IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received 21 September 2023, accepted 8 October 2023, date of current version 26 October 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3325326

RESEARCH ARTICLE

A Longitudinal/Lateral Coupled Neural Network Model Predictive Controller for Path Tracking of Self-Driving Vehicle

SIBING YANG AND CONG GENG

Beijing Key Laboratory of Powertrain for New Energy Vehicle, School of Mechanical, Electronic, and Control Engineering, Beijing Jiaotong University, Beijing 100044, China

Corresponding author: Cong Geng (cgeng@bjtu.edu.cn)

This work was supported by the National Key Research and Development Program under Grant 2017YFB103701.

ABSTRACT In recent years, the model predictive control (MPC) algorithm has been increasingly applied to the path tracking of self-driving vehicles due to its capacity to deal with dynamic constraints explicitly. The control performance of MPC is highly dependent on the accuracy dynamic model; however, as vehicles are strongly coupled nonlinear systems, the prediction accuracy of the classical mechanism model decreases significantly at high-speed conditions, leading to increased control errors. This paper proposes replacing the classical mechanism model with a recurrent neural network (RNN) for vehicle dynamical state prediction under the framework of MPC to achieve higher control effects under high speed steering processes. The RNN vehicle dynamic model uses historical data of control and state variables to predict future states. Based on this novel model, longitudinal/lateral coupled model predictive control is realized. The differential evolution algorithm is proposed to solve the optimization problem in the controller. Finally, the prediction accuracy of the RNN model is verified on the real vehicle dataset and compared with linear/nonlinear mechanism models. The control algorithm proposed in this paper is compared with classical MPC against low and high speeds (10m/s and 30m/s) on the ADAMS/Python/Simulink joint simulation platform. The results show that the control accuracy and stability of the longitudinal/lateral coupled neural network MPC are higher than classical MPC, especially at high speed.

INDEX TERMS Model predictive control (MPC), recurrent neural network (RNN), longitudinal/lateral coupled control, path tracking.

I. INTRODUCTION

With the rapid development of the automobile industry, the increase in traffic accidents has become an important social problem. Among them, when a traffic accident occurs at high speed, it is difficult for the driver to make an adequate response, which often brings more severe consequences. Self-driving technology has developed rapidly in recent years, and how to use self-driving technology to reduce traffic accidents has become a hot spot for research [1], [2], [3], [4], [5].

Self-driving technology can be mainly divided into four parts: environment sensing, decision making, motion planning, and path tracking. The path tracking control module receives the reference path output from the path planning module and calculates the control variables of the chassis actuator through the control algorithm. Ultimately, the vehicle's driving stability is ensured while minimizing the tracking error [6].

The lateral control of self-driving vehicles has been studied since the 1950s and continued until now, and the control methods have gone through classical control, modern control, and intelligent control [7], [8], [9], [10], [11]. Among them, model predictive control has become a hot research topic due to its strong robustness and ability to provide vehicle control and state constraints explicitly [12], [13], [14], [15], [16], [17], [18]. Yu et al. [19] introduced a novel framework for path tracking control in autonomous vehicles, which integrates tube model predictive control with time-delayed

The associate editor coordinating the review of this manuscript and approving it for publication was Wei Quan.

motion prediction. Within this study, they presented a method for handling path tracking in the presence of signal delays and employed a vehicle kinematics model to anticipate alterations in vehicle position and yaw. Cheng et al. [20] developed a dynamic model that accounts for nonlinear tire characteristics and varying vehicle speeds. Using this model, they designed an MPC-based vehicle path tracking controller capable of robustly handling uncertainties in system parameters.

However, there are still some unsolved problems in the field of model predictive control of vehicle path tracking. A large number of the path tracking control algorithms use a linearized two-degree-of-freedom or three-degree-of-freedom bicycle model. The prediction accuracy will be greatly reduced due to the nonlinearity of the vehicle significantly increasing in extreme working conditions such as high speeds [21]. Therefore, some researchers started to use nonlinear models directly in the MPC algorithm for state prediction without linearization [22], [23], [24]. It makes a higher accuracy of the state prediction, meanwhile still brings some problems. On the one hand, it takes a longer time in the procedure of optimization [25], which brings a significant impact on the real-time performance of the algorithm. On the other hand, The wire control chassis of the intelligent vehicle is a complex system with electro-mechanical-hydraulic coupling and multiple nonlinear components. It is hard for the mechanism model to express all the nonlinear features of the real vehicle. Moreover, most of the path tracking control algorithms use the front wheel angle and tire force as the control variables. Still, there is a complex transfer function from these variables to the actual actuator, which is challenging to express explicitly. It leads to a decrease in control accuracy [26].

Additionally, The vehicle is a complex, strongly coupled, variable parameter nonlinear system. Its longitudinal and lateral movements are coupled and interact with each other. Especially when the vehicle travels at high speed on the uneven road surface, its coupling characteristics are more obvious. In the past, the longitudinal and lateral control of the vehicle was accomplished by two independent controllers without considering the coupling influence, which led to a significant controller error and affected the control accuracy [27].

A number of scholars have been working on the above problems. Among them, algorithms combining neural network models with model predictive control have great advantages in dealing with nonlinear control problems. Spielberg and colleagues [28], [29] introduced a neural network MPC system. Remarkably, this system outperformed the conventional counterparts that relied on physics-based predictive models, despite not utilizing any tire-pavement friction information. Rokonuzzaman and his team [24] constructed a feed-forward neural network with two hidden layers. This network was designed to forecast the time derivatives of lateral slip velocity and yaw speed. The integral of the network's output was employed to generate the system state. Utilizing this network architecture, they

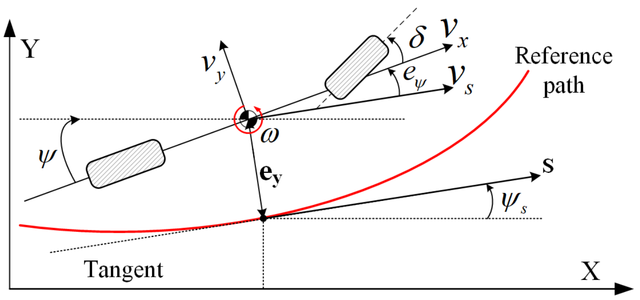

FIGURE 1. Vehicle dynamics model in a road-aligned coordinate frame.

introduced a novel Inverse Optimal Control (IOC) algorithm. Montanaro et al. [30] offered an outline of the state estimator framework and sensor fusion system utilized in path tracking algorithms based on connected vehicle Model Predictive Control (MPC). They integrated a system that gathers data regarding the communication with the leading vehicle. Consequently, this information can be shared with the trailing vehicle, providing crucial insights into the tire-road friction coefficient ahead.

Although these studies demonstrated improved performance using the NNMPC, they are only tested at relatively low speeds with different friction surfaces, the effects of vehicle lateral-longitudinal coupling effects are not considered, and use simpler controllers in the longitudinal direction. In addition, the front wheel angle and the ground tire force are used as control variables in these studies. However, the tracking accuracy will be degraded since the actuator system has nonlinear characteristics, which are difficult to model.

This paper uses the data generated from the driving process to establish a deep learning vehicle dynamics model, and based on this, a longitudinal/ lateral coupled model predictive controller for path tracking is realized. The configuration of this paper is as follows. Section II mainly describes the establishment process of the RNN vehicle dynamics model. In section III, a path tracking controller that realizes the lateral/longitudinal coupled control based on the prediction of the RNN model is established, and a differential evolution algorithm is introduced to solve the optimization problem. In Section IV, the model built in sectionII is firstly verified using real car datasets and compared with the mechanism model. The path tracking effect of the controller built in sectionIII is verified at low and high speeds. Section V presents the conclusion of this paper.

II. DYNAMIC MODELING

A. MECHANISM DYNAMIC MODELING

A mechanism-based model, which is currently the most used, is established as a benchmark [31], [32], [33]. The mechanism-based model is first analyzed to illustrate its limitations at the theoretical level. In the later part of this paper, the MPC algorithm based on the RNN model is compared with the MPC algorithm based on this mechanism model. Figure1 is a vehicle dynamics model considering only the longitudinal, lateral, and yaw, ignoring the load transfer

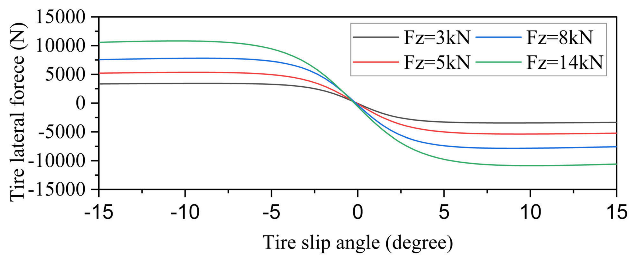

FIGURE 2. Lateral force of tires under different vertical loads.

and aerodynamics in each direction. The body dynamics equilibrium equations in the three degrees can be expressed as equation(1):

$$\begin{aligned} m\dot{v}_x &= m\dot{y}\omega + 2(F_{xf}\cos\delta - F_{yf}\sin\delta) + 2F_{xr} \\ m\dot{v}_y &= -m\dot{x}\omega + 2(F_{xf}\sin\delta + F_{yf}\cos\delta) + 2F_{yr} \\ I_z\dot{\omega} &= 2a(F_{xf}\sin\delta + F_{yf}\cos\delta) - 2bF_{yr} \end{aligned} \tag{1}$$

where v_x/v_y is the longitudinal/lateral speed of the vehicle. ω is the yaw speed. δ is the equivalent front wheel steering angle. a, b denotes the distance from the center of mass to the front and rear axles, respectively. $F_{xf}, F_{xr}, F_{yf}, F_{yr}$ are the longitudinal and lateral forces on tires of front and rear axles. m is the gross mass of the vehicle. I_z is the vehicle moment of inertia about the z-axis.

Using the magic formula Proposed by Pacejka to calculate the "lateral force-tire slip angle" curves at different loads (3KN, 5KN, 8KN, 14KN) for the 235-40R18 tires, which will be used in the subsequent study, the result is shown in the figure2.

Under the small angle assumption, the front and rear tire slip angle are assumed to be approximately equal to 0, the tire force in the x-direction is approximately equal to the tire longitudinal force, and the tire force in the y-direction is approximately equal to the tire lateral force. The tire lateral force can be expressed as a linear function of the tire slip angle, i.e.:

$$F_c = \bar{C}_\alpha \alpha \tag{2}$$

where $\bar{C}_\alpha$ is linear tire cornering stiffness, α is tire slip angle.

Combining equation(1) and equation(2), under the small angle assumption, the nonlinear dynamic model of the vehicle can be established as:

$$\begin{aligned} m\dot{v}_x &= mv_y\omega + 2[\bar{C}_{lf}s_f + \bar{C}_{\alpha f}(\delta - \frac{v_y + l_f\omega}{v_x})\delta + \bar{C}_{lr}s_r] \\ m\dot{v}_y &= -mv_x\omega + 2[\bar{C}_{\alpha f}(\delta - \frac{v_y + l_f\omega}{v_x}) + \bar{C}_{\alpha r}\frac{l_r\omega - v_y}{v_x}] \\ I_z\dot{\omega} &= 2[l_f\bar{C}_{\alpha f}(\delta - \frac{v_y + l_f\omega}{v_x}) - l_f\bar{C}_{\alpha r}\frac{l_r\omega - v_y}{v_x}] \\ \dot{X} &= v_x\cos\varphi - v_y\sin\varphi \\ \dot{Y} &= v_x\sin\varphi + v_y\cos\varphi \end{aligned} \tag{3}$$

where$\bar{C}_{lf}$ and $\bar{C}_{lr}$ are longitudinal stiffness of front/ rear tire. s_f and s_f are indicates tire slip rate.

It can be seen from figure2: the tire lateral force curve can be seen as linear when the tire slip angle is small. But when the vehicle is driven at high speed, the tire slip angle will further increase, resulting in the tire lateral force showing nonlinear characteristics. In addition, the vehicle is accompanied by constant load transfer during operation, which affects the tire lateral force. It is difficult for the mechanism model to model this nonlinear characteristic.

In addition, by observing equation(3), it can be found that all three output parameters of the vehicle system are not determined by a single directional input only. There are complex coupling relationships. Moreover, for dynamic coupling effects, such as the lateral force of a tire, there is a component in the longitudinal direction that affects the longitudinal acceleration. Lateral and longitudinal forces in tires are also coupled to each other. For a given coefficient of friction of the tire pavement, the lateral and longitudinal forces acting on each tire limit each other so that the combined force does not exceed the attachment limit. Load transfer also has a significant coupling effect. When load transfer is caused by longitudinal acceleration, the redistribution of vertical loads between the front and rear tires will significantly affect lateral dynamics. It can be concluded that the longitudinal and lateral coupling is firm and not easy to decouple by the analytic method.

In order to solve the above problems, this paper takes advantage of the neural network model in nonlinear modeling to replace the mechanism model in traditional model predictive control for state prediction and achieve higher prediction accuracy. In addition, the longitudinal and lateral of the vehicle are predicted with the same neural network, which in turn establishes a controller to realize coupled control of the vehicle in the longitudinal and lateral.

B. RECURRENT NEURAL NETWORK DYNAMICS MODELING

Hornik et al. proposed the "universal approximation theorem" in 1989. This principle indicates that "a feedforward neural network with a linear output layer and at least one hidden layer with an activation function that is 'squeezed' in nature can approximate any function from one finite-dimensional space to another with arbitrary accuracy." Throughout the training process, a prediction error propagates backward, and the loss function continues to decline in the direction of the gradient to reach the local optimal value or a small value meeting the demand.

Due to the universal fitting capacity of neural networks for nonlinear systems and the fact that they do not require complex modeling mechanism studies of the system, they have been increasingly used for the establishment of various nonlinear models in recent years [34], [35], [36]. Compared with the vehicle dynamics mechanism model, after a large number of assumptions and simplifications, the neural network-based dynamics model makes full use of the dynamics data during the vehicle driving process and is modeled based on machine learning, which avoids the need to study the complex electromechanical-liquid coupling

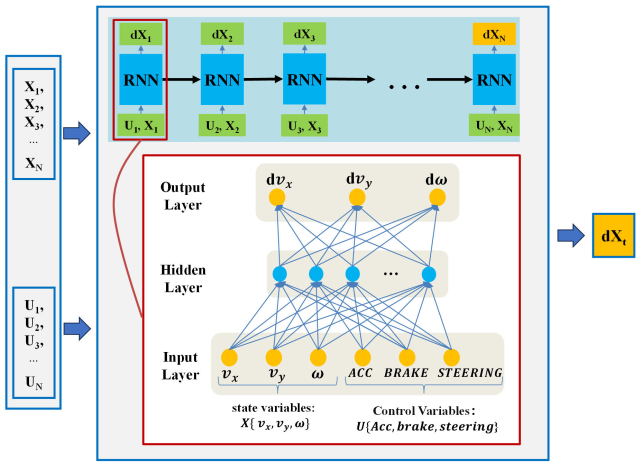

FIGURE 3. Structure of RNN vehicle dynamics prediction model. The historical state variables and control variables are used to predict the change rate of state variables to the next time step.

nonlinear mechanism in the by-wire-chassis system, and improves the prediction accuracy of the model.

Recurrent Neural Network (RNN) is often used to deal with the prediction of time-series data. Time-series data generally refers to the existence of some kind of chronological correlation features in the data, and the vehicle dynamics state data studied in this paper is typical time-series data. The nodes in the hidden layer of the basic artificial neural network cannot effectively transfer information between them, and the time-step correlation information on the time-sequential data can not be transferred to the next hidden layer node, resulting in the network being insensitive to the time sequence of the data. The network model of a recurrent neural network with time series data as input, which has a chain loop network structure, and the data time-step correlation information is realized to be transferred, can effectively solve the above problems. A more accurate prediction of the future state variables of the vehicle can be realized by using the time series of historical data.

There also exist many gated RNNs, for example, LSTMs, and GRUs. Most of these networks are used to solve the problem of gradient explosion and gradient disappearance in long-term dependence. However, in this paper, only a short sequence of driving data is needed to predict the future state. In the test, the two problems mentioned above did not appear, in contrast, The general RNN model has higher prediction accuracy and shorter prediction time compared to the other gated RNNs. Therefore, the general RNN prediction model was chosen.

Based on the above analysis, with reference to the state variables and longitudinal and lateral control variables of the vehicle mechanism analysis model, the RNN vehicle dynamics model is established, and its specific model structure is shown in figure3.

The model is a recurrent neural network of sequence length N. The network inputs state and control variables for N time steps and outputs the predicted value of the change rate of the state variables from the last time step to the next time step, i.e.:

$$dX_N = f_{RNN}(X_1, X_2, \ldots, X_N, U_1, U_2, \ldots, U_N,) \qquad (4)$$

The reason why the change rate of the state variables is used as the output of the network instead of directly outputting the state variables of the next time step is that during the testing process, we found that due to the short time interval of the prediction process, the change of the state variables in the interval is very small from, which leads to the difficulty of the network to learn the trend of the future change.

Every single network in the RNN sequence, i.e., the part in the red box in figure3, inputs the state and control variables of that step and outputs the predicted change rate of the state variables from that step to the next step. Eventually, the change rate of the state variables calculated at the last time step is used as the output of the whole network.

The state variables can be expressed as:

$$X\left\{v_x, v_y, \omega\right\} \qquad (5)$$

The control variables are expressed as:

$$U\,\{acc, brake, steer\} \qquad (6)$$

where "*acc*" is the percentage of accelerator pedal input, "*brake*" is the brake oil pressure (which is set to 0 when not braking), and "*steer*" is the steering wheel angle.

Both longitudinal and lateral control variables are included to provide a basis for the subsequent realization of longitudinal and lateral coupling control. The above control variables can be directly controlled by the actuator in order to improve the real control accuracy. For example, the steering wheel angle instead of the equivalent front wheel Angle was selected as the lateral control variables, which avoids the control error caused by the complex nonlinear transmission system between the two.

The output variables are expressed as:

$$Y = \dot{X}\left\{dv_x, dv_y, \dot{\omega}\right\} \qquad (7)$$

That is, the differential of the state variables, where dv_x is the differential of the longitudinal velocity, dv_y is the differential of the lateral velocity, and $\dot{\omega}$ is the differential of the yaw speed.

The authors trained the neural network with a single hiding layer and the neural network with multiple hiding layers (2 and 3 layers), respectively, recorded the lowest value of loss function decline after 5000 epochs of training, and counted the time required for forward propagation of neural networks with different hiding layers. The result shows that the loss function of the single-hidden layer neural network also drops to a very low level, which is similar to that of the multi-hidden layer neural network. Still, the time required for forward propagation (only 0.4 ms) is much less than that of the multi-hidden layer neural network. In order to ensure the real-time performance of the model, the single hidden layer neural network was finally chosen. "TANH" was selected

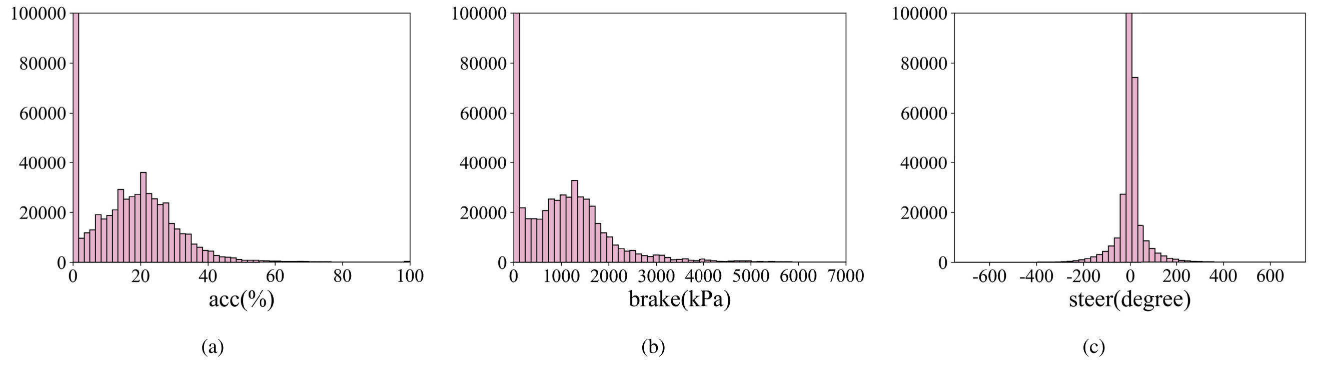

FIGURE 4. The control instruction distribution of HDD datasets. There is a large amount of parking data in the datasets.

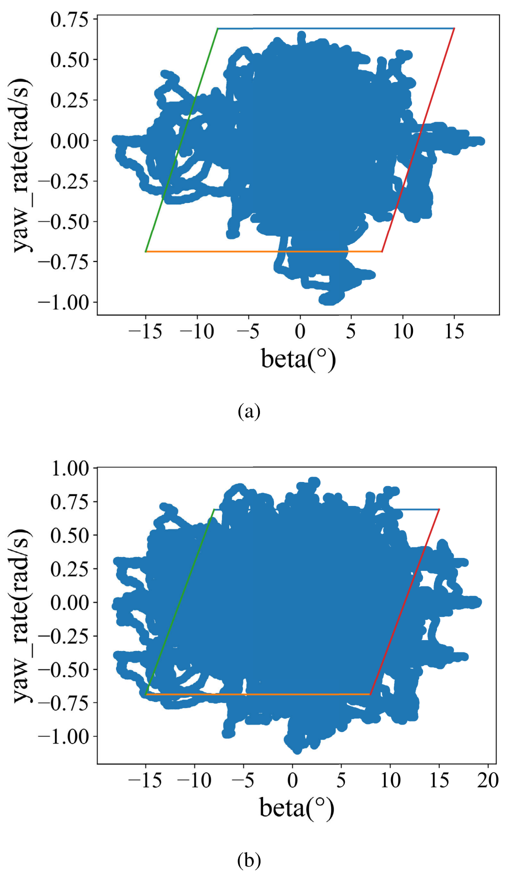

FIGURE 5. Distribution of the datasets within the stable envelope region. (a) is the distribution before supplementing. (b) is the distribution after supplementing.

as the hidden layer activation function to introduce nonlinear factors into the model.

To sum up, the data is propagated forward through RNN, and the continuous time state data series generated during vehicle operation can be used to calculate the current state change rate of the vehicle. The forward propagation calculation process is shown as equation(8):

$$\begin{aligned} a^t &= b + Wh^{t-1} + Ux^t \\ h^t &= \tanh(a^t) \\ Y^t &= c + Vh^t \end{aligned} \tag{8}$$

where the bias vectors b and c of the parameters, together with the weight matrices U, V, and W, correspond to the connections of the input layer to the hidden layer, the hidden layer to the output layer and the hidden layer to the hidden layer respectively. a is the input to the hidden layer activation function. h is the hidden layer output and Y is the model output.

C. MEODEL TRAINING

This paper trained the model using the "Honda Research Institute Driving Dataset (HDD)" [37], which consists of 104 hours of real vehicle driving data collected in the San Francisco Bay Area using instrument vehicles equipped with different sensors, including multi-sensor acquisition data such as CAN bus data, camera data, Lidar data, etc. Data collection includes high-speed, low-speed, dry road surface, wet road surface, city, highway, suburban, and other scenarios. This data set is used to train the RNN dynamic model established above, so that the model can predict the dynamic state of various speeds and various environments.

The data set needs to be processed before RNN can be trained using this data set. Because the sampling frequency and sampling time of different sensors are different in the original data, the sampling time should be unified when entering the model. If the time difference is less than 0.05ms, it is considered as simultaneous data frames, and if there are multiple frames matching, they are arranged in order. The brake oil pressure sensor with the largest sampling step (0.02s) is used as the time step of the final input data. In addition, there are some outliers in the data set, so it is necessary to remove abnormal data such as jumping, exceeding the input range, and null data frames, and perform low-pass filtering on the data to eliminate the jitter of the data collected by the sensor.

The control input distribution of the processed data set is analyzed. Figure4 shows the distribution of control instructions in the HDD data set, and it can be seen that the data on parking accounts for a large proportion of the data set. To avoid the imbalance of the data set, which makes it difficult for the model to learn the vehicle dynamics characteristics under extreme working conditions such as

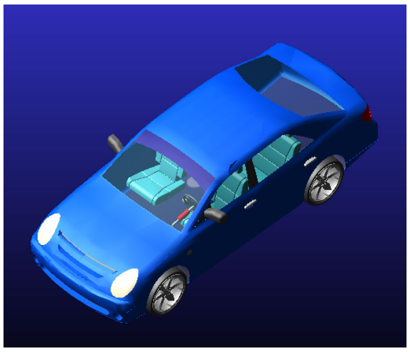

FIGURE 6. ADAMS multi-body dynamics model, modified from the "sedan FWD" template in the official document of Adams2020.

TABLE 1. Some important parameters of ADAMS model.

Parameter	Value	Unit
Front tread	1552	*mm*
rear tread	1560	*mm*
curb weight	1792	*kg*
sprung mass	1090	*kg*
height of center of mass	436	*mm*
wheelbase	2559	*mm*
axle load ratio	0.41	N/A
tire type	235/40r18	N/A

high speed, some of them are removed during processing to make the working conditions distribution of the data set more balanced. After processing, a total of about 720,000 pieces of data were obtained.

In this paper, the phase-plane analysis proposed by Erlien et al. [38] is used to draw the vehicle stability envelope to analyze the distribution of the processed data within the range of the stability envelope. As shown in figure5 (a), the distribution of the data set in the "yaw speed-side slip angle" plane is analyzed. The parallelogram depicts the vehicle kinematic stability envelope, and the data distribution of the HDD data set species is shown in blue. It can be seen that the data set only covers a part of the stable driving area, and it is necessary to supplement the data set to avoid overfitting of the model prediction model.

In this paper, the data set is supplemented by simulation, so a more accurate model needs to be established. As a mature commercial software, the model established by the ADAMS model is a full degree of freedom model, which can reflect the nonlinear characteristics of the real vehicle parts. Based on the ADAMS/car module, this paper establishes a multi-body dynamic vehicle model corresponding to the data set, as shown in figure6. With the ADAMS model, virtual environment simulation is carried out, and new data sets are generated to supplement the training data. The model is also used in the subsequent joint simulation platform construction, which provides a basis for verifying the neural network dynamic model and control algorithm in the simulation environment.

As shown in Figure6, this model is modified based on the "sedan FWD" template in the official document of Adams2020, and some parameters are shown in table1.

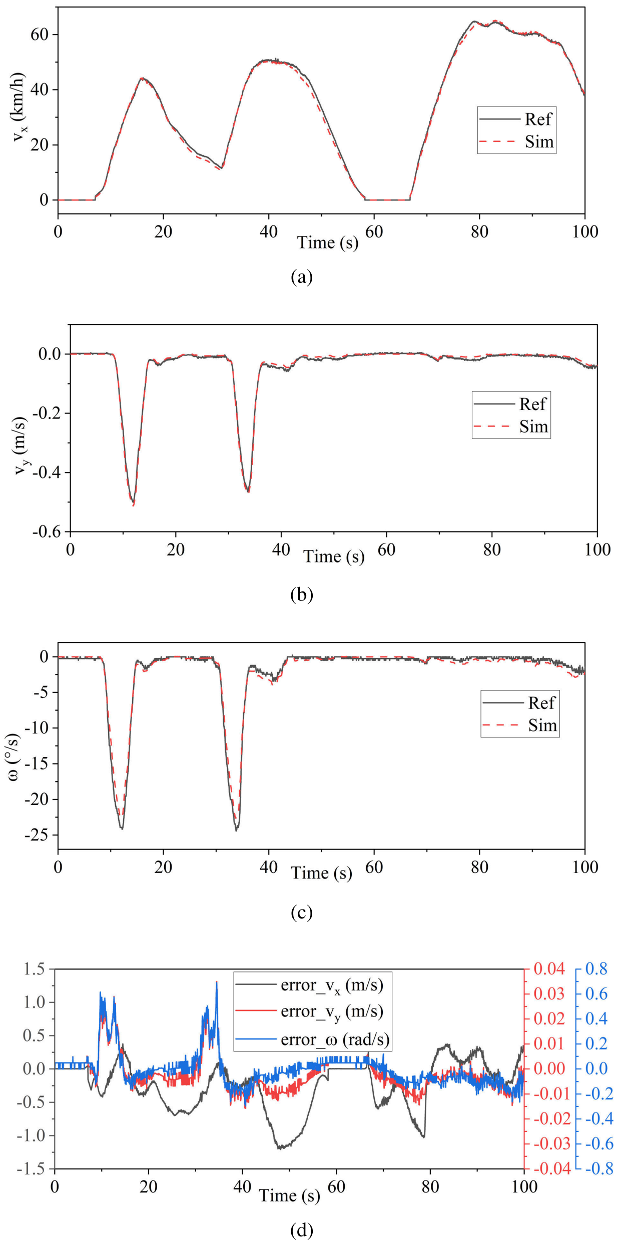

FIGURE 7. Comparison between the ADAMS model simulation output and the sensor data of the real vehicle dataset. (a) is the comparison of longitudinal speed. (b) is the comparison of lateral speed. (c) is the comparison of yaw rate. (d) is the error of longitudinal speed, lateral speed, and yaw rate during the test.

The accuracy of the ADAMS model is demonstrated by comparing it with the real sensor data in the dataset. The control variables of the data sets are input into the ADAMS model, and the state variables output of the simulation platform is compared with the state variables in the datasets. Since this paper mainly focuses on the vehicle dynamic characteristics, the three dynamic outputs(longitudinal velocity, lateral velocity, and yaw rate) mainly used in the following work are compared and verified. In figure7, the simulation results of the ADAMS model are compared with a segment of 100s real sensor data from HDD datasets, where the black solid line is the real vehicle reference of the data set, and the red dashed line is the simulation output of the ADAMS model. In the test segment, the longitudinal speed varies constantly on straights and curves, and the max speed is 68.6 km/h. The error of the compared variables is shown in figure7(d). The maximum error of v_x is $1.21 \times 10^0 m/s$, the maximum error of v_y is $3.50 \times 10^{-2} m/s$,and the maximum error of ω is $6.91 \times 10^{-1} rad/s$. The variation trend of the error is similar to that of the value being compared, and the maximum values appear around the peak of each compared variables but are still in an acceptable range. Furthermore, all the 720, 000 sets of data obtained above were used to test the ADAMS model. Within the all-data test, the MAE of longitudinal speed is $4.45 \times 10^{-1} m/s$, the MAE of lateral speed is $7.52 \times 10^{-3} m/s$, and the MAE of yaw speed is $1.36 \times 10^{-1} rad/s$, all within a small range. It is shown that the simulation platform can be used in the dynamics areas to represent real vehicles in the described dataset.

Based on the ADAMS multi-body dynamics model established above, the data generation module based on the ADAMs-MATLAB/ SIMULINK co-simulation platform is built. The authors used this platform to supplement the dataset. First, the initial state is determined at an appropriate position within the stability envelope area of the vehicle. Then, based on the initial state, uniform random control sampling is carried out, that is, the control variables of the vehicle is a reasonable random number. After the calculation of the model, the vehicle state data is obtained, so that the vehicle state can cover the entire state space of the stability area as far as possible. Figure5 (b) shows the distribution of the supplemented data set inside the stable envelope. It can be seen that the supplemented data set almost covers the entire stable region, indicating that the data set meets the needs of neural network training. Finally, data beyond the stable area is then removed.

The data set generated by the ADAMS model was merged with the real data in the HDD data set, and the time step was 0.02s. Each data of one time step included vehicle state variables and control variables, and the label was set as the change rate of state variables. For each time step, the current time step data, and the 9 historical time steps before the time step data, a total of 10 time steps are merged and extracted into a group. The RNN prediction model uses the data of 10 time steps as the model input to predict the change rate of the state variables from the current time step to the future time step. Some duplicate data is deleted to balance the distribution of the data set. A total of around 960,000 sets of data were obtained after the collation. The datasets were used for training the RNN vehicle dynamics model, with 70% as the training set and 30% as the test set.

The process of training the model using the constructed dataset is the process of driving the network to learn the intrinsic connections of the data, and the key to this is to design a correct loss function. During the training process, the internal parameters of the network are adjusted through

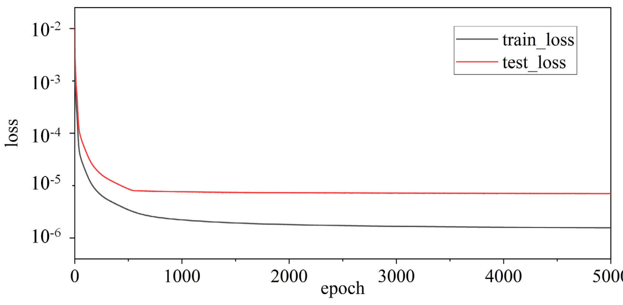

FIGURE 8. Decline of the loss function on the training and test sets.

error back-propagation to make the loss value decrease, and the smaller the loss value, the higher the accuracy of the network's fit to the data. The training loss function using the mean square error is defined as equation(9):

$$\min_{W,b} J = \frac{1}{N_e}\sum_{i=1}^{N_e} \left\| \begin{bmatrix} \dot{U}_x \\ \dot{U}_y \\ \dot{\omega} \end{bmatrix}_t - \begin{bmatrix} \hat{\dot{U}}_x \\ \hat{\dot{U}}_y \\ \hat{\dot{\omega}} \end{bmatrix}_t \right\|_2 \tag{9}$$

where $\dot{U}_x$, $\dot{U}_y$, $\dot{\omega}$ is reference value, $\hat{\dot{U}}_x$, $\hat{\dot{U}}_y$, $\hat{\dot{\omega}}$ is output value of the model, N_e is the epoch of training

"Adam" is chosen as the optimization algorithm in this paper. The algorithm has the advantages of high computational efficiency, less memory requirement, and large sample data size. In the process of vehicle dynamics model training, the initial learning rate is set to 0.001. Due to the large amount of data in the constructed vehicle dynamics data set, 1000 batch samples were selected for each training iteration.

RNN network model programming is based on Python and is implemented by the Pytorch deep learning library. The hardware configuration of the training environment is as follows: (1) CPU: Intel(R)Core(TM)i7 79900K CPU@2.59GHz; (2) GPU: NVIDIA T2000; (3) Memory: DDR4 16G.

RNNs with different numbers of neurons in the hidden layer were trained for 5000 epochs with the same data set, and the structure of 64 neurons in the hidden layer was finally selected after considering the prediction accuracy and prediction time. Figure8 shows the decline of the loss function on the training set and the test set during the training process. After 5000 epochs of training with a batch size of 1000, the loss curves of both the training set and test set dropped smoothly to 10^{-6} orders of magnitude. It indicates that this RNN model has a high accuracy and is not overfitted. The single forward propagation time of the network, after the completion of training, is only 0.4ms, which can meet the real-time requirements.

III. RNN MODEL PREDICTIVE CONTROLLER BUILDING

Based on the RNN predictive model established above, the improved model predictive controller for path tracking is built. The prediction model of the MPC algorithm is replaced

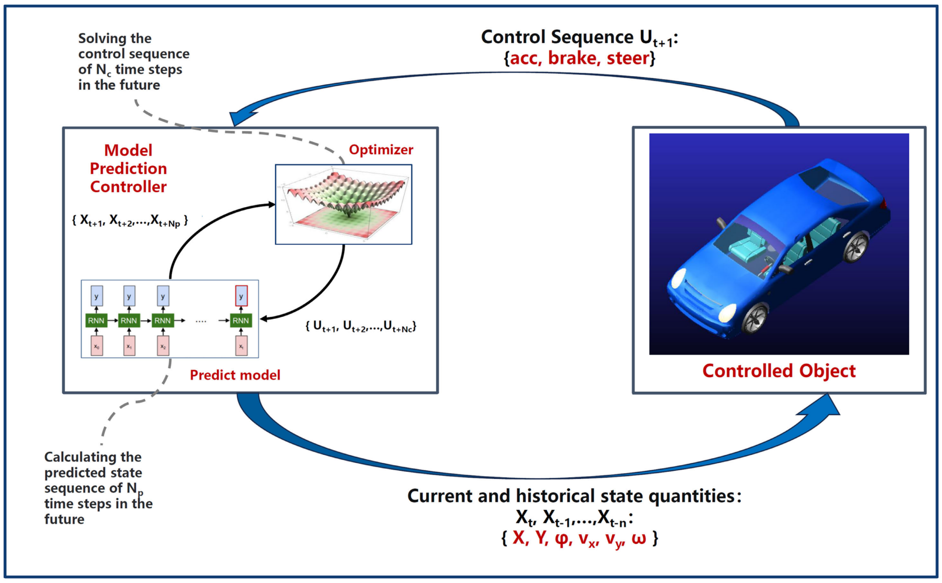
Solving the control sequence of N_c time steps in the future
Control Sequence U_{t+1}: {acc, brake, steer}
Model Prediction Controller
Optimizer
{ $X_{t+1}, X_{t+2}, \ldots, X_{t+Np}$ }
RNN
{ $U_{t+1}, U_{t+2}, \ldots, U_{t+Nc}$ }
Predict model
Controlled Object
Calculating the predicted state sequence of N_p time steps in the future
Current and historical state quantities:
$X_t, X_{t-1}, \ldots, X_{t-n}$:
{ X, Y, φ, v_x, v_y, ω }

FIGURE 9. RNN model predictive control algorithm framework.

by an RNN model to improve the prediction accuracy of future system output in nonlinear regions. The input of the prediction model contains both longitudinal and lateral control variables, based on which the coupling control in longitudinal and lateral is realized in the subsequent control algorithm.

As shown in Figure9, the neural network model predictive control algorithm is mainly divided into two parts: the model predictive controller and the controlled object. The controlled object sends the current and historical state variables to the model prediction controller. After the solution of the controller, the control sequence at the next time is sent to the controlled object. The neural network prediction model in the controller is responsible for predicting the state variables of future N_p time steps according to the current and historical state variables and control variables given by the controlled object, and sends the prediction result to the optimizer. After receiving the future state variables, the optimizer calculates the optimization objective function value. After solved by the optimization algorithm, the control sequence is given when the optimization objective function reaches the minimum value, and the control sequence at the latter moment is output to the controlled object. The state variables are $X\{v_x, v_y, \omega\}$, and the control variables are $U\{acc, brake, steer\}$, referred to in sectionII.

In the model prediction controller, the ADAMS model is used as the controlled object, and the RNN prediction model has been built above. The subsequent work is mainly the design and implementation of the optimizer.

A. PREDICTION MODEL

In the MPC algorithm, the predictive model is the basis of model predictive control. It is able to predict the future

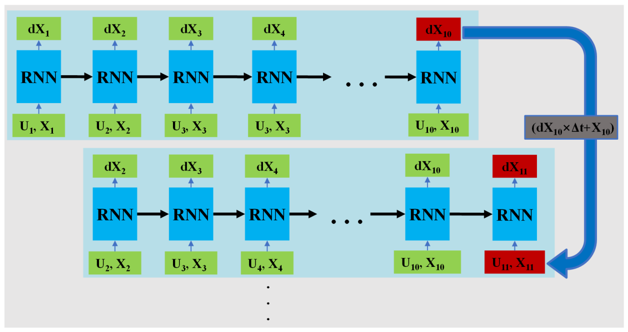

FIGURE 10. Rolling Call Process for RNN Predictive Models.

output of the system based on historical information and control inputs. An essential idea of the MPC algorithm is to predict the future state variables for N_p time steps and use the optimization algorithm to obtain the control variables in the control horizon. Therefore, multiple rolling calls to the RNN prediction model designed in this paper are made to predict the state variables at N_p time steps.

Since a single call to the network outputs the change rate of the state variables, but the values of state variables are needed, the Euler integration is utilized to calculate the state variables at the next moment:

$$X_{t+1} = X_t + \Delta t \times dX_t \tag{10}$$

During the rolling call, the predicted values of the state variables are added to the end of the input sequence. Then, the first data of the input sequence is deleted, so a new set of predicted input data sequences is formed. The RNN dynamics model is called again to predict the latter time step state variables. The predicted values of the state quantities for the next N_p time steps can be obtained after multiple calls, as shown in figure10 and equation(11).

$$\begin{aligned} d\hat{X}_t &= f_{RNN}\left(X_t, X_{t-1}, \ldots, X_{t-9}, U_t, U_{t-1}, \ldots, U_{t-9}\right) \\ \hat{X}_{t+1} &= X_t + \Delta t \times d\hat{X}_t \\ d\hat{X}_{t+1} &= f_{RNN}\left(X_{t+1}, X_t, \ldots, X_{t-8}, U_{t+1}, U_t, \ldots, U_{t-8}\right) \\ \hat{X}_{t+2} &= X_{t+1} + \Delta t \times d\hat{X}_{t+1} \end{aligned} \tag{11}$$

B. CONSTRAINT CONDITION

Control variables constraints and the side slip angle constraints are considered in the dynamics-based model prediction controller design.

1) Control variables constraints. Limit values for accelerator pedal percentage, brake oil pressure, and steering wheel angle in the control algorithm are set based on the control variables input information in the dataset. The accelerator pedal output value ranges from 0 to 100%. The brake oil pressure ranges from 0 to 7000 kPa (the oil pressure is set to 0 when not braking) in the range, and the steering wheel turning angle ranges from −720° to 720°. In addition, constraining the increment of the control variables and limiting the control increment in each sampling cycle within a reasonable range can avoid sudden changes in the control variables and ensure the continuity of the control variables. Since the accelerator pedal and brake pedal cannot be output simultaneously, an additional constraint is added. The control variables constraint is expressed as follows:

$$\begin{aligned} acc \times brake &= 0 \\ acc &\in [0, 100] \\ brake &\in [0, brake_{\max}] \\ steer &\in [steer_{\min}, steer_{\max}] \\ \Delta acc &\in [\Delta acc_{\min}, \Delta acc_{\max}] \\ \Delta brake &\in [\Delta brake_{\min}, \Delta brake_{\max}] \\ \Delta steer &\in [\Delta steer_{\min}, \Delta steer_{\max}] \end{aligned} \tag{12}$$

2) Side slip angle constraints. The side slip angle significantly impacts the vehicle's stability, so the side slip angle must be limited to a reasonable range. BOSCH conducted by the vehicle stability study results [39] show that: in good adhesion on dry asphalt, the limit of the side slip angle can reach +12 ° during stable driving, that is:

$$-12^\circ < \beta < 12^\circ \tag{13}$$

C. TRACKING ERROR MODEL

In the MPC algorithm, it is necessary to calculate the error value according to the predicted state and the reference state to determine the control input. Since the controller controls both the longitudinal and lateral of the vehicle, the longitudinal and lateral errors need to be calculated separately in the error model. As shown in figure1, the yaw angle error is defined as the angle between the vehicle yaw angle and the yaw angle of the reference track. The lateral error is defined as the length of the vertical line segment from the vehicle centroid to the path of the reference track. The longitudinal error is defined as the deviation value between the reference longitudinal speed and the vehicle longitudinal speed. The error value can be calculated by equation(14).

$$\begin{aligned} \dot{e}^t_\psi &= \dot{\psi}^t - \dot{\psi}^t_s \\ \dot{e}^t_y &= V^t{}_x \sin(e^t{}_\psi) + V^t{}_y \cos(e^t{}_\psi) \\ e^t{}_v &= \left|V^t{}_x - V^t{}_{ref}\right| \end{aligned} \tag{14}$$

In the equation, the superscript t represents the value of the variable at time step t. The value of the variable at the future time step is predicted using the RNN model proposed above.

D. OPTIMIZED OBJECTIVE FUNCTION

Reasonably designing the optimized objective function in the model predictive control algorithm is an important component to ensure that the intelligent vehicle tracks the reference path quickly and stably. When designing the RNN-based model predictive control, the increment of the control variables is also added to the optimization objective function while considering the state variables error, with the aim of preventing excessive acceleration and shock. The optimization objective function is shown in equation(15).

$$\begin{aligned} &\textit{Minimize } J \\ &J = \sum_{t=t_0}^{t=N_p} \left(Q_{v_x}\left(e^t{}_{v_x}\right)^2 + Q_y\left(e^t{}_y\right)^2 + Q_\psi\left(e^t{}_\psi\right)^2\right) \\ &\quad + \sum_{t=t_0}^{t=N_c} \Big(Q_{\Delta acc}(\Delta acc)^2 + Q_{\Delta brake}(\Delta brake)^2 \\ &\quad + Q_{\Delta steer}(\Delta steer)^2\Big) \end{aligned} \tag{15}$$

where N_p is the prediction horizon and N_c is the control horizon. Throttle input increment Δacc, brake input increment $\Delta brake$, and steering wheel angle control increment $\Delta steer$ are used as optimization variables. Q_{v_x}, Q_y, and Q_ψ are the weight matrices of the errors. $Q_{\Delta acc}$, $Q_{\Delta brake}$, and $Q_{\Delta steer}$ are the weight matrix of the control increment.

The first term of equation(15) indicates the ability of the system to follow the reference path, which requires that the vehicle track the reference path with the smallest possible longitudinal speed error, lateral error, and heading deviation to enhance the path tracking effect. The second term indicates the requirement for control increment constraints to ensure that the control variables change smoothly. The overall goal of the objective function is to enable fast, accurate, and stable path tracking of the controlled object.

By solving a nonlinear optimization problem in a particular control horizon, the sequence of control increments in that period is solved:

$$\Delta u(t) = [\Delta u(t), \Delta u(t+1), \ldots, \Delta u(t+N_c-1)] \tag{16}$$

The first value in the solved sequence of control increments is used to calculate the control variables for the next time step:

$$u(t) = \Delta u(t-1) + u(t-1) \tag{17}$$

E. OPTIMIZER BASED ON DIFFERENTIAL EVOLUTION ALGORITHM

In the classical MPC algorithm, the predicted values of the state variables are calculated by the explicit set of dynamics equations, and there is only one control variable, the front wheel angle. As shown in equation(11),(14) and (15), the predicted values in the optimization problem involved in this paper come from the output of the RNN neural network, and the control variables include a total of three in the longitudinal and lateral directions, which is more than classical MPC. This brings the following problems to the optimization solution:

1) Classical MPC generally utilizes the interior point method or active set method to solve nonlinear optimization problems. The gradient of the optimization objective function is needed in these two methods. However, due to the RNN neural network inside the optimization objective function, its expression is not explicit, and the gradient solution is complicated.

2) As there are 3 control quantities (only one control variable in the classical MPC), the constraints and optimization objective function become more complicated. After testing, the optimization solution algorithm with classical MPC very easily falls into the local optimum, reducing the control accuracy significantly.

Based on the above two reasons, the optimization problems involved in this paper are difficult to solve by traditional optimization algorithms, so it is proposed to solve the optimization problems in this paper by using "differential evolution(DE)" Algorithm.

Differential evolution algorithm originates from the improvement of genetic annealing algorithm, does not need to provide the gradient information of the optimization objective function, can effectively avoid falling into the local optimum, and the convergence speed is the fastest among the intelligent optimization algorithms [40]. Therefore, it is suitable for using in the control of vehicle path tracking. The population evolves to the next generation through mutation, crossover, and selection operations. It repeats the cycle until the algorithm reaches a predetermined maximum number of iterations or the optimal solution of the population reaches a predetermined error accuracy.

The primary hyperparameters of the differential evolution algorithm include the population size n_p, the variation factor F, the crossover rate CR, and the number of evolutionary generations G. Since path tracking is a task with high real-time requirements, it is necessary to make a trade-off between the convergence speed and the optimization accuracy. Among them, the population size greatly impacts real-time performance. In order to ensure the real-time performance

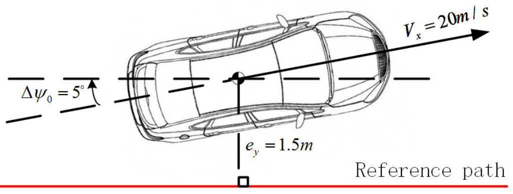

FIGURE 11. A simple experiment to test the performance of DE at different evolutionary generations to determine the final evolutionary generations.

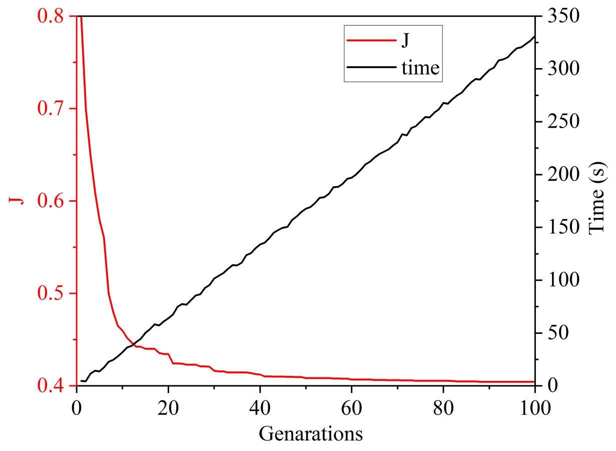

FIGURE 12. Optimization objective function values and computing time with evolutionary generations.

TABLE 2. Parameters of RNN model predictive control.

Parameter	Value	Unit
Prediction horizon(N_p)	25	N/A
Control horizon(N_c)	15	N/A
Constrain of acc	[0, 100]	%
Constrain of $brake$	[0, 7000]	kPa
Constrain of $steer$	[-11.3, 11.3]	rad
Constrain of Δacc	[-5, 5]	%
Constrain of $\Delta brake$	[-350, 350]	kPa
Constrain of $\Delta steer$	[-0.35, 0.35]	rad
Weight:[$x, y, yaw, \Delta acc, \Delta brake, \Delta steer$]	[1,1,0.5,0.1,0.1,0.2]	N/A
Population size of DE (n_p)	20	N/A
Variation factor of DE (F)	0.5	N/A
Crossover probability of DE (CR)	0.7	N/A
Evolutionary generations of DE (G)	20	N/A

of the algorithm, the population size is reduced as much as possible. A large maximum number of evolutionary generations(1000) is fixed to determine the population size, variance factor, and crossover probability using orthogonal experimentation.

In addition, the idea of "early stopping" is utilized to prioritize the real-time performance of the algorithm by setting the maximum number of evolutionary generations to limit the convergence time. The larger the number of evolution generations, the higher the optimization accuracy of the objective function, but at the same time, it will take more computation time. The differential evolution algorithm with different evolutionary generations is simulated to track

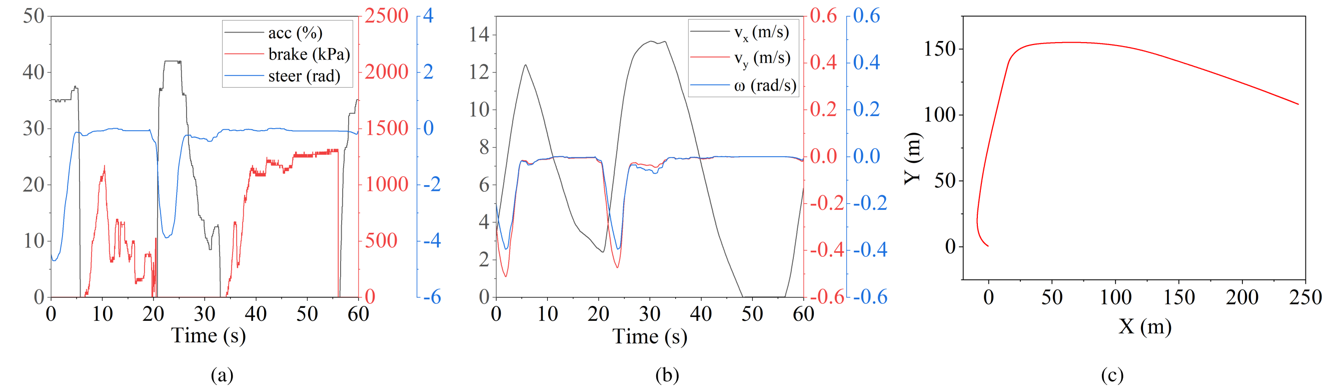

FIGURE 13. A segment of driving data of a human driver on an urban road with variable speeds is used to test the RNN dynamic state predictive model. (a) shows the driving instructions from the human driver. (b) shows longitudinal/lateral speed and yaw rate during the test process. (c) shows the driving path throughout the test.

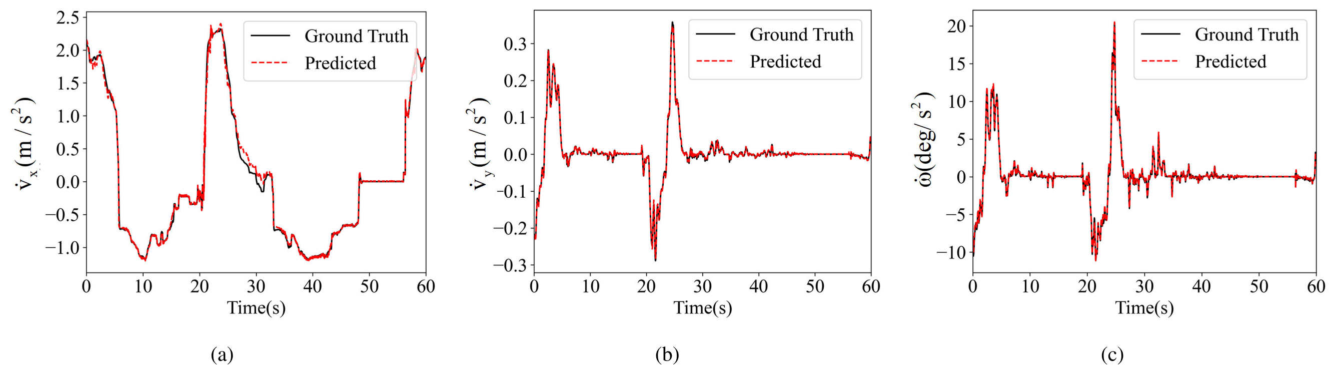

FIGURE 14. The model outputs ($\dot{v}_x$, $\dot{v}_y$, $\dot{\omega}$) are compared with the data collected by real car sensor. (a) is the comparison of longitudinal speed change rate. (b) is the comparison of lateral speed change rate. (c) is the comparison of yaw speed change rate.

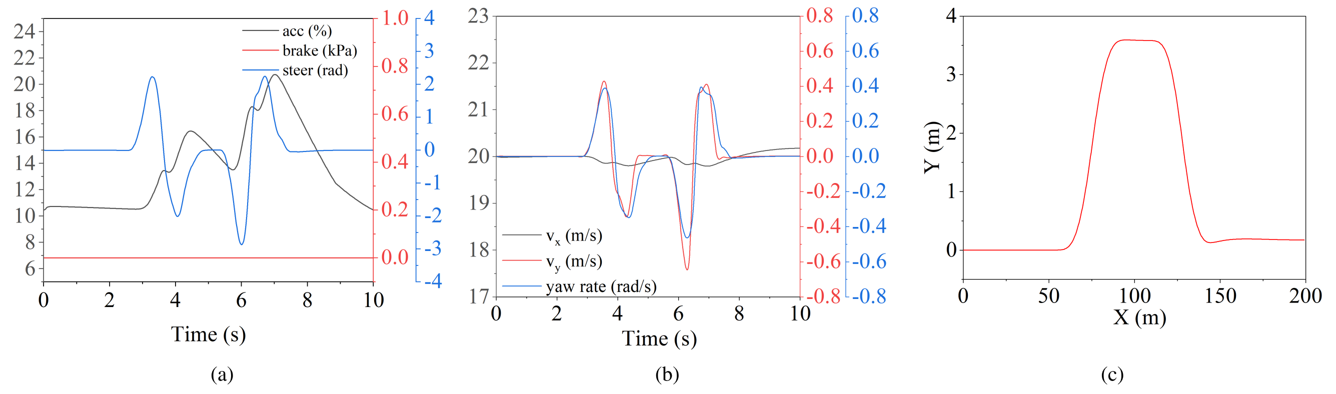

FIGURE 15. The predictive effect of RNN dynamic model and linear/nonlinear mechanism model is compared under double lane change condition. (a) shows the control variables sequence. (b) shows longitudinal/lateral speed and yaw rate during the double lane change test. (c) shows the path throughout the double lane change test.

the same path at the same speed to select the best evolutionary generations.

The simple experiment shown in figure11 is designed to analyze the performance of the control algorithm with different evolutionary generations under the GPU parallel computing environment, and the appropriate evolutionary generations are selected. The vehicle travels at a speed of $20m/s$, with a lateral deviation of $1m$ and a heading deviation of 5°. Figure12 shows the value of the objective function and the optimization solution time at different evolutionary generations during the simulation. It can be seen that the objective optimization function decreases quickly before 20 generations of evolutionary generations. After 20 generations, the objective function value decreases

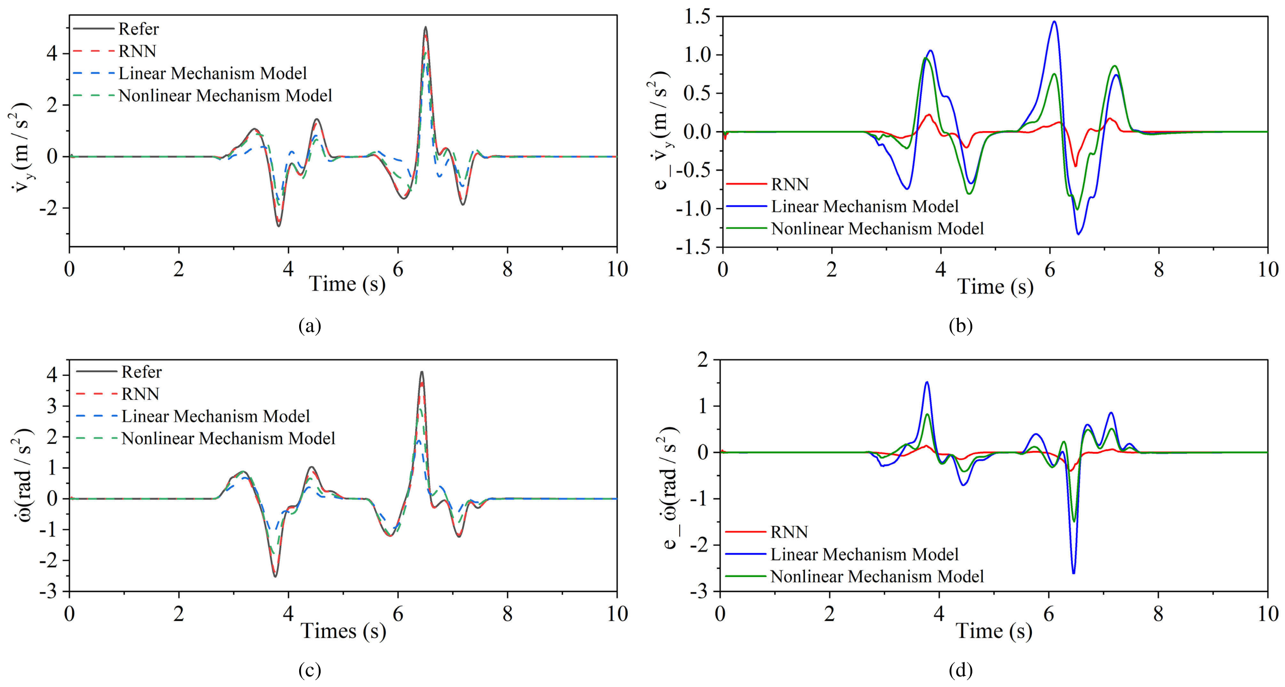

FIGURE 16. The model outputs($\dot{v}_y$ and $\dot{\omega}$) of the RNN model are compared with that of the mechanism-based model, and the predictive errors are shown in (b) and (d). The black line is the actual output data of the vehicle, the red line is the performance of the RNN model, the blue line is the performance of the linear mechanism model, and the green line is the performance of the nonlinear mechanism model. (a) is the lateral speed change rate. (b) is the error of lateral speed change rate. (c) is the yaw speed change rate. (d) is the error of yaw speed change rate.

insignificantly. However, the time consumed by the algorithm always increases linearly with the number of evolutionary generations. Therefore, 20 is finally chosen as the evolutionary generation, at which point the algorithm operation consumes 63.9ms.

IV. EXPERIMENT

The RNN dynamic model and control algorithm established above are tested in the last section of this paper. Firstly, the prediction accuracy of the RNN vehicle dynamics model developed in sectionII is validated. The validation is performed on a segment of a continuous real vehicle dataset. Then, the RNN model is compared with a classical linear vehicle dynamics model and a nonlinear vehicle dynamics model (with a nonlinear tire model). The controller built in sectionIII is then validated by simulation at low (10m/s) and high (30m/s) speeds on the co-simulation platform, and compared to the mechanism model-based MPC control algorithm whose lateral and longitudinal control are separated (longitudinal using a PID controller). Double lane change is selected as the simulation condition. Table2 shows the control parameters. The ADAMS multi-body dynamics model established in sectionII is utilized to build the ADAMS/ Python/ Simulink joint simulation model. The ADAMS model is used as the controlled object, and the RNN prediction model and controller code are written based on Python. Finally, the integration is carried out in Simulink to complete the data transfer.

A. MODEL VALIDATION

The RNN prediction model is validated using data from the HDD real vehicles dataset. A segment of 3000 continuous time steps (60s) from the HDD dataset that is not added to the training set is extracted to test the model. The test uses driving data of a human driver on an urban road with variable speeds, including turning and going straight. Control instructions $(acc, brake, steer)$ from the dataset are input to the RNN model, and then the model outputs $\{dv_x, dv_y, d\omega\}$ are compared with the real vehicle sensor data in the dataset. Figure13 shows the control variables$(acc, brake, steer)$, longitudinal/lateral velocity, yaw rate, and trajectory during the test segment.

The comparison result of the model output and real sensor data is shown in figure14. In order to prove the accuracy of the model prediction, the mean square error between the model output and the real data is calculated. The MSE of $dv_x(m/s^2)$ is 7.28×10^{-3}, the MSE of $dv_y(m/s^2)$ is 8.94×10^{-5}, and the MSE of $d\omega(deg/s^2)$ is 1.01×10^{-2}. This result shows that, in the general driving scene, the prediction errors of the state variables for all three directions(longitudinal, lateral, and heading) are minor and within the acceptable range. Additionally, good performance on the dataset that is not added to the training set indicates that the model is not overfitting.

Moreover, the advantage of the RNN vehicle dynamics model compared with the classical mechanism model is tested. The vehicle is run under the double lane change

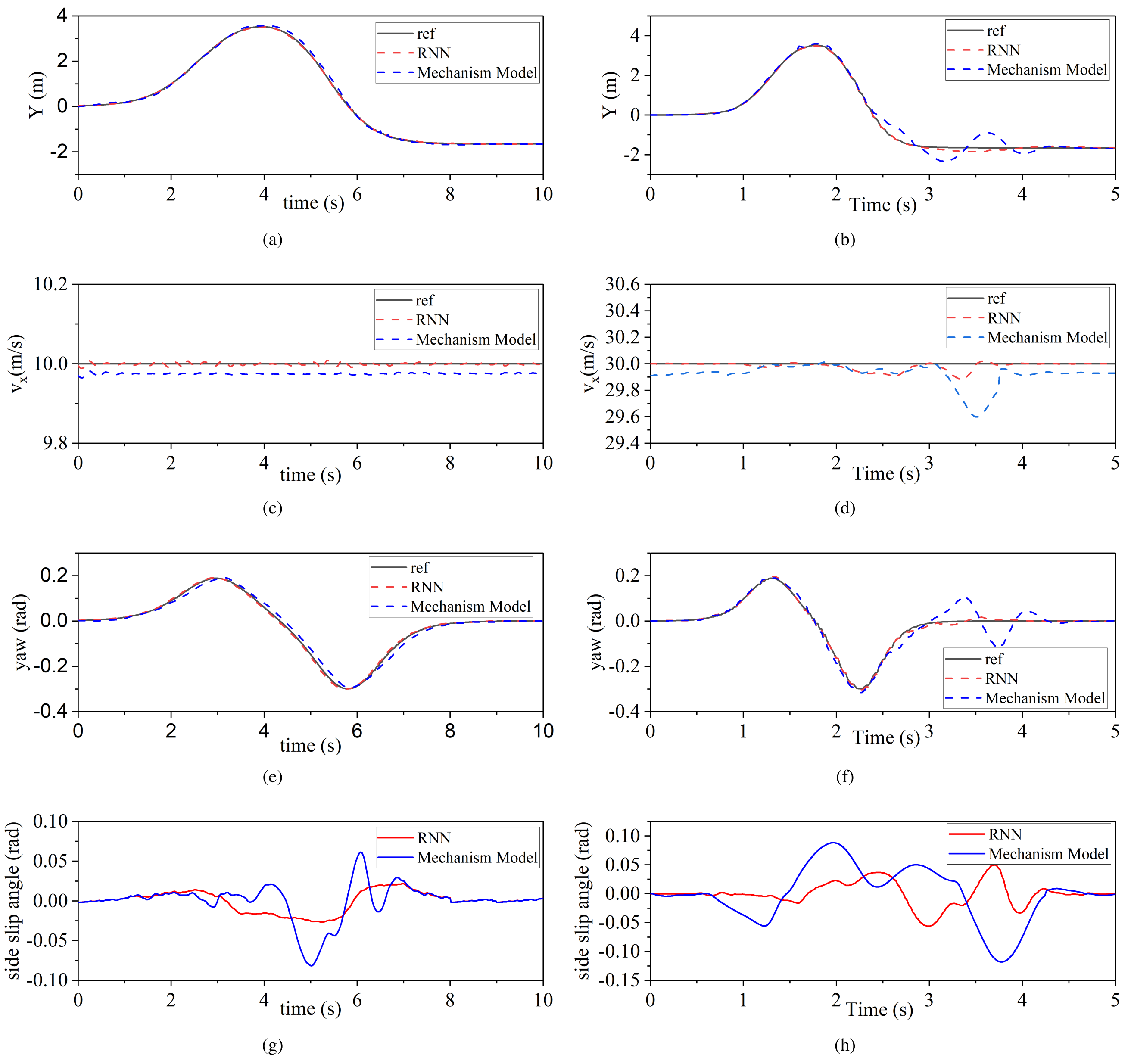

FIGURE 17. The longitudinal/lateral coupled RNN model predictive control algorithm is compared with the mechanism-model-based model predictive control algorithm. The black line is the reference path, the red line is the performance of the RNN model, the blue line is the performance of the mechanism model. (a), (c), (e), and (g) are tested under a relatively low speed(10m/s), and (b), (d), (f), and (h) are tested under a relatively high speed(30m/s). (a) and (b) is the tracking performance of the two for lateral displacements. (c) and (d) is the tracking performance of the two for longitudinal speed. (e) and (f) is the tracking performance of the two for yaw angle. (g) and (h) is the comparison of the side slip angle.

test condition built into ADAMS/Car(maintain a speed of $20m/s$). The control variables(acc, $brake$, $steer$), longitudinal/lateral velocity, yaw rate, and trajectory are shown in figure15.

In this simulation experiment, different models were used to predict the change reate of lateral velocity and the change rate of yaw rate. The linear mechanism model, nonlinear mechanism model, and the RNN model were compared. As shown in figure16, under the double lane change condition, the prediction accuracy of the RNN model is significantly better than that of the linear/nonlinear mechanistic model for both lateral velocity and yaw speed. As shown in figure16(b), the maximum error of the RNN model when predicting the lateral velocity in this test is $4.52 \times 10^{-1} m/s$, while that of the linear model is $1.44 \times 10^{0} m/s$, and $1.01 \times 10^{0} m/s$ for the nonlinear model. In figure16(d), the maximum error of the RNN model when predicting the yaw rate is $3.98 \times 10^{-1} rad/s$, while that of the linear model is $2.61 \times 10^{0} rad/s$, and $1.50 \times 10^{0} rad/s$ for the nonlinear model. Especially around the 4s and 6s,

TABLE 3. Comparison of control results between longitudinal/lateral coupled RNN-based MPC and classical MPC agianst low and high speed(10 m/s and 30 m/s). Th mean square error (MSE), max error of state variables (v_x, v_y, and *yaw*), and the max side slip angle is compared. Length unit: m, angular unit: rad, time unit: s.

Item	RNN(10m/s)	Mechanism(10m/s)	RNN(30m/s)	Mechanism(30m/s)
E_{v_x}(MSE)	1.23×10^{-5}	6.46×10^{-4}	1.20×10^{-3}	1.29×10^{-2}
E_y(MSE)	3.41×10^{-4}	5.57×10^{-3}	4.44×10^{-3}	7.40×10^{-2}
E_{yaw}(MSE)	2.99×10^{-5}	1.78×10^{-4}	3.85×10^{-5}	1.11×10^{-3}
E_{v_x}(MAX)	1.15×10^{-2}	3.55×10^{-2}	1.13×10^{-1}	4.01×10^{-1}
E_y(MAX)	8.83×10^{-2}	1.99×10^{-1}	1.99×10^{-1}	7.64×10^{-1}
E_{yaw}(MAX)	1.45×10^{-2}	2.86×10^{-2}	1.81×10^{-2}	1.15×10^{-1}
β(MAX)	2.19×10^{-2}	6.16×10^{-2}	5.03×10^{-2}	8.84×10^{-2}

when the lateral velocity is higher, the error of the mechanism model increases significantly, while the error of the RNN model is much smaller. The prediction accuracy of the nonlinear mechanism model is still much lower than that of the RNN model, although it is higher when compared to the linear mechanism model. These results prove that the RNN vehicle dynamic model developed in this paper performs better than mechanical models under high lateral velocity.

B. CONTROL ALGORITHM TESTING

The classical MPC, which uses a mechanism-based model for lateral tracking and a PID controller for longitudinal tracking, is compared with the longitudinal/lateral coupled RNN-based MPC, as shown in figure17. The tracking effect of the two kinds of controllers on lateral displacement, yaw angle, and longitudinal velocity at low speed (longitudinal velocity ($10m/s$)) and high speed (longitudinal velocity ($30m/s$)) conditions are compared, respectively. Since a smaller side slip angle is further away from the edge of the stability envelope, it indicates good stability. The side slip angle during vehicle travel at both conditions is also compared to determine the travel stability of the two during the control process. Table3 shows the mean square error and the maximum error of the two controllers in different directions, as well as the maximum side slip angle during the simulation.

At low speeds($10m/s$), as shown in figure17(a),(c),(e), although both controllers have a low tracking error, the tracking error of the MPC based on the RNN prediction model is slightly lower than that of the MPC based on the classical mechanism model. Moreover, there is a small amount of uncanceled steady-state error in the following of the longitudinal speed by the PID controller. As shown in the figure17(g), when using the MPC based on RNN model, the max side slip angle is $2.65 \times 10^{-2} rad$, while that of the MPC based on mechanism model is $8.16 \times 10^{-2} rad$. This indicates that the MPC based on RNN model has a higher stability.

At high speeds($30m/s$), as shown in figure17(b),(d),(f), the mean square tracking error and maximum tracking error in both longitudinal, lateral, and yaw directions increase a lot. Especially when the vehicle is performing the third steering at around 3s, due to the increase of lateral velocity and yaw velocity, the prediction accuracy of both models is affected, which leads to the decrease of the control accuracy. The MPC based on the mechanism model has a strong oscillation, while the MPC based on the RNN remained within a small range control error. As shown in table3, mean square error and maximum error of RNN-based controller proposed in this paper are far less than the classical MPC controller. Additionally, Throughout the process, the side slip angle is smaller under the RNN-based MPC control. The maximum side slip angle is $5.03 \times 10^{-2} rad$ within the vehicle stability envelope, and the controller control effect meets the vehicle control stability requirements.

In conclusion, the longitudinal/laterral coupled RNN model predictive controller established in this paper has higher control accuracy and stability than the model predictive controller based on the classical mechanism model, espechially at high speeds.

V. CONCLUSION

This paper proposes to replace the traditional state prediction model in model predictive control with an RNN model, which improves the accuracy of dynamic state prediction. This model is trained in the framework of deep learning using a large amount of data generated during driving, avoiding the need to analyze the complex electromechanical-hydraulic coupling characteristics of the vehicle. In addition, the output of the RNN model includes both longitudinal and lateral dynamics parameters, based on which the controller is designed to realize longitudinal and lateral coupled control. For the problem that the gradient of the objective optimization function is challenging to compute and easily falls into the local optimum during the optimization solution in the MPC control process, it is proposed to solve the problem with a differential evolution algorithm. Finally, the prediction accuracy of the RNN model and the control effect of the controller proposed in this paper are verified. The RNN prediction model is validated on the real vehicle data set and compared with linear/nonlinear mechanism models under typical working conditions. The results show that the prediction accuracy of the RNN model is much higher than that of the mechanism model, and the single prediction time is only 0.4 ms. Moreover, a joint ADAMS/Simulink/Python simulation platform is built to verify the control accuracy and stability of the RNN-based MPC control algorithm against the mechanism model-based MPC. The results show that the control accuracy and stability of the RNN-based MPC controller are higher than that of the MPC based on the mechanical model at high speed, and the real-time performance of the algorithm is guaranteed.

ACKNOWLEDGMENT

The authors would like to thank the Honda Research Institute, USA, for sharing the HRI Driving Dataset (HDD) and some key parameters of the experimental vehicle.

REFERENCES

[1] Q. Yao, Y. Tian, Q. Wang, and S. Wang, "Control strategies on path tracking for autonomous vehicle: State of the art and future challenges," *IEEE Access*, vol. 8, pp. 161211–161222, 2020.

[2] H. Wang, B. Liu, X. Ping, and Q. An, "Path tracking control for autonomous vehicles based on an improved MPC," *IEEE Access*, vol. 7, pp. 161064–161073, 2019.

[3] C. Sun, X. Zhang, Q. Zhou, and Y. Tian, "A model predictive controller with switched tracking error for autonomous vehicle path tracking," *IEEE Access*, vol. 7, pp. 53103–53114, 2019.

[4] P. Stano, U. Montanaro, D. Tavernini, M. Tufo, G. Fiengo, L. Novella, and A. Sorniotti, "Model predictive path tracking control for automated road vehicles: A review," *Annu. Rev. Control*, vol. 55, pp. 194–236, Dec. 2023.

[5] Z. Yang, X. Pei, J. Xu, X. Zhang, and W. Xi, "Decision-making in autonomous driving by reinforcement learning combined with planning & control," in *Proc. 6th CAA Int. Conf. Veh. Control Intell. (CVCI)*, Nanjing, China, Oct. 2022, pp. 1–6, doi: 10.1109/CVCI56766.2022.9964691.

[6] M. Rokonuzzaman, N. Mohajer, S. Nahavandi, and S. Mohamed, "Review and performance evaluation of path tracking controllers of autonomous vehicles," *IET Intell. Transp. Syst.*, vol. 15, no. 5, pp. 646–670, May 2021.

[7] H. Peng and M. Tomizuka, "Vehicle lateral control for highway automation," in *Proc. Amer. Control Conf.*, May 1990, pp. 788–794, doi: 10.23919/ACC.1990.4790840.

[8] R. C. Rafaila and G. Livint, "Nonlinear model predictive control of autonomous vehicle steering," in *Proc. 19th Int. Conf. Syst. Theory, Control Comput. (ICSTCC)*, Oct. 2015, pp. 466–471, doi: 10.1109/ICSTCC.2015.7321337.

[9] U. Rosolia and F. Borrelli, "Learning how to autonomously race a car: A predictive control approach," *IEEE Trans. Control Syst. Technol.*, vol. 28, no. 6, pp. 2713–2719, Nov. 2020, doi: 10.1109/TCST.2019.2948135.

[10] G. Bai, Y. Meng, L. Liu, W. Luo, Q. Gu, and L. Liu, "Review and comparison of path tracking based on model predictive control," *Electronics*, vol. 8, no. 10, p. 1077, Sep. 2019.

[11] J. C. Simotwo, S. I. Kamau, and P. K. Hinga, "Control strategies to steer and drive an autonomous 4WS4WD ground vehicle: A review of MPC approaches," in *Proc. Sustain. Res. Innov. Conf.*, JKUAT Main Campus, Kenya, Apr. 2019, pp. 267–274.

[12] H. Peng, W. Wang, Q. An, C. Xiang, and L. Li, "Path tracking and direct yaw moment coordinated control based on robust MPC with the finite time horizon for autonomous independent-drive vehicles," *IEEE Trans. Veh. Technol.*, vol. 69, no. 6, pp. 6053–6066, Jun. 2020, doi: 10.1109/TVT.2020.2981619.

[13] P. Falcone, F. Borrelli, and J. Asgari, "Predictive active steering control for autonomous vehicle systems," in *IEEE Trans. Control Syst. Technol.*, vol. 15, no. 3, pp. 566–580, May 2007, doi: 10.1109/TCST.2007.894653.

[14] T. Nayl, G. Nikolakopoulos, and T. Gustafsson, "Path following for an articulated vehicle based on switching model predictive control under varying speeds and slip angles," in *Proc. IEEE 17th Int. Conf. Emerg. Technol. Factory Autom. (ETFA)*, Sep. 2012, pp. 1–7, doi: 10.1109/ETFA.2012.6489635.

[15] B.-C. Chen, B.-C. Luan, and K. Lee, "Design of lane keeping system using adaptive model predictive control," in *Proc. IEEE Int. Conf. Autom. Sci. Eng. (CASE)*, Aug. 2014, pp. 922–926, doi: 10.1109/CoASE.2014.6899436.

[16] J. Hanema, M. Lazar, and R. Tóth, "Stabilizing tube-based model predictive control: Terminal set and cost construction for LPV systems," *Automatica*, vol. 85, pp. 137–144, Nov. 2017.

[17] R. Hajiloo, M. Abroshan, A. Khajepour, A. Kasaiezadeh, and S. K. Chen, "Integrated steering and differential braking for emergency collision avoidance in autonomous vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 5, pp. 3167–3178, May 2021, doi: 10.1109/TITS.2020.2984210.

[18] Z. Cui, X. Xia, and X. Pei, "A modified vehicle following control system on the curved road based on model predictive control," in *Proc. 33rd Chin. Control Decis. Conf. (CCDC)*, May 2021, pp. 1098–1103, doi: 10.1109/CCDC52312.2021.9602682.

[19] J. Yu, X. Guo, X. Pei, Z. Chen, W. Zhou, M. Zhu, and C. Wu, "Path tracking control based on tube MPC and time delay motion prediction," *IET Intell. Transp. Syst.*, vol. 14, no. 1, pp. 1–12, Jan. 2020.

[20] S. Cheng, L. Li, X. Chen, J. Wu, and H.-d. Wang, "Model-predictive-control-based path tracking controller of autonomous vehicle considering parametric uncertainties and velocity-varying," *IEEE Trans. Ind. Electron.*, vol. 68, no. 9, pp. 8698–8707, Sep. 2021, doi: 10.1109/TIE.2020.3009585.

[21] P. Fang, Y. Cai, L. Chen, H. Wang, Y. Li, M. A. Sotelo, and Z. Li, "A high-performance neural network vehicle dynamics model for trajectory tracking control," *Proc. Inst. Mech. Eng., D, J. Automobile Eng.*, vol. 237, no. 7, pp. 1695–1709, May 2022, doi: 10.1177/09544070221095660.

[22] Y. Qi, Z. Zhang, C. Hu, X. Zhou, L. Xie, and H. Su, "An MPC-based controller framework for agile maneuvering of autonomous vehicles," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jul. 2021, pp. 1228–1234, doi: 10.1109/IV48863.2021.9575134.

[23] N. Chowdhri, L. Ferranti, and F. S. Iribarren, "Integrated nonlinear model predictive control for automated driving," *Control Eng. Pract.*, vol. 106, Jan. 2021, Art. no. 104654, doi: 10.1016/j.conengprac.2020.104654.

[24] M. Rokonuzzaman, N. Mohajer, S. Nahavandi, and S. Mohamed, "Learning-based model predictive control for path tracking control of autonomous vehicle," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2020, pp. 2913–2918, doi: 10.1109/SMC42975.2020.9283293.

[25] Q. Wang, J. He, C. Lu, C. Wang, H. Lin, H. Yang, H. Li, and Z. Wu, "Modelling and control methods in path tracking control for autonomous agricultural vehicles: A review of state of the art and challenges," *Appl. Sci.*, vol. 13, no. 12, p. 7155, Jun. 2023.

[26] M. Rokonuzzaman, N. Mohajer, S. Nahavandi, and S. Mohamed, "Model predictive control with learned vehicle dynamics for autonomous vehicle path tracking," *IEEE Access*, vol. 9, pp. 128233–128249, 2021, doi: 10.1109/ACCESS.2021.3112560.

[27] H. Zhang and W. Zhao, "Decoupling control of steering and driving system for in-wheel-motor-drive electric vehicle," *Mech. Syst. Signal Process.*, vol. 101, pp. 389–404, Feb. 2018, doi: 10.1016/j.ymssp.2017.08.042.

[28] N. A. Spielberg, M. Brown, N. R. Kapania, J. C. Kegelman, and J. C. Gerdes, "Neural network vehicle models for high-performance automated driving," *Sci. Robot.*, vol. 4, no. 28, Mar. 2019, Art. no. eaaw1975, doi: 10.1126/scirobotics.aaw1975.

[29] N. A. Spielberg, M. Brown, and J. C. Gerdes, "Neural network model predictive motion control applied to automated driving with unknown friction," *IEEE Trans. Control Syst. Technol.*, vol. 30, no. 5, pp. 1934–1945, Sep. 2022, doi: 10.1109/TCST.2021.3130225.

[30] S. Dixit, U. Montanaro, M. Dianati, D. Oxtoby, T. Mizutani, A. Mouzakitis, and S. Fallah, "Trajectory planning for autonomous high-speed overtaking in structured environments using robust MPC," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 6, pp. 2310–2323, Jun. 2020, doi: 10.1109/TITS.2019.2916354.

[31] W. Zhang, Z. Wang, L. Drugge, and M. Nybacka, "Evaluating model predictive path following and yaw stability controllers for over-actuated autonomous electric vehicles," *IEEE Trans. Veh. Technol.*, vol. 69, no. 11, pp. 12807–12821, Nov. 2020, doi: 10.1109/TVT.2020.3030863.

[32] Z. Luan, J. Zhang, W. Zhao, and C. Wang, "Trajectory tracking control of autonomous vehicle with random network delay," *IEEE Trans. Veh. Technol.*, vol. 69, no. 8, pp. 8140–8150, Aug. 2020, doi: 10.1109/TVT.2020.2995408.

[33] K. Yuan, H. Shu, Y. Huang, Y. Zhang, A. Khajepour, and L. Zhang, "Mixed local motion planning and tracking control framework for autonomous vehicles based on model predictive control," *IET Intell. Transp. Syst.*, vol. 13, no. 6, pp. 950–959, Jun. 2019.

[34] R. Skulstad, G. Li, T. I. Fossen, B. Vik, and H. Zhang, "A hybrid approach to motion prediction for ship docking—Integration of a neural network model into the ship dynamic model," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–11, 2021, doi: 10.1109/TIM.2020.3018568.

[35] A. Wunderlich and E. Santi, "Digital twin models of power electronic converters using dynamic neural networks," in *Proc. IEEE Appl. Power Electron. Conf. Expo. (APEC)*, Jun. 2021, pp. 2369–2376, doi: 10.1109/APEC42165.2021.9487201.

[36] Y. Ren, Z. Zhao, C. Zhang, Q. Yang, and K.-S. Hong, "Adaptive neural-network boundary control for a flexible manipulator with input constraints and model uncertainties," *IEEE Trans. Cybern.*, vol. 51, no. 10, pp. 4796–4807, Oct. 2021, doi: 10.1109/TCYB.2020.3021069.

[37] V. Ramanishka, Y.-T. Chen, T. Misu, and K. Saenko, "Toward driving scene understanding: A dataset for learning driver behavior and causal reasoning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7699–7707.

[38] S. M. Erlien, S. Fujita, and J. C. Gerdes, "Shared steering control using safe envelopes for obstacle avoidance and vehicle stability," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 2, pp. 441–451, Feb. 2016, doi: 10.1109/TITS.2015.2453404.

[39] A. Zanten, R. Erhardt, and K. Landesfeind, "VDC systems development and perspective," *J. Passenger Cars*, vol. 107, no. 6, pp. 424–444, 1998.

[40] M. Pant, H. Zaheer, L. Garcia-Hernandez, and A. Abraham, "Differential evolution: A review of more than two decades of research," *Eng. Appl. Artif. Intell.*, vol. 90, Apr. 2020, Art. no. 103479.

SIBING YANG received the dual B.S. degree in energy and power engineering and in computer science from Beijing Jiaotong University, Beijing, China, in 2021, where he is currently pursuing the M.S. degree in power machinery and engineering. His current research interests include predictive control, path tracking, and environment sensing of intelligent vehicles.

CONG GENG received the B.S. degree in vehicle body design and the M.S. degree in vehicle design and manufacturing from Jilin University, Changchun, China, in 1993 and 1996, respectively, and the Ph.D. degree in electrical engineering from Tokyo University, Tokyo, Japan, in 2009. She is currently an Associate Professor with the School of Mechanical Engineering, Beijing Jiaotong University. Her current research interests include vehicle dynamics and control, vehicle energy management, and intelligent vehicle.