

Data-driven Vehicle Modeling for Path Tracking Based on the Combination of a Neural Network and Kinematics Model

Abstract

Autonomous driving systems must safely navigate in increasingly diverse and challenging conditions, which necessitate the incorporation of vehicle dynamic models capable of accurately capturing a vehicle's behavior in diverse conditions. Moreover, these models need to be easily and rapidly developed to meet the needs of rapid autonomous driving software updates. Currently used models have limited accuracy, require extensive parameter tuning, and cannot meet these demands. This paper introduces the Combination of Neural Network and Kinematics Model (CNKM). A neural network is utilized to model the nonlinear characteristics of vehicle subsystems (powertrain, braking, steering, tires) and various unknown factors. It ultimately outputs accelerations that are fed into a planar kinematics model to derive the vehicle states. The neural network is trained using a dataset collected from natural driving. A weighting formula suitable for natural driving data is proposed to mitigate the impact of an uneven dataset distribution. This model is compared with commonly used models under typical and high lateral acceleration scenarios, and the position and heading errors of CNKM are 15.34% and 14.71% of those of the nonlinear dynamic model, respectively.

Keywords Vehicle Modeling · Autonomous Driving · Path Tracking · Nonlinear Characteristics · Natural Driving Dataset · Neural Network

Abbreviations

CNKM	Combination of Neural Network and Kinematics Model
DNN	Deep Neural Network
GRU	Gated Recurrent Units
LSTM	Long Short-Term Memory
LTl	Linear Time Invariant
NDD	Natural Driving Dataset
RNN	Recurrent Neural Network

1 Introduction

Autonomous driving is considered as an effective technical solution of traffic congestion, pollution, and road safety [1]. With the significant improvement of artificial intelligence technology and increased computing power, the autonomous driving is making rapid progress [2, 3].

In autonomous driving systems, precise vehicle models are essential for planning and tracking subsystems to generate safe trajectories and minimize trajectory-following errors [4, 5], and extensive simulation is necessary to validate software functionality before testing in real-world environments [6]. This highlights the importance of an accurate vehicle model to simulate a car's behavior as realistically as possible [7, 8].

As autonomous driving systems continue to advance, they must operate safely in unconventional conditions (e.g., emergency braking, sharp turns with high lateral acceleration) [9], and the control subsystems must perform well in these scenarios [10, 11]. To validate algorithms before real vehicle testing, abundant unconventional condition tests must be considered [12], resulting in the demand for easily constructed models

that accurately portray a vehicle's behavior in emergencies. A straightforward and sufficiently precise model that captures the nonlinear behavior of vehicles would be advantageous from both aspects.

1.1 Related Works

Physics-derived models are widely applied in vehicle control and simulation due to their ease of understanding. These models, based on physical principles and the modeler's empirical insights into the application environment, include kinematic models, linear and nonlinear single-track dynamic models [13], as well as dual-track and multi-body dynamics models. When constructing a physical model, it is crucial to identify which factors to consider and which to disregard based on their significance in the operating environment [14]. For instance, an accurate tire model is vital for the precision of a nonlinear vehicle model, as the forces originating at the tire-road contact patch primarily affect the vehicle motion. Commonly used tire models include the Pacejka, Fiala, brush, and Magic Formula models [15]. Numerous parameters need to be determined before creating a physical model, such as the tire model and the driving and braking system. Obtaining the precise values of the parameters requires a laborious and tedious calibration; some of them are often acquired through specialized experiments and are generally not subject to dynamic adjustment. Allowing for the dynamic adjustment of parameters can enhance accuracy in specific operating conditions [16]. A more accurate model should consider more dynamic factors, which increases complexity [17]. Furthermore, higher-precision physical models generally require more calibration parameters. In summary, constructing a high-quality physics-based vehicle model requires meticulous

consideration of specific factors and extensive parameter calibration.

The intelligence level of mass-produced vehicles has continuously improved, and the abundant driving data has been generated and easily obtained [18]. This makes a data-driven approach possible to identify critical parameters, thereby eliminating the need for extensive calibration [11, 19]. Sebastian et al. [20] verified the feasibility of using long-term natural driving data for vehicle model parameter identification by collecting one year of natural driving data and estimating vehicle model parameters using least squares. The results indicated that long-term natural driving datasets are adequate to identify vehicle dynamics characteristics. Furthermore, once the amount of data used for identification reaches a certain threshold, increasing the size of the dataset further does not significantly improve model accuracy. Hyunki et al. [21] introduced a model parameter identification method based on hyperparameter optimization to enhance the control performance of autonomous vehicles at high speeds. The proposed approach effectively identified parameters such as tire models and engine curves compared to traditional methods. To compare the predictive capabilities of data-driven parameter identification models with physics-based models, Bernardo et al. [22] collected 108 km of driving data from a real vehicle to identify an LTI state-space model. The simple data-driven model outperformed a complex nonlinear physical model under normal driving conditions. However, the aforementioned studies are linear state-space models with a small assumption space and cannot reflect more accurate vehicle dynamics [23].

Neural network technology has advanced rapidly and has been mathematically proven as a universal approximator, capable of approximating arbitrary models given sufficient data [24, 25]. Hence, research has been conducted on vehicle models based on neural networks [26, 27]. In their design of vehicle control algorithms approaching the friction limit, Spielberg et al. [28] developed a DNN-based vehicle dynamics model to circumvent the parameter calibration of the physical model. The model was trained using datasets collected from dry and snow-covered roads and had better prediction accuracy than the physical model on both types of roads. To showcase the modelling capability of neural networks, parametric dynamic models of varying precision have been utilized to produce training data. As the accuracy of the data generation model improves, the error of the neural network model is notably less than that of the identified linear model, indicating a powerful modelling capability [23]. To enhance the precision of vehicle models required for simulation testing in highly dynamic scenarios for autonomous driving systems, Leonhard et al. [9] proposed to use a vehicle model based on GRU networks. The training data was obtained from an autonomous race car. The proposed model could reflect high-order vehicle dynamic features without explicit modelling. Shu et al. [7] proposed an integrated vehicle model architecture combining basic models with

residual networks to build a precise vehicle model in an autonomous driving simulation platform. They tested various residual network architectures, including CNN, DCNN, and Transformer, which significantly improved the prediction accuracy. Yongjun et al. [17] and Mauro et al. [29] constructed longitudinal neural network-based dynamics models which outperformed physical rule-based models. Xiaobo et al. [30] constructed DNN-based longitudinal and lateral vehicle models and compared their predictions with those of a multibody dynamics model. Sibling et al. [31] employed RNN models to couple vehicles' longitudinal and lateral dynamics, which enhanced the predictive accuracy during high-speed steering manoeuvres.

1.2 Motivations and Contributions

Commonly used physics-based vehicle models cannot capture the complex characteristics of vehicles, while detailed physics-based models that consider more dynamic factors are unsuitable for rapid development due to the amount of the parameter calibration. Recent research has shown that data-driven approaches can construct accurate vehicle models with ample driving data. However, vehicle models based on the LTI state space for parameter identification have a limited assumption space and are prone to overfitting when there is a large amount of driving data. With advancements in artificial intelligence, neural networks show potential in vehicle modelling. However, their direct application does not match the characteristics of vehicles, as the inputs have complex nonlinear relationships with the outputs, and there are integral relationships between certain state variables, which face difficulty in fitting. Additionally, contemporary vehicle models using neural networks incorporate multiple historical states as parts of the inputs. Typically, the state variables of vehicles do not exhibit substantial changes within the sampling interval. The connection weights related to the inputs of the historical state tend to be relatively large in the trained network, while connection weights linked to other inputs are easily optimized to 0.

Based on the above analysis, a vehicle modelling method based on the Combination of Neural Network and Kinematics Model (CNKM) is proposed, which considers the vehicle's characteristics and the neural network's modelling capabilities. A natural driving dataset is gathered to train the model's neural network. To mitigate the bias resulting from the uneven distribution of the dataset, weights are assigned to training samples based on the attachment characteristics of the vehicles in natural driving. The proposed model is compared with commonly used vehicle models in normal and close to the adhesion limit conditions. The main contributions of this paper are as follows:

- A vehicle dynamics model that integrates a neural network and a kinematic model is proposed, considering vehicle characteristics and the modelling capabilities of neural networks.

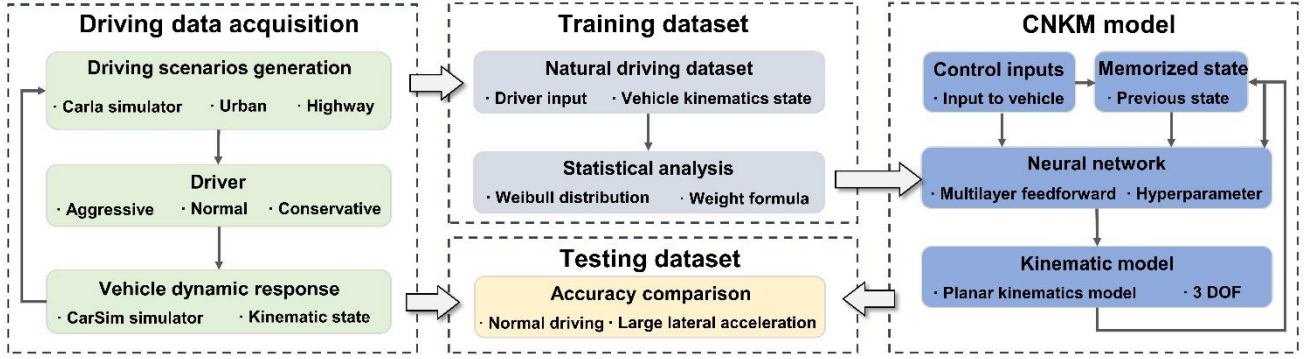


Fig.1 Overview of CNKM and Training Process

- A long-term natural driving dataset is collected and weighting formulas are developed based on distribution characteristics. This addresses model mismatches resulting from non-uniform dataset distributions, thus enabling the use of the natural driving dataset for all-regime vehicle model training.
- The prediction errors of CNKM and commonly used vehicle models are compared under normal and close to vehicle adhesion limits conditions.

The remainder of this paper is structured as follows. Section 2 covers the problem statement. Section 3 introduces the proposed CNKM, as shown on the right side of Fig. 1. Section 4 describes the natural driving dataset acquisition process and the training of CNKM, as shown at the left side of Fig. 1. In Section 5, the performance of the model is compared with commonly used models. The paper concludes with Section 6.

2 Problem Statement

The vehicle dynamics model can be generally described as

$$x_{t+1} = f(x_t, u_t) \quad t = 0, 1, \dots, N \quad (1)$$

where t is the sampling step, $x_t \in R^n$ is the n -dimensional state vector, $u_t \in R^m$ is the m -dimensional control input vector, $f(\cdot)$ is the state transition function of the vehicle system, and N is the total number of predicted steps.

Given initial state x_0 and control sequence $u_t (t = 0, 1, \dots, N)$, iteratively applying the state transition function yields the predicted state sequence $x_t (t = 1, 2, \dots, N+1)$.

The process of vehicle dynamics modelling is to find the best system state transition function $f(\cdot)$ in a given application environment, so that the error between the predicted state sequence x_t and the real vehicle state is the smallest.

The vehicle's vertical motion is discarded in this paper, and the motion is simplified to a two-dimensional plane

with three degrees of freedom. The state vector can be expressed as

$$x = [X, Y, \varphi, v_x, v_y, r] \quad (2)$$

where X, Y represent the position of the vehicle in the inertial system. φ is the heading angle, and v_x, v_y, r are the respective longitudinal velocity, lateral velocity, and yaw rate in the body coordinate system. The control input vector is

$$u = [\delta, T, P] \quad (3)$$

where δ is the steering wheel angle, T is the drive torque request, and P is the pressure of the master cylinder in the braking system.

3 CNKM Model

Based on the theory of vehicle dynamics, the driver's control inputs are processed through the vehicle's different subsystems to generate forces on the tires' contact patches with the ground, which primarily impact the vehicle's movement. The vehicle's relative movement with the air leads to additional aerodynamic forces, which together constitute the overall external forces influencing the motion of the vehicle disregarding the impact of longitudinal incline. The ratio of the cumulative external forces at the center of mass to the total vehicle mass determines the vehicle's acceleration, and the ratio of the torque around the center of mass to the vehicle's moment of inertia determines its angular acceleration. Integrating the acceleration offers an accurate evaluation of the vehicle's state.

The relationship between the driver's control inputs and the tire forces is inherently complex and nonlinear, influenced by a combination of the powertrain, braking system, steering system, suspension, and tire dynamics. Developing mathematical models for each subsystem presents significant challenges, often leading to the neglect of some contributing factors. The most commonly used approach is based on fitting observed data from specific experiments, such as the Magic Formula tire models. Integrating these diverse subsystems inevitably involves simplifications, which introduce potential sources of error.

In contrast, once the quantities of acceleration and angular acceleration are obtained, the derivation of

vehicle states involves an integration process with no unknown or challenging-to-model factors at play. Any errors primarily arise from the integration method itself.

CNKM is proposed based on the analysis of fundamental dynamics mentioned above. Initially, a neural network model is employed to capture the complex nonlinear mapping between the driver's input and the overall vehicle acceleration. The end-to-end model encompasses all interactions between the vehicle subsystems and the ground. Upon obtaining the acceleration quantities, the vehicle's state variables are derived through the integration of the planar kinematic vehicle model.

3.1 Neural Network Part

Given that the vehicle system itself is an inertial system, the states at the current and past moment are closely related. Therefore, the input of the neural network model includes the states of multiple previous time steps. The input for a single time step is expressed as

$$h_t = [v_{x,t}, v_{y,t}, r_t, \delta_t, T_t, P_t] \quad (4)$$

The overall input and output of the neural network model are expressed as

$$a_t = f_{nn}([h_t, h_{t-1}, h_{t-2}, \dots, h_{t-n}]) \quad (5)$$

$$a_t = [a_{x,t}, a_{y,t}, a_{vz,t}] \quad (6)$$

where $f_{nn}(\cdot)$ is a multilayer feedforward neural network, and a_t is the output, which is a vector composed of the longitudinal acceleration $a_{x,t}$, lateral acceleration $a_{y,t}$, and yaw angular acceleration $a_{vz,t}$.

A fully connected (FC) layer is expressed as

$$a_i = \text{act}(W_i a_{i-1} + b_i) \quad (7)$$

Where a_i , a_{i-1} , W_i and b_i are the output, input, weight matrix, and bias vector, of layer i , respectively, and $\text{act}(\cdot)$ is an activation function, chosen as Leaky ReLU, as shown in Fig. 2. When the input is less than zero, the slope of the activation function is nonzero, ensuring that neurons will not die. In addition, the calculation is faster than that of the nonlinear activation function.

$f_{nn}(\cdot)$ consists of multiple FC layers. The number of layers and neurons is determined during hyperparameter tuning (see Section 4).

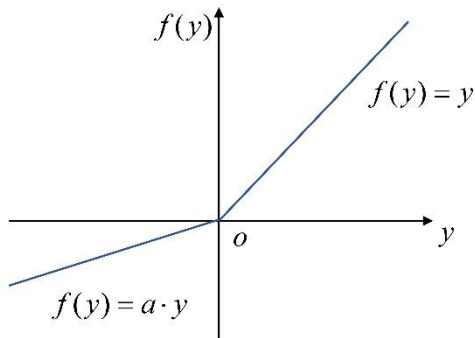


Fig.2 Leaky ReLU activation function

3.2 Kinematics Model Part

When the height of the vehicle's center of mass is ignored, the vehicle is regarded as a rigid body moving in a two-dimensional plane (see Fig. 3). Its states of motion are described by Eq. (2), and the relationships between each state are described as

$$\begin{aligned} \dot{X} &= v_x \cos \varphi - v_y \sin \varphi \\ \dot{Y} &= v_x \sin \varphi + v_y \cos \varphi \\ \dot{\varphi} &= r \\ \dot{v}_x &= r v_y + a_x \\ \dot{v}_y &= -r v_x + a_y \\ \dot{r} &= a_{vz} \end{aligned} \quad (8)$$

where a_x , a_y , a_{vz} are inputs to the model. Eq. (8) can be simplified to

$$\dot{x}_t = f_{km}(x_t, a_t) \quad (9)$$

which can be discretized with Euler integration to yield the following equation, used to determine the state x_{t+1} given both a_t and x_t

$$x_{t+1} = x_t + \Delta t \cdot f_{km}(x_t, a_t) \quad (10)$$

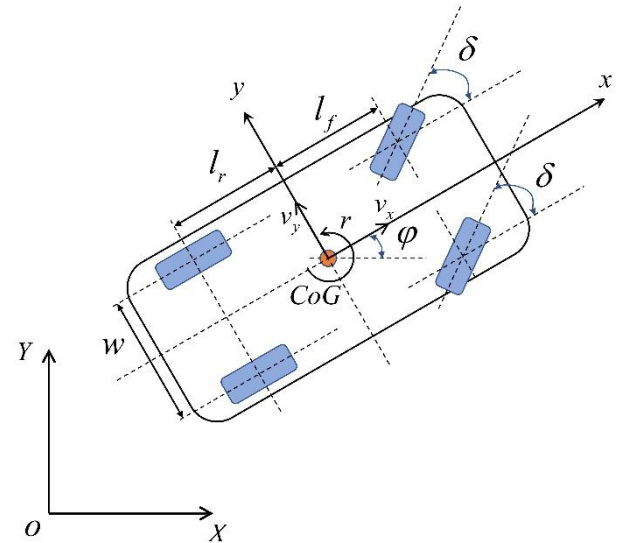


Fig.3 Schematic diagram of planar vehicle model

3.3 Structure of CNKM

Fig. 4 shows the structure of the CNKM model. The current control input $u_t = [\delta, T, P]$ is concatenated with the velocity to generate state h_t , which is then concatenated with the previously stored state $[h_{t-1}, h_{t-2}, \dots, h_{t-n}]$ as the input of the neural network.

The output of $f_{nn}(\cdot)$ is the acceleration vector $[a_x, a_y, a_{yz}]$, which is then input to the kinematics model. The state at the next moment is calculated using Euler integration.

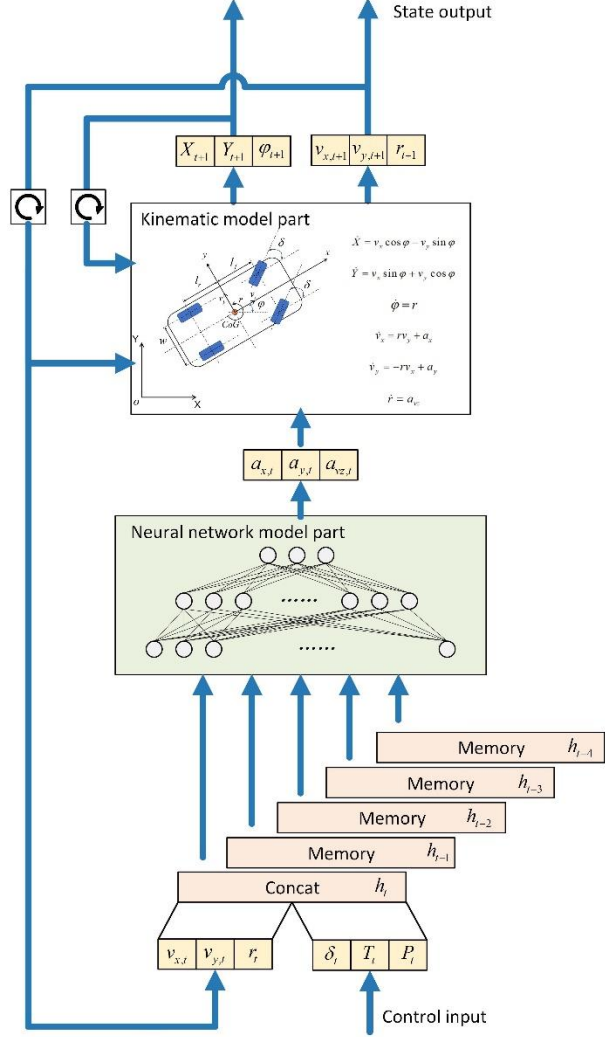


Fig.4 Structure of CNKM

Combining Eq. (5) and (10) results in the recursive dynamics equation

$$\begin{aligned} x_{t+1} &= f_{CNKM}(x_t, u_t) \\ &= x_t + \Delta t \cdot f_{km}(x_t, f_{nn}([h_t, h_{t-1}, \dots, h_{t-n}])) \end{aligned} \quad (11)$$

4 Data Collection and Model Training

In this section, the collection of the training data and the training process of the neural network in CNKM are introduced. Firstly, the approach to gather the natural driving dataset (NDD) is outlined. Then, the characteristics of the collected NDD are analyzed, and a suitable weighting formula is proposed. This formula adapts the weights of training examples based on ground adhesive ratio features, facilitating the use of imbalanced NDD in vehicle dynamic modeling. The best hyperparameters are chosen through grid search and are used to train the network.

4.1 Data Collection Method

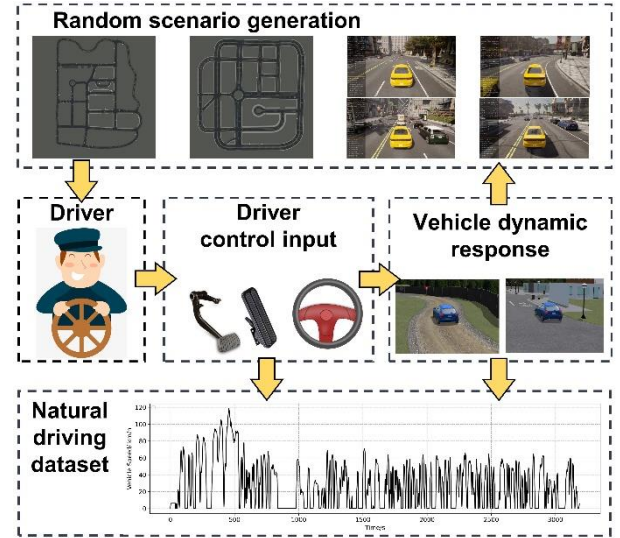


Fig.5 NDD data acquisition process

A natural driving data collection platform was built using the autonomous driving simulation software Carla 0.9.13 [32], CarSim 2019.0, and a driving simulator (see Fig. 5). Carla was employed to create random driving scenarios, which were visualized and presented on the driving simulator. The driver's interactions in the simulator were inputted to CarSim to simulate the vehicle's dynamic response, which was returned to Carla to visualize the actual vehicle posture. The driver's inputs and the vehicle's dynamic response were recorded as the natural driving dataset.

Table 1. Basic Vehicle Parameters

Parameter	Value	Unit
Mass	1270.000	kg
Wheelbase	2.910	m
Distance from front axle to CoG	1.015	m
Distance from rear axle to CoG	1.895	m
Centroid height	0.540	m
Front track	1.675	m
Rear track	1.675	m
Yaw moment of inertia	1536.700	kg·m ²
Steering ratio	17.500	-

The driving scenarios encompassed urban roads and highways, with other traffic participants showing the random behavior. Table 1 shows the basic parameters of the vehicle model, while the gathered physical quantities are listed in Table 2, with a sampling frequency of 2000 Hz. Three drivers, representing aggressive, conservative, and normal driving styles were chosen for data collection, each with approximately 36 hours of driving duration.

Table 2. Recorded Physical Quantities

Physical Quantities	Symbol	Unit
Steering wheel angle	δ	$^{\circ}$
Drive request torque	T	N·m
Brake master cylinder pressure	P	Mpa
Longitudinal speed	v_x	km/h
Lateral speed	v_y	km/h
Yaw rate	r	$^{\circ}/s$
Longitudinal acceleration	a_x	g
Lateral acceleration	a_y	g
Yaw acceleration	a_z	rad/s^2

4.2 NDD Statistical Analysis

The total duration of the dataset was 106.7 hours. Fig.6 illustrates the probability density distribution with respect to longitudinal and lateral acceleration, where the black envelope denotes the vehicle's adhesion limit. The data distribution is unbalanced, with a higher probability density observed closer to the origin. Furthermore, approximately 90% of the data falls within the lateral acceleration range of $\pm 0.25g$, as shown in the green area of Fig.6 .

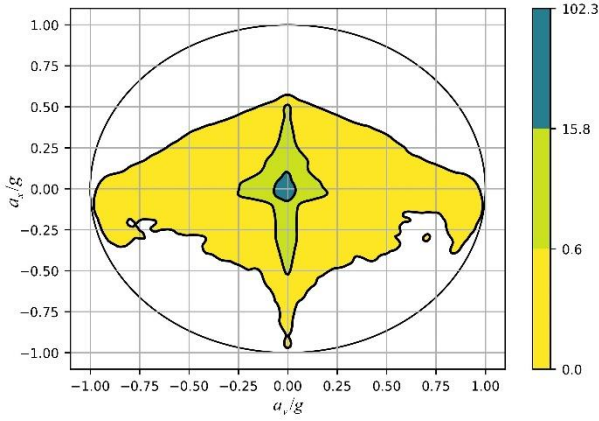


Fig.6 Distribution density of dataset with longitudinal and lateral acceleration

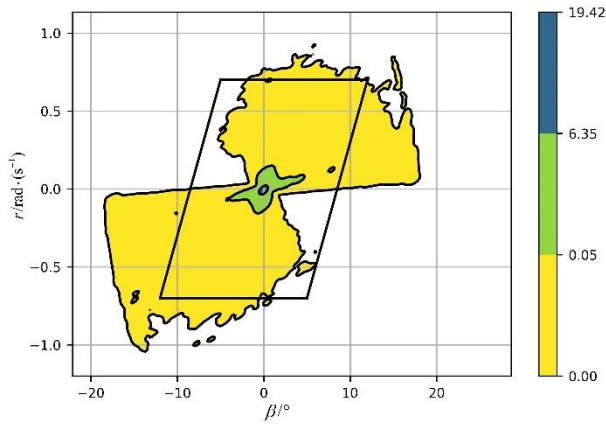


Fig.7 Distribution density of dataset within stable envelope region

Fig. 7 illustrates the dataset distribution with respect to the sideslip angle and yaw rate. The black envelope represents the vehicle's stable handling region. It is evident that some of the data points fall beyond this, and approximately 90% of the data falls in the green area.

4.3 NDD Weighting Formula

The above analysis indicates that the distribution of NDD is uneven. Directly training a model on this dataset may result in a bias toward small longitudinal and lateral acceleration. To address this issue, we propose a weighting formula based on the road adhesion ratio to assign weights to each training sample. On a road surface with a consistent coefficient of friction, the road adhesive ratio is directly proportional to the combined longitudinal and lateral forces and, hence, to the combined acceleration (the vector sum of the longitudinal and lateral acceleration).

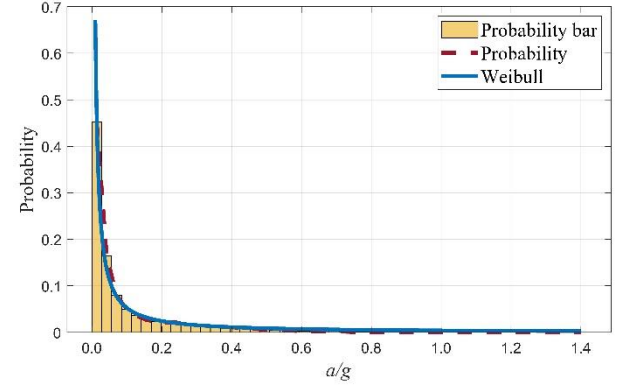


Fig.8 Distribution probability of training data with vehicle acceleration

Fig. 8 illustrates the probability distribution of the training data with respect to combined acceleration. The range of combined acceleration is divided into 50 equidistant bins. The samples within each bin are counted, and their distribution frequency is calculated. The red line is the distribution frequency curve, which is fit with a Weibull distribution:

$$f(x, \lambda, k) = \begin{cases} \frac{k}{\lambda} \left(\frac{x}{\lambda}\right)^{k-1} & x \geq 0 \\ 0 & x < 0 \end{cases} \quad (12)$$

The fitting result is $k = 0.04707$, and $1/\lambda^k = 3.812$. The blue line in Fig. 8 is the fitted Weibull distribution curve. Then, the weighting formula is derived as

$$w = \frac{1}{f(a)} \quad (13)$$

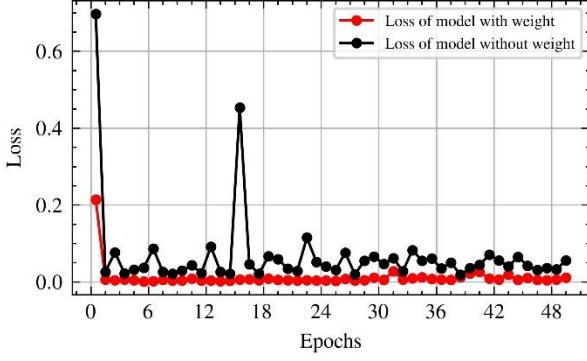


Fig.9 Training process with and without weights

Fig. 9 shows the training processes for models with and without weights. It is apparent that the assignment of weights to samples leads to a notable reduction in the mean squared error on the test set and maintains it at a lower level. The average loss is merely 17.27% of that without the application of weights.

4.4 Model Training

The model was implemented and trained using PyTorch 2.0 with Python version 3.7. The hardware was equipped with AMD R7 4800H CPU, Nvidia 2060 GPU, and 16 GB of DDR4 memory.

The NDD was partitioned into 5,915,401 samples, with 15% reserved for validation. The loss function was

$$J = \frac{1}{N} \sum_{i=1}^N (a_i - \hat{a}_i)^T (a_i - \hat{a}_i) \quad (14)$$

Model training used the Adam optimizer. Hyperparameters, such as the number of hidden layers, units per layer, and learning rate, were chosen through grid search. Table 3 displays the final model parameters.

Table 3. Neural Network Hyperparameter Settings

Hyperparameter Group	Hyperparameter	Value
Neural Network Architecture	hidden layers	3
	units on first layer	256
	units on second layer	512
	units on third layer	256
	number of parameters	272,899
	activation function	Leaky ReLU
Training Process	batch size	64
	initial learning rate	0.001
	early stop patience	20
Training Algorithm	gradient decay factor	0.900
	squared gradient decay factor	0.990
	bias	1×10^{-8}

After hyperparameter tuning, the model was retrained on the complete dataset. The final training process is shown in Fig. 10.

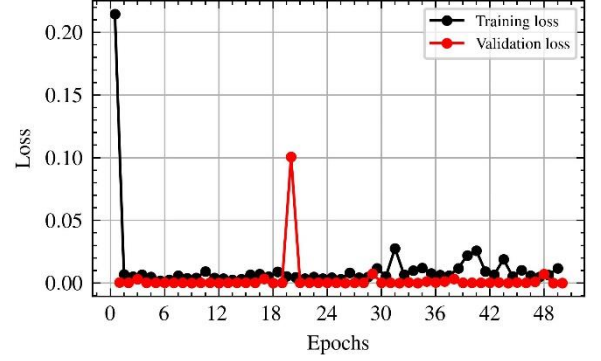


Fig.10 Final training process of neural network

5 Model Evaluation

CNKM was evaluated using both open-loop and receding horizon methods, and compared with linear and nonlinear models, as well as LSTM-based, GRU-based, and NN-based models. Furthermore, the effects of lateral acceleration, longitudinal velocity, and sampling time on model performance were investigated. Below is a brief introduction of the baseline models.

The linear dynamics model assumes a small steering wheel angle, a small slip angle, and a linear tire model, and is expressed in a standard state space form as

$$\dot{x} = Ax + Bu \quad (15)$$

$$A = \begin{bmatrix} \frac{2(C_f + C_r)}{mv_x} & \frac{2(C_f l_f + C_r l_r)}{mv_x} & 0 \\ \frac{2(C_f l_f + C_r l_r)}{I_z v_x} & \frac{2(C_f l_f^2 + C_r l_r^2)}{I_z v_x} & 0 \\ 0 & 0 & \frac{\bar{\kappa}_D}{m} \end{bmatrix} \quad (16)$$

$$B = \begin{bmatrix} \frac{\kappa_e}{m} & -\frac{\kappa_b}{2m} & -\frac{2C_f}{m} & 0 & 0 \\ \frac{\kappa_e l_f}{I_z} & -\frac{\kappa_b l_f}{2I_z} & -\frac{l_f C_f}{I_z} & 0 & 0 \\ 0 & 0 & 0 & \frac{\kappa_e}{m} & -\frac{\kappa_b}{m} \end{bmatrix} \quad (17)$$

with the state vector

$$x = [v_y, r, v_x]^T \quad (18)$$

And the input vector

$$u = [T_e \delta, P_b \delta, \delta, T_e, P_b]^T \quad (19)$$

where v_y and v_x are the respective lateral and longitudinal velocity at the center of mass of the vehicle, r is the yaw rate, T_e is the driving torque at the wheel, P_b is the brake pressure, and δ is the front wheel steering angle.

The nonlinear dynamic model considers the conditions at large tire slip angles and integrates a nonlinear tire model to couple the longitudinal and lateral characters. The nonlinear model is described as

$$\begin{aligned}
\dot{v}_x &= \frac{1}{m} ((F_{x,fl} + F_{x,fr}) \cos \delta + F_{x,rl} + F_{x,rr} \\
&\quad + (F_{y,fl} + F_{y,fr}) \sin \delta - F_D) + v_y r \\
\dot{v}_y &= \frac{1}{m} ((F_{y,fl} + F_{y,fr}) \cos \delta + F_{y,rl} + F_{y,rr} \\
&\quad + (F_{x,fl} + F_{x,fr}) \sin \delta) - v_x r \\
\dot{r} &= \frac{1}{I_z} (((F_{y,fl} + F_{y,fr}) \cos \delta + (F_{x,fl} + F_{x,fr}) \sin \delta) l_f \\
&\quad - (F_{y,rl} + F_{y,rr}) l_r + (F_{x,fr} - F_{x,fl} + F_{x,rr} - F_{x,rl}) \frac{W_b}{2})
\end{aligned} \quad (20)$$

where W_b is the wheelbase, and $F_{(x,y),ij}$ is the longitudinal or lateral force of the corresponding wheel, which is calculated using the Dugoff tire model [33].

The LSTM and GRU models are suitable for processing sequential data and their inputs and outputs are represented as

$$x_{t+1} = f_{\text{Seq}}([h_t, h_{t-1}, h_{t-2}, h_{t-3}, h_{t-4}]) \quad (21)$$

where the definitions of x and h are consistent with CNKM. The number of cells and layers are determined by the hyperparameters tuning [34].

An end-to-end NN-based model has been developed for comparison. The model's input and output are as shown as

$$x_{t+1} = f_{\text{NNE2E}}([h_t, h_{t-1}, h_{t-2}, h_{t-3}, h_{t-4}]) \quad (22)$$

where the input is consistent with the CNKM model and the output represents the required state variables rather than the acceleration intermediates used in the CNKM model. Thus, the contribution of the kinematic model part to the overall performance of the CNKM model is highlighted. The internal structure of the model consists of fully connected hidden layers, with the number of hidden layers and units in each layer determined by the hyperparameter tuning.

5.1 Open-loop Evaluation

The open-loop comparison uses a model to forecast the system's state given the initial state and a sequence of control inputs over a specific period and then compares the predicted and actual states. Separate assessments of each model's longitudinal and lateral modeling capabilities were first conducted, followed by the selection of a segment from the test dataset to simultaneously evaluate their longitudinal and lateral performance.

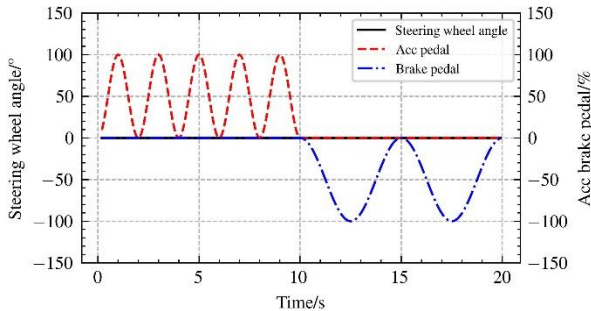


Fig.11 Inputs of longitudinal evaluation condition

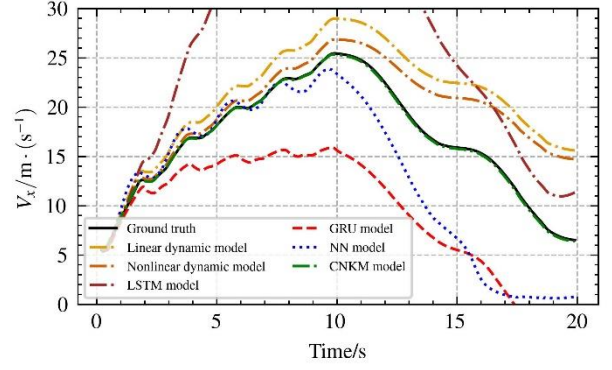


Fig.12 Validation of longitudinal velocity in longitudinal evaluation condition

5.1.1 Longitudinal Dynamics

Fig.11 illustrates the longitudinal evaluation condition with the steering wheel input set at 0. The initial velocity is set at 20 km/h. The first 10 seconds demonstrates an acceleration segment with five cycles of sinusoidal fluctuation in the accelerator pedal, followed by a deceleration segment including two cycles of the brake pedal fluctuation in the subsequent 10 seconds.

Fig.12 presents the predicted longitudinal velocity for each model. The LSTM, GRU, and NN models show marked deviations from the actual values due to their sensitivity to cumulative error. The linear and nonlinear models display a bias in their predicted values compared to the actual values, which is attributed to omitting critical factors in the modelling process. CNKM strongly aligns with the actual velocity curve and delivers precise predictions.

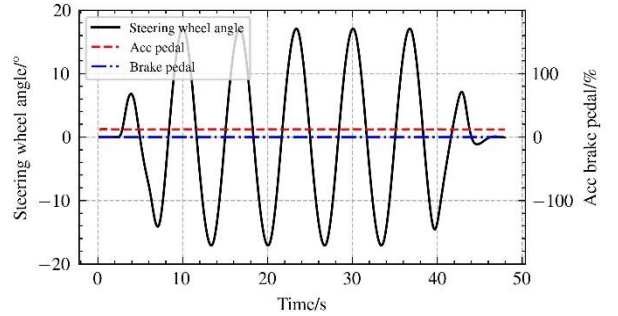


Fig.13 Inputs of lateral evaluation condition

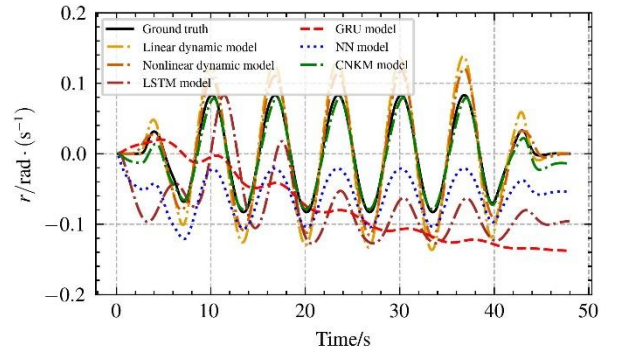


Fig.14 Validation of yaw rate in lateral evaluation condition

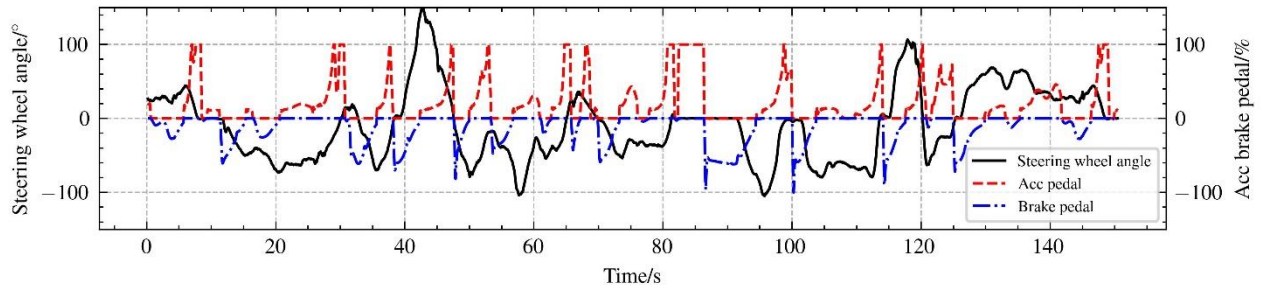


Fig.15 The input for combined longitudinal and lateral evaluation

5.1.2 Lateral Dynamics

To evaluate the lateral dynamics modelling capability, a S-shaped slalom maneuver was performed at a speed of 65 km/h, as illustrated in Fig. 13, with the steering wheel angle showing a sinusoidal oscillation pattern. The accelerator pedal was opened slightly to maintain a constant longitudinal velocity.

Fig.14 displays the predicted yaw rate for each model. The LSTM, GRU and NN models, which are susceptible to cumulative error, quickly show substantial deviation after the initial segment. The linear dynamics model, which assumes a small steering wheel angle, generates large peaks. The nonlinear dynamics model performs better than the linear model. CNKM effectively forecasts yaw rate values, with its trend closely matching the actual values.

5.1.3 Combination of Longitudinal and Lateral

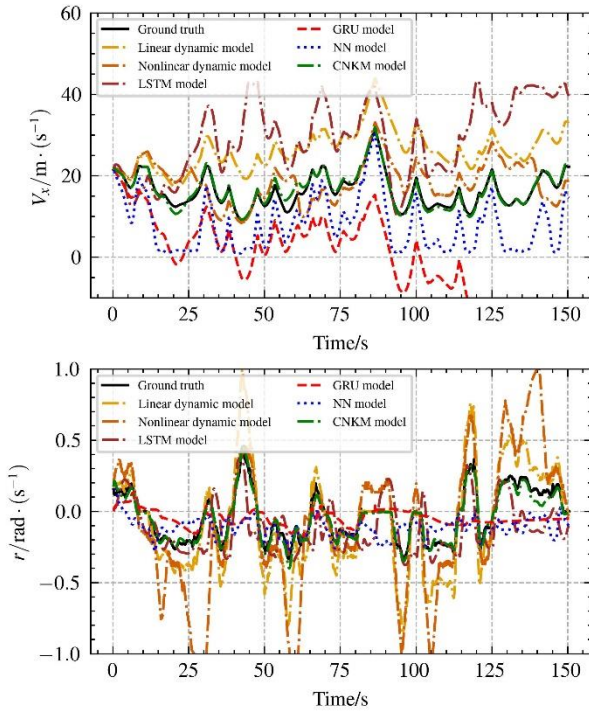


Fig.16 Validation in longitudinal velocity and yaw rate

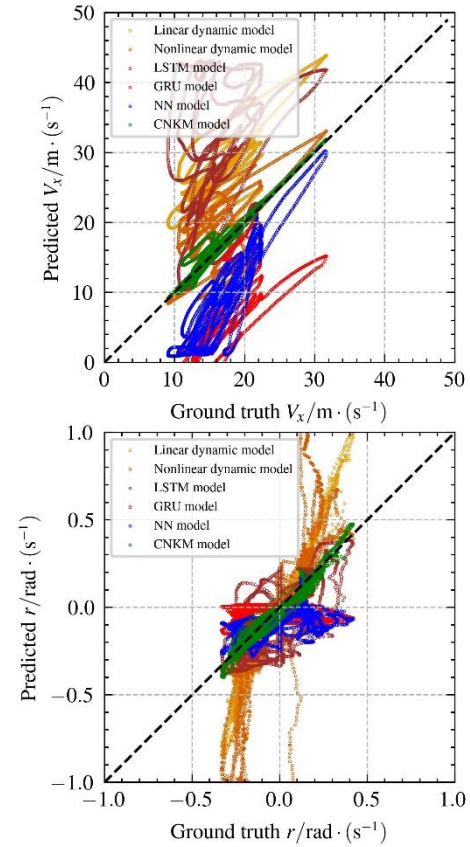


Fig.17 Correlation between ground truth and predicted states over test lap

A 150-second segment from the test dataset was selected to assess the capability of longitudinal and lateral modelling simultaneously. The input of the vehicles is depicted in Fig.15.

Errors in the longitudinal velocity and yaw rate were assessed. Fig.16 clearly shows that CNKM accurately estimates the longitudinal velocity and yaw rate, representing the longitudinal and lateral characteristics, respectively. The LSTM, GRU and NN models exhibit notable errors attributed to error accumulation and an inadequate longitudinal modelling capability. The fundamental reason is that the three models output the state values of the next moment. Since a car is an inertial object, state values between consecutive moments exhibit minimal changes. This leads to the strong dependency on the previous state in the models and the heightened sensitivity to cumulative errors. Due to unaccounted factors in longitudinal modelling, both linear and nonlinear models show biases compared to

actual values. Notably, the nonlinear model demonstrates superior performance over the linear model.

In Fig.17, a correlation plot illustrates the association between each model's predicted and actual values. Table 4 displays the mean errors for each model over the test lap.

Table 4. Mean Errors of Each Model

Model	Longitudinal velocity	Yaw Rate
Linear Dynamic Model	9.92433	0.0365092
Nonlinear Dynamic Model	2.27361	0.0740135
LSTM Model	12.86140	0.0774324
GRU Model	19.82180	0.0133235
NN Model	8.37460	0.0780873
CNKM	0.143614	0.0122606

5.2 Receding Horizon Evaluation

The model performance within the receding horizon was evaluated, which is a common pattern in model predictive control. The vehicle was operated by the driver in specific scenarios, and its control commands, actual responses, and driving trajectory were recorded. Then, the vehicle's initial state and future control actions were input to the model offline to simulate in a receding horizon fashion over a horizon of $T = 2.5s$. The predicted and actual trajectories of the vehicle were compared. The model operated with a time interval of 50ms. The model performance was thoroughly evaluated in urban driving and track racing scenarios.

5.2.1 Evaluation in Urban Driving

In urban driving scenarios, the vehicle maintained a speed below 70 km/h, with the longitudinal and lateral acceleration less than 0.2g. Fig.18 illustrates the actual trajectory and speed of the vehicle under typical driving conditions, where the speed is visually represented through color variation. Two points were chosen to illustrate the error between the predicted and actual trajectories.

Fig.19 illustrates the positional and yaw errors at the 20th step. It is apparent that, aside from the linear dynamic model, the forecast errors of the remaining models are insignificant. Of particularly notable is the abrupt surge in the prediction error of the linear dynamic model at the U-shaped road section, attributed to the large steering wheel angle of the vehicle at this location, which leads to a significant error induced by the linear tire model and the small steering angle assumption.

By the time the prediction reaches the 50th step (Fig.20), the errors of each model show notable variations, primarily due to cumulative errors. The LSTM, GRU and NN models also show a substantial increase in position error, attributed to their constrained modelling capacity in the longitudinal motion.

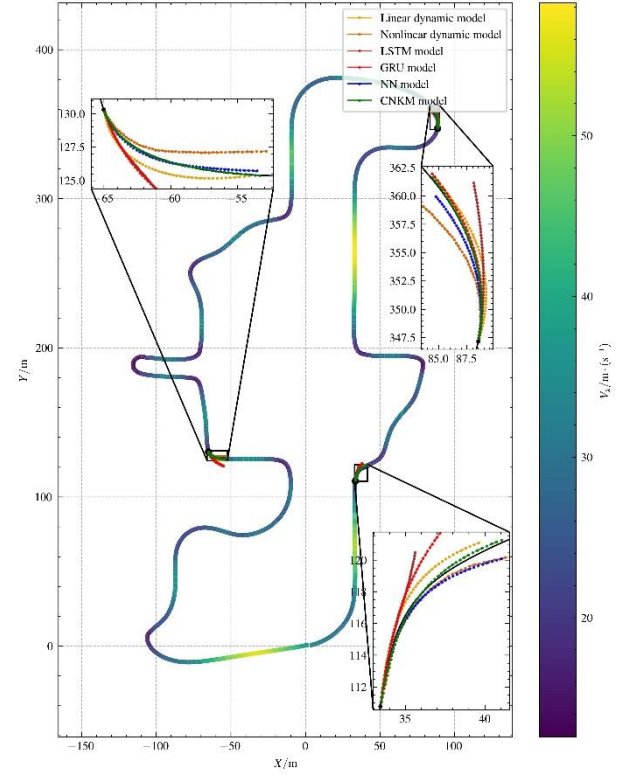


Fig.18 Actual trajectory and speed of vehicle under urban driving scenario

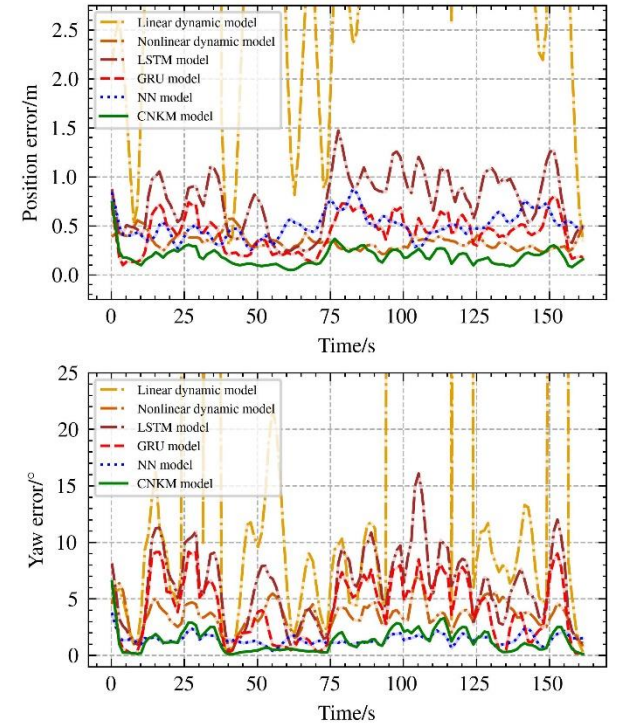


Fig.19 Errors of position and heading at 20th step

In summary, except for the linear dynamic model, the prediction errors of the models are generally comparable. The performance of CNKM closely approximates that of nonlinear vehicle dynamic models.

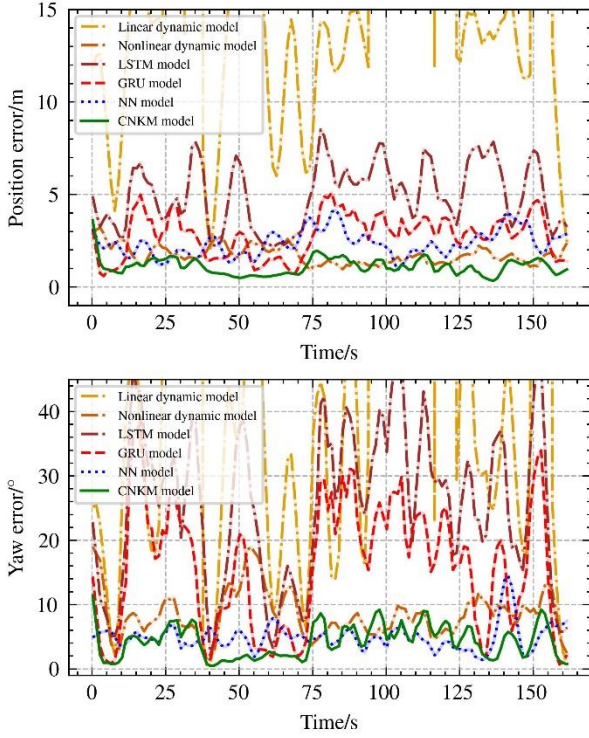


Fig.20 Errors of position and heading at 50th step

5.2.2 Evaluation in Large Lateral Acceleration Scenario

In track racing scenarios, the maximum speed approaches 130 km/h, and the maximum longitudinal acceleration can reach 1g, with the maximum lateral acceleration at $\pm 0.7g$. Fig. 21 depicts the actual trajectory and speed of the vehicle in this large lateral acceleration condition.

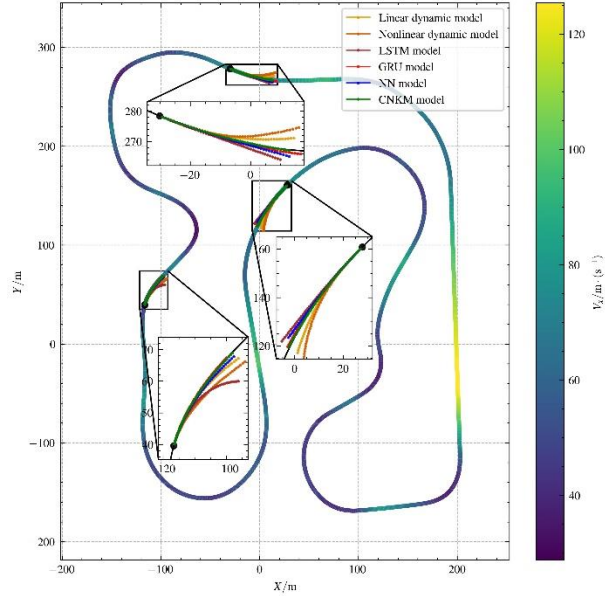


Fig.21 Actual trajectory and speed of vehicle under large lateral acceleration

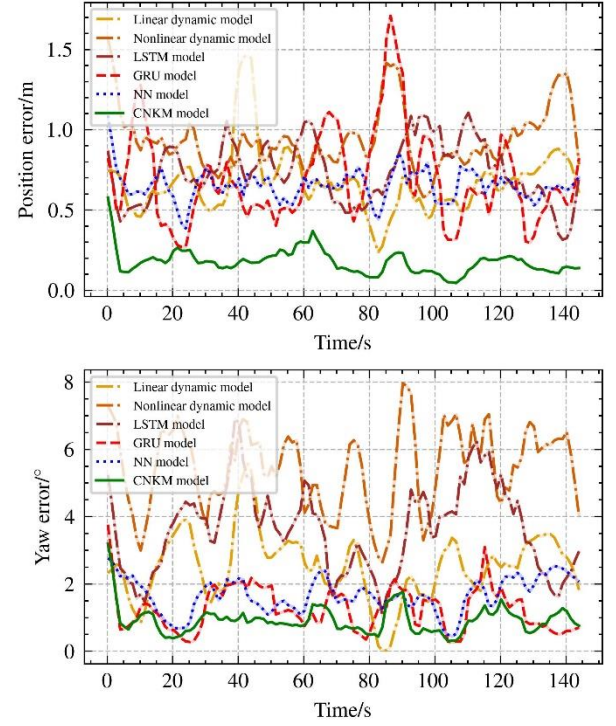


Fig.22 Errors of position and heading at 20th step

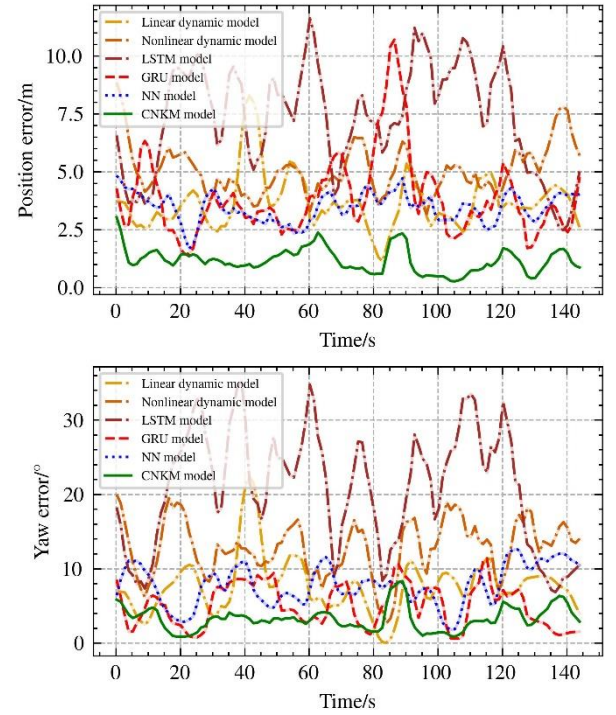


Fig.23 Errors of position and heading at 50th step

Fig.22 illustrates the positional and yaw error at the 20th step. The errors of the proposed model are smaller than those of the other models. The LSTM, GRU and NN models demonstrate larger positional errors. However, the higher vehicle speed and smaller steering angles in this scenario result in a return to normally predictive accuracy for the linear dynamic model. As the model predicts to the 50th step (Fig.23), the absolute prediction

error increases due to the influence of the cumulative error, accompanied by noticeable fluctuations.

In summary, when the vehicle operates in conditions with significant nonlinear characteristics, the commonly used models based on physical principles show large prediction errors. The proposed vehicle model outperforms the other models in such conditions.

5.3 Effect of Lateral Acceleration

Utilizing the data from the above scenarios, a statistical analysis of the prediction errors concerning lateral acceleration was performed, as displayed in Fig. 24.. It is evident that as the lateral acceleration increases, the prediction errors for both the position and heading of the linear dynamic model also increase. Once the lateral acceleration surpasses 0.1g, the errors of the linear dynamic model increase significantly, making them incomparable with the errors of other models. When the lateral acceleration exceeds 0.4g, the heading error of these four models increases, among which CNKM has a comparatively lower level. The results indicate that CNKM effectively captures the vehicle's nonlinear characteristics under high lateral acceleration.

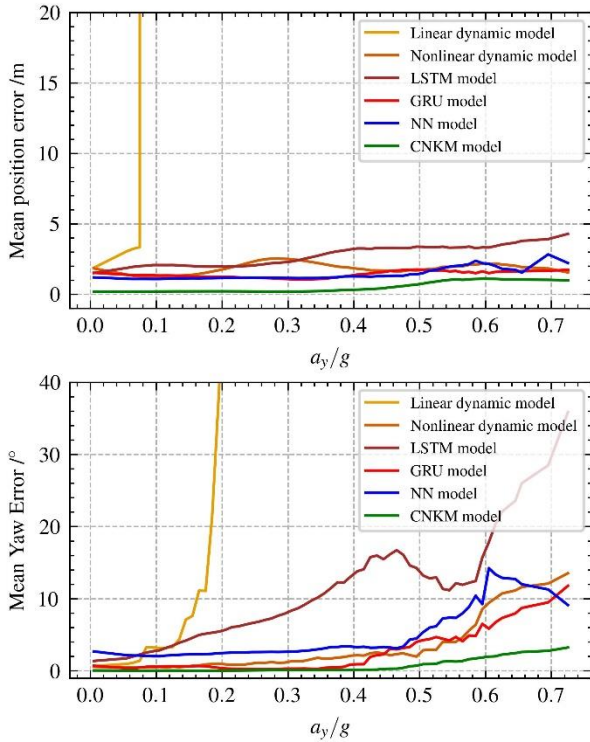


Fig.24 The relationship between error and lateral acceleration

5.4 Effect of Longitudinal Velocity

As illustrated in Fig. 25, the impact of longitudinal velocity on the prediction errors was analyzed. Estimation errors are relatively large across all models at low speeds, with the linear vehicle model exhibiting the highest error. This is primarily due to the significant steering wheel inputs often associated with low-speed

driving, causing the linearization assumption of the linear model invalid. As longitudinal velocity increases, the prediction errors of all models tend to stabilize. Notably, the errors of the CNKM model are minimally affected by longitudinal velocity, maintaining consistent stability, which indicates that the CNKM model effectively captured the distinct dynamic characteristics of the vehicle at both high and low speeds.

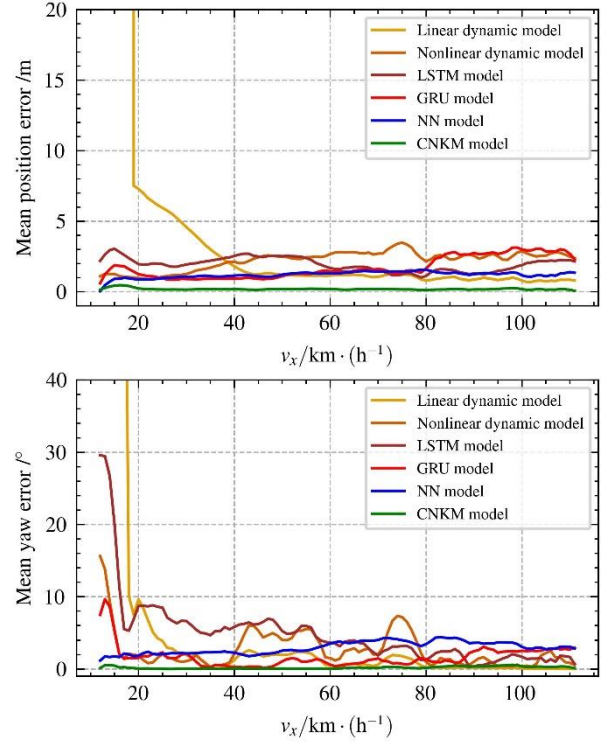


Fig.25 The relationship between error and longitudinal velocity

5.5 Effect of Sampling Time

To analyze the error caused by discretization, two additional sampling times were utilized to calculate each model's average prediction error.

Table 5. Mean Position Errors of Each Model (m)

Model		10ms	50ms	100ms
Linear Model	Dynamic	0.06188360	6.5×10^{32}	2.6×10^{58}
Nonlinear Model	Dynamic	0.09509120	1.530350	6.622470
LSTM Model		0.06994860	2.563176	7.829292
GRU Model		0.03799730	1.413220	4.961330
NN Model		0.01564880	1.329190	2.937680
CNKM		0.00897374	0.234747	1.412802

Tables 5 and 6 present the average prediction errors of the four models under three sampling intervals. It is evident that as the sampling interval increases, the prediction error increases exponentially. When the sampling time is 10ms, the linear dynamic model

demonstrates a slight performance advantage over the nonlinear dynamic model. As the sampling time increases, the gap between them gradually diminishes, and the nonlinear dynamic model significantly outperforms the linear dynamic model. CNKM is similarly influenced by the sampling time. Nevertheless, its performance remains better than the other four at each interval.

Table 6. Mean Yaw Errors of Each Model (°)

Model		10ms	50ms	100ms
Linear Model	Dynamic	0.6209860	98.649100	1570.46000
Nonlinear Model	Dynamic	0.9521300	6.269310	16.63410
LSTM Model		0.5581470	10.341400	20.41420
GRU Model		0.3732240	4.641810	14.61200
NN Model		0.1588710	3.091620	15.77820
CNKM		0.0428300	0.921988	3.67879

6 Conclusion

This paper introduces a data-driven vehicle modelling scheme that integrates neural networks and kinematics models. The neural network captures the vehicle's nonlinear characteristics and various unknown factors. Training based on natural driving data substitutes the conventional parametric vehicle modelling approach. The acceleration values produced by the neural network model are converted into state quantities through a planar kinematics model.

A weighting formula is proposed based on the statistical characteristics of natural driving data. Different weights are applied to training samples to address dataset imbalances and to reduce errors in the neural network, which allows the use of extensive unbalanced natural driving datasets for vehicle model training. Assigning weights to training samples significantly reduced the model's training error to only 17.27% of the original and remained stable.

In open-loop tests, the CNKM showed prediction errors of 6.32% for longitudinal velocity and 16.57% for yaw rate compared to the nonlinear model. In normal driving and large lateral acceleration conditions, the CNKM consistently demonstrated low prediction errors, maintaining performance stability even under significant lateral forces. The position and heading errors of the CNKM were 15.34% and 14.71%, respectively, compared to those of the conventional nonlinear dynamic model. These statistical results underscore CNKM's proficiency in accurately capturing the nonlinear dynamics of vehicles.

Compliance with Ethical Standards

Conflict of interest On behalf of all the authors, the corresponding author states that there is no conflict of interest.

References

1. Chen L, Li Y, Huang C, et al (2023) Milestones in Autonomous Driving and Intelligent Vehicles: Survey of Surveys. *IEEE Trans Intell Veh* 8:1046–1056. <https://doi.org/10.1109/TIV.2022.3223131>
2. Muhammad K, Ullah A, Lloret J, et al (2021) Deep Learning for Safe Autonomous Driving: Current Challenges and Future Directions. *IEEE Trans Intell Transport Syst* 22:4316–4336. <https://doi.org/10.1109/TITS.2020.3032227>
3. Wang H, Feng Y, Tian Y, et al (2024) Towards the Next Level of Vehicle Automation Through Cooperative Driving: A Roadmap From Planning and Control Perspective. *IEEE Transactions on Intelligent Vehicles* 9:4335–4347. <https://doi.org/10.1109/TIV.2024.3363873>
4. Teng S, Hu X, Deng P, et al (2023) Motion Planning for Autonomous Driving: The State of the Art and Future Perspectives. *IEEE Trans Intell Veh* 8:3692–3711. <https://doi.org/10.1109/TIV.2023.3274536>
5. Paden B, Cap M, Yong SZ, et al (2016) A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles. *IEEE Trans Intell Veh* 1:33–55. <https://doi.org/10.1109/TIV.2016.2578706>
6. Varga B, Doba D, Tettamanti T (2023) Optimizing vehicle dynamics co-simulation performance by introducing mesoscopic traffic simulation. *Simulation Modelling Practice and Theory* 125:102739. <https://doi.org/10.1016/j.simpat.2023.102739>
7. Jiang S, Lin W, Cao Y, et al (2021) Learning-Based Vehicle Dynamics Residual Correction Model for Autonomous Driving Simulation. In: 2021 IEEE International Intelligent Transportation Systems Conference (ITSC). pp 782–789
8. Wang H, Lai J, Zhang X, et al (2022) Make space to change lane: A cooperative adaptive cruise control lane change controller. *Transportation Research Part C: Emerging Technologies* 143:103847. <https://doi.org/10.1016/j.trc.2022.103847>
9. Hermansdorfer L, Trauth R, Betz J, Lienkamp M (2020) End-to-End Neural Network for Vehicle Dynamics Modeling. In: 2020 6th IEEE Congress

on Information Science and Technology (CiSt).
IEEE, Agadir - Essaouira, Morocco, pp 407–412

10. Betz J, Zheng H, Liniger A, et al (2022) Autonomous Vehicles on the Edge: A Survey on Autonomous Vehicle Racing. *IEEE Open J Intell Transp Syst* 3:458–488.
<https://doi.org/10.1109/OJITS.2022.3181510>
11. Bascetta L, Ferretti G (2022) LFT-Based Identification of Lateral Vehicle Dynamics. *IEEE Transactions on Vehicular Technology* 71:1349–1362. <https://doi.org/10.1109/TVT.2021.3134591>
12. Omeiza D, Webb H, Jirotko M, Kunze L (2022) Explanations in Autonomous Driving: A Survey. *IEEE Trans Intell Transport Syst* 23:10142–10162.
<https://doi.org/10.1109/TITS.2021.3122865>
13. Rokonuzzaman M, Mohajer N, Nahavandi S, Mohamed S (2021) Review and performance evaluation of path tracking controllers of autonomous vehicles. *IET Intelligent Trans Sys* 15:646–670. <https://doi.org/10.1049/itr2.12051>
14. Hu J, Zhang N, Wang H, et al (2024) Vehicles Swarm Intelligence: Cooperation in both Longitudinal and Lateral Dimensions. *IEEE Transactions on Intelligent Vehicles* 1–10.
<https://doi.org/10.1109/TIV.2024.3412130>
15. Xu N, Yang Y, Guo K (2020) A Discrete Tire Model for Cornering Properties Considering Rubber Friction. *Automot Innov* 3:133–146.
<https://doi.org/10.1007/s42154-020-00097-y>
16. Zhou Z, Wang Y, Ji Q, et al (2021) A Hybrid Lateral Dynamics Model Combining Data-driven and Physical Models for Vehicle Control Applications. *IFAC-PapersOnLine* 54:617–623.
<https://doi.org/10.1016/j.ifacol.2021.11.240>
17. Pan Y, Nie X, Li Z, Gu S (2021) Data-driven vehicle modeling of longitudinal dynamics based on a multibody model and deep neural networks. *Measurement* 180:109541.
<https://doi.org/10.1016/j.measurement.2021.109541>
18. Li Z, Zhao P, Jiang C, et al (2022) A Learning-Based Model Predictive Trajectory Planning Controller for Automated Driving in Unstructured Dynamic Environments. *IEEE Transactions on Vehicular Technology* 71:5944–5959.
<https://doi.org/10.1109/TVT.2022.3159994>
19. Wang Y, Lian R, He H, et al (2024) Auto-tuning Dynamics Parameters of Intelligent Electric Vehicles via Bayesian Optimization. *IEEE Trans Transp Electrific* 1–1.
<https://doi.org/10.1109/TTE.2023.3346874>
20. Reicherts S, Hesse BS, Schramm D (2021) Use of Naturalistic Driving Studies for Identification of Vehicle Dynamics. *IEEE Open J Intell Transp Syst* 2:195–206.
<https://doi.org/10.1109/OJITS.2021.3093712>
21. Seong H, Chung C, Shim DH (2023) Model Parameter Identification via a Hyperparameter Optimization Scheme for Autonomous Racing Systems. *IEEE Control Systems Letters* 7:1652–1657.
<https://doi.org/10.1109/LCSYS.2023.3267041>
22. Vicente BAH, James SS, Anderson SR (2021) Linear System Identification Versus Physical Modeling of Lateral–Longitudinal Vehicle Dynamics. *IEEE Trans Contr Syst Technol* 29:1380–1387.
<https://doi.org/10.1109/TCST.2020.2994120>
23. Spielberg NA, Brown M, Kapania NR, et al (2019) Neural network vehicle models for high-performance automated driving. *Sci Robot* 4:eaaw1975.
<https://doi.org/10.1126/scirobotics.aaw1975>
24. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. *Nature* 521:436–444.
<https://doi.org/10.1038/nature14539>
25. Xiao Y, Zhang X, Xu X, et al (2023) Deep Neural Networks With Koopman Operators for Modeling and Control of Autonomous Vehicles. *IEEE Trans Intell Veh* 8:135–146.
<https://doi.org/10.1109/TIV.2022.3180337>
26. Nguyen CM, Nguyen A-T, Delprat S (2023) Neural-Network-Based Fuzzy Observer With Data-Driven Uncertainty Identification for Vehicle Dynamics Estimation Under Extreme Driving Conditions: Theory and Experimental Results. *IEEE Transactions on Vehicular Technology* 72:8686–8696.
<https://doi.org/10.1109/tvt.2023.3249832>
27. Lusch B, Kutz JN, Brunton SL (2018) Deep learning for universal linear embeddings of nonlinear dynamics. *Nat Commun* 9:4950.
<https://doi.org/10.1038/s41467-018-07210-0>
28. Spielberg NA, Brown M, Gerdes JC (2022) Neural Network Model Predictive Motion Control Applied to Automated Driving With Unknown Friction. *IEEE Trans Contr Syst Technol* 30:1934–1945.
<https://doi.org/10.1109/TCST.2021.3130225>

29. Da Lio M, Bortoluzzi D, Rosati Papini GP (2020) Modelling longitudinal vehicle dynamics with neural networks. *Vehicle System Dynamics* 58:1675–1693.
<https://doi.org/10.1080/00423114.2019.1638947>
30. Nie X, Min C, Pan Y, et al (2022) Deep-Neural-Network-Based Modelling of Longitudinal-Lateral Dynamics to Predict the Vehicle States for Autonomous Driving. *Sensors* 22:2013.
<https://doi.org/10.3390/s22052013>
31. Yang S, Geng C (2023) A Longitudinal/Lateral Coupled Neural Network Model Predictive Controller for Path Tracking of Self-Driving Vehicle. *IEEE Access* 11:117121–117136.
<https://doi.org/10.1109/ACCESS.2023.3325326>
32. Gómez -Huélamo C, Del Egidio J, Bergasa LM, et al (2022) Train here, drive there: ROS based end-to-end autonomous-driving pipeline validation in CARLA simulator using the NHTSA typology. *Multimed Tools Appl* 81:4213–4240.
<https://doi.org/10.1007/s11042-021-11681-7>
33. Bhorkar A, Sakthivel P (2017) A review and a comparison of Dugoff and modified Dugoff formula with Magic formula. In: 2017 International Conference on Nascent Technologies in Engineering (ICNTE). IEEE, Vashi, Navi Mumbai, India, pp 1–4
34. Cho K, van Merriënboer B, Gulcehre C, et al (2014) Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation



DSPACE

<https://dspace.org/>

Data-driven vehicle modeling for path tracking based on the Combination of a Neural Network and Kinematics Model

Gao, Zhenhai; Wen, Wenhao; Chen, Guoying; Xing, Yang; Sun, Tianjun

2025-08

Springer

<https://doi.org/10.1007/s42154-024-00320-0>

Gao Z, Wen W, Chen G, et al., (2025) Data-driven vehicle modeling for path tracking based on the Combination of a Neural Network and Kinematics Model. Automotive Innovation, Volume 8, August 2025, pp. 696-711