2025-01-7324 Published 31 Dec 2025

Real-Time Multidimensional Vehicle Dynamic Stability Domain Calculation and Its Application in Intelligent Vehicles

Chengye Wang, Yu Zhang, Xuepeng Hu, Haipeng Qin, Guoli Wang, and Yechen Qin Beijing Institute of Technology

Citation: Wang, C., Zhang, Y., Hu, X., Qin, H. et al., "Real-Time Multidimensional Vehicle Dynamic Stability Domain Calculation and Its Application in Intelligent Vehicles," SAE Technical Paper 2025-01-7324, 2025, doi:10.4271/2025-01-7324.

Received: 28 Jun 2025 Revised: 26 Aug 2025 Accepted: 11 Nov 2025

Abstract

Vehicle stability is fundamental to the safe operation of intelligent vehicles, and real-time, high-accuracy calculation of the stability domain is crucial for maintaining control across the full range of driving conditions. Because the real stability domain is difficult to parameterize accurately and is shaped by multiple driving factors including vehicle-dynamics parameters and environmental conditions, existing approaches fail to capture the multidimensional couplings between time-varying driving inputs and the resulting stability boundaries. Moreover, these methods remain overly conservative owing to algorithmic limitations and cautious design assumptions, thereby restricting dynamic performance in complex scenarios. To address these limitations, this paper introduces a multidimensional vehicle dynamic stability region calculation framework under time-varying driving conditions and apply it into path tracking controller of intelligent vehicle. Sum-of-squares programming (SOSP) is enhanced with iterative shape functions to have a more precise description of the stability domain across discrete operating conditions. A feature analysis of the driving factors including key vehicle parameters and road conditions that influence stability-domain variation is conducted based on the SOSP. A multi-input-multi-output neural network is then used to continuously maps time-dependent driving factors to their stability-domain characteristics. Furthermore, a linear interpolation is adopted to parametrically represent the stability-domain boundary. Verification on a hardware-in-the-loop (HiL) platform demonstrates that the proposed method accurately captures the stability-domain characteristics in time-varying environments subject to multiple factors. Furthermore, the path-tracking controller equipped with the proposed model computes the stability-domain boundary in real time, improves maneuverability by limiting unnecessary interventions, and markedly reduces maximum tracking error.

Introduction

Stable driving is critical for accurate path tracking of intelligent vehicles in complex scenarios [1, 2], since it relies on clearly defined vehicle stability boundaries [3]. However, determining these boundaries involves considering time-varying factors and stability mechanisms of the automotive systems [4]. Therefore, accurately identifying stability boundaries in real-time as driving parameters variate is challenging but essential for effective vehicle stability control.

Two mainstream approaches define a vehicle's stability-domain: the empirical phase-plane method [5] and the Lyapunov Function Method (LFM) [6]. Specifically, the phase-plane method approximates the vehicle's dynamic stability-domain based on geometric characteristics of the phase plane [7]. Since the primary method relies on identifying saddle points, the stability region becomes indeterminate once these points vanish under complex driving scenarios [8, 9]. In contrast, LFM constructs a nonlinear system, employing Lyapunov's second method. Consequently, it clearly explains the causes of instability from the system's stability mechanisms perspective. Moreover, a SOSP-based LFM is widely applied in practice to estimate the attraction domains of nonlinear systems [10]. Nevertheless, due to simplistic shape functions, the SOSP method does not effectively estimate the stability-domain through iterations.

Real-time stability-domain estimation follows three paths: portrait phase method, dynamics-constraint method, and graphical fitting method. Portrait phase method represents the stability domain continuously by approximating phase-plane regions with simple linear segments and employing offline look-up tables for real-time computation [11]; however, the lack of a detailed analysis of how the stability boundaries evolve constrains its applicability in real-time driving conditions. Stability

controllers commonly impose vehicle dynamics constraints derived from the rear wheels' linear operating region [12, 13], yet this conservative approach is inadequate for real-time calculations in extreme scenarios, because it fails to fully exploit the vehicle's dynamic potential. Graphical fitting method approximates with predefined shapes by constructing ellipse-parameter functions that use driving conditions as independent variables [14]; nevertheless, it falters when handling the complex driving conditions, as multidimensional relationships between scenarios and stability-domain make the actual stability boundaries highly irregular.

Based on the above issues, this paper presents a neural network-based real-time stability-domain representation model incorporating vehicle dynamics (DSR-NET), and evaluates its time-critical responsiveness and computational efficiency by integrating DSR-NET into a path-tracking controller on a HiL platform. To improve the conservative domain estimated by SOSP, iterative shape function is used to improve the accuracy. Furthermore, stability-domain features responsive to driving factors are extracted, and a neural network maps these factors to domain features for real-time representation. Moreover, DSR-NET is integrated into a stability controller and validated on a HiL platform. Compared with traditional methods, the framework computes stability boundaries on the fly, cuts unnecessary controller interventions, and markedly improves tracking precision.

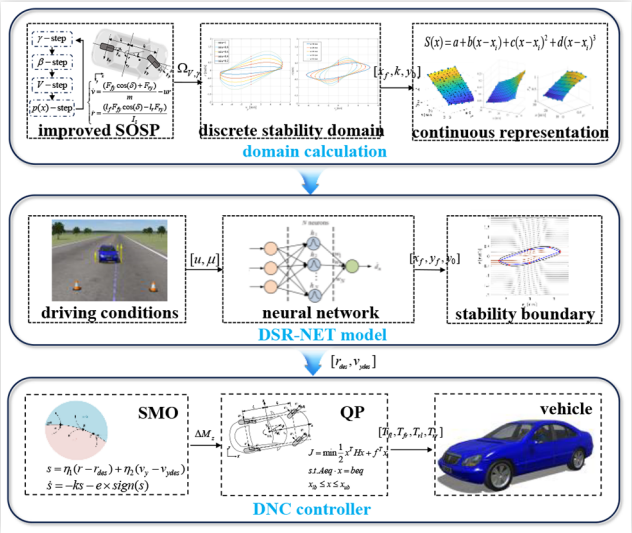
The innovations of this paper are as follows:

1. An enhanced SOSP-based method for estimating a vehicle's stability domain is presented. By introducing iterative shape functions, the method overcomes the limitations of a single shape function and substantially enlarges the domain.
2. A data-driven, multidimensional mapping model is established through the construction of a neural network, enabling the real-time capture of stability-domain characteristics under complex driving conditions.
3. A driving scenario based on HiL platform is constructed to validate effectiveness and real-time performance of the DSR-NET integrated with path tracking controller.

System Dynamics and Improved SOSP

The structure of the DSR-NET is presented in Figure 1, divided into three parts: the construction of improved-SOSP method for stability boundary calculation, the establishment of DSR-NET for real-time stability boundary representation, and the validation of DSR-NET in vehicle control.

FIGURE 1 Structure of DSR-NET



This section introduces the application of a nonlinear 2-DOF vehicle model and a nonlinear tire model of the SOSP-based system.

Vehicle Dynamics Model

The parameters of the vehicle model (1) and are as follows:

$$\begin{cases} m\left(\dot{v}_y + ur\right) = F_{fy}\cos\delta + F_{ry} \\ I_z\dot{r} = l_f F_{fy}\cos\delta - l_r F_{ry} \end{cases} \tag{1}$$

where $m, I_z, l_f, l_r, \delta, u, v_y, F_{yf}, F_{yr}$ denote the vehicle's weight, vehicle's moment of inertia, distances from the front and rear axle of the vehicle to the center, steering angle, longitudinal and lateral velocities, lateral forces of the tire, respectively.

For the vehicle tire model, the side slip angles of the front and rear tires can be expressed as:

$$\begin{cases} \alpha_f = \dfrac{v_y + l_f r}{v_x} - \delta \\ \alpha_r = \dfrac{v_y - l_r r}{v_x} \end{cases} \tag{2}$$

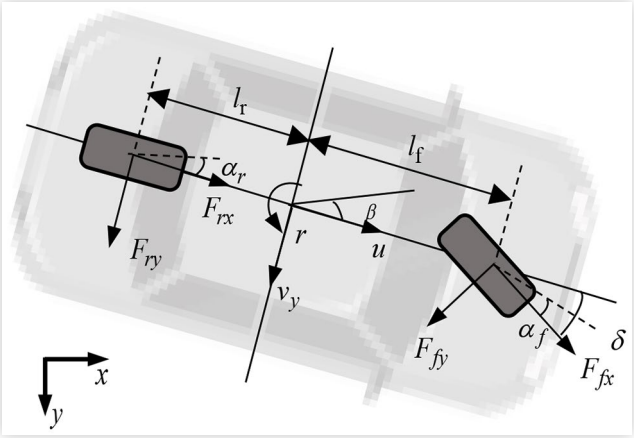
where front and rear side slip angles are denoted by α_f and α_r. In this paper, the magic tire model [15] is used to express the lateral force, the basic form as follows:

$$F_{yi} = D_i \sin\left(C_i \arctan\left(B_i\left(1-E_i\right)\alpha_i + E_i \arctan\left(B_i\alpha_i\right)\right)\right) \tag{3}$$

where D_i, C_i, B_i, E_i are the fitting coefficients in the magic formula, $i = f, r$, variable F_{yi} represent the lateral forces, parameter α_i are side slip angles.

The nonlinear system composed of the above dynamics model and tire model will be used for SOSP later.

FIGURE 2 The 2-DOFs vehicle model



Improved SOS Programming

SOSP derives a nonlinear system's attraction domain by constructing a Lyapunov function and solving it iteratively. In terms of procedure, the algorithm executes three successive steps—V step, γ step and β step [16]. Moreover, the iteration stops after three cycles when the β no longer changes [17].

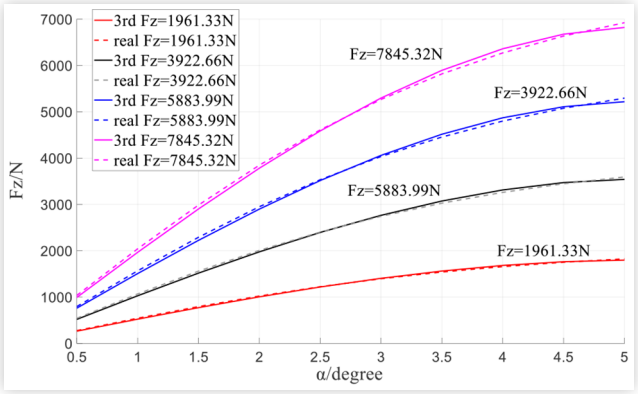
Nonlinear System Model of SOSP Construct the nonlinear system model for the SOSP and rewrite (1) in the form of (4). Due to SOSP can only solve polynomial nonlinear systems, the trigonometric Magic-Formula tire model is approximated by a cubic polynomial. Currently, stability-domain estimation usually assumes fully saturated tire forces [18] by constraining tire forces to the linear range [19]. Therefore, a polynomial description of tire behavior from the linear to weak-nonlinear region is required. The resulting polynomial tire model is (5).

$$\begin{cases} \dot{v} = \dfrac{\left(F_{fy}\cos\left(\delta\right)+F_{ry}\right)}{m} - ur \\ \dot{r} = \dfrac{\left(l_f F_{fy}\cos\left(\delta\right)-l_r F_{ry}\right)}{I_z} \end{cases} \tag{4}$$

$$F_y = F_z\left(-k_1\alpha_i + k_2\alpha_i^3\right)\ i = f,r \tag{5}$$

The tire fitting database uses the tire test database, and the fitting results are shown in Figure 3. This polynomial tire model provides a relatively accurate fit within the required range for the maximum fitting error is 77N.

FIGURE 3 Polynomial tire model for SOSP



Implementation and Improvement of SOSP The key parameters of the SOSP are defined as follows:

TABLE 1 Parameters of the improved SOSP

parameter	description
f	the nonlinear vehicle system
$V(x)$	the Lyapunov function
$\nabla V(x)f$	$\dfrac{\partial V\left(x\right)}{\partial t}$
$p(x)$	iterative shape function
$s_1(x),s_2(x)$	SOS polynomials
l_1,l_2	fixed positive definite polynomial
γ,β	the size of $V(x)$and$p(x)$
$\sum[x]$	the set of SOS polynomials

1. γ-step: Within the range of $\nabla V(x)f < 0$, find the$V(x)$ that maximizes the value γ.
2. β-step: Within the range of $V(x) < \gamma$, find the $p(x)$ that maximizes the β.
3. V-step: Within the above iterative γ and β, iteratively solve for the new$V(x)$, ensuring $V(x) > 0$ and $\nabla V(x)f < 0$.

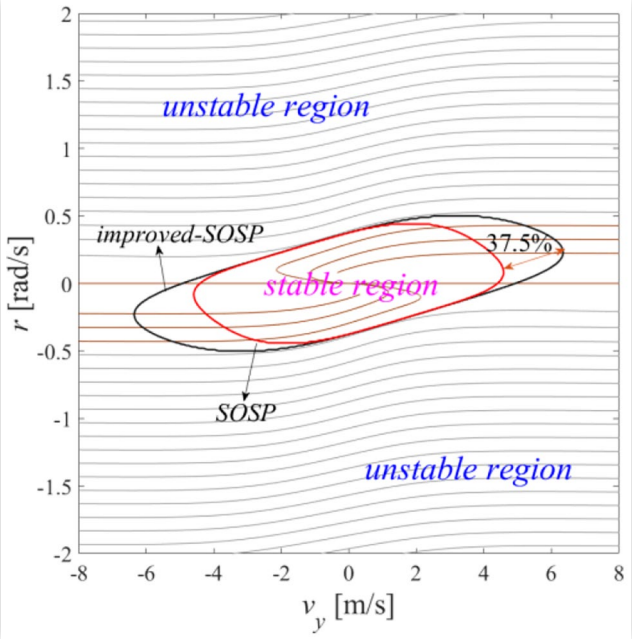
Based on the $p(x)$ framework proposed in [20], the improved SOSP introduces a $p(x)$-step iteration to enrich the shape functions and applies it repeatedly for real-time SOS estimation of the vehicle stability-domain. Once the iteration converges, the latest $V(x)$ is fed back as the initial $p(x)$ for the next SOSP cycle. Consequently, the enhanced SOSP takes vehicle and driving parameters as inputs and outputs the estimated vehicle-dynamics stability-domain $\Omega_{V,\gamma}$.

ALGORITHM 1 Improved SOSP for Stability-domain Estimation

Input: vehicle parameters, u , μ
Output: $\Omega_{V,\gamma}$
Initialization: $V(x)$, $i=j=k=g=1$
while $\beta_g-\beta_{g-1}<1e^{-5}$
  γ **-step:**
    while $\gamma_i-\gamma_{i-1}>\varepsilon_\gamma$
      $\exists s_{2,i}(x)\in\sum[x]$
      $-\nabla V(x)f-\varphi_2(x)+\ldots$
      $s_{2,i}(x)\left(V(x)-\gamma_i\right)\in\sum[x]$
      $i=i+1$
    end
    keep $V(x)$, $s_2(x)$, γ
  β **-step:**
    while $\beta_j-\beta_{j-1}>\varepsilon_\beta$
      $\exists s_{1,j}(x)\in\sum[x]$
      $s_{1,j}(x)\left(p(x)-\beta_j\right)-\left(V(x)-\gamma\right)\in\sum[x]$
      $j=j+1$
    end
    $\beta_g=\beta_j$
    $g=g+1$
    keep $s_1(x)$, $s_2(x)$, γ, β
  V **-step:**
    while $\nexists V(x)$
      $V_k(x)-l_1\in\sum[x]$
      $s_1(x)\left(p(x)-\beta\right)-\left(V_k(x)-\gamma\right)\in\sum[x]$
      $-\nabla_x V(x)f-l_2+\ldots$
      $s_2(x)\left(V_k(x)-\gamma\right)\in\sum[x]$
      $k=k+1$
    end
end
$\Omega_{V,\gamma}=\{V\le\gamma\}$
return $V(x)$ $\Omega_{V,\gamma}$
$p(x)$ **-step:**
  $p(x)$ for the next iteration $=V(x)$
next iteration:
while $\beta_g-\beta_{g-1}<1e^{-5}$
  γ **-step**
  β **-step**
  V **-step**
end
$\Omega_{V,\gamma}=\{V\le\gamma\}$
return $V(x)$, $\Omega_{V,\gamma}$

Under fixed driving conditions (u=30m/s ,μ=0.4), the stability-domain estimated by the improved SOSP (black line) closely matches the phase-plane boundary, confirming the method's accuracy. Moreover, as illustrated in Figure 4, the improved SOSP enlarges the estimated stability-domain by over 35 % compared with the traditional SOSP (red line).

FIGURE 4 Comparison of stability-domain calculation



This section employs an improved SOSP that combines nonlinear stability theory with a vehicle dynamics model to estimate the stability-domain under given driving conditions. Moreover, iterative shape-function optimization further enlarges the estimated domain. Furthermore, the next sections use this enhanced SOSP to build a stability-domain database.

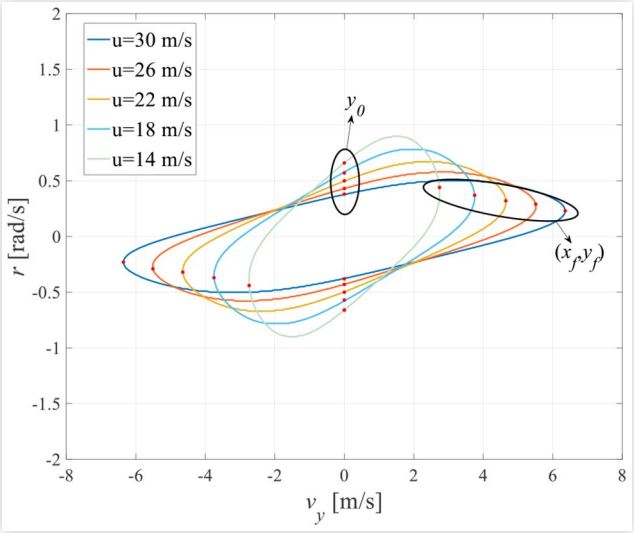
The Method of Real-time Vehicle Stability-domain Representation

First, the improved SOSP builds a stability-domain database under typical driving conditions. Afterwards, based on this database, this section analyzes how the domain evolves with vehicle speed and road adhesion and extract the most sensitive features. Finally, a neural network maps driving conditions to the stability-domain characteristics, enabling continuous representation of the discrete stability-domain.

Vehicle Stability-Domain Feature Analysis

The key factors of stability domain include coefficient of road adhesion μ and longitudinal velocity u which are discussed in [18]. This section applies the improved SOSP to build a stability-domain dataset across $u\in[14:4:30]$ m/s and $\mu\in[0.2:0.2:1]$. Specifically, Figure 5 shows the

FIGURE 5 Variation patterns under different vehicle speeds

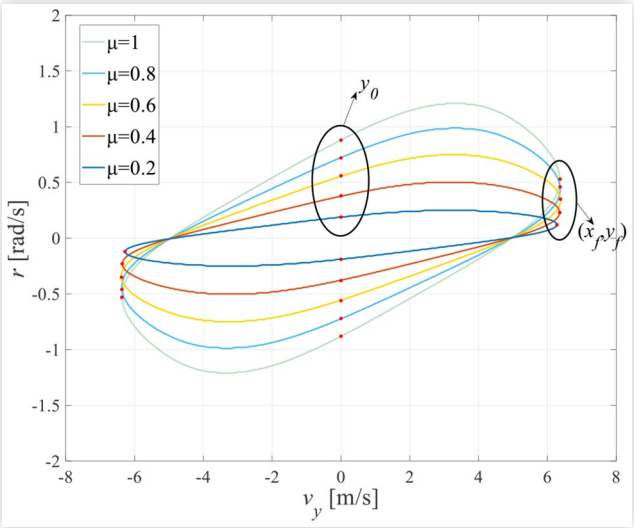


domains at various speeds with constantμ = 0.4, whereas Figure 6 depicts the domains at different μ with constant$u$ = 30m/s. Based on these results, the conclusions are as follows:

1. The evolution of the stability domain under changing driving factors can be characterized by its length, width, and orientation angle.
2. The domain is symmetric about the origin because steering inputs are not considered.

Based on the above conclusions, DSR-NET extracts the coordinate point(x_f,y_f)—the point farthest from the origin in Euclidean distance as the representative feature. x_f characteristics domain length, the ratio $\frac{y_f}{x_f}$ captures its inclination, while the vertical-axis intercept y_0 denotes domain width. As shown in Figure 5 and Figure 6, these features will serve as neural-network outputs in the next section to construct a continuous stability-domain calculation model.

FIGURE 6 Variation patterns under different adhesion coefficients



Neural Network-Based Representation Method

Since SOSP can estimate the stability domain only at discrete operating points, DSR-NET employs a neural network to build a continuous mapping from driving conditions to stability-domain characteristics.

Continuous Processing of Feature Targets The stability domain is characterized by three values—x_f,$k = \frac{y_f}{x_f}$, y_0. To ensure smooth local transitions, a cubic-spline interpolation (6) with fitting function$S_i(x)$is applied to the discrete data. Consequently, continuous functions with respect to the driving conditions are obtained shown in Figure 7. Moreover, the clear variation trends of these functions make them ideal output variables for neural-network training.

$$\begin{aligned} &S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3 \\ &S_i(x_i) = f(x_i) \\ &S_i''(x_i) = S_{i+1}''(x_{i+1}) \\ &S''(x_0) = 0, S''(x_n) = 0 \end{aligned} \tag{6}$$

A Neural Network-Based Model for Stability-Domain Nowadays, neural network–based methods are widely used to model complex mappings because they efficiently capture coupled nonlinear relationships [21] and maintain real-time performance [22]. A feedforward neural network (FNN) iteratively refines its weights through a selected training function, thereby minimizing error and capturing complex mappings. Here the network trained with

FIGURE 7 Interpolation function

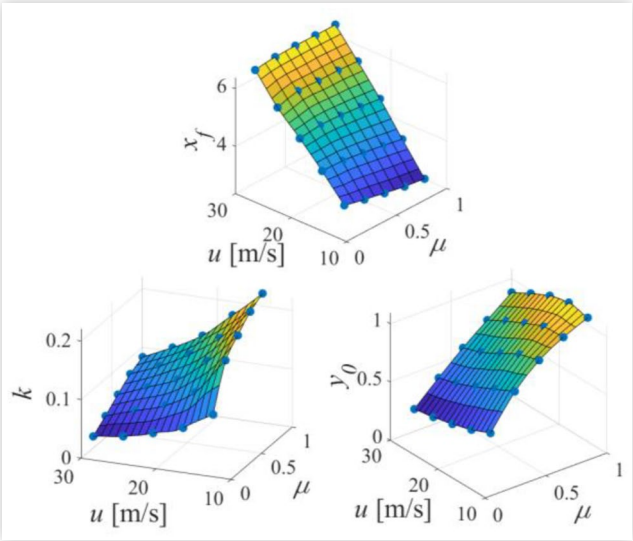


TABLE 2 FNN model parameters

parameters	description	value
n_0	input layer	2,$[\mu,u]^T$
n_l	hidden layer	5
n_L	output layer	3,$[x_f,y_f,y_0]^T$
θ	the error threshold	$1e^{-8}$
N	Maximum iterations	1000
f	training function	Levenberg-Marquardt

the Levenberg–Marquardt [23] algorithm is adopted, which is well suited to small-medium scale datasets. Table 2 lists the model parameters. A mapping model with input $X = [\mu,u]$ and output $Y = [x_f,y_f,y_0]$is constructed as Algorithm 2: define w_{ij} as the weight from x_i to neuron j, b_j as the bias of neuron j and η as the learning rate.

ALGORITHM 2 FNN model

Input: $X^T=[\mu,u]^T$, $Y^T=[x_f,y_f,y_0]^T$

Output: $Y_{predict}$

Initialization: n_0, n_l, n_L, N, θ.

for 1: N
  for x_i in X
    forward propagation:
      weighted calculation: $z_j=\sum_{i=1}^{n} w_{ij}x_i+b_j$
      activation function: $a_j=f(z_j)$
    inter-layer transmission:
      the output layer predicts the final result $\hat{y}_i$
      loss function: $L_{MSE}=\frac{1}{N}\sum_{i=1}^{N}(y_i-\hat{y}_i)^2$
  end
  for x_i in X
    backward propagation:
      $\delta_j=\sum_{k=1}^{k=K}\delta_k w_{jk}\cdot f'(z_j)$
    update weights and biases:
      $w_{ij}\leftarrow w_{ij}-\eta\frac{\partial L_{MSE}}{\partial w_{ij}}$
      $b_j\leftarrow b_j-\eta\frac{\partial L_{MSE}}{\partial b_j}$ \
  end
  if $L_{MSE}<\theta$
    break;
  end
end
return FNN model, training progress

Figure 8 confirms that the neural-network model closely matches the measured data. Specifically, the maximum errors of the predictions fo rx_fk, and y_0 are 0.05, 0.05, and 0.03 respectively.

In terms of explicit boundary representation, six feature points (x_f,y_f),$\left(\frac{2x_f}{3},2y_f\right)$,$(0,y_0)$,$(-x_f,-y_f)$,$\left(-\frac{2x_f}{3},-2y_f\right)$,

FIGURE 8 The deviation between the predicted values and the actual values.

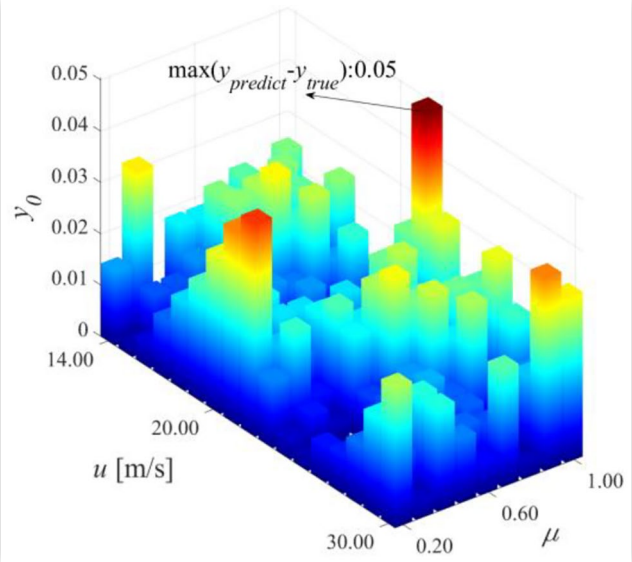
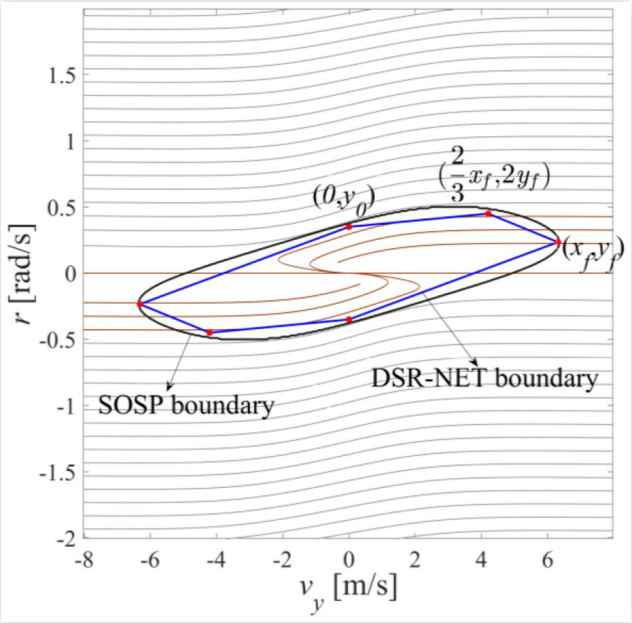


FIGURE 9 Stability-domain boundary fitting method



$(0,-y_0)$ are linearly interpolate. Consequently, the fitted curve in Figure 9 reproduces 90% of the SOSP-calculated domain, accurately capturing its boundary features. Thus, DSR-NET provides a real-time, explicit stability-boundary framework.

Design of A Path Tracking Controller

This section incorporates DSR-NET into a stability controller (DSR-NET Controller, DNC). Initially, an upper-layer sliding mode observer (SMO) computes the

additional yaw moment; subsequently, a lower-layer quadratic-programming (QP) routine assigns optimal four-wheels torques; ultimately, the DNC is deployed on a hardware-in-the-loop platform to evaluate path tracking performance.

Hierarchical Stability Controller Design

To obtain the additional yaw moment, the vehicle dynamics model can be rewritten as (7). The parameters of the test vehicle are shown as Table 3.

$$\begin{cases} m\left(\dot{v}_y + ur\right) = F_{fy}\cos\delta + F_{ry} \\ I_z\dot{r} = l_f F_{fy}\cos\delta - l_r F_{ry} + \Delta M_z \end{cases} \tag{7}$$

where ΔM_z is the additional yaw moment, generated by longitudinal forces of each wheel.

$$\Delta M_z = \left(F_{xfr} + F_{xrr} - F_{xfl} - F_{xrl}\right)\frac{B}{2} \tag{8}$$

where F_{xij} are the longitudinal forces of the tires,$i, j = f, r$.

SMO estimates the additional yaw moment by calculating the real-time deviation between actual yaw rate r and the desired value r_{des}, as well as the deviation between v_y and v_{ydes} bounded by DSR-NET. And the pre-control approach [24] is triggered once the threshold is exceeded. Accordingly, The SMO sliding surface is defined as (9), and the approaching law is given by (10).

$$s = \eta_1\left(r - r_{des}\right) + \eta_2\left(v_y - v_{ydes}\right) \tag{9}$$

$$\dot{s} = -ks - e \times sign\left(s\right) \tag{10}$$

η_1 and η_2 represent the weight for $r - r_{des}$ and $v_y - v_{ydes}$, as shown in [16].

The lower layer computes F_{xij} via QP (11). Specifically, it minimizes the four-wheels forces while satisfying the tire-adhesion ellipse constraints.

$$\begin{aligned} &J = \min\frac{1}{2}x^T Hx + f^T x \\ &\text{s.t.} A \cdot x = b \\ &x_{lb} \le x \le x_{ub} \end{aligned} \tag{11}$$

TABLE 3 Parameters of the simulation vehicle

parameters	description	value
m	total mass (kg)	2053
I_z	vehicle moment of inertia (kgm^2)	4095
I_{tire}	tire moment of inertia (kgm^2)	2
B	track width (m)	1.605
R	rolling radius of the wheel (m)	0.353
l_f, l_r	distance from the front/rear axle to C.G. (m)	1.265,1.895
preview time	horizon length (s)	5

where $A = \left[-\cos\delta\cdot\frac{B}{2} + \sin\delta\cdot l_f, \cos\delta\cdot\frac{B}{2} + \sin\delta\cdot l_f, -\frac{B}{2}, \frac{B}{2}\right]$, $x = [F_{xfl}, F_{xfr}, F_{xrl}, F_{xrr}], b = \Delta M_z, x_{ub} = \sqrt{\left(\mu F_{zij}\right)^2 - F_{yij}^{\ 2}}, x_{lb} = -x_{ub}$.

The vertical and lateral forces of four wheels are denoted by F_{zij} and F_{yij},$i, j = f, r$.

The optimal torques of the four wheels are calculated by (12) and are executed smoothly by the actuators [25].

$$T_{ij} = F_{xij}R + I_{tire}\dot{w}_{ij} \tag{12}$$

where T_{ij} is the output torque, variable w_{ij} is the wheel speed.

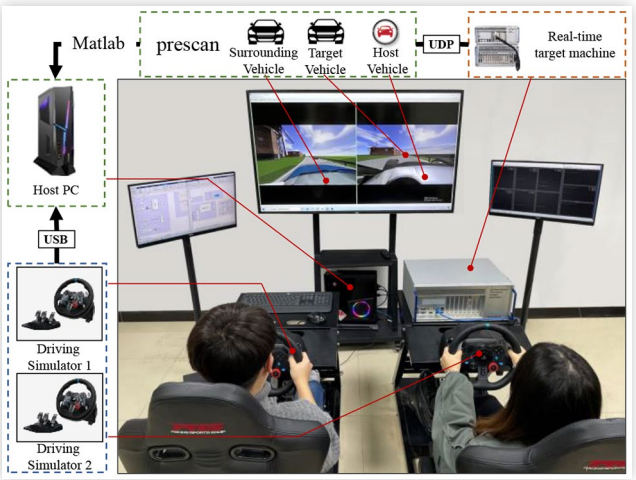
DSR-NET Based Path Tracking Method Validation

By building the HiL platform, this section validates a DNC-integrated path-tracking controller based on active front-wheel steering [26] and confirms its real-time performance and stability-maintenance capability during trajectory tracking.

HiL System Platform Figure 10 outlines the HiL platform, which comprises a host computer, a Speedgoat real-time target, and a driving simulator. The host runs the vehicle model and traffic scenarios, while the Speedgoat unit computes control inputs in real time, logs each step's computation time, and returns commands to the host via UDP and CAN. Meanwhile, the driving simulator allows the driver to change between the DNC and controller based on constraints of the linear region of rear wheels (LC) [13] during double-lane-change (DLC) tests at u = 100km/h, μ = 0.6. Finally, the test vehicle and controller parameters are listed in Table 3. The constraints of LC are defined as (13).

$$\begin{cases} rl_r - s_{sat}u \le v_y \le rl_r + s_{sat}u \\ \dfrac{-\mu g}{u} \le r \le \dfrac{\mu g}{u} \end{cases} \tag{13}$$

FIGURE 10 Hil platform



where s_{sat} is the linear range of the rear tire's side slip angle.

Validation Results Figure 11 shows the phase-plane trajectories under DLC. Without a stability controller, the vehicle briefly crosses the safety boundary, whereas the controller keeps it within safe limits at all times. Moreover, compared to LC, the DNC both preserves safety and exploits a wider dynamic envelope by 66.8%.

As Figure 12 shows, the LC and DNC both activate at the lane-change to maintain stability. But compared to LC, DNC reduces the maximum optimized torques by over 50% under extreme maneuverability.

FIGURE 11 The stability performance of vehicle tracking

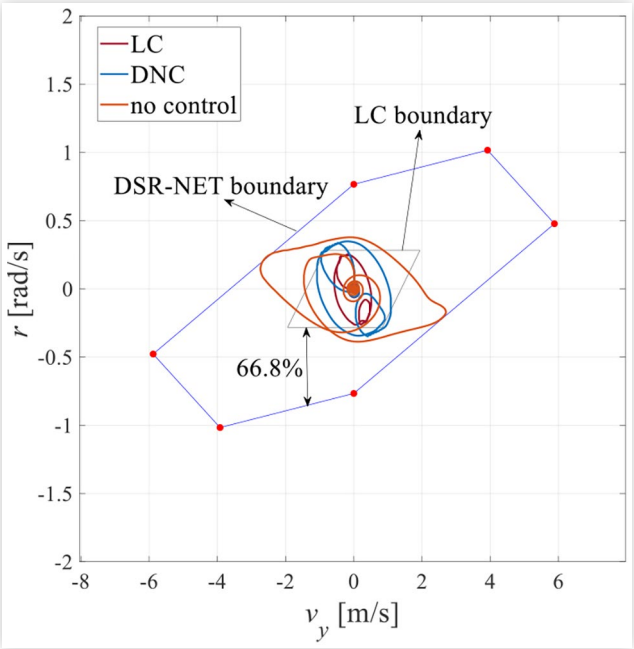


FIGURE 12 The optimal wheel torques

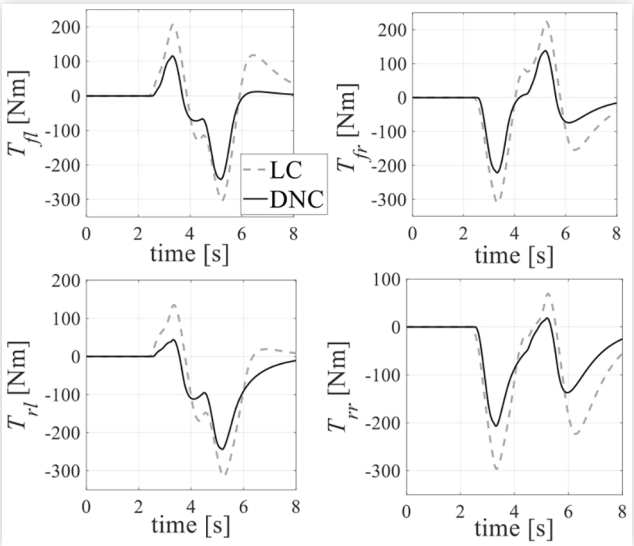


FIGURE 13 States responses of the vehicle tracking trajectory

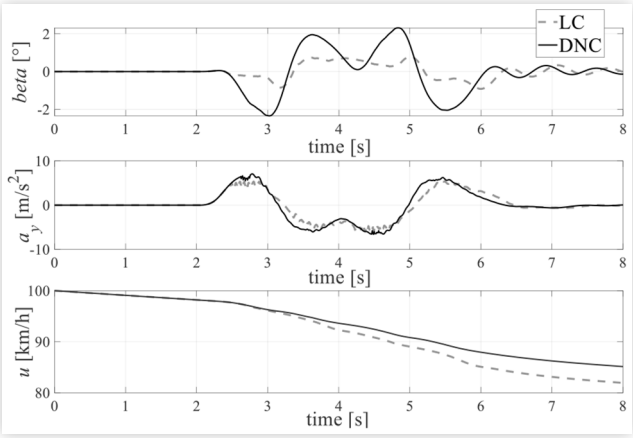


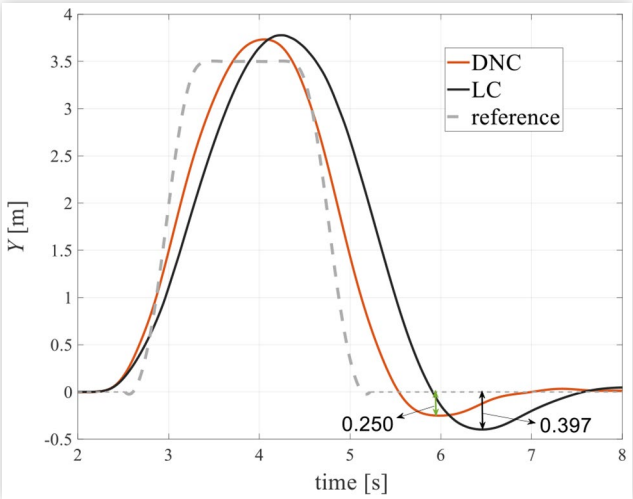
TABLE 4 Comparisons of key vehicle state parameters

state parameters	LC_{min},LC_{max}	DNC_{min},DNC_{max}
β(°)	-0.91,0.74	-2.33,2.31
a_y(m/s²)	-5.,5.2	-6.53,7.03
u(km/h)	81.1,100	85.2,100

Figure 13 reveals the state responses of the vehicle tracking trajectory under DLC.

During the lane-change operation from 2.2 s to 6 s, the vehicle's states fluctuate appreciably, prompting activation of the stability controller. Because the LC has a lower stability threshold, it intervenes earlier and remains active longer. Specifically, the LC engages at 2.3 s and deactivates later, whereas the DNC does not engage until 2.6 s and releases control sooner. Relative to the DNC, the LC causes pronounced longitudinal-speed fluctuations between 3 s and 8 s because it commands larger optimized wheel torques. As shown in Table 4, since the states of LC-based vehicle fluctuate only within a limited range, the states β and a_y can't follow their desired trajectories

FIGURE 14 Comparisons of lateral deviation in path tracking



effectively. By contrast, the DNC intervenes less and optimize smaller four-wheels torques within the stability boundary, thereby enhancing path-tracking accuracy. Consequently, Figure 14 illustrates that LC lags and suffers higher peak errors, whereas DNC maintains stability and achieves markedly smaller tracking errors from 0.397 m to 0.250 m.

The above section validates the effectiveness of the DNC in intelligent vehicle path tracking. Specifically, while maintaining stability, DNC reduces peak tracking error by 35% and boosted maneuverability by reducing controller interventions compared with the LC.

Conclusions

This paper introduces DSR-NET, a real-time calculator of the vehicle stability-domain in complex driving environments, and applies it into DNC to improve performance during extreme scenarios. By adding an iterative shape-function to the traditional SOSP, the estimated stability-domain is developed. An FNN-based mapping model is constructed to represent the connections between driving factors and stability boundaries based on the analysis of stability-domain features extracted by the improved SOSP to achieve a continuous stability-domain description. A HiL platform is built to validate the proposed model, showing that the DNC halves peak tracking error yet maintains stability due to a larger stability threshold for vehicle motion, demonstrating the practical value of DSR-NET for real-time control and path planning. Future work will aim to demonstrate the practical value of the DNC through full-vehicle tests, integrate advanced path-tracking control method such as model predictive control, and apply machine-learning methods to capture the characteristics of the extended stability domain under varying control inputs.

References

1. Zhang, Yu, Hu, Yuxuan, Hu, Xuepeng, Qin, Yechen, Wang, Zhenfeng, Dong, Mingming, and Hashemi, Ehsan. "Adaptive Crash-Avoidance Predictive Control under Multi-Vehicle Dynamic Environment for Intelligent Vehicles." *IEEE Transactions on Intelligent Vehicles* (2024).
2. Zhang, Y., Tang, X., Qin, Y., and Dong, M., "DMIC: Decision-Making Integrated Predictive Control for Intelligent Vehicles," *IEEE Transactions on Systems, Man, and Cybernetics: Systems* (2025).
3. Zhu, Z., Tang, X., Qin, Y., Huang, Y. et al., "A Survey of Lateral Stability Criterion and Control Application for Autonomous Vehicles," *IEEE Transactions on Intelligent Transportation Systems* 24, no. 10 (2023): 10382-10399.
4. Zhang, Y., Mingfan, X., Qin, Y., Dong, M. et al., "MILE: Multiobjective Integrated Model Predictive Adaptive Cruise Control for Intelligent Vehicle," *IEEE Transactions on Industrial Informatics* 19, no. 7 (2022): 8539-8548.
5. Li, S.E., Chen, H., Li, R., Liu, Z. et al., "Predictive Lateral Control to Stabilise Highly Automated Vehicles at Tire-Road Friction Limits," *Vehicle System Dynamics* 58, no. 5 (2020): 768-786.
6. Huang, Y., and Chen, Y., "Lateral Stability Control of Autonomous Ground Vehicles Considering Stability Margins and State Estimation Errors," in *2018 Annual American Control Conference (ACC)*, IEEE, 2018.
7. Chung, T. and Yi, K., "Design and Evaluation of Side Slip Angle-Based Vehicle Stability Control Scheme on a Virtual Test Track," *IEEE Transactions on Control Systems Technology* 14, no. 2 (2006): 224-234.
8. Tristano, M. and Lenzo, B., "Analytical Solution of Vehicle Phase-Plane Equilibria through the Root-Rational Tyre Model," *Vehicle System Dynamics* (2025): 1-19.
9. Jiang, Z., Pan, W., Liu, J., Han, Y. et al., "Enhancing Autonomous Vehicle Safety Based on Operational Design Domain Definition, Monitoring, and Functional Degradation: A Case Study on Lane Keeping System," *IEEE Transactions on Intelligent Vehicles* (2024).
10. Pitarch, J.L., Rakhshan, M., Mardani, M.M., and Shasadeghi, M., "Distributed Saturated Control for a Class of Semilinear PDE Systems: An SOS Approach," *IEEE Transactions on Fuzzy Systems* 26, no. 2 (2017): 749-760.
11. Liu, W., Lu, X., Leng, B., Meng, H. et al., "Vehicle Stability Criterion Research Based on Phase Plane Method," SAE Technical Paper 2017-01-1560 (2017).
12. Beal, C.E. and Christian Gerdes, J., "Model Predictive Control for Vehicle Stabilization at the Limits of Handling," *IEEE Transactions on Control Systems Technology* 21, no. 4 (2012): 1258-1269.
13. Zhang, Y., Lin, Y., Qin, Y., Dong, M. et al., "A New Adaptive Cruise Control Considering Crash Avoidance for Intelligent Vehicle," *IEEE Transactions on Industrial Electronics* 71, no. 1 (2023): 688-696.
14. Peng, D., Xia, Z., Wang, L., and Liu, Q., "A Novel Method to Assess 4WS Vehicle Stability Based on Vehicle Sideslip Angle and Angular Velocity Phase Plane Method," SAE Technical Paper 2024-01-5004 (2024).
15. Pacejka, H., *Tire and Vehicle Dynamics* (Elsevier, 2005)
16. Zhu, Z., Zhang, Y., Huang, Y., and Qin, Y., "Sum-of-Squares Based Vehicle Dynamic Stability Method and Its Applications in ADAS," in *2022 IEEE Intelligent Vehicles Symposium (IV)*, 247-254, IEEE, 2022.
17. Imani Masouleh, M. and Limebeer, D.J.N., "Region of Attraction Analysis for Nonlinear Vehicle Lateral Dynamics Using Sum-of-Squares Programming," *Vehicle System Dynamics* 56, no. 7 (2018): 1118-1138.
18. Wang, F., Shen, T., Zhao, M., Ren, Y. et al., "Lane-Change Trajectory Planning and Control Based on Stability Region for Distributed Drive Electric Vehicle," *IEEE Transactions on Vehicular Technology* 73, no. 1 (2023): 504-521.
19. Zanelli, A., Domahidi, A., Jerez, J., and Morari, M., "FORCES NLP: An Efficient Implementation of Interior-Point Methods for Multistage Nonlinear Nonconvex Programs," *International Journal of Control* 93, no. 1 (2020): 13-29.

20. He, Z., Meng, F., Wang, G. et al., "Sum-of-Squares Method in Design and Analysis of Nonlinear Systems," *Journal of Electric Machines and Control (In Chinese)* 17, no. 6 (2013): 6-12.
21. Vaghefi, S.Y., Monir, F.S.-E., Fekri-Ershad, S., and Vaghefi, S.M.M., "From One-Dimensional to Multidimensional Map Neural Networks," *Applied Soft Computing* 167 (2024): 112457.
22. Huang, G.-B., Zhu, Q.-Y., and Siew, C.-K., "Real-Time Learning Capability of Neural Networks," *IEEE Transactions on Neural Networks* 17, no. 4 (2024): 863-878.
23. Rokonuzzaman, M., Saifur Rahman, M.A., Hannan, M.K., Mishu, W.-S.T. et al., "Levenberg-Marquardt Algorithm-Based Solar PV Energy Integrated Internet of Home Energy Management System," *Applied Energy* 378 (2025): 124407.
24. Chung, T. and Yi, K., "Design and Evaluation of Side Slip Angle-Based Vehicle Stability Control Scheme on a Virtual Test Track," *IEEE Transactions on Control Systems Technology* 14, no. 2 (2006): 224-234.
25. Qin, Y., Hashemi, E., and Khajepour, A., "Integrated Crash Avoidance and Mitigation Algorithm for Autonomous Vehicles," *IEEE Transactions on Industrial Informatics* 17, no. 11 (2021): 7246-7255.
26. Hoffmann, G.M., Tomlin, C.J., Montemerlo, M., and Thrun, S., "Autonomous Automobile Trajectory Tracking for Off-Road Driving: Controller Design, Experimental Validation and Racing," in *2007 American Control Conference*, 2296-2301, IEEE, 2007.

Contact Information

Yechen Qin
School of Mechanical Engineering
Beijing Institute of Technology
5 South Zhongguancun Street, Haidian District
Beijing, 10081, P. R. China
qinyechen@bit.edu.cn

Acknowledgments

The authors acknowledge the support of the National Natural Science Foundation of China under Grant 52272386, 52522218 and 52502495, Fundamental Research Funds for the Central Universities.

Definitions/Abbreviations

DNC - DSR-NET-controller
LC - Linear constraints based controller
SOSP - Sum of Squares Programming
LFM - Lyapunov Function Method
HiL - Hardware-in-the-Loop
SMO - Sliding Mode Observer
QP - Quadratic Programming
DSR-NET - A neural network-based dynamic stability region
FNN - Feedforwad Neural Network

This paper is based upon a presentation at the *SAE 2025 Intelligent and Connected Vehicles Symposium.*

ISSN 0148-7191