

Article

Path Tracking for Car-like Robots Based on Neural Networks with NMPC as Learning Samples

Guoxing Bai ^{1,2}, Yu Meng ^{1,*}, Li Liu ¹, Qing Gu ^{1,2}, Jianxiu Huang ³, Guodong Liang ¹, Guodong Wang ¹, Li Liu ⁴, Xinrui Chang ¹ and Xin Gan ⁵

¹ School of Mechanical Engineering, University of Science and Technology Beijing, Beijing 100083, China
² Shunde Graduate School, University of Science and Technology Beijing, Foshan 528399, China
³ Hohhot Branch, Inner Mongolia Co., Ltd., China Mobile Communications Group, Hohhot 010020, China
⁴ Artificial Intelligence Center, Haomo Technology Co., Ltd., Beijing 100073, China
⁵ SAIC-GM-Wuling Automobile Co., Ltd., Liuzhou 545007, China
* Correspondence: myu@ustb.edu.cn; Tel.: +86-1851-017-3375

Abstract: In the field of path tracking for car-like robots, although nonlinear model predictive control (NMPC) can handle the system constraints well, its real-time performance is poor. To solve this problem, a neural network control method with NMPC as the learning sample is proposed. The design process of this control method includes establishing the NMPC controller based on the time-varying local model, generating learning samples based on this NMPC controller, and training to obtain the neural network controller. The proposed controller is tested by a joint simulation of MATLAB and Carsim and compared with other controllers. According to the simulation results, the accuracy of the NN controller is close to that of the NMPC controller and far better than that of the Stanley controller. In all simulations, the absolute value of displacement error of the NN controller does not exceed 0.2854 m, and the absolute value of heading error does not exceed 0.2279 rad. In addition, the real-time performance of the NN controller is better than that of the NMPC controller. The maximum time cost and average time cost of the NN controller are, respectively, 40.91% and 22.37% smaller than those of the NMPC controller under the same conditions.

Keywords: car-like robot; model predictive control; neural network; path tracking; real-time



Citation: Bai, G.; Meng, Y.; Liu, L.; Gu, Q.; Huang, J.; Liang, G.; Wang, G.; Liu, L.; Chang, X.; Gan, X. Path Tracking for Car-like Robots Based on Neural Networks with NMPC as Learning Samples. *Electronics* **2022**, *11*, 4232. <https://doi.org/10.3390/electronics11244232>

Academic Editor: Felipe Jiménez

Received: 25 November 2022

Accepted: 16 December 2022

Published: 19 December 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Car-like robots are a common type of wheeled mobile robot. For mobile robots, path tracking is often an essential feature. The function of path tracking is to make the robot travel along a reference path and reduce the deviation between its travel path and the reference path. The path tracking of car-like robots has some similarities with the path tracking of driverless vehicles, such as the work of Ortega et al. [1], but it also faces different challenges. In general, car-like robots typically travel at low speeds, but track reference paths that have large curvature and dramatic curvature changes. Therefore, some researchers, such as Naderi Samani et al., are interested in the path tracking of car-like robots and have published some papers on this topic [2–12]. In 2016, Naderi Samani et al. designed controllers for the parallel parking path tracking of car-like robots based on feedback linearization control and sliding mode control, respectively [2]. Hwang proposed a path tracking model and algorithm for car-like robots based on hierarchical variable structure control [3]. Abd Latip and Omar developed proportional-based and proportional-differential-based path tracking controllers for car-like robots and evaluated the performance of both controllers [4]. In 2018, Ballinas et al. proposed an automatic parallel parking system for a car-like robot, using fuzzy control for the path tracking part [5]. Ghaffari and Homaeinezhad also proposed a path tracking method for car-like robots based on fuzzy control [6]. In 2019, Ljungqvist et al. developed a path tracking system for systems with a car-like tractor and two trailers to solve the reversing challenges

of such multi-body systems [7]. Kamran et al. proposed a reinforcement learning-based path tracking algorithm for car-like robots [8]. Wu developed an algorithm for path tracking of car-like robots under saturated input and sudden pulse disturbance [9]. In 2020, Mohan Rayguru et al. designed a sliding mode controller based on a robust observer to achieve path tracking for car-like robots in the presence of missing state quantities and faulty sensors [10]. Prokopyev and Sofronova compare several path tracking methods for car-like robots [11]. In 2021, Elobaid et al. proposed a transverse feedback linearization-based path tracking method for car-like robots [12]. In 2022, Yeom proposed a vehicle-like robot path tracking method using deep neural networks to adjust parameters [13]. There is no doubt that these research efforts are positive. However, these studies of car-like robots have not considered the effect of system constraints.

For the path tracking of driverless vehicles, Gong et al. pointed out that the system constraints, such as the front wheel turning angle constraint and front wheel turning angle speed constraint of the vehicle, have an impact on the path tracking [14]. Recently, several review papers pointed out the advantages of model predictive control (MPC) in dealing with system constraints [15–19]. Further, Bai et al. also pointed out that among multiple types of MPC, nonlinear MPC (NMPC) can more effectively utilize the reference path information in front of the robot to obtain better path tracking accuracy relative to other types of MPC [15,17,18]. Benefiting from the above advantages, NMPC has achieved a vast range of applications in path tracking. In 2017, Wang et al. proposed an MPC path tracking method for field vehicles [20]. Between 2019 and 2022, based on NMPC, Bai et al. attempted to solve the problems faced in path tracking for a variety of mobile devices, such as tractor-trailer vehicles [21], car-like vehicles [22–24], and differential wheeled robots [25,26]. In 2020, Yin et al. designed an NMPC-based path tracking controller using discrete previewed points in the inertial system [27]. Barzegar et al. used NMPC to solve the trajectory tracking problem for driverless vehicles [28]. In 2021, Farag developed an easy-to-port NMPC-based path tracking controller for driverless vehicles [29]. Chen et al. proposed an NMPC-based controller to improve path tracking accuracy and vehicle driving stability [30]. Franco et al. proposed a driverless system that includes lane line recognition and path tracking, where the path tracking is based on NMPC [31]. In 2022, Hang et al. proposed a driverless vehicle control framework that includes longitudinal and lateral control using NMPC [32]. Lee and Choi designed an NMPC-based path tracking controller with integrated braking and steering [33]. These studies also show that NMPC has significant advantages in path tracking accuracy and dealing with system constraints.

However, due to high computation costs, NMPC has the significant drawback of poor real-time performance. Bai et al. investigated the relationship between the parameters of NMPC and real-time performance to solve the real-time problem and pointed out that the real-time performance of the NMPC-based controller is better when the control horizon is small [34,35]. Wang et al. also attempted to optimize the real-time performance of the NMPC-based path tracking controller using continuous/generalized minimum residual (C/GMRES) [36]. Unfortunately, even after optimization, the computation cost of NMPC-based controllers is still high, limiting the application of such controllers. The work of Gómez-Ortega et al. provides further inspiration. In 1994, they pioneered using an NMPC-based controller as a learning sample to train a neural network controller [37]. In 1996, based on their previous work, they proposed a neural network control algorithm for two-wheeled differential robots considering obstacle information [38,39]. In 2019, Liu further investigated the effect of input layer parameters and network type on the performance of the path tracking controller [40]. The performance of these controllers in terms of real-time performance is exhilarating, but it is slightly unfortunate that these controllers are still weaker than the NMPC controller in terms of accuracy. By analyzing the above research, we found that the learning samples they use usually contain information about only a single reference point on the reference path. However, NMPC can consider information about multiple previewed points on the reference path in front of the robot. The difference

in the number of reference points may be the reason for the inferior accuracy performance of these neural network controllers compared to the NMPC-based controllers.

Therefore, we conducted the following study to obtain a path tracking control method that can combine both accuracy and real-time. First, we analyzed the relationship between the reference point in front of the robot and the coordinate system. Second, we built NMPC-based path tracking controllers as learning samples. Finally, we designed the input and output of the neural network, built the neural network controller, and tested it with a joint simulation of MATLAB and Carsim. In summary, our contribution in this paper is to propose a neural network path tracking control method capable of considering multiple reference points on the forward reference path of a car-like robot.

2. Reference Path and Coordinate System

For path tracking, the reference path is very important. Most current research on path tracking gives the reference path in the global coordinate system. However, as shown in the upper part of Figure 1, the heading and coordinates of the reference path may be different even if the relative positions of the car-like robot and the reference path are the same. These differences are not conducive to finding a correspondence between the reference path and the control inputs generated by the path tracking controller. Converting the reference path from the global coordinate system into the car-like robot coordinate system can solve the above problem. As shown in the lower part of Figure 1, the two reference paths on the left and right sides of Figure 1, which are entirely different in the global coordinate system, are the same when transferred into the robot coordinate system.

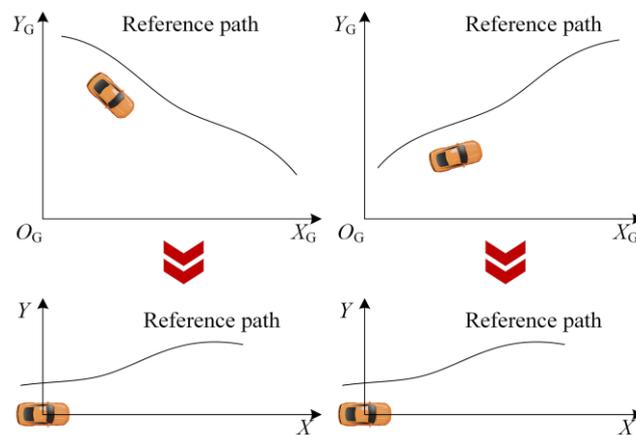


Figure 1. The schematic diagram of the reference path in different coordinate systems.

Therefore, the reference points on the reference path that need to be considered in each control period need to be transferred to the robot coordinate system. The starting point of these reference points is the closest point on the reference path to the car-like robot. The reference points are then taken further ahead of the car-like robot. The length of the reference path between each reference point and the next reference point is equal to the product of the set speed of the car-like robot and the control period.

Then, the reference points in the global coordinate system can be obtained:

$$\mathbf{x}_{rG}(i|t) = [X_{rG}(i|t) \quad Y_{rG}(i|t) \quad \theta_{rG}(i|t)]^T, \quad i = 1, 2, \dots, n \tag{1}$$

where \mathbf{x} is the reference point vector, X is the horizontal coordinate, Y is the vertical coordinate, θ is the heading angle, the subscript r indicates the information on the reference path, and the subscript G indicates the information in the global coordinate system, $(i|t)$ denotes the i -th message at moment t , and n denotes the prediction horizon of the NMPC controller.

In each control period, the positioning system provides the coordinates and heading angle of the car-like robot in the global coordinate system, so the position information of the car-like robot can be expressed as:

$$\mathbf{x}_G(t) = [X_G(t) \ Y_G(t) \ \theta_G(t)]^T \tag{2}$$

Then, the coordinates of the reference points in (1) are transformed:

$$\begin{cases} X_r(i|t) = (X_{rG}(i|t) - X_G(t)) \cos \theta_G(t) + (Y_{rG}(i|t) - Y_G(t)) \sin \theta_G(t) \\ Y_r(i|t) = (Y_{rG}(i|t) - Y_G(t)) \cos \theta_G(t) - (X_{rG}(i|t) - X_G(t)) \sin \theta_G(t) \\ \theta_r(i|t) = \theta_{rG}(i|t) - \theta_G(t) \end{cases}, \quad i = 1, 2, \dots, n \tag{3}$$

The reference points in the car-like robot coordinate system can be abstracted as:

$$\mathbf{x}_r(i|t) = [X_r(i|t) \ Y_r(i|t) \ \theta_r(i|t)]^T, \quad i = 1, 2, \dots, n \tag{4}$$

3. NMPC Controller Based on the Time-Varying Local Model

The NMPC controller design process is well known, so this section will be as brief as possible. The basis of all NMPC controllers is a mathematical model, and the mathematical model used here is a two-degree-of-freedom dynamics model of the car-like robot, where the input to the model is the front wheel turning angle:

$$\mathbf{u} = [\delta] \tag{5}$$

The output is:

$$\mathbf{x} = [X \ Y \ \theta \ v_y \ \omega]^T \tag{6}$$

where v_y is the lateral speed and ω is the heading angle speed.

A more detailed modeling process can be found in our previous work [24], the tire model of the car-like robot model uses the Magic Formula [41,42], and the final abstract model can be expressed as follows:

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) \tag{7}$$

The prediction model for the NMPC controller is obtained by discretizing (7):

$$\begin{cases} \mathbf{x}(1|t) = \mathbf{x}(0|t) + Tf(\mathbf{x}(0|t), \mathbf{u}(1|t)) \\ \vdots \\ \mathbf{x}(i|t) = \mathbf{x}(i-1|t) + Tf(\mathbf{x}(i-1|t), \mathbf{u}(i|t)) \\ \vdots \\ \mathbf{x}(m|t) = \mathbf{x}(m-1|t) + Tf(\mathbf{x}(m-1|t), \mathbf{u}(m|t)) \\ \vdots \\ \mathbf{x}(n|t) = \mathbf{x}(n-1|t) + Tf(\mathbf{x}(n-1|t), \mathbf{u}(n|t)) \end{cases} \tag{8}$$

where m is the control horizon, T is the control period, and in particular, $\mathbf{x}(0|t)$ is the starting pose of the prediction model.

The most significant difference between the NMPC controller based on the time-varying local model and the traditional NMPC controller is the difference in the starting poses of the prediction model. In the traditional NMPC controller, the starting pose is the current pose of the car-like robot in the global coordinate system. In contrast, in the NMPC controller based on the time-varying local model, the starting pose is:

$$\mathbf{x}(0|t) = [0 \ 0 \ 0 \ v_y(t) \ \omega(t)]^T \tag{9}$$

where $v_y(t)$ is the current lateral speed of the car-like robot and $\omega(t)$ is the current heading angle speed of the car-like robot.

Bringing (9) into (8), we can obtain all the predicted positional states of the car-like robot in the prediction horizon. Since the reference path has only three-dimensional information, the output vector of (9) can be set as:

$$\mathbf{x}_o(i|t) = [X_o(i|t) \quad Y_o(i|t) \quad \theta_o(i|t)]^T, \quad i = 1, 2, \dots, n \tag{10}$$

where the subscript o indicates the output information.

The relationship between the output state and the predicted state is:

$$\mathbf{x}_o(t + i|t) \subset \mathbf{x}(t + i|t), \quad i = 1, 2, \dots, n \tag{11}$$

The optimization objective function of the NMPC controller can be designed as:

$$\begin{aligned} \min_{\mathbf{u}(i|t)} J &= \sum_{i=1}^n \|\mathbf{x}_o(i|t) - \mathbf{x}_r(i|t)\|_{\mathbf{Q}}^2 + \sum_{i=0}^{m-1} \|\mathbf{u}(i+1|t) - \mathbf{u}(i|t)\|_{\mathbf{R}}^2 \\ \text{s.t.} \quad & -\delta_{\text{lim}} \leq \delta(i) \leq \delta_{\text{lim}} \\ & -T\dot{\delta}_{\text{lim}} \leq \Delta\delta(i) \leq T\dot{\delta}_{\text{lim}} \end{aligned} \tag{12}$$

where \mathbf{Q} and \mathbf{R} are weight matrices and the subscript lim denotes the limit.

The first value of the input sequence obtained by solving (12) is the control input in this control period.

4. Neural Network Controller with NMPC as Learning Samples

Due to the success of feed-forward neural networks in numerous fields [43,44], we chose this neural network as the basis for building the controller. The training principle of the neural network path tracking controller is shown in Figure 2.

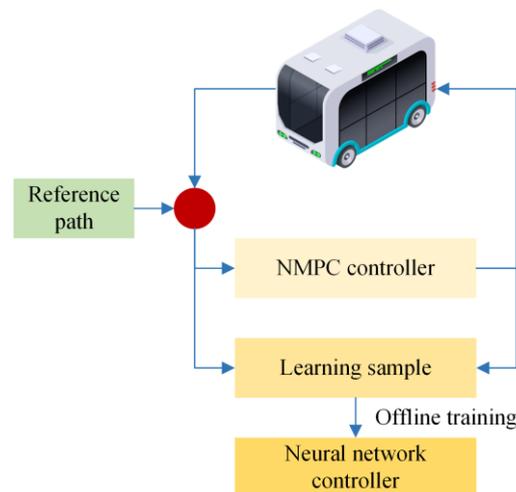


Figure 2. The training principle of the neural network path tracking controller.

The NMPC controller in the joint simulation of MATLAB and Carsim generates the learning samples. The input and output of the neural network are the same as those of the NMPC controller. The information entered into the NMPC controller includes the front wheel turning angle before the start of the control period and the positional state of reference points on the reference path in the robot coordinate system:

$$\xi = [\delta(0|t) \quad \mathbf{x}_r(1|t)^T \quad \mathbf{x}_r(2|t)^T \quad \dots \quad \mathbf{x}_r(i|t)^T \quad \dots \quad \mathbf{x}_r(n|t)^T]^T \tag{13}$$

The output information is the front wheel turning angle generated during that control period:

$$\zeta = [\delta(1|t)] \tag{14}$$

Considering that a neural network with two hidden layers can theoretically fit arbitrary functions [45], we can set the number of hidden layers of the neural network to 2. The structure of the neural network controller is shown in Figure 3.

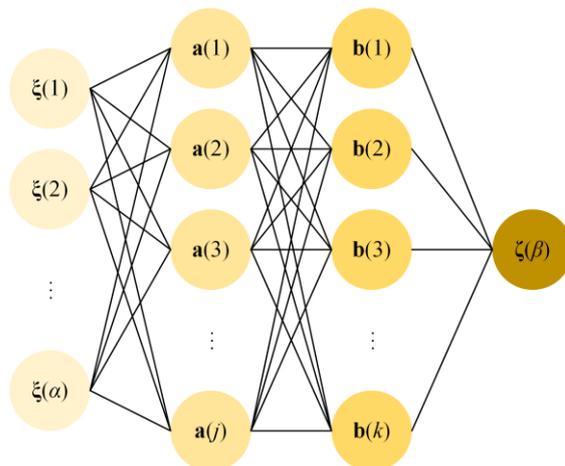


Figure 3. The structure of the neural network.

The number of nodes in the hidden layer can be calculated by the empirical formula:

$$\begin{cases} j = \sqrt{\alpha + \beta} + \lambda_j \\ k = \sqrt{\alpha + \beta} + \lambda_k \end{cases} \tag{15}$$

where α is the number of input layer nodes with a value of $3n + 1$, β is the number of output layer nodes with a value of 1, and λ_j and λ_k are the practical adjuster.

In addition, since the neural network controller cannot handle constraints strictly, the range of constraints of the simulated system can be reduced when generating samples to reduce the impact of system constraints on the neural network controller:

$$\begin{cases} \delta_{slim} = \eta_1 \delta_{lim} \\ \dot{\delta}_{slim} = \eta_2 \dot{\delta}_{lim} \end{cases} \tag{16}$$

where η_1 and η_2 are range reduction factors.

5. Joint Simulation and Results

The joint simulation software includes MATLAB R2020a and Carsim 2017.1. Carsim provides the controlled car-like robot model with the parameters shown in Table 1. The simulation operates on a Dell G15 5510 laptop computer with an Intel(R) Core(TM) i5-10200H 2.40 GHz processor, 16 GB of RAM, and 512 GB of a solid-state drive. The control period is set to 20 ms.

Table 1. Parameters of the car-like robot.

Parameter	Value	Parameter	Value
Wheelbase	1 m	Distance from the center of mass to the front axle	0.5 m
Mass	1230 kg	Rotational inertia around the vertical direction	1343.1 kg·m ⁻²
δ_{lim}	30°	Ground adhesion coefficient	0.8
$\dot{\delta}_{lim}$	15°/s	Tire type	185/65 R15

The simulations are divided into three groups. The first group is used to compare the performance of the proposed controller with other controllers under the same condition, and the second and third groups are used to test the performance of the proposed controller under different conditions.

5.1. Comparison with Other Controllers

The proposed neural network (NN) controller is compared with the NMPC and Stanley controller in this set of simulations. The parameters of the NN controller and NMPC controller are shown in Table 2. The data generated by NMPC tracking reference paths are the learning samples of NN. The interval between every two data sets is 0.05 s, and there are 884 sets. The gain of the Stanley controller is 2. The speed of the car-like robot is 3 m/s.

Table 2. Parameters of the controllers.

Parameter	Value	Parameter	Value
n	30	m	2
\mathbf{Q}	$diag\{10, 10, 1\}$	\mathbf{R}	[0.01]
j	20	k	10
η_1	0.85	η_2	0.73

Figure 4a shows the reference path, which is a U-shaped curve. U-turns are a common car-like robot work path and are a key component of complex paths. We generate the reference path in the form of a discrete point column, specifically, the coordinates and heading of each reference point on the reference path employing a function whose independent variable is the mileage. Figure 4b shows the driving trajectory of the car-like robot. It can be seen from the figure that both the NN and NMPC controller can control the car-like robot to track the reference path, but the Stanley controller fails.

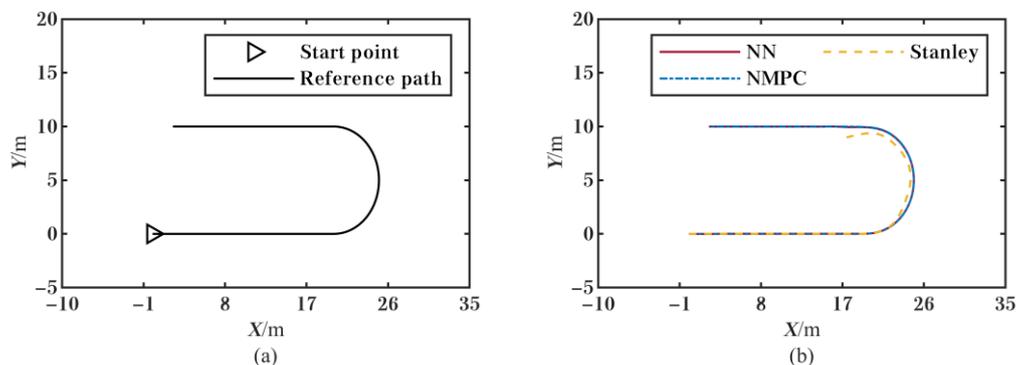


Figure 4. The reference path and the trajectory of the comparison simulation: (a) the reference path; (b) the trajectory.

Figure 5 shows the errors. The NN controller and the NMPC controller have minor errors and eventually converge. The maximum absolute values of displacement error and heading error of the NN controller are 0.1314 m and 0.2186 rad, respectively. These values for the NMPC controller are 0.1181 m and 0.2137 rad. It can be seen that the NN controller has very similar accuracy to the NMPC controller. However, in the same conditions, the displacement error of the Stanley controller is significantly divergent, and the Stanley controller is considered to fail when its displacement error is too large.

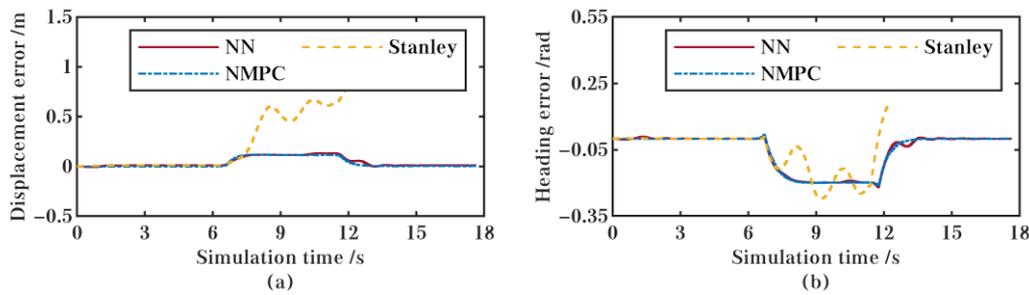


Figure 5. The errors of the comparison simulation: (a) the displacement error; (b) the heading error.

Figures 6 and 7 show the front wheel turning angle (FWTA) and the front wheel turning angle speed (FWTAS), respectively. The FWTA generated by the NN controller is close to that executed by the system. The FWTA generated by the NMPC controller is the same as that executed by the system, and the FWTA generated by the Stanley controller and executed by the system have significant differences. It is further known from the FWTAS that the control inputs generated by the NN controller will slightly exceed the front wheel turning angle speed constraint. The control inputs generated by the NMPC controller will not exceed it, and those generated by the Stanley controller will exceed it significantly. Based on these phenomena, it is clear that the NMPC controller can handle the system constraints well, the NN controller is less affected by the system constraints, and the Stanley controller is highly affected by the system constraints.

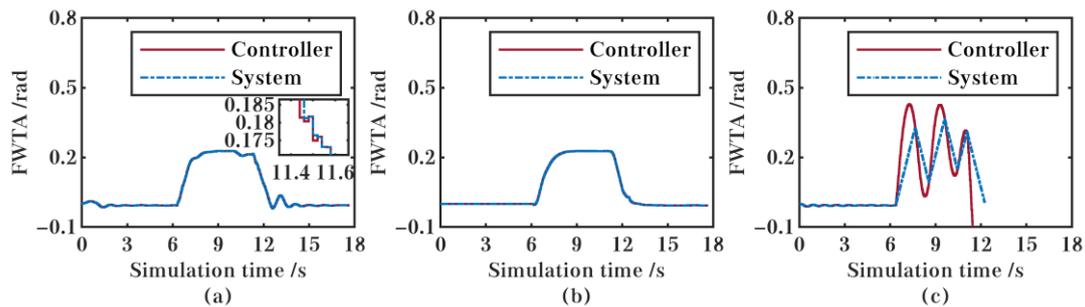


Figure 6. The front wheel turning angle of the comparison simulation: (a) the NN controller; (b) the NMPC controller; (c) the Stanley controller.

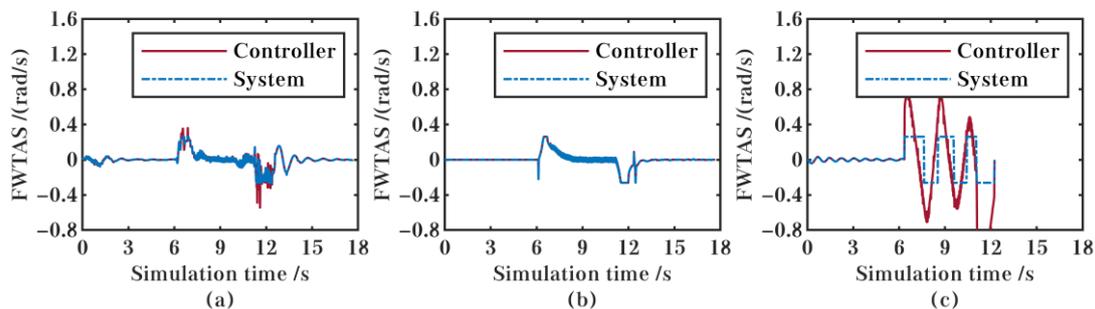


Figure 7. The front wheel turning angle speed of the comparison simulation: (a) the NN controller; (b) the NMPC controller; (c) the Stanley controller.

Figure 8 shows the real-time performance, i.e., the computation time of each controller in each control period and its statistical values. Intuitively, the real-time performance of the NN controller is better than that of the NMPC controller. Specifically, in each control period, the maximum time cost of the NN controller is 13 ms, and the average time cost is 8.19 ms, which are 40.91% and 22.37% less than those values of the NMPC controller, respectively. In addition, by combining Figures 8a and 6, we can see that the NMPC controller has a

considerable time cost during the turn. Even in some control cycles, the time cost exceeds the control period. Although this situation has little impact on the simulation, in the actual control system, it means that the car-like robot cannot receive the control input generated by the controller in that control period. Therefore, there is a significant risk of loss of control. Thus, in summary, the NN controller significantly outperforms the NMPC controller in real-time performance.

Of course, the Stanley controller has better real-time performance than the NN controller. Still, the massive disadvantage in the accuracy of the Stanley controller prevents it from winning the competition with the NN controller.

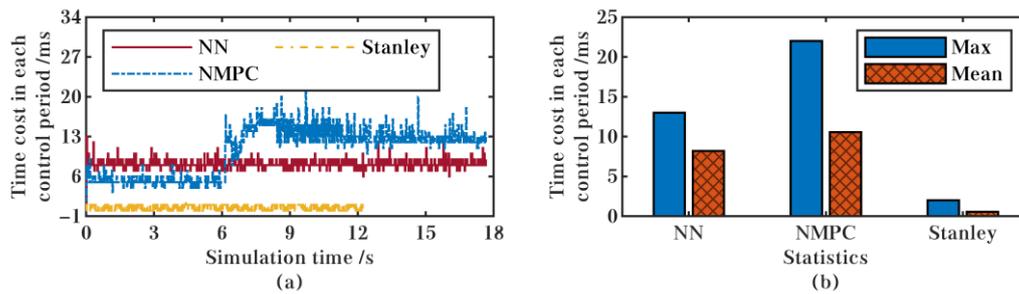


Figure 8. The real-time performance of the comparison simulation: (a) the time-series diagram; (b) the statistical diagram.

5.2. Simulation at Different Speeds

This set of simulations is designed to test the performance of the NN controller at different speeds. The speed of the car-like robot is set to 2 m/s, 3 m/s, and 4 m/s, respectively. Considering the different optimal prediction horizons at different speeds [22], the prediction horizon is set to 28 for a speed of 2 m/s and 50 for a speed of 4 m/s.

The trajectory of the simulation at different speeds is shown in Figure 9. According to Figure 9, the speed variation has a negligible effect on the NN controller.

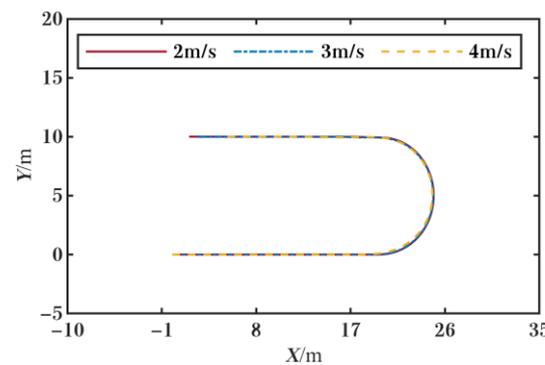


Figure 9. The trajectory of the simulation at different speeds.

The error is shown in Figure 10. The displacement error and heading error of the NN controller are small and eventually converge at different speeds. The absolute value of the displacement error does not exceed 0.2854 m, and the absolute value of the heading error does not exceed 0.2279 rad.

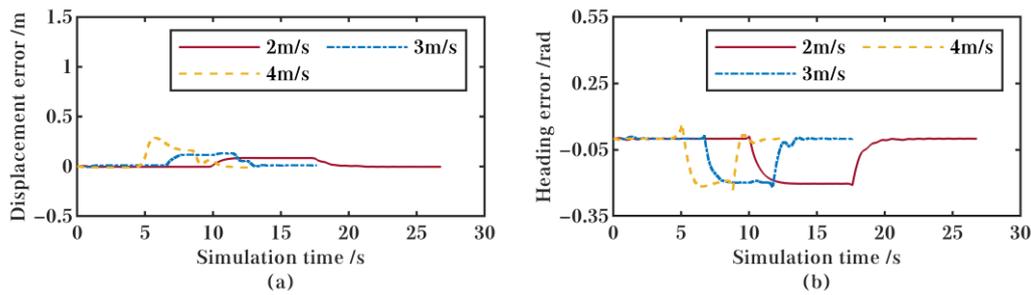


Figure 10. The errors of the simulation at different speeds: (a) the displacement error; (b) the heading error.

Figures 11 and 12 show the FWTA and the FWTAS, respectively. As can be seen in the figure, the NN controller is less affected by the system constraints. In addition, the FWTAS generated from the NN controller is closer to that executed by the system when the speed is 4 m/s. So, it can be inferred that increasing the prediction horizon can further mitigate the effect of the system constraint on the NN controller.

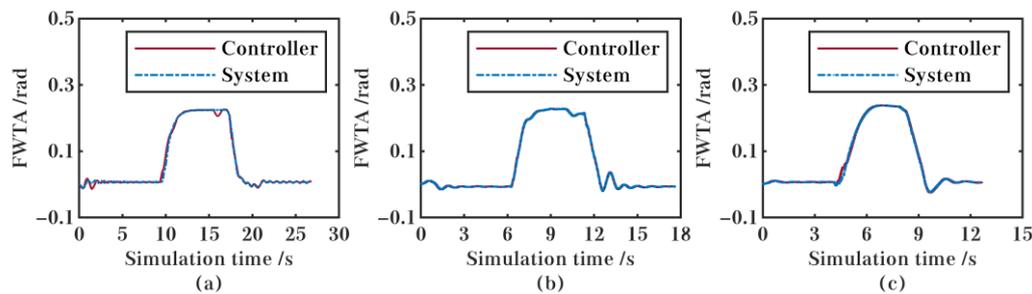


Figure 11. The front wheel turning angle of the simulation at different speeds: (a) at 2 m/s; (b) at 3 m/s; (c) at 4 m/s.

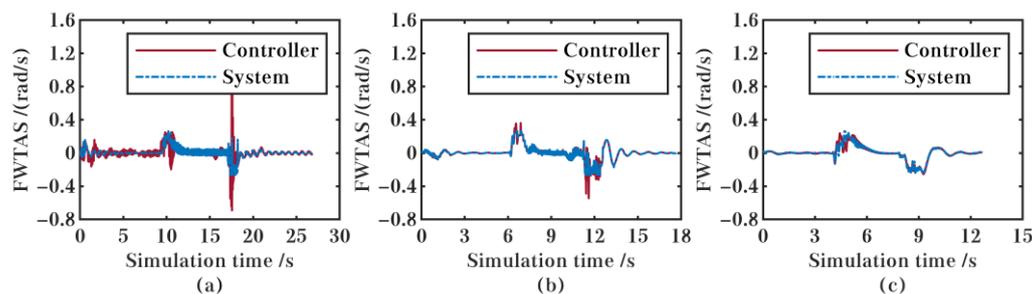


Figure 12. The front wheel turning angle speed of the simulation at different speeds: (a) at 2 m/s; (b) at 3 m/s; (c) at 4 m/s.

Figure 13 shows the real-time performance. As can be seen from the figure, the real-time performance of the NN controller does not vary with speed and is less influenced by the prediction horizon. For three sets of simulations with different speeds, the maximum time cost in each control period is 14 ms, and the maximum value of the average computation cost is 8.487 ms.

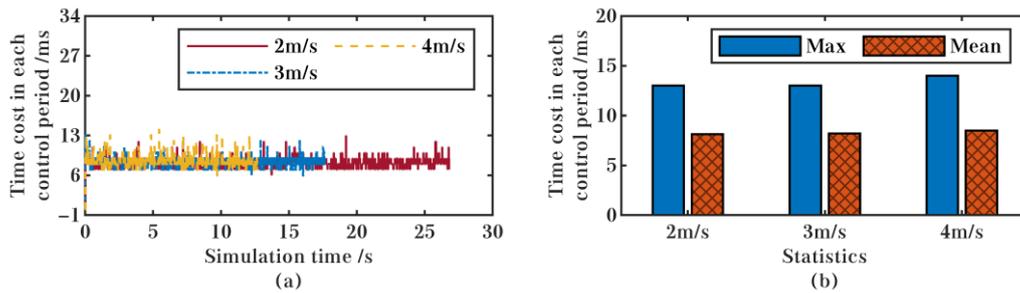


Figure 13. The real-time performance of the simulation at different speeds: (a) the time-series diagram; (b) the statistical diagram.

5.3. Simulation When Tracking the Complex Reference Path

This set of simulations is designed to test the performance of the NN controller when tracking the complex reference path. The learning samples of NN in this set of simulations are the sum of the samples of a right turn and the samples of the first set of simulations. The reference path of the right turn samples is the same as that in the first set of simulations, except that the endpoint is used as the start point. The speed of the car-like robot is set to 2 m/s.

The trajectory of the simulation is shown in Figure 14, and it can be seen that the complex reference path has a negligible effect on the NN controller.

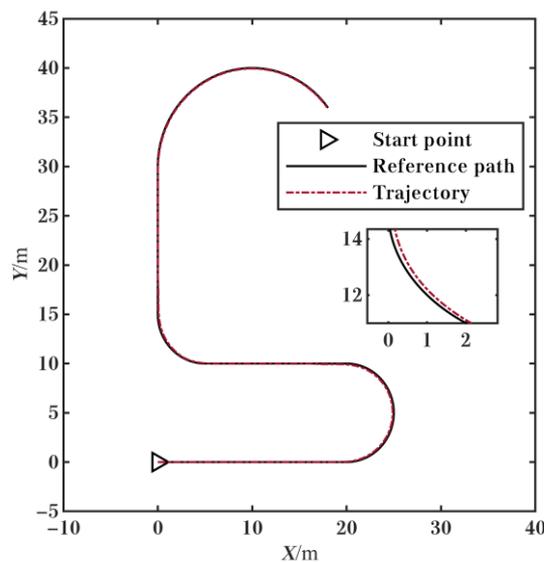


Figure 14. The reference path and the trajectory of the simulation when tracking the complex reference path.

The error is shown in Figure 15. The displacement error and heading error of the NN controller are small and eventually converge at different speeds. The absolute value of the displacement error does not exceed 0.1330 m, and the absolute value of the heading error does not exceed 0.2186 rad.

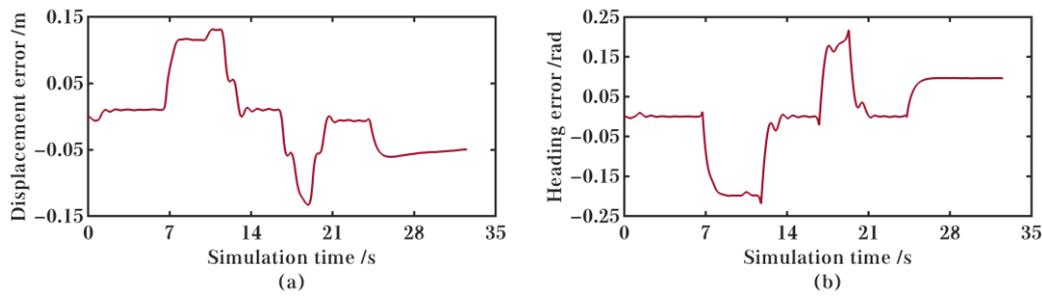


Figure 15. The errors of the simulation when tracking the complex reference path: (a) displacement error; (b) heading error.

Figure 16 shows the FWTA and the FWTAS, respectively. As can be seen in the figure, the NN controller is less affected by the system constraints.

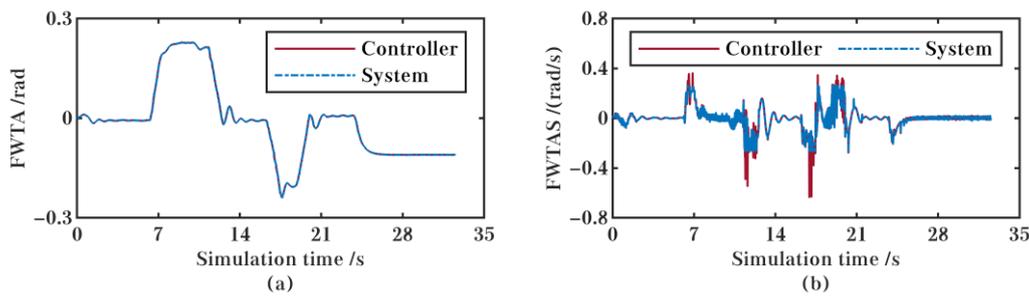


Figure 16. The front wheel turning angle and the front wheel turning angle speed of the simulation when tracking the complex reference path: (a) FWTA; (b) FWTAS.

As shown in Figure 17, the real-time performance of the NN controller is less influenced by the reference path. In this set of simulations, the maximum time cost in each control period is 12 ms and the maximum value of the average time cost is 8.181 ms.

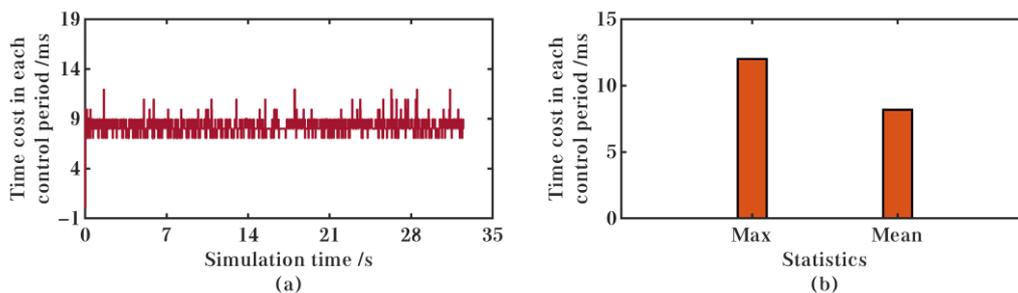


Figure 17. The real-time performance of the simulation when tracking the complex reference path: (a) the time-series diagram; (b) the statistical diagram.

6. Conclusions

To obtain a path tracking controller for car-like robots that can combine both accuracy and real-time, we proposed a neural network control method using an NMPC controller based on the time-varying local model as the learning sample. The controller was tested by joint simulation of MATLAB and Carsim and compared with other controllers. Based on the simulation results, the following conclusions can be obtained.

First, the proposed NN controller has high accuracy. The maximum absolute value of displacement error is 0.2854 m, and the maximum absolute value of heading error is 0.2279 rad in all simulations. The accuracy of the NN controller is very close to that of the NMPC controller under the same condition. Furthermore, the Stanley controller fails in this condition.

Second, the proposed NN controller has excellent real-time performance. In all simulations, the maximum time cost of this controller in each control period is 14 ms, and the average time cost does not exceed 8.487 ms. In the same condition, the maximum time cost and average time cost of the NN controller are 40.91% and 22.37% smaller than those of the NMPC, respectively.

In the near future, we will conduct further research on real-world robot experiments, including improving the robustness of the control method to perturbations and porting the algorithm to a car-like robot platform.

Author Contributions: Conceptualization, G.B., L.L. (Li Liu, qianli12101993@163.com), and X.G.; methodology, G.B., Y.M., L.L. (Li Liu, liliu@ustb.edu.cn), and Q.G.; software, G.B. and J.H.; validation, G.B., G.L., G.W., and X.C.; formal analysis, G.B.; investigation, G.B. and L.L. (Li Liu, qianli12101993@163.com); resources, G.B.; data curation, G.B.; writing—original draft preparation, G.B.; writing—review and editing, Y.M., L.L. (Li Liu, liliu@ustb.edu.cn), and Q.G.; visualization, G.B., J.H., G.L., G.W., and X.C.; supervision, G.B. and Y.M.; project administration, G.B., Y.M., L.L. (Li Liu, liliu@ustb.edu.cn), and Q.G.; funding acquisition, G.B., Y.M., L.L. (Li Liu, liliu@ustb.edu.cn), and Q.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research is funded by the National Natural Science Foundation of China (52202505), the National Key Research and Development Program of China (2019YFC0605300), the China Postdoctoral Science Foundation (2022M710354) and the Fundamental Research Funds for the Central Universities (FRF-TP-20-052A1).

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

- Ortega, J.; Lengyel, H.; Szalay, Z. Overtaking maneuver scenario building for autonomous vehicles with PreScan software. *Transp. Eng.* **2020**, *2*, 100029. [[CrossRef](#)]
- Naderi Samani, N.; Danesh, M.; Ghaisari, J. Parallel parking of a car-like mobile robot based on the P-domain path tracking controllers. *IET Control Theory Appl.* **2016**, *10*, 564–572. [[CrossRef](#)]
- Hwang, C.L. Comparison of path tracking control of a car-like mobile robot with and without motor dynamics. *IEEE ASME Trans. Mechatron.* **2016**, *21*, 1801–1811. [[CrossRef](#)]
- Abd Latip, N.B.; Omar, R. Car-like robot path tracker with kinematic constraints. In Proceedings of the 2016 6th IEEE International Conference on Control System, Computing and Engineering (ICCSCE), Penang, Malaysia, 25–27 November 2016.
- Ballinas, E.; Montiel, O.; Castillo, O.; Rubio, Y.; Aguilar, L.T. Automatic Parallel Parking Algorithm for a Carlike Robot using Fuzzy PD+ I Control. *Eng. Lett.* **2018**, *26*, 447–454.
- Ghaffari, S.; Homaeinezhad, M.R. Autonomous path following by fuzzy adaptive curvature-based point selection algorithm for four-wheel-steering car-like mobile robot. *Proc. Inst. Mech. Eng. C J. Eng. Mech. Eng. Sci.* **2018**, *232*, 2655–2665. [[CrossRef](#)]
- Ljungqvist, O.; Evestedt, N.; Axehill, D.; Cirillo, M.; Pettersson, H. A path planning and path-following control framework for a general 2-trailer with a car-like tractor. *J. Field Robot.* **2019**, *36*, 1345–1377. [[CrossRef](#)]
- Kamran, D.; Zhu, J.; Lauer, M. Learning path tracking for real car-like mobile robots from simulation. In Proceedings of the 2019 European Conference on Mobile Robots (ECMR), Prague, Czech Republic, 4–6 September 2019.
- Wu, H.M. Nonlinear trajectory-tracking control of a car-like mobile robot in the presence of input saturations and a pulse disturbance. In Proceedings of the 2019 58th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE), Hiroshima, Japan, 10–13 September 2019.
- Mohan Rayguru, M.; Rajesh Elara, M.; Ramalingam, B.; Muthugala, M.V.J.; Samarakoon, S.B.P. A path tracking strategy for car like robots with sensor unpredictability and measurement errors. *Sensors* **2020**, *20*, 3077. [[CrossRef](#)]
- Prokopyev, I.; Sofronova, E. Study on synthesis methods for real-time control of car-like mobile robot. In *Artificial Intelligence and Evolutionary Computations in Engineering Systems*; Dash, S., Lakshmi, C., Das, S., Panigrahi, B., Eds.; Springer: Singapore, 2020; pp. 431–441.
- Elobaid, M.; Mattioni, M.; Monaco, S.; Normand-Cyrot, D. Digital path-following for a car-like robot. *IFAC-PapersOnLine* **2021**, *54*, 174–179. [[CrossRef](#)]
- Yeom, K. Design of deep neural network based model predictive controller for a car-like mobile robot. *Int. J. Mech. Eng. Robot. Res.* **2022**, *11*, 606–613. [[CrossRef](#)]
- Gong, J.W.; Xu, W.; Jiang, Y.; Liu, K.; Guo, H.F.; Sun, Y.J. Multi-constrained model predictive control for autonomous ground vehicle trajectory tracking. *J. Beijing. Inst. Technol.* **2015**, *24*, 441–443.
- Bai, G.; Meng, Y.; Liu, L.; Luo, W.; Gu, Q.; Liu, L. Review and comparison of path tracking based on model predictive control. *Electronics* **2019**, *8*, 1077. [[CrossRef](#)]

16. Rokonuzzaman, M.; Mohajer, N.; Nahavandi, S.; Mohamed, S. Review and performance evaluation of path tracking controllers of autonomous vehicles. *IET Intell. Transp. Syst.* **2021**, *15*, 646–670. [[CrossRef](#)]
17. Bai, G.X.; Luo, W.D.; Liu, L.; Meng, Y.; Gu, Q.; Li, K.L. Current status and progress of path tracking control of mining articulated vehicles. *Chin. J. Eng.* **2021**, *43*, 193–204. (In Chinese)
18. Bai, G.X.; Meng, Y.; Liu, L.; Gu, Q.; Wang, G.D.; Zhou, B.N. Current status of path tracking control of unmanned driving vehicles. *Chin. J. Eng.* **2021**, *43*, 475–485. (In Chinese)
19. Zhang, K.; Wang, J.; Xin, X.; Li, X.; Sun, C.; Huang, J.; Kong, W. A survey on learning-based model predictive control: Toward path tracking control of mobile platforms. *Appl. Sci.* **2022**, *12*, 1995. [[CrossRef](#)]
20. Wang, X.; Taghia, J.; Katupitiya, J. Adaptive min-max model predictive control for field vehicle guidance in the presence of wheel slip. In *Robotics and Mechatronics for Agriculture*; Zhang, D., Wei, B., Eds.; CRC Press: Boca Raton, FL, USA, 2017; pp. 157–184.
21. Bai, G.; Liang, C.; Meng, Y.; Liu, L.; Luo, W.; Gu, Q. Obstacle avoidance of semi-trailers based on nonlinear model predictive control. *World Electr. Veh. J.* **2019**, *10*, 72. [[CrossRef](#)]
22. Bai, G.; Meng, Y.; Liu, L.; Gu, Q.; Luo, W.; Gan, X. Path tracking control of vehicles based on variable prediction horizon and velocity. *China Mech. Eng.* **2020**, *31*, 1277–1284. (In Chinese)
23. Bai, G.; Liu, L.; Meng, Y.; Gu, Q.; Wang, G.; Dong, G. Path tracking of vehicles on urban roads based on fuzzy control and NMPC. In Proceedings of the 2021 China Automation Congress (CAC), Beijing, China, 22–24 October 2021.
24. Bai, G.X.; Zhou, L.; Meng, Y.; Liu, L.; Gu, Q.; Wang, G.D. Path tracking of unmanned vehicles based on the time-varying local model. *Chin. J. Eng.* **2022**, *in press*. (In Chinese)
25. Bai, G.; Meng, Y.; Liu, L.; Luo, W.; Gu, Q.; Li, K. Anti-sideslip path tracking of wheeled mobile robots based on fuzzy model predictive control. *Electron. Lett.* **2020**, *56*, 490–493. [[CrossRef](#)]
26. Bai, G.; Meng, Y.; Gu, Q.; Wang, G.; Dong, G.; Zhou, L. An anti-sideslip path tracking control method of wheeled mobile robots. In Proceedings of the International Conference on Intelligent Robotics and Applications, Harbin, China, 1–3 August 2022.
27. Yin, C.; Xu, B.; Chen, X.; Qin, Z.; Bian, Y.; Sun, N. Nonlinear model predictive control for path tracking using discrete previewed points. In Proceedings of the 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), Rhodes, Greece, 20–23 September 2020.
28. Barzegar, A.; Doukhi, O.; Lee, D.J.; Jo, Y.H. Nonlinear model predictive control for self-driving cars trajectory tracking in GNSS-denied environments. In Proceedings of the 2020 20th International Conference on Control, Automation and Systems (ICCAS), Busan, Republic of Korea, 13–16 October 2020.
29. Farag, W. Real-time NMPC path tracker for autonomous vehicles. *Asian J. Control* **2021**, *23*, 1952–1965. [[CrossRef](#)]
30. Chen, L.; Zhou, K.; Teng, C.; Sun, X.; Wang, H. Longitudinal and lateral comprehensive trajectory tracking control of intelligent vehicles based on NMPC. *Autom. Eng.* **2021**, *43*, 153–161. (In Chinese)
31. Franco, I.J.P.B.; Ribeiro, T.T.; Conceição, A.G.S. A novel visual lane line detection system for a NMPC-based path following control scheme. *J. Intell. Rob. Syst. Theor. Appl.* **2021**, *101*, 12. [[CrossRef](#)]
32. Hang, P.; Lou, B.; Lv, C. Nonlinear predictive motion control for autonomous mobile robots considering active fault-tolerant control and regenerative braking. *Sensors* **2022**, *22*, 3939. [[CrossRef](#)]
33. Lee, J.; Choi, S. Nonlinear model predictive control for path tracking in high-speed corner entry situations. *Int. J. Automot. Technol.* **2022**, *23*, 1373–1381. [[CrossRef](#)]
34. Bai, G.X.; Liu, L.; Meng, Y.; Liu, S.Y.; Liu, L.; Luo, W.D. Real-time path tracking of mobile robot based on nonlinear model predictive control. *Trans. Chin. Soc. Agric. Mach.* **2020**, *51*, 47–52, 60. (In Chinese)
35. Bai, G.; Meng, Y.; Gu, Q.; Li, K.; Li, S. Some rules for setting the horizon parameters of NMPC-based vehicle path tracking. In Proceedings of the 2020 Chinese Automation Congress (CAC), Shanghai, China, 6–8 November 2020.
36. Wang, H.W.; Liu, C.Y.; Li, L.; Zhang, H.T. Research on trajectory tracking control of unmanned vehicle based on efficient NMPC algorithm. *Autom. Eng.* **2022**, *44*, 1494–1502, 1618. (In Chinese)
37. Gómez-Ortega, J.; Camacho, E.F. Neural network MBPC for mobile robot path tracking. *Rob. Comput. Integr. Manuf.* **1994**, *11*, 271–278. [[CrossRef](#)]
38. Gómez-Ortega, J.; Camacho, E.F. Neural predictive control for mobile robot navigation in a partially structured static environment. *IFAC Proc.* **1996**, *29*, 8125–8130. [[CrossRef](#)]
39. Gómez-Ortega, J.; Camacho, E.F. Mobile robot navigation in a partially structured static environment, using neural predictive control. *Control Eng. Pract.* **1996**, *4*, 1669–1679. [[CrossRef](#)]
40. Liu, L. Path Tracking Control of Handling Robot Based on Neural Network. Master's Thesis, University of Science and Technology Beijing, Beijing, China, 11 December 2019. (In Chinese).
41. Ji, X.; He, X.; Lv, C.; Liu, Y.; Wu, J. A vehicle stability control strategy with adaptive neural network sliding mode theory based on system uncertainty approximation. *Veh. Syst. Dyn.* **2018**, *56*, 923–946. [[CrossRef](#)]
42. Ji, X.; Yang, K.; Na, X.; Lv, C.; Liu, Y.; Liu, Y. Feedback game-based shared control scheme design for emergency collision avoidance: A fuzzy-linear quadratic regulator approach. *J. Dyn. Syst. Meas. Control Trans. ASME* **2019**, *141*, 081005. [[CrossRef](#)]
43. Chen, Y.; Song, L.; Liu, Y.; Yang, L.; Li, D. A review of the artificial neural network models for water quality prediction. *Appl. Sci.* **2020**, *10*, 5776. [[CrossRef](#)]

-
44. Hemeida, A.M.; Hassan, S.A.; Mohamed, A.A.A.; Alkhalaf, S.; Mahmoud, M.M.; Senjyu, T.; El-Din, A.B. Nature-inspired algorithms for feed-forward neural network classifiers: A survey of one decade of research. *Ain Shams Eng. J.* **2020**, *11*, 659–675. [[CrossRef](#)]
 45. Lippmann, R. An introduction to computing with neural nets. *IEEE ASSP Mag.* **1987**, *4*, 4–22. [[CrossRef](#)]