracking error. The powerful ability of PID to control the tracking error stems from the direct feedback on tracking error. However, the results of jerks show that the jerk will be amplified and the oscillation that is difficult to restore may happen because of improper weights for PID. It is a challenge to properly pre-adjust the weights to guarantee both tracking error and ride quality.

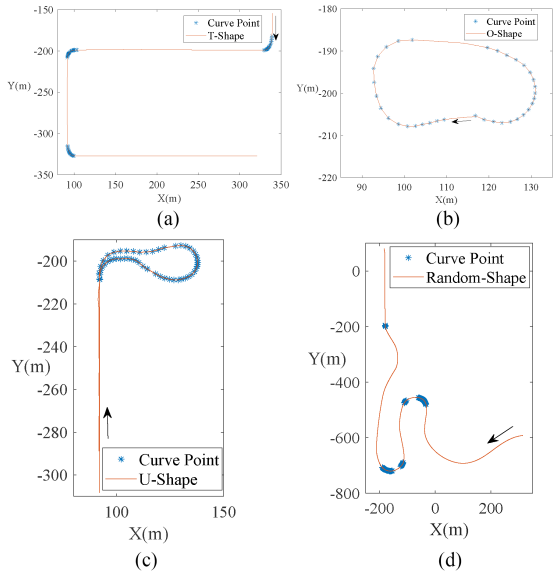

Fig. 16. Four test scenarios for test the proposed tracking scheme. The black arrow indicates the start point and driving direction. (a) T-shape. (b) O-shape. (c) U-shape. (d) random-shape.

TABLE I
STATISTIC RESULTS OF TRACKING APPROACH APPLIED TO THE TRAINING SCENARIO AND THE TEST SCENARIOS

scenario Shape	Tracking Error (mean, std, max)	Jerk (mean, std, max)
S shape(training)	(0.0878, 0.1061, 0.3707)	(0.0505, 0.0661, 0.3679)
T shape(test)	(0.0141, 0.0378, 0.2523)	(0.0131, 0.0299, 0.3088)
U shape(test)	(0.0421, 0.02, 0.3719)	(0.0283, 0.0497, 0.3335)
O shape(test)	(0.0508, 0.0837, 0.3801)	(0.0274, 0.0405, 0.2528)

Nevertheless, RL_PP_PID provides a solution to the weight-adjustment problem of PP and PID by the RL model. While RL_PP_PID does not perform best when separately comparing the results of tracking error or jerk, it is better than manual adjustment for balancing between tracking error for safety and jerk for ride quality. In the RL model, the actor model of PPO, which is trained to capture the characteristics of smoothness and accuracy for the path tracking task, can automatically select proper weights. The customized reward function can reward actions that decrease jerk and punish those that increase jerk or tracking error. A clearer adaptive mechanism of RL_PP_PID is illustrated in Fig. 17. The change of weight with the locations in the three scenarios shows the adaptive-adjustment ability of the RL model. In Fig. 17, the brightness of color is used to show the change of K_{pid}. A detailed observation of the three scenarios shows that the segments to be tracked with more curve points are brighter. The amplified tracking error in the curves triggers the RL model to increase K_{pid} to control the tracking error. Once the tracking error is under control, K_{pid} will be reduced, and the influence of PP will be increased correspondingly.

E. High-Speed Steering Performance

We conducted a test to compare RL_PP_PID to the Stanley method [40], MPC, and LQR on a publicly available predefined

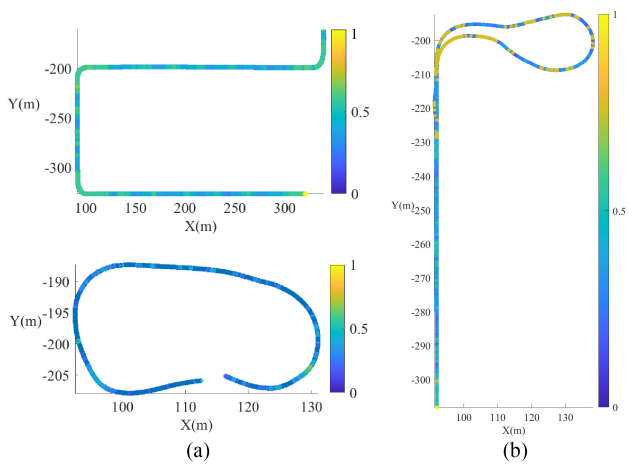

Fig. 17. Adaptive weights for three tracking scenarios. K_{pid} is represented by RGB color. (a) Adaptive weights in T-shape and O-shape tracking scenario. (b) Adaptive weights in U-shape tracking scenario.

TABLE II
PARAMETERS FOR THREE COMPARISION COUNTERPARTS

Method	Parameters	value
Stanley	Proportional Gain K	0.5
MPC	Sample time	0.03(s)
	Prediction horizon	20(time steps)
	Control horizon	4(time steps)
LQR	Q	$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$
	R	1

path (Fig. 16(d)).[2] For the comparison methods, we try our best to make them maintain a good state in our test, several important parameters in these methods are described in detail in Table II. Moreover, a well-known bicycle model [41] with four states $[x, y, \theta, v]$, as shown in (24), is used in both MPC and LQR methods.

$$\begin{aligned} \dot{x} &= vcos(\theta + \beta) \\ \dot{y} &= vsin(\theta + \beta) \\ \dot{\theta} &= \frac{v}{l_r + l_f} sin(\beta) \\ \dot{v} &= a \end{aligned} \tag{24}$$

where $\beta = tan^{-1}(\frac{l_r}{l_r+l_f} tan(\delta))$ denotes the slip angle, l_r and l_f are the distances from rear and front axles to the center of vehicles, respectively. δ and a are the steering angle of front wheels and acceleration, respectively.

All the controllers ran with the speed profile that was recorded when collecting the path. Although 60 km/h is enough for urban driving, we want to further investigate the high-speed performance (up to 80 km/h) in this test. In addition, in order to show the generalization, the used RL_PP_PID was not retrained for high speed scenarios. The test results are shown in Fig. 18. The first observation is that the Stanley method cannot perform well both in tracking error and jerk control. Although LQR did well in error control, poor ability was shown in the jerk. Moreover, both RL_PP_PID and MPC achieved similar good tracking performance, and RL_PP_PID was better than MPC at tracking error control. The jerk is shown in Fig. 18(b), and we can observe that for jerk control, compared to RL_PP_PID, MPC keeps the jerk in a narrower range but has jitters, while RL_PP_PID maintains smooth jerks. Comparing Fig. 18(a) and Fig. 18(b), we can conclude that a reason for RL_PP_PID increasing jerks is to well control the tracking error. In brief, by combining two simple controllers and training simply, RL_PP_PID can achieve performance similar to kinematic-model-based controllers, even in high-speed scenarios. Moreover, since that the used RL_PP_PID in this test is only trained in low-speed scenarios, the performance will be improved given more training datasets with high-speed scenarios.

[2]https://github.com/paulyehtw/Lane-Keeping-Assist-on-CARLA

F. Robustness to Sensor Noises

We add sensor noises to investigate the robustness of the proposed tracking scheme. In this test, the typical scenario is used, and four Gaussian while noises are added, including $\sigma_1 = 0.03$, $\sigma_2 = 0.05$, $\sigma_3 = 0.07$ and $\sigma_4 = 0.1$. The maximum noise is limited in 0.3 m ($0 + 3\sigma_4$), which coincides with the positioning accuracy of differential GPS in real-life autonomous driving systems. However, the output of positioning system is continuous, and our noises are randomly added, which increases the difficulty of the simulation. Fig. 19 shows statistical results of the tracking error and jerk. The first observation is that the tracking error is still acceptable even under the maximum noise condition (Fig. 19(a)). Compared with the tracking error, the noises have more influence on the jerk (Fig. 19(c)). The main reason for that is the random noises make the input control cannot obtain ground-truth feedback and may suddenly steer to correct the tracking error. The second observation is that RL_PP_PID is more robust to the noise with $\sigma \leq 0.05$. When a noise greater than 0.05 is applied, the tracking error increases dramatically and is more discretized (Fig. 19(b)). For RL_PP_PID, in order to increase the robustness to high noises, augmenting more training data with noises should be a reasonable solution.

G. Effects of the Proposed Speed Adaptation Method

The impact of speed cannot be ignored in the tracking process. To validate the effectiveness of the proposed fuzzy speed adaptation method, the typical tracking scenario (Fig. 16(a)) is used in this test, and the maximum speed is set to 50 km/h. In our tracking scheme, the target speed depends on the static geometry of the predefined path and real-time tracking performance. Fig. 22 analyzes the impact of rough offline adaptation and online refinement. With carful observation of Fig. 22(a), one can see that the original RL_PP_PID without speed adaptation will fluctuate after turning a corner. Improper speed in a curve causes steering overshoot, which is difficult to correct. It is to be noted that the test vehicle had to stop at 20.7 s because of a collision caused by excessive tracking error. To weaken the influence of the geometry of the predefined path, it

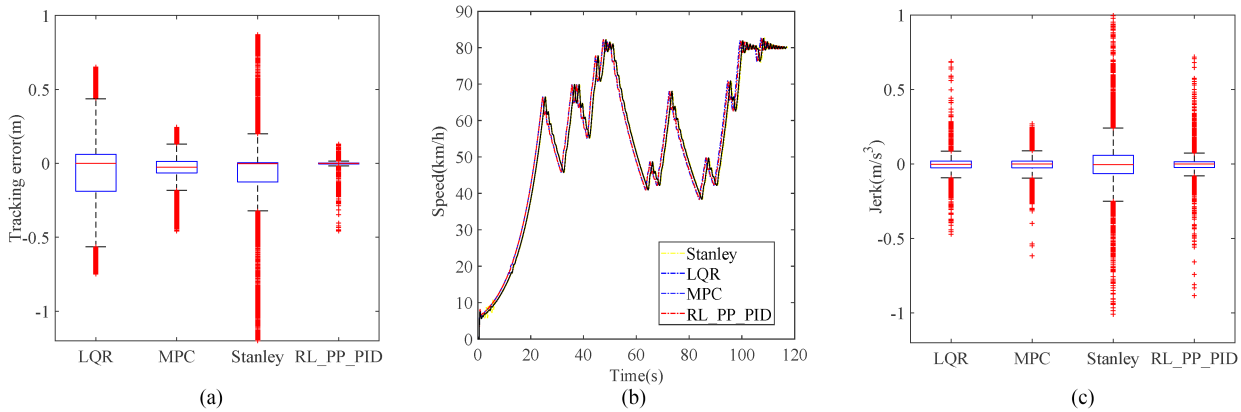

Fig. 18. Comparison results of RL_PP_PID, Stanley, MPC and LQR methods with the same speed profile. (a) Tracking error. (b) Jerk. (c) Speed.

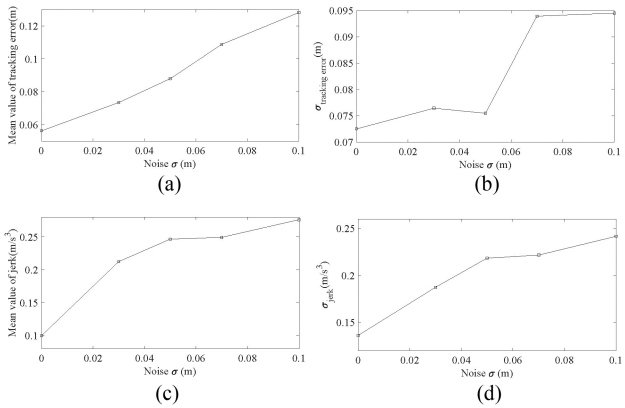

Fig. 19. Four Gaussian white noises are added into the position of test vehicle to test the robustness of RL_PP_PID to noises. (a) Mean value of tracking error. (b) Standard derivation of tracking error. (c) Mean value of jerk. (d) Standard derivation of jerk.

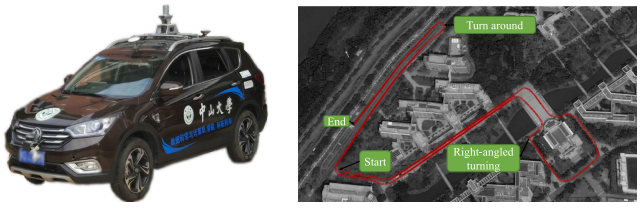

Fig. 20. The used autonomous vehicle for test and the GPS path to be tracked. (a) Autonomous vehicle. (b) Path to be tracked.

is reasonable to change the speed with the geometry. While we can observe that the overshooting problem was alleviated before the vehicle went into the second corner (the vehicle could safely execute the first turn with offline speed adaptation), steering overshoot was inevitable after the second turn, as shown in Fig. 22(b). Moreover, the test vehicle stopped near the third turn. Although the vehicle had decelerated before entering the turn (Fig. 22(c)), the expected speed for a successful turn was not realized, as the offline adaptation method did not consider the impact of real-time tracking error, especially lateral acceleration, which was still unstable (Fig. 22(b)). The unstable acceleration increased the difficulty of lateral control, which may be the main reason for the collision in the third turn. Therefore, it is necessary to consider both the impact of path geometry and the real-time state. Better performance was realized by RL_PP_PID with full speed adaptation. With the proposed speed adaptation method, the vehicle successfully achieved the tracking task of a typical scenario. The speed was adjusted more frequently than by the offline adaptation method in the turn segment to adapt to the changes of tracking error and the lateral acceleration (Fig. 22(c)).

H. Field Test

In order to test RL_PP_PID, we carry out real-life vehicle test in our campus. The used autonomous vehicle and the path to be tracked are shown in Fig. 20. For the path, we collected it by a driver and make it challenging for the tracking method by including U-turn and right-angled turning scenarios. In these two scenarios, large maneuvers are inevitable, which will lead to larger tracking error and jerks. Fig. 21 shows the statistic results of tracking error, jerk and the corresponding speed. For the sake of safety, we limit the maximum speed to 30 km/h, and modify the proposed speed adaptation method to automatically adjust the speed and try to keep constant speed when driving straightly. From Fig. 21, the first observation is that both the tracking error and jerk of field tests are greater than the simulation results. For the possible reasons, besides the challenging scenarios, the dynamics of real-life vehicle is more complicated than the simple dynamics used in CARLA, which cannot reflect the real movement of real vehicles. Complicated dynamics will raise the difficult of control and lead to greater error. Moreover, retraining RL_PP_PID with a real-life vehicle is difficult in consideration of possible dangers due to unpredicted control generated by controllers. Therefore, we do not retrain RL_PP_PID to adapt to the changes in the tracking error and jerk, and directly apply pretrained model by CARLA on our real-life vehicle. With such a transfer, although worse performance will be demonstrated than simulation results due to the error gap between the simulator and real-life vehicle, adaptive capacity should work as simulations. We can observe that despite greater tracking error is achieved than simulations,

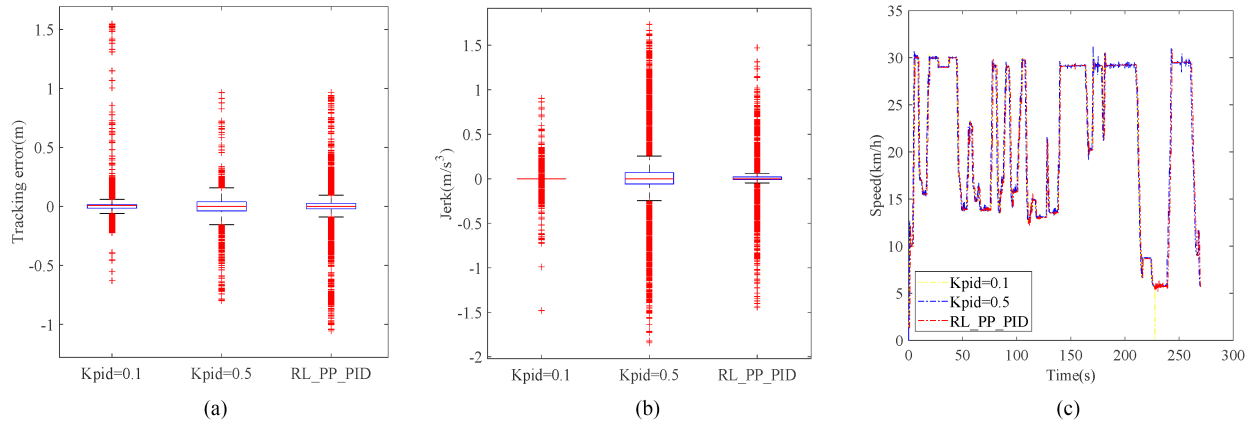

Fig. 21. The tracking results of field tests. (a) Tracking error. (b) Jerk. (c) Speed.

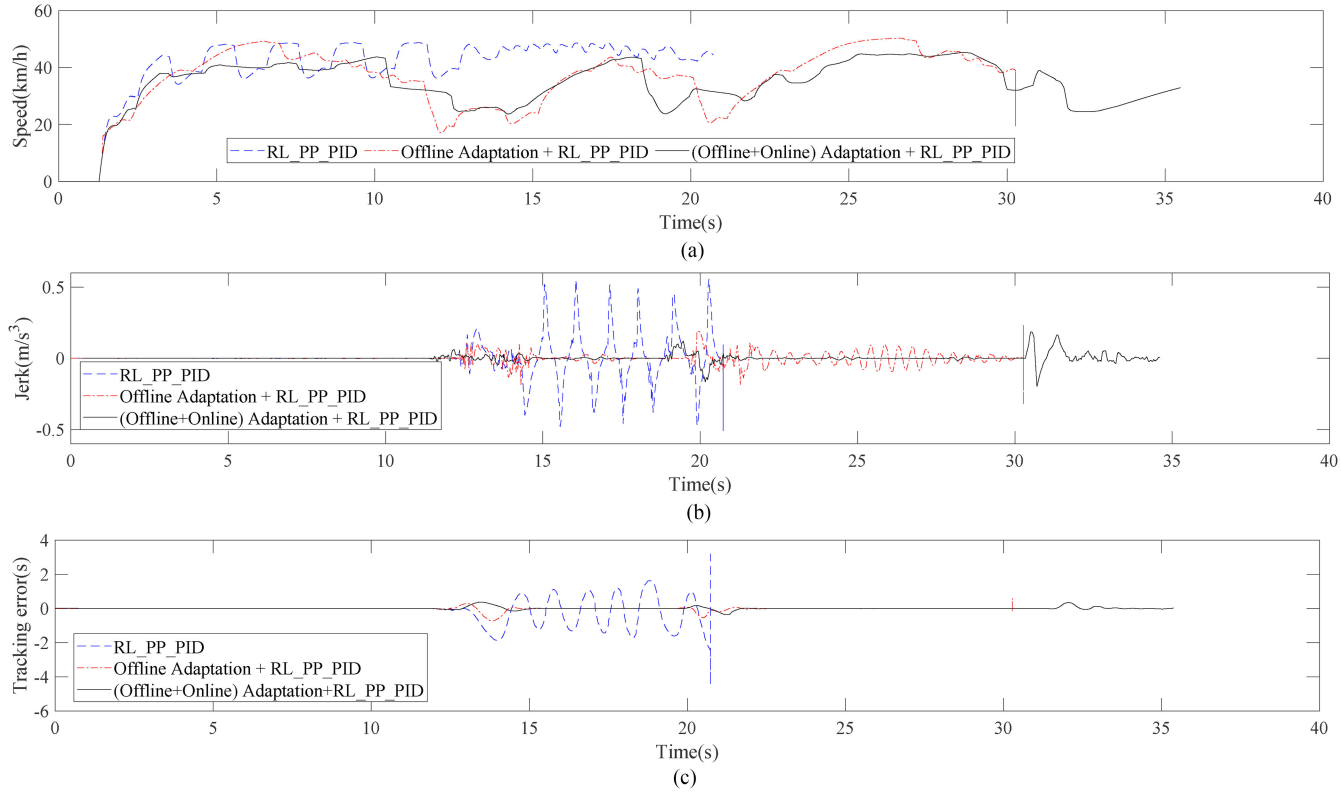

Fig. 22. The comparing results of RL_PP_PID with maximum speed, RL_PP_PID with offline speed adaptation method and RL_PP_PID with offline and online speed adaptation method. The reason of several shorter data sequences is that the test vehicle crashed and the data collecting process has to stop. (a) Tracking error. (b) Jerk. (c) Speed.

consistent with the simulation results (Fig. 18), RL_PP_PID still shows better balancing performance in tracking error and smoothness, which further validates the balancing ability and practicability. The second observation stems from Fig. 21(c) is that PP_PID with smaller weight of PID ($Kpid = 0.1$) cannot achieve U-turn, and the vehicle has to be stopped by us (between 200 and 250 s)before crashing road boundaries. Nevertheless, it maintains a smoother control process in most of scenarios, which further proves that decreasing the weight of $Kpid$ may improve the smoothness.

VI. Conclusion

In this paper, we proposed an adaptive path-tracking approach to simultaneously track a predefined path with high accuracy and a good riding experience. For lateral control, a steering method based on the fusion of the RL model with a combination of two simple controllers, PP and PID, was designed to adapt to various tracking scenarios. The velocity was also considered in the tracking process. A rough-to-fine speed adaptation method was deployed to change the velocity with the geometry of the predefined path and the real-time state of the vehicle. both

simulation and field tests validated the effect of our proposed tracking method. Nevertheless, training a RL model which can better adapt to real-life vehicles should be emphasized in our further research.

REFERENCES

[1] Q. Li, L. Chen, M. Li, S.-L. Shaw, and A. Nüchter, "A sensor-fusion drivable-region and lane-detection system for autonomous vehicle navigation in challenging road scenarios," *IEEE Trans. Veh. Technol.*, vol. 63, no. 2, pp. 540–555, Feb. 2014.

[2] L. Chen, L. Fan, G. Xie, K. Huang, and A. Nüchter, "Moving-object detection from consecutive stereo pairs using slanted plane smoothing," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 11, pp. 3093–3102, Nov. 2017.

[3] L. Chen, Y. Shan, W. Tian, B. Li, and D. Cao, "A fast and efficient double-tree RRT*-like sampling-based planner applying on mobile robotic systems," *IEEE/ASME Trans. Mechatronics*, vol. 23, no. 6, pp. 2568–2578, Dec. 2018.

[4] Y. Shan, B. Li, J. Zhou, and Y. Zhang, "An approach to speed up RRT," in *Proc. IEEE Intell. Vehicles Symp.*, 2014, pp. 594–598.

[5] S. Xu and H. Peng, "Design, analysis, and experiments of preview path tracking control for autonomous vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 1, pp. 48–58, Jan. 2020.

[6] N. H. Amer, H. Zamzuri, K. Hudha, V. R. Aparow, Z. A. Kadir, and A. F. Z. Abidin, "Modelling and trajectory following of an armoured vehicle," in *Proc. IEEE SICE Int. Symp. Control Syst.*, 2016, pp. 1–6.

[7] P. Zhao, J. Chen, Y. Song, X. Tao, T. Xu, and T. Mei, "Design of a control system for an autonomous vehicle based on adaptive-pid," *Int. J. Adv. Robotic Syst.*, vol. 9, no. 2, pp. 1–11, 2012.

[8] M. A. Zakaria, "Trajectory tracking algorithm for autonomous ground vehicle," Ph.D. dissertation, Dept. Autom. Control, Universiti Teknologi Malaysia, Skudai, Malaysia 2015.

[9] A. S. Yamashita, P. M. Alexandre, A. C. Zanin, and D. Odloak, "Reference trajectory tuning of model predictive control," *Control Eng. Pract.*, vol. 50, pp. 1–11, 2016.

[10] H. Merabti, K. Belarbi, and B. Bouchemal, "Nonlinear predictive control of a mobile robot: a solution using metaheuristcs," *J. Chin. Inst. Engineers*, vol. 39, no. 3, pp. 282–290, 2016.

[11] A. L. Rankin, C. D. Crane, and D. G. Armstrong, "Evaluating a pid, pure pursuit, and weighted steering controller for an autonomous land vehicle," in *Proc. Mobile Robots XII*, 1998, vol. 3210, pp. 1–13.

[12] Y. Chen, Y. Shan, L. Chen, K. Huang, and D. Cao, "Optimization of pure pursuit controller based on pid controller and low-pass filter," in *Proc. IEEE 21st Int. Conf. Intell. Transp. Syst.* 2018, pp. 3294–3299.

[13] N. R. Kapania and J. C. Gerdes, "Design of a feedback-feedforward steering controller for accurate path tracking and stability at the limits of handling," *Vehicle Syst. Dyn.*, vol. 53, no. 12, pp. 1687–1704, 2015.

[14] N. H. Amer, H. Zamzuri, K. Hudha, and Z. A. Kadir, "Modelling and control strategies in path tracking control for autonomous ground vehicles: A review of state of the art and challenges," *J. Intell. Robotic Syst.*, vol. 86, no. 2, pp. 225–254, 2017.

[15] Y. Kuwata, J. Teo, G. Fiore, S. Karaman, E. Frazzoli, and J. P. How, "Real-time motion planning with applications to autonomous urban driving," *IEEE Trans. Control Syst. Technol.*, vol. 17, no. 5, pp. 1105–1118, Sep. 2009.

[16] Y. Shan, W. Yang, C. Chen, J. Zhou, L. Zheng, and B. Li, "CF-pursuit: A pursuit method with a clothoid fitting and a fuzzy controller for autonomous vehicles," *Int. J. Adv. Robotic Syst.*, vol. 12, no. 9, pp. 1–13, 2015.

[17] G. M. Hoffmann, C. J. Tomlin, M. Montemerlo, and S. Thrun, "Autonomous automobile trajectory tracking for off-road driving: Controller design, experimental validation and racing," in *Proc. IEEE Amer. Control Conf.*, 2007, pp. 2296–2301.

[18] Q. Zhu, Z. Huang, D. Liu, and B. Dai, "An adaptive path tracking method for autonomous land vehicle based on neural dynamic programming," in *Proc. IEEE Int. Conf. Mechatronics Automat.*, 2016, pp. 1429–1434.

[19] M. Park, S. Lee, and W. Han, "Development of steering control system for autonomous vehicle using geometry-based path tracking algorithm," *Etri J.*, vol. 37, no. 3, pp. 617–625, 2015.

[20] A. Ollero and O. Amidi, "Predictive path tracking of mobile robots. application to the CMU Navlab," in *Proc. 5th Int. Conf. Adv. Robot.*, 1991, vol. 91, pp. 1081–1086.

[21] I. Prodan *et al.*, "Predictive control for path-following. from trajectory generation to the parametrization of the discrete tracking sequences," in *Developments in Model-Based Optimization and Control*. Berlin, Germany: Springer, 2015, pp. 161–181.

[22] T. Tomatsu, K. Nonaka, K. Sekiguchi, and K. Suzuki, "Model predictive trajectory tracking control for hydraulic excavator on digging operation," in *Proc. IEEE Conf. Control Appl.*, 2015, pp. 1136–1141.

[23] C. E. Beal, *Applications of Model Predictive Control to Vehicle Dynamics for Active Safety and Stability*. Stanford, CA, USA: Stanford Univ. Press, 2011.

[24] J. Mattingley and S. Boyd, "CVXGEN: A code generator for embedded convex optimization," *Optim. Eng.*, vol. 13, no. 1, pp. 1–27, 2012.

[25] J. M. Snider *et al.*, "Automatic steering methods for autonomous automobile path tracking," Robotics Institute, Pittsburgh, PA, USA, Tech. Rep. CMU-RITR-09-08, 2009.

[26] R. Sharp, D. Casanova, and P. Symonds, "A mathematical model for driver steering control, with design, tuning and performance results," *Vehicle Syst. Dyn.*, vol. 33, no. 5, pp. 289–326, 2000.

[27] S. E. Shladover, "Automatic vehicle control developments in the path program," *IEEE Trans. Veh. Technol.*, vol. 40, no. 1, pp. 114–130, Feb. 1991.

[28] T. Kim and H. J. Kim, "Path tracking control and identification of tire parameters using on-line model-based reinforcement learning," in *Proc. IEEE 16th Int. Conf. Control, Automat. Syst.*, 2016, pp. 215–219.

[29] G. Garimella, J. Funke, C. Wang, and M. Kobilarov, "Neural network modeling for steering control of an autonomous vehicle," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 2609–2615.

[30] C. Chen, Y. Jia, M. Shu, and Y. Wang, "Hierarchical adaptive path-tracking control for autonomous vehicles," *IEEE Trans. on Intell. Transp. Syst.*, vol. 16, no. 5, pp. 2900–2912, Oct. 2015.

[31] S. A. Ahmed and M. G. Petrov, "Trajectory control of mobile robots using type-2 fuzzy-neural pid controller," *IFAC-PapersOnLine*, vol. 48, no. 24, pp. 138–143, 2015.

[32] A. Bakdi, A. Hentout, H. Boutami, A. Maoudj, O. Hachour, and B. Bouzouia, "Optimal path planning and execution for mobile robots using genetic algorithm and adaptive fuzzy-logic control," *Robot. Auton. Syst.*, vol. 89, pp. 95–109, 2017.

[33] P. Jardine, "A reinforcement learning approach to predictive control design: Autonomous vehicle applications," Ph.D. dissertation, Dept. Elect. Comput. Eng., Queens Univ. Kingston, Ontario, Canada 2018.

[34] N. R. Kapania and J. C. Gerdes, "Path tracking of highly dynamic autonomous vehicle trajectories via iterative learning control," in *Proc. IEEE Amer. Control Conf.*, 2015, pp. 2753–2758.

[35] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.

[36] C. G. Serna and Y. Ruichek, "Dynamic speed adaptation for path tracking based on curvature information and speed limits," *Sensors*, vol. 17, no. 6, 2017, Art. no. 1383.

[37] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Proc. Conf. Robot. Learn.*, 2017, pp. 1–16.

[38] M. Yamakado, K. Nagatsuka, and J. Takahashi, "A yaw-moment control method based on a vehicless̈ lateral jerk information," *Vehicle Syst. Dyn.*, vol. 52, no. 10, pp. 1233–1253, 2014.

[39] J. Eriksson and L. Svensson, *Tuning for Ride Quality in Autonomous Vehicle*. Uppsala, Sweden: Uppsala University, 2015.

[40] S. Thrun *et al.*, "Stanley: The robot that won the DARPA grand challenge," *J. Field Robot.*, vol. 23, no. 9, pp. 661–692, 2006.

[41] R. Rajamani, *Vehicle Dynamics and Control. Springer*, 2011.

Yunxiao Shan received the M.Sc. degree in machinery manufacturing and automation from the Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun, China, and Ph.D. degree in photogrammetry and remote sensing from Wuhan University, Wuhan, China, in 2010 and 2018, respectively. From 2015 to 2016, he trained at Rutgers University, New Brunswick, NJ, USA. He currently works as an Associate Researcher with the School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China. His research interests include the planning and control of autonomous vehicles, water surface intelligence, and research on autonomous surface vehicles. He has contributed more than 20 publications on the research of autonomous systems.

Boli Zheng received B.S. degree in computer science and technology from Guangzhou University, Guanghzhou, China, in 2018. He is currently working toward the M.S. degree in computer technology with School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China. His research of interests focus on motion planning and the control of autonomous vehicles.

Longsheng Chen received the B.Sc. degree in communication engineering from South China Agricultural University, Guangzhou, China, and the M.Sc. degree in software engineering from Sun Yet-sen University, Guangzhou, China, in 2017 and 2019, respectively. His research interests focus on the control of autonomous vehicles.

Long Chen (Senior Member, IEEE) received the B.Sc. degree in communication engineering and the Ph.D. degree in signal and information processing from Wuhan University, Wuhan, China, in 2007 and 2013, respectively. From October 2010 to November 2012, he was a Ph.D. Student at the National University of Singapore. He is currently an Associate Professor with the School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China. He received the IEEE Vehicular Technology Society 2018 Best Land Transportation Paper Award, IEEE Intelligent Vehicle Symposium 2018 Best Student Paper Award, and Best Workshop Paper Award. His areas of interest include autonomous driving, robotics, and artificial intelligence, where he has contributed more than 70 publications. He serves as an Associate Editor for IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS IEEE Technical Committee on Cyber-Physical Systems newsletter, and Guest Editor for IEEE TRANSACTIONS ON INTELLIGENT VEHICLES, IEEE INTERNET OF THINGS JOURNAL.

De Chen received the M.Eng. degree in mechanical manufacture and automation from China Three Gorges University, Yichang, China, in 2014. He has been the Head of the Pre-Research Department of Research-Institute, Guangdong Lyric Robot Automation Company Limited, China. His research interest includes environmental perception, path planning and tracing of unmanned ground vehicle.