IEEE Open Journal of Intelligent Transportation Systems

Received 22 December 2025; revised 27 March 2026; accepted 14 April 2026. Date of publication 17 April 2026; date of current version 4 May 2026.

Digital Object Identifier 10.1109/OJITS.2026.3685078

Vehicle Dynamics Learning From Physics Priors With Model-Structured Neural Networks

ANIELLO MUNGIELLO[1] (Graduate Student Member, IEEE), FELIX JAHNCKE[2] (Graduate Student Member, IEEE), STEFANIA SANTINI[1], JOHANNES BETZ[3] (Member, IEEE), GASTONE PIETRO ROSATI PAPINI[4] (Member, IEEE), AND MATTIA PICCININI[2] (Member, IEEE)

[1]University of Naples Federico II, 80125 Naples, Italy
[2]Professorship of Autonomous Vehicle Systems, Technical University of Munich, 85748 Garching, Germany
[3]Munich Institute of Robotics and Machine Intelligence (MIRMI), Technical University of Munich, 85748 Garching, Germany
[4]Department of Industrial Engineering, University of Trento, 38123 Trento, Italy

CORRESPONDING AUTHOR: M. PICCININI (mattia.piccinini@tum.de)

This work was supported in part by Italian FIS 2 Call (Fondo Italiano per la Scienza), Grant Assignment Decree No. 1236 (adopted on 01/08/2023) by Italian Ministry of University and Research (MUR), for the Project FIS-2023-03684 "Structured Neural Network Framework for Modeling and Control of Autonomous Systems-Neu4mes" under Grant CUP E53C24003800001; in part by MUR through Piano Nazionale di Ripresa e Resilienza (PNRR) Project "Centro Nazionale per la Mobilitá Sostenibile (CNMS)" under Grant CUP E63C22000930007; and in part by the Alexander von Humboldt Foundation through a Humboldt Postdoctoral Fellowship.

ABSTRACT Modeling the vehicle dynamics near the handling limits is crucial for autonomous driving and racing. However, building accurate models remains challenging due to strong nonlinearities, limited data availability, and changing environmental conditions. Physics-based models offer interpretability but require costly parameter identification, while general-purpose neural networks typically need large datasets to generalize. This paper introduces a new model-structured neural network (MS-NN-full) that learns the coupled lateral-longitudinal vehicle dynamics by embedding physical knowledge into its internal architecture. MS-NN-full combines physics-inspired neuro-fuzzy models with data-driven components to capture the quasi-steady-state and transient behavior, as well as the mutual coupling between the lateral and longitudinal dynamics. Experimental results on a 1:10-scale autonomous vehicle show that MS-NN-full outperforms existing MS-NN baselines and general-purpose neural networks in accuracy and generalization, using less than two minutes of training data. The model also demonstrates rapid adaptation to new tire configurations and increased vehicle mass with minimal fine-tuning. We release our implementation and datasets to support further research in physics-guided learning for autonomous systems.

INDEX TERMS Model-based neural networks, physics-guided learning, robot dynamics learning, vehicle dynamics, autonomous racing.

I. INTRODUCTION

LEARNING generalizable models of a vehicle's dynamics from limited real-world data is challenging, especially when the vehicle operates near its handling limits [1]. In autonomous driving and racing [2], accurate dynamics models are critical for motion planning [3], prediction, and control [4]. *Physics-based vehicle models*, such as single- and double-track formulations [5], are interpretable and have the potential to generalize in unseen scenarios, but identifying their parameters requires extensive experiments and costly setups to characterize tires [6], [7], suspensions, aerodynamics, powertrain, and inertial properties [8]. Moreover, capturing the nuances of the nonlinear coupled dynamics with analytical models and rapidly adapting to changing conditions remain open challenges. *Neural Network (NN) models* can learn complex nonlinear dynamics directly from data [9], without knowing the underlying physics. However, they often require large, high-quality training datasets, and they may lack interpretability and physical consistency, leading to unrealistic predictions in unseen conditions [10].

Model-Structured Neural Networks (MS-NNs), recently introduced in [11], [12], [13], and [14], aim to bridge this gap by embedding physical insights into NN architectures, resulting in lightweight models that potentially generalize better from limited data. While existing MS-NNs have shown promise to model the lateral [15] or longitudinal [14] dynamics, they have not yet been developed for the coupled

The review of this article was arranged by Associate Editor Hongsheng Qi.

dynamics, validated on real vehicles, or evaluated for rapid adaptation to changing conditions. This paper addresses these open issues, as outlined in Sec. I-B.

A. RELATED WORK

This section reviews the literature on vehicle dynamics modeling, with a focus on physics-based, learning-based, and hybrid physics-guided learning approaches.

1) FROM PHYSICS-BASED TO LEARNING-BASED VEHICLE MODELS

Physics-based vehicle models have been studied for decades, yielding formulations with varying levels of detail [5], [8]. Thanks to their relatively simple formulation, kinematic and dynamic single-track models are widely used in simulation [16], [17], [18], motion planning [19], [20], [21], and control [22], [23], [24], [25]. In [24] and [26], the tire parameters of nonlinear single-track models were identified with dedicated vehicle maneuvers, that required large testing areas. However, single-track models neglect the lateral load transfer and struggle to capture the transient nonlinear lateral-longitudinal dynamics [27]. Double-track models address these effects but require more complex and costly parameter identification [8], [28]. Moreover, modeling all relevant physical phenomena remains difficult, especially during highly dynamic maneuvers in uncertain environmental conditions [1].

To address these limitations, recent work has leveraged neural networks and data-driven methods to model and control the vehicle dynamics directly from data. Sparse regression was used in [29] to identify a data-driven dynamics model. Devineau et al. [30] proposed feedforward and convolutional networks for path tracking at constant speed, while [31] employed an LSTM to model the combined lateral dynamics near the handling limits. Similarly, [4], [32] learned the lateral dynamics with a shallow feedforward NN. Neural vehicle models have also been integrated into NMPC for vehicle guidance [33], [34]. Jerk-optimal maneuvers were learned via a feedforward NN in [35], and LSTMs were trained for autonomous racing control in [36]. A transformer-based controller capable of handling multiple vehicle platforms was introduced in [37], yet requiring extensive training data. Using a RoboRacer vehicle, [38] performed online adaptation to changing friction with a two-layer NN. Several works learned residual dynamics on top of physics-based models using NNs [39], transformers [40], local affine models [41], and Gaussian processes [42]. Hermansdorfer et al. [9] used GRUs to learn the longitudinal-lateral dynamics, highlighting that deep NNs often struggle to generalize beyond the training domain. This generalization issue, also noted in [13], [14], [38], [39], and [40], stems from the lack of physical insights in general-purpose NN architectures.

2) HYBRID PHYSICS-GUIDED LEARNING APPROACHES

Recent work on physics-guided learning aims to embed prior knowledge of a system's dynamics into neural models, to improve explainability [43], safety [44], and sample efficiency. Two main approaches have emerged. The first uses neural networks to solve known partial differential equations (PDEs), known as physics-informed neural networks (PINNs) [45], [46]. PINNs have physics-based loss functions enforcing the PDE residuals, while their architectures remain conventional feedforward NNs. Examples of PINNs in vehicle applications include [47], which predicted the energy consumption of electric vehicles, and [48], which learned the dynamics of a tractor-trailer system. A concise survey of PINNs in this domain is [49].

The second approach designs NNs whose internal structure encodes physical or model-based principles, here termed *model-structured neural networks* (MS-NNs). These architectures embed or extend dynamics knowledge to improve generalization and interpretability. In [12], [14], and [15], MS-NNs were developed to learn the decoupled longitudinal and lateral vehicle dynamics, far from the handling limits. Their MS-NNs incorporate principles such as forces superposition and steering characteristics derived from analytical single-track models. Similar ideas have been applied to steering control near the handling limits [13], [50]. In [51] and [52], single-track model parameters were learned with NNs and constrained within physically meaningful ranges. Other works integrated neural tire models into physics-based dynamics [53], [54], [55], or developed MS-NNs for automated parking [56], optimal maneuver generation [57], and lateral speed estimation [58]. These studies report improved interpretability, accuracy, and data efficiency over generic neural networks [59].

However, MS-NNs have not yet been developed to learn the coupled lateral-longitudinal vehicle dynamics, nor tested on real vehicles or under varying operating conditions to assess generalization and fast adaptation.

B. CONTRIBUTIONS AND STRUCTURE OF THE PAPER

The main contributions of this paper are as follows:

- A new model-structured neural network, MS-NN-full, that learns the coupled lateral-longitudinal vehicle dynamics by embedding and generalizing physical knowledge.
- An experimental validation on a 1:10-scale autonomous vehicle (RoboRacer), including comparisons with literature benchmarks and ablation models, demonstrating improved accuracy and generalization using less than 2 minutes of driving data.
- An analysis of the generalization and rapid adaptation capabilities under new vehicle configurations, including two tire types never seen during training and a mass variation.

II. MODEL-STRUCTURED NEURAL NETWORKS TO LEARN THE VEHICLE DYNAMICS

A. OVERALL ARCHITECTURE: MS-NN-FULL

Fig. 1a shows the overall architecture of the proposed model-structured neural network, named *MS-NN-full*, to learn the combined lateral-longitudinal nonlinear vehicle dynamics. MS-NN-full takes as inputs windows of past measurements

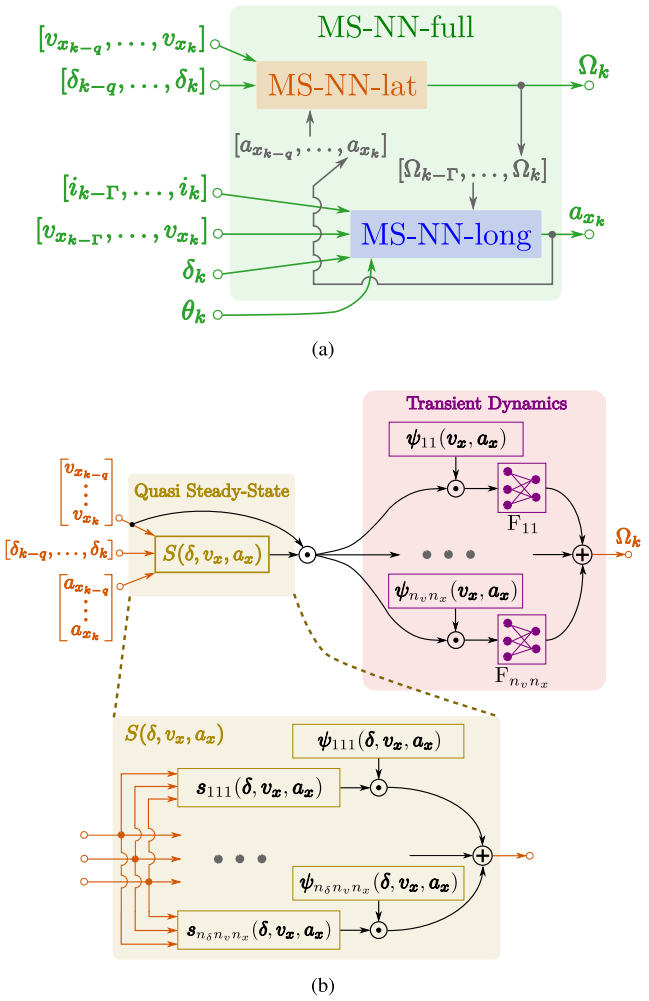


FIGURE 1. (a) Architecture of our MS-NN-full. Green parts indicate the inputs (past windows of vehicle speed v_x, steering angle δ, electric motor current i, local road slope θ) and outputs (predicted yaw rate Ω, longitudinal acceleration a_x). Gray signals are internal variables linking the lateral (MS-NN-lat) and longitudinal (MS-NN-long) networks. (b) MS-NN-lat, learning the combined lateral vehicle dynamics. MS-NN-long is shown in Fig. 3.

of the longitudinal speed v_x, steering angle δ, electric motor current i, and road slope θ, and outputs the predicted yaw rate Ω and longitudinal acceleration a_x.

MS-NN-full consists of two interconnected sub-networks (Fig. 1a), called *MS-NN-lat* and *MS-NN-long*. The former learns the lateral dynamics, predicting the yaw rate Ω, while the latter models the longitudinal dynamics, estimating the longitudinal acceleration a_x. The two sub-networks are interconnected to capture the mutual influence between lateral and longitudinal dynamics: the output of MS-NN-long is fed to MS-NN-lat, and viceversa (gray signals in Fig. 1a).

In the following subsections, MS-NN-lat and MS-NN-long are presented separately, followed by a discussion on their interconnection.

B. COMBINED LATERAL DYNAMICS: MS-NN-LAT

The proposed architecture of MS-NN-lat, depicted in Fig. 1b, is conceived to capture the combined lateral vehicle dynamics. The model predicts the yaw rate Ω_k for the current k-th time step, using as inputs the following windows of past measurements:

$$\begin{cases} \boldsymbol{v_x}_k = \left[v_{x_{k-q}}, \ldots, v_{x_k}\right] & \text{(1a)} \\ \boldsymbol{\delta}_k = \left[\delta_{k-q}, \ldots, \delta_k\right] & \text{(1b)} \\ \boldsymbol{a_x}_k = \left[a_{x_{k-q}}, \ldots, a_{x_k}\right] & \text{(1c)} \end{cases}$$

with v_x being the longitudinal speed, δ the steering angle, a_x the longitudinal acceleration, and q the tunable number of past time steps. The discretization time step in Eq.(1) is another tunable parameter, denoted as T_s, and set to 0.05 s in our experiments.

As shown in Fig. 1b, our MS-NN-lat is composed of two main parts: the first captures the quasi steady-state (QSS) behavior, while the second models the transient dynamics.

Quasi Steady-State Behavior: Modeling the QSS lateral dynamics involves capturing the relation among yaw rate Ω, steering angle δ, and longitudinal speed v_x under steady-state cornering with non-zero a_x. Insights into this relationship are provided in Fig. 2a, which shows the trajectory curvature $\rho = \Omega / v_x$ versus δ for different v_x ranges. The diagram is obtained with our 1:10-scale vehicle through sinusoidal steering maneuvers at varying speeds and accelerations. It reveals an almost linear δ-ρ relation, with a slope that changes with v_x. The effect of a_x is more complex and not shown in the plot. Overall, the shape of this diagram is influenced by several factors, including tire characteristics and load transfer. We learn this diagram with a neuro-fuzzy approach, with local linear models defined as:

$$\begin{cases} s_{jlp}(\delta, v_x, a_x) = \kappa_{1_{jlp}} + \kappa_{2_j}(\delta - \delta_{0_j}) + & \text{(2a)} \\ \qquad + \kappa_{3_l}(v_x - v_{x0_l}) + \kappa_{4_p}(a_x - a_{x0_p}) \\ \rho = \dfrac{\Omega}{v_x} = s_{jlp}(\delta, v_x, a_x) & \text{(2b)} \\ \qquad \Longrightarrow \Omega = v_x \cdot s_{jlp}(\delta, v_x, a_x)\,, \end{cases}$$

where $\{\delta_{0_j}, v_{x0_l}, a_{x0_p}\}$ denote the centers of the local models, and $\kappa_{1_{jlp}}$, κ_{2_j}, κ_{3_l}, and κ_{4_p} are their learnable parameters. Eq. (2b) expresses the yaw rate Ω computed by the local model $s_{jlp}(\delta, v_x, a_x)$ around its center. The combination of all local models yields the yaw rate Ω in QSS conditions:

$$\begin{cases} S(\delta, v_x, a_x) = \displaystyle\sum_{j=1}^{n_{\delta,\text{lat}}} \sum_{l=1}^{n_{v,\text{lat}}} \sum_{p=1}^{n_{x,\text{lat}}} s_{jlp}(\delta, v_x, a_x) \cdot & \text{(3a)} \\ \qquad \cdot\, \psi_{jlp}(\delta, v_x, a_x) \\ \Omega = v_x \cdot S(\delta, v_x, a_x)\,, & \text{(3b)} \end{cases}$$

with $n_{\delta,\text{lat}}$, $n_{v,\text{lat}}$, and $n_{x,\text{lat}}$ being the number of local models for the steering angle, vehicle speed, and longitudinal acceleration, respectively. The functions $\psi_{jlp}(\delta, v_x, a_x)$ activate locally the corresponding models, and they are defined as:

$$\psi_{jlp}(\delta, v_x, a_x) = \psi_j(\delta) \cdot \psi_l(v_x) \cdot \psi_p(a_x)\,. \quad (4)$$

The individual functions $\psi_j(\delta)$, $\psi_l(v_x)$, and $\psi_p(a_x)$ are plotted in Fig. 2b. We design them as piecewise linear, by imposing

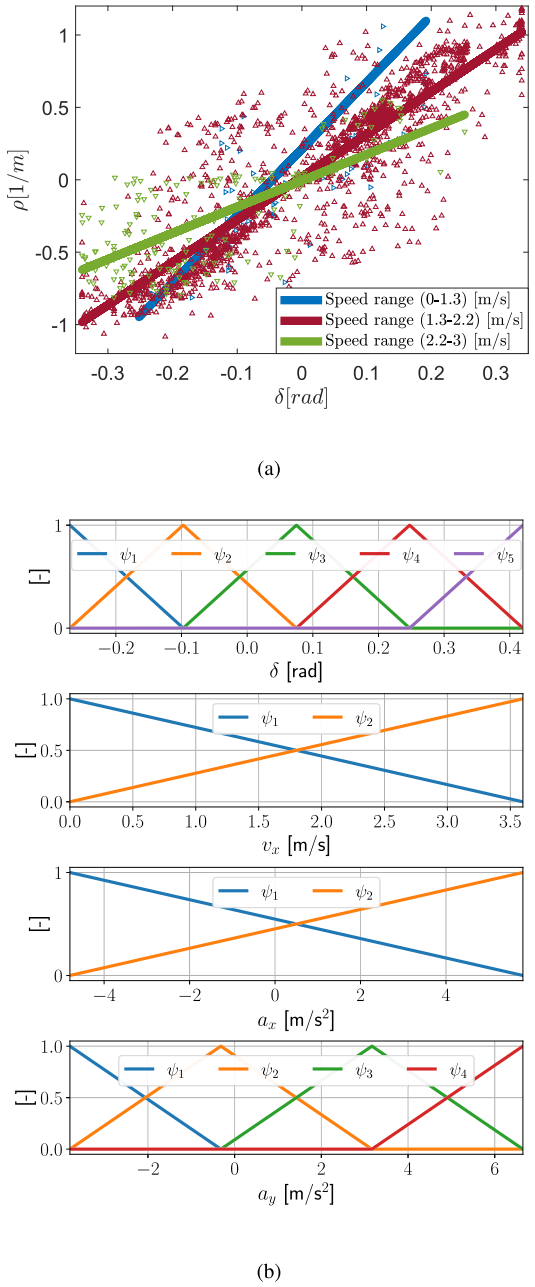


FIGURE 2. (a) Relation between the steering angle δ and the vehicle trajectory curvature $\rho = \Omega/v_x$ in different ranges of vehicle speed v_x, where the solid lines are linear fits of the data in three ranges of v_x. (b) Activation functions $\psi(\cdot)$ used in our MS-NN-lat and MS-NN-long to locally activate the corresponding neuro-fuzzy models, as a function of the steering angle δ, vehicle speed v_x, longitudinal acceleration a_x, and lateral acceleration a_y.

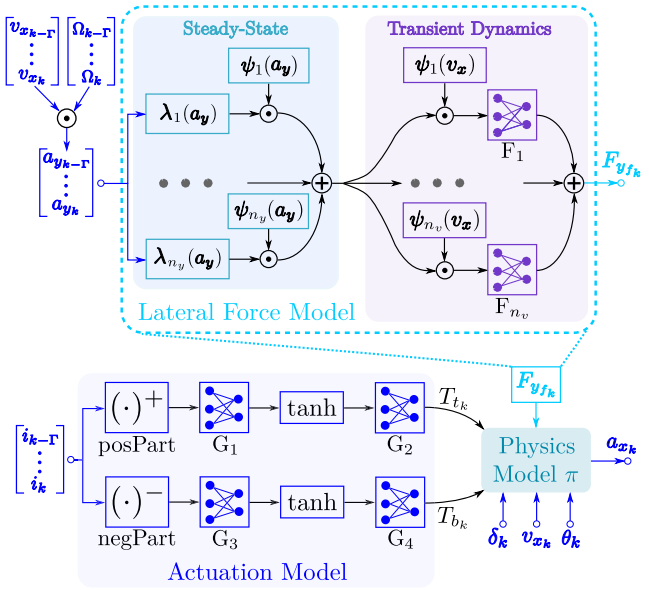


FIGURE 3. Architecture of our MS-NN-long, learning the combined longitudinal vehicle dynamics. The symbol ⊙ denotes the Hadamard (element-wise) vector product.

that $\psi_j(\delta) = 1$ for $\delta = \delta_{0_j}$, and $\sum_{j=1}^{n_{\delta,\text{lat}}} \psi_j(\delta) = 1, \forall \delta$. The centers of these activation functions are evenly spaced within the ranges of the corresponding variables, to ensure complete coverage of the input space.

Transient Dynamics: To capture the transient lateral dynamics, we extend the QSS model by including windows of past input measurements, as defined in Eq. (1). Indeed, a dynamical system's output depends on both current and past inputs, and the past window length is chosen according to the transient dynamics' timescale (Sec. IV-B), which is typically 1-2 s for vehicles. The complete formulation of our MS-NN-lat is given by:

$$\Omega_k = \boldsymbol{v_{xk}} \odot \sum_{l=1}^{n_{v,\text{lat}}} \sum_{p=1}^{n_{x,\text{lat}}} \Bigg(\underbrace{\boldsymbol{S}(\boldsymbol{\delta_k}, \boldsymbol{v_{xk}}, \boldsymbol{a_{xk}})}_{\text{quasi steady-state}} \odot \overbrace{\odot\ \boldsymbol{\psi}_{lp}(\boldsymbol{v_{xk}}, \boldsymbol{a_{xk}}) \Bigg)}^{\text{transient}} \cdot \begin{bmatrix} F_{lp_1} \\ \vdots \\ F_{lp_{q+1}} \end{bmatrix}, \tag{5}$$

where $\boldsymbol{S}(\cdot)$ is the vector form of $S(\cdot)$ in Eq. (3a), and $\odot$ denotes the Hadamard (element-wise) product. Following the theories of vehicle dynamics [8], the transient lateral response varies with the longitudinal speed v_x and acceleration a_x. Building on this insight, we design a neuro-fuzzy architecture in which the transient dynamics are captured by local fully connected layers $\boldsymbol{F}_{lp}$ (purple blocks in Fig. 1b), with $l \in \{1, \ldots, n_{v,\text{lat}}\}$ and $p \in \{1, \ldots, n_{x,\text{lat}}\}$. Each layer learns linear combinations of the $\boldsymbol{S}(\cdot)$ entries over specific v_x-a_x regions, and is activated by $\boldsymbol{\psi}_{lp}(\boldsymbol{v_{xk}}, \boldsymbol{a_{xk}})$, the vector form of Eq. (4), sharing the same shape and properties as its scalar counterparts in Fig. 2b.

Learnable Parameters in MS-NN-lat: In summary, the learnable parameters of MS-NN-lat lie in the local models Eq. (2a)–Eq. (3a) capturing the QSS behavior (4 parameters per model) and in the layers modeling the transient dynamics ($q+1$ weights per local fully connected layer):

$$N_{\text{pars,lat}} = \overbrace{4 \cdot n_{\delta,\text{lat}} \cdot n_{v,\text{lat}} \cdot n_{x,\text{lat}}}^{\text{quasi steady-state}} + \overbrace{(q+1) \cdot n_{v,\text{lat}} \cdot n_{x,\text{lat}}}^{\text{transient}} \tag{6}$$

C. COMBINED LONGITUDINAL DYNAMICS: MS-NN-LONG

The neural model MS-NN-long, depicted in Fig. 3, predicts the longitudinal acceleration a_x using as inputs the following windows of past measurements and instantaneous values:

$$\begin{cases} \boldsymbol{v_{xk}} = \left[v_{x_{k-\Gamma}}, \ldots, v_{x_k}\right] & \text{(7a)} \\ \boldsymbol{a_{yk}} = \left[a_{y_{k-\Gamma}}, \ldots, a_{y_k}\right] = \boldsymbol{\Omega}_k \odot \boldsymbol{v_{xk}} & \text{(7b)} \\ \boldsymbol{i}_k = \left[i_{k-\Gamma}, \ldots, i_k\right] & \text{(7c)} \\ \delta_k,\ \theta_k & \text{(7d)} \end{cases}$$

where a_y is the lateral acceleration, given by the product of yaw rate Ω and longitudinal speed v_x, while i and θ are respectively the electric current used to control the vehicle's motor and the local road slope. Note that the tunable number of samples in the vectors in Eq. (7) is $\Gamma+1$, which may differ from the $q+1$ used in Eq. (1) for MS-NN-lat. This is because the timescales of the longitudinal and lateral dynamics may differ.

Physics-based Model: To design MS-NN-long, we start from the Newton law for the longitudinal vehicle dynamics:

$$\begin{aligned} &m\left(a_x + c_r\, g\, \cos(\theta) + g\, \sin(\theta)\right) \\ &= F_{x_f} + F_{x_r} - k_a\, v_x^2 - c_l\, v_x - F_{y_f} \sin(\delta)\,, \end{aligned} \tag{8}$$

with m being the vehicle mass, g the gravitational acceleration, c_r the rolling resistance coefficient, k_a and c_l the aerodynamic and laminar drag coefficients. F_{x_f} and F_{x_r} are the longitudinal forces at the front and rear axles, while F_{yf} is the lateral force at the front axle. We derive F_{x_f} and F_{x_r} from the rotational wheel dynamics of the corresponding axles[1]:

$$2\, I_{w_j}\, \dot{\omega}_j = -r\, F_{x_j} + T_j \Longrightarrow F_{x_j} = \frac{T_j - 2\, I_{w_j}\, \dot{\omega}_j}{r}\,, \tag{9}$$

where $j \in \{f, r\}$ denotes the front and rear axles; I_w, r, ω and T are the wheel inertias, radii, angular velocities, and torques, respectively. The wheel acceleration $\dot{\omega}$ can be expressed as a function of the vehicle longitudinal acceleration a_x through:

$$\dot{\omega}_j \approx \frac{\dot{v}_x}{r} = \frac{a_x}{r}\,, \tag{10}$$

assuming small longitudinal slips. Moreover, we use the following auxiliary relation:

$$T_f + T_r = T_t + T_b\,, \tag{11}$$

which links the sum of the front and rear wheel torques to the total traction (T_t) and braking (T_b) torques. By plugging Eq. (10)–(11) in Eq. (9) and substituting the resulting expressions of F_{x_f} and F_{x_r} into Eq. (8), we isolate a_x as:

$$\begin{cases} N_k = \dfrac{1}{m}\left(\dfrac{T_{t_k} + T_{b_k}}{r} - k_a\, v_{x_k}^2 - c_l\, v_{x_k} - F_{y_{f_k}} \sin(\delta_k)\right) + \\ \qquad - c_r\, g\, \cos(\theta_k) - g \sin(\theta_k) & \text{(12a)} \\ D_k = 1 + \dfrac{2\,(I_{w_f} + I_{w_r})}{m\, r^2} & \text{(12b)} \\ a_{x_k} = \pi\left(T_{t_k}, T_{b_k}, \delta_k, v_{x_k}, F_{y_{f_k}}, \theta_k\right) = \dfrac{N_k}{D_k} & \text{(12c)} \end{cases}$$

[1]Eq. (9) assumes the angular velocities of the left and right wheels on the same axle are equal, but it can be easily extended to the more general case.

which is the physics-based model $\pi(\cdot)$ shown in Fig. 3 and used in our MS-NN-long to predict a_{x_k}.

Actuation Model: To connect the traction and braking torques $\{T_t, T_b\}$ in Eq. (12c) to the control input i (electric motor current), we introduce an actuation model within MS-NN-long. This relation is in general highly nonlinear, motor-dependent and hard to model from first principles. Hence, we learn the actuation model with a general-purpose neural architecture, as shown in Fig. 3. We create two separate branches to learn the traction and brake torques $\{T_t, T_b\}$. The traction branch takes as input the vector $\boldsymbol{i}_k$ of past current measurements, extracts the positive part (corresponding to traction commands), and feeds it to a 2-layer fully connected network. The first layer G_1 uses η neurons and a tanh activation function, while the second layer G_2 is linear and outputs the learned T_{t_k}. The brake branch has the same architecture used for traction, but it extracts the negative part of $\boldsymbol{i}_k$, outputting T_{b_k}.

Lateral Force Model: Finally, MS-NN-long learns the effect of the lateral force F_{y_f} appearing in Eq. (12c). As shown in Fig. 3, we design a neuro-fuzzy structure to learn the steady-state and transient components of F_{y_f}:

$$F_{y_{f_k}} = \sum_{\iota=1}^{n_{y,\text{long}}} \Bigg(\underbrace{\boldsymbol{\Lambda}(\boldsymbol{a_{yk}})}_{\text{steady-state}} \odot \boldsymbol{\psi}_\iota(\boldsymbol{v_{xk}}) \Bigg) \cdot \overbrace{\begin{bmatrix} F_{\iota_1} \\ \vdots \\ F_{\iota_{\Gamma+1}} \end{bmatrix}}^{\text{transient}} \tag{13}$$

For the steady-state (SS) part, we rely on the physical insight that the lateral force depends primarily on the lateral acceleration a_y [5], [8]. Thus, we learn the SS behavior with local linear models depending on a_y, with the function $\boldsymbol{\Lambda}(\boldsymbol{a_{yk}})$ defined as:

$$\begin{cases} \boldsymbol{\lambda}_\nu(\boldsymbol{a_{y_k}}) = q_{1_\nu} + q_{2_\nu}\left(\boldsymbol{a_{y_k}} - a_{y0_\nu}\right) & \text{(14a)} \\ \boldsymbol{\Lambda}(\boldsymbol{a_{y_k}}) = \displaystyle\sum_{\nu=1}^{n_{y,\text{long}}} \boldsymbol{\lambda}_\nu(\boldsymbol{a_{y_k}}) \odot \boldsymbol{\psi}_\nu(\boldsymbol{a_{y_k}}) & \text{(14b)} \end{cases}$$

The local linear models $\boldsymbol{\lambda}_\nu(\boldsymbol{a_{yk}})$ in Eq. (14a) depend on the learnable parameters $\{q_{1_\nu}, q_{2_\nu}\}$, with a_{y0_ν} being the (predefined) center of the local model. The activation functions $\boldsymbol{\psi}_\nu(\boldsymbol{a_{yk}})$ share the same properties as those defined in Eq. (4), and are shown in Fig. 2b.

Finally, as explained in Sec. II-B, the transient evolution of the lateral forces changes with the longitudinal speed v_x [8]. Thus, we model the transient component of F_{y_f} with local fully connected layers $\boldsymbol{F}_\iota$ in Eq. (13) (violet blocks in Fig. 3), with $\iota \in \{1, \ldots, n_{y,\text{long}}\}$. Each layer learns linear combinations of the $\boldsymbol{\Lambda}(\boldsymbol{a_{yk}})$ vector entries over specific v_x regions, and is activated by $\boldsymbol{\psi}_\iota(\boldsymbol{v_{xk}})$, thus resulting in a neuro-fuzzy structure.

Learnable Parameters in MS-NN-long: Overall, the learnable parameters of MS-NN-long are contained in: 1) the physics-based model in Eq. (12) (its 7 physical parameters are treated as learnable scalar weights); 2) the actuation model (2 neural networks with $(\Gamma+1)\cdot\eta + 2\,\eta$ parameters each); 3) the steady-state lateral force model in Eq. (14a)

(a) (b)

FIGURE 4. RoboRacer vehicle used in our experiments (a), and configurable track layout (b).

($2\,n_{y,\text{long}}$ parameters); and 4) the transient lateral force model in Eq. (13) ($(\Gamma+1)\cdot n_{v,\text{long}}$ parameters):

$$N_{\text{pars,long}} = \overbrace{7}^{m,c_r,c_l,k_a,I_{w_f},I_{w_r},r \text{ in (12)}} + \overbrace{2\cdot\big((\Gamma+1)\cdot\eta+2\,\eta\big)}^{\text{actuation model}} + \underbrace{2\,n_{y,\text{long}}}_{\text{steady-state } F_y} + \underbrace{(\Gamma+1)\cdot n_{v,\text{long}}}_{\text{transient } F_y} \tag{15}$$

D. INTERCONNECTION OF MS-NN-LAT AND MS-NN-LONG INTO MS-NN-FULL

As shown in Fig. 1a, MS-NN-lat and MS-NN-long are interconnected to capture the mutual influence between lateral and longitudinal dynamics. Specifically, the yaw rate Ω predicted by MS-NN-lat is fed to MS-NN-long to compute the lateral acceleration a_y in Eq. (7b), while the longitudinal acceleration a_x estimated by MS-NN-long is passed back to MS-NN-lat. This closed-loop interconnection allows the overall model MS-NN-full to capture the complex coupling between lateral and longitudinal dynamics, which is particularly relevant during aggressive maneuvers, such as high-acceleration cornering or rapid acceleration/deceleration transitions.

III. EXPERIMENTAL SETUP AND TRAINING

A. ROBORACER PLATFORM AND TEST TRACK

We apply the proposed architecture to learn the dynamics of a 1:10-scale robotic vehicle, the RoboRacer,[2] shown in Fig. 4a. The RoboRacer is being extensively used in the context of robotics research and autonomous racing competitions, providing a relatively low-cost platform for rapid prototyping of autonomous driving algorithms.

The vehicle employs an all-wheel-drive (AWD) electric powertrain based on the Velineon 3351R/3500 brushless DC (BLDC) motor. The motor is characterized by a Kv rating of 3500 RPM/V and delivers a peak power of approximately 300 W. Motor control is performed by a Veddar Electronic Speed Controller (VESC), which integrates an Inertial Measurement Unit (IMU) to enable real-time dynamic feedback and improved control performance. The platform is equipped with a Hokuyo 2D LiDAR sensor and a ZED2 stereo vision camera mounted at the front of the vehicle, along with an inertial measurement unit (IMU). These sensors provide real-time data essential for localization, mapping, and control. The onboard computation is handled by a NVIDIA Jetson Orin Nano running a Ubuntu-based operating system. We use the Robot Operating System 2 (ROS2) middleware to implement the full autonomous software stack.

Measurements of longitudinal acceleration, lateral acceleration, and yaw rate are acquired from the IMU sensor. We feed the requested steering angle as a control input to the steering actuator, while the measured motor current and vehicle speed are provided by the VESC motor controller.

B. TRAINING DATASET

The training and validation datasets are generated by driving autonomously the RoboRacer vehicle around the test track in Fig. 4b. We use a longitudinal speed-tracking controller and three lateral controllers—Stanley, proportional-integral-derivative (PID), and pure pursuit—to execute pre-computed racelines with varying acceleration limits. The vehicle is pushed close to its handling limits, performing aggressive maneuvers with lateral accelerations up to 7 m/s^2 and longitudinal accelerations beyond 6 m/s^2.

The training set consists of approximately **2 minutes** of recorded data, while the validation set comprises around 1 minute of driving data. Thus, the size of the dataset is relatively small: we aim to explore the learning and generalization capabilities under limited data conditions, which reflects real-world robotics use cases. All signals are post-processed and resampled to obtain a unified sampling rate of 20 Hz.

C. TRAINING METHOD

We perform the training with supervised learning, using the mean squared error (MSE) between the predicted and actual outputs as the loss function. Our training follows two steps. First, we independently train the combined lateral and longitudinal models (MS-NN-lat and MS-NN-long) using their respective input-output pairs. Next, we integrate them into the unified closed-loop model MS-NN-full of Fig. 1a and fine-tune the entire architecture. In MS-NN-full, the yaw rate returned by MS-NN-lat is passed to MS-NN-long, while the longitudinal acceleration from MS-NN-long is fed back to MS-NN-lat, forming a closed-loop interconnection. Our two-step process aims to enhance the training efficiency and stability. In the first step, the absence of closed-loop autoregressive connections allows for a more efficient and stable training of MS-NN-lat and MS-NN-long. The second step then focuses on fine-tuning the complete closed-loop system (MS-NN-full), which would be significantly harder and more computationally demanding to train from scratch. It is also worth emphasizing that both MS-NN-lat and MS-NN-long are designed to capture the coupling effects between lateral and longitudinal dynamics, as they include the relevant variables (e.g., a_x in MS-NN-lat and a_y in MS-NN-long) as inputs. Thus, even when trained independently, the coupling effects are not neglected, but rather learned within each model. The second step of fine-tuning the closed-loop MS-NN-full then allows for further refinement of the coupling effects, as the interconnected architecture captures the mutual

[2]https://roboracer.ai

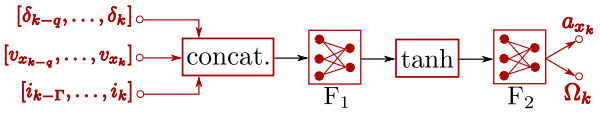


FIGURE 5. Architecture of the benchmark general-purpose NN (G-NN).

influence between lateral and longitudinal dynamics by feeding the outputs of each model into the other.

Training uses the Adam optimizer [60]. MS-NN-lat and MS-NN-long employ a learning rate of 10^{-3}, a batch size of 100, and 3000 epochs. Finally, our closed-loop training of MS-NN-full is performed for 1000 epochs. Early stopping is used to prevent overfitting by monitoring the MSE on the validation set, halting training when no improvement is observed for 300 consecutive epochs.

1) IMPLEMENTATION WITH THE `nnodely` OPEN FRAMEWORK

We use the open-source framework `nnodely`[3] to implement and train our neural architectures. `nnodely` is a PyTorch-based library designed to simplify the development, training, and deployment of model-structured and physics-guided NNs. Unlike general-purpose deep learning tools, it offers higher-level abstractions and custom layers tailored for MS-NNs. By streamlining the integration of physical models and deployment, it makes these techniques more accessible to users with classical modeling expertise, such as engineers and physicists.

IV. RESULTS

This section analyzes the experimental results obtained by applying the proposed MS-NN architecture to the RoboRacer platform. Sec. IV-A introduces the benchmark models used for comparison, while Sec. IV-B discusses the hyperparameter tuning process. The performance and generalization capabilities of the proposed architectures are evaluated in Sec. IV-C, focusing on the nominal vehicle setup. Finally, Sec. IV-D assesses the generalization under varying vehicle setups, including changes in vehicle mass and tire configurations, and the capability of rapid adaptation through fine-tuning.

Our implementation code and datasets are available at:
https://github.com/tonegas/nnodely-applications/tree/main/vehicle/model_combined_vehicle_dynamics_RoboRacer

A. BENCHMARKS

We compare our MS-NN-full against the following benchmarks, which share the same inputs and compute the same outputs as our MS-NN-full.

1) G-NN: GENERAL-PURPOSE NEURAL NETWORK

This benchmark, shown in Fig. 5, employs a general-purpose architecture with two fully connected layers. Its structure is adapted from [14, Sec. V-A]. The first layer (F_1) uses a tanh activation function, while the second layer (F_2) is linear.

[3]https://github.com/tonegas/nnodely.git

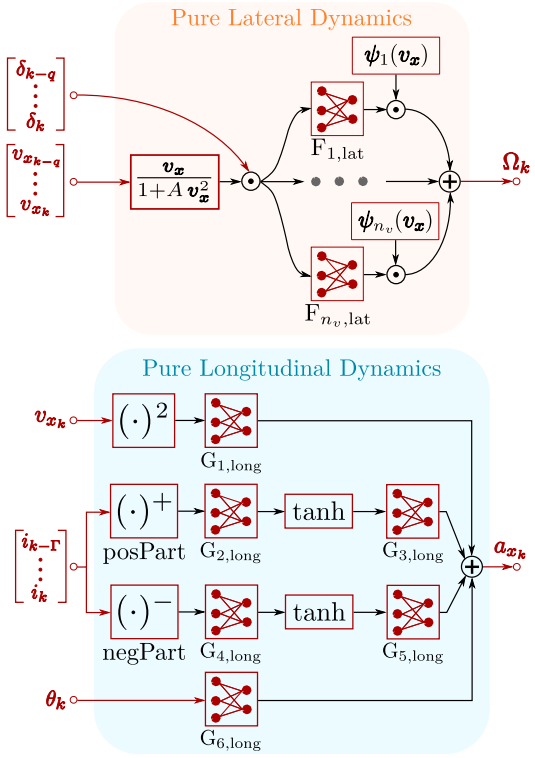


FIGURE 6. Architecture of the benchmark MS-NN-bench, derived from [14] and [15] and capturing the pure lateral and longitudinal vehicle dynamics.

2) MS-NN-BENCH: PURE LATERAL AND LONGITUDINAL DYNAMICS

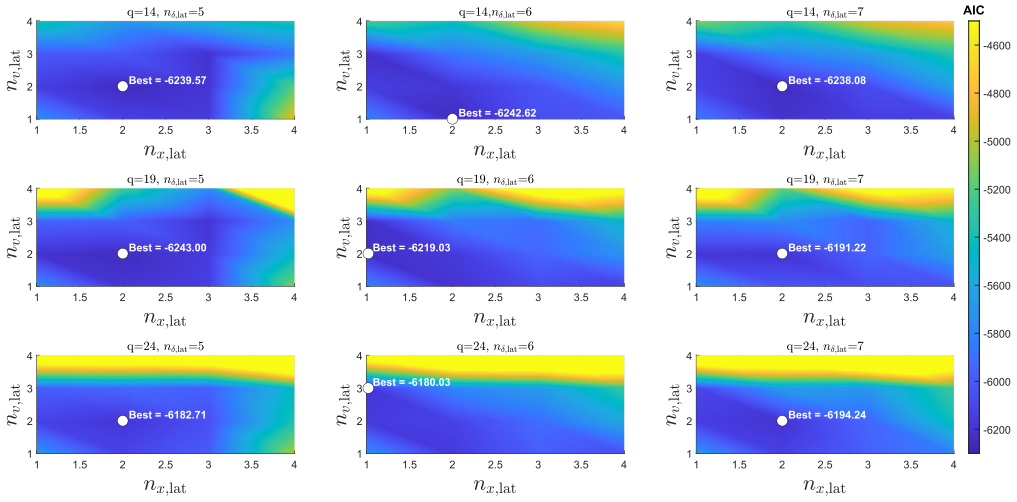
The benchmark MS-NN-bench, shown in Fig. 6, learns the lateral dynamics with the MS-NN of [15, Sec. III-A], and the longitudinal dynamics with the MS-NN from [14, Sec. V-B]. It is a simpler alternative to our MS-NN-full, as it omits lateral-longitudinal coupling and uses a less structured architecture, while still embedding key physical insights. In the lateral block (left side of Fig. 6), the parameter A encodes the understeering gradient, and the layers $F_{1,\text{lat}},\ldots,F_{n_v,\text{lat}}$ learn the transient lateral response across different speed ranges. In the longitudinal block (right side), $G_{1,\text{long}}$ learns the aerodynamic drag; $G_{2,\text{long}},\ldots,G_{5,\text{long}}$ capture the motor actuation dynamics; and $G_{6,\text{long}}$ models the road slope effect.

3) ABLATION MODELS

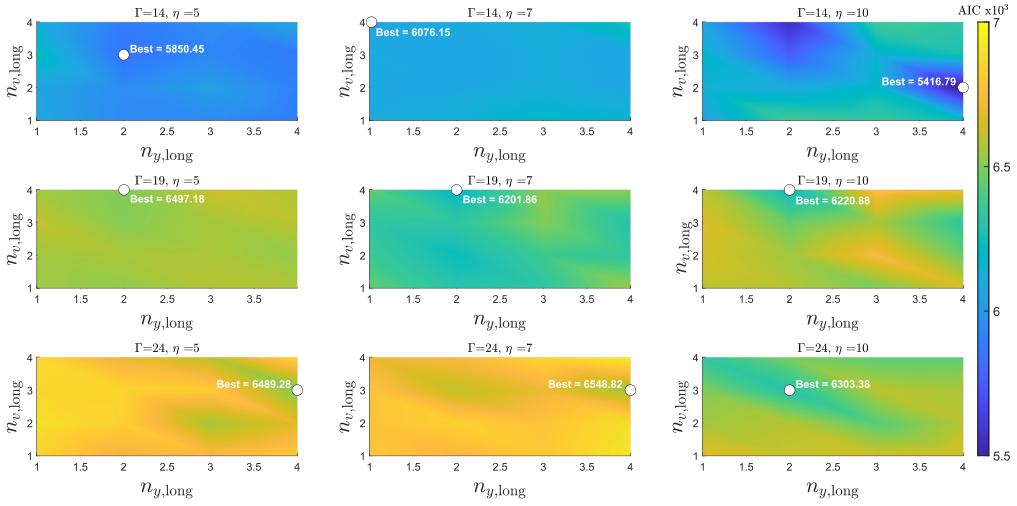
To assess the contribution of individual components within the proposed MS-NN-full architecture, we design specific ablations for the lateral and longitudinal dynamics.

MS-NN-Ablation 1: The first ablation model acts on the transient dynamics part of MS-NN-lat (Fig. 1b), and omits the dependency on the longitudinal acceleration $\boldsymbol{a}_{x_k}$ in the neuro-fuzzy models $\boldsymbol{\psi}_{lp}(\boldsymbol{v}_{xk},\boldsymbol{a}_{xk})$ of Eq. (5). Hence, the local models $\boldsymbol{\psi}_{lp}(\boldsymbol{v}_{xk},\boldsymbol{a}_{xk})$ are solely dependent on $\boldsymbol{v}_{x_k}$, reducing to $\boldsymbol{\psi}_l(\boldsymbol{v}_{xk})$, $l\in\{1,\ldots,n_{v,\text{lat}}\}$.

MS-NN-Ablation 2: The second ablation extends the first by also removing the dependency on $\boldsymbol{a}_{x_k}$ in the $\boldsymbol{S}(\cdot)$ function of Eq. (5), which represents the quasi-steady-state part of



(a)



(b)

FIGURE 7. **Hyperparametrs tuning for the sub-models MS-NN-lat (a) and MS-NN-long (b). Comparing different configurations through the AIC metric, where lower values indicate better trade-offs between model accuracy and complexity (number of learnable parameters).**

MS-NN-lat (Fig. 1b). Thus, Eq. (5) becomes a function of $\boldsymbol{\delta_k}$ and $\boldsymbol{v_{x_k}}$ only, without any influence from the longitudinal acceleration.

MS-NN-Ablation 3: The third ablation model acts on the longitudinal dynamics (MS-NN-long), and removes the coupling with the lateral dynamics. Specifically, the longitudinal physics-based model $\pi(\cdot)$ in Eq. (12c) omits the lateral contribution $F_{y_{f_k}} \sin(\delta_k)$.

B. HYPERPARAMETERS TUNING

To tune the hyperparameters of the proposed MS-NN-full architecture, we adopt a grid search strategy, where the metric used to evaluate each configuration is the Akaike Information Criterion (AIC) [61]. The AIC evaluates the trade-off between model accuracy and complexity (number of learnable parameters), where lower values indicate better sample efficiency. The hyperparameters for the two sub-models MS-NN-lat and MS-NN-long are tuned separately, focusing on the yaw rate and longitudinal acceleration prediction errors, respectively.

We define the hyperparameter search ranges from preliminary data analyses and vehicle-dynamics insights. The numbers of local models $n_{x,\text{lat}}$, $n_{\delta,\text{lat}}$, $n_{v,\text{lat}}$, $n_{v,\text{long}}$, and $n_{y,\text{long}}$ are chosen to balance expressiveness and data coverage, so that each local model is trained on sufficiently informative samples within its operating region. The input window lengths q and Γ are related to the data sampling time (0.05 s), and define the history needed to capture the transient vehicle dynamics. We notice a significant performance variation for $q, \Gamma \in [14, 24]$, corresponding to time windows of 0.75−1.25 s.

MS-NN-lat: We vary the number of local models for the steering angle $n_{\delta,\text{lat}} \in \{5, 6, 7\}$, the longitudinal acceleration $n_{x,\text{lat}} \in \{1, 2, 3, 4\}$, the longitudinal velocity $n_{v,\text{lat}} \in \{1, 2, 3, 4\}$, and the length of the windows of past inputs $q \in \{14, 19, 24\}$. Fig. 7a shows the AIC values for each configuration on the

TABLE 1. Prediction of the combined lateral dynamics (yaw rate Ω): comparison of the proposed MS-NN-full with the benchmarks (G-NN, MS-NN-bench) and two ablations, evaluated over three training set sizes—large (100%), medium (80%), and small (60%). (Bold: best; Underlined: second best).

		Large Train. Set		Medium Train. Set		Small Train. Set	
Models	Metrics	Train.	Valid.	Train.	Valid.	Train.	Valid.
G-NN	RMSE ↓ [rad/s]	0.102	0.142	0.135	0.172	0.134	0.224
	FVU ↓ [-]	0.022	0.053	0.030	0.072	0.037	0.155
MS-NN-bench [15]	RMSE ↓ [rad/s]	0.147	0.154	0.159	0.158	0.181	0.181
	FVU ↓ [-]	0.056	0.060	0.059	0.063	0.088	0.066
MS-NN-Ablation 1	RMSE ↓ [rad/s]	0.081	0.122	0.079	0.110	0.088	0.134
	FVU ↓ [-]	0.014	0.033	0.012	0.032	0.019	0.053
MS-NN-Ablation 2	RMSE ↓ [rad/s]	0.085	0.116	0.084	0.116	0.094	**0.124**
	FVU ↓ [-]	0.016	0.036	0.014	0.035	0.022	**0.041**
MS-NN-full (ours)	RMSE ↓ [rad/s]	**0.076**	**0.109**	**0.076**	**0.106**	**0.085**	0.137
	FVU ↓ [-]	**0.012**	**0.031**	**0.012**	**0.030**	**0.019**	0.061

validation set, with a white dot indicating local minima of the AIC surface. The global minimum, corresponding to the best trade-off between accuracy and parsimony, is obtained with: $n_{\delta,\text{lat}} = 5$, $n_{x,\text{lat}} = 2$, $n_{v,\text{lat}} = 2$, $q = 19$. Using Eq. (6), the total number of learnable parameters is 160.

MS-NN-long: The same analysis is conducted for MS-NN-long, by varying the numbers of neurons in the actuation model $\eta \in \{5, 7, 10\}$, the past windows' length $\Gamma \in \{14, 19, 24\}$, the numbers of local models $n_{v,\text{long}} \in \{1, 2, 3, 4\}$, $n_{y,\text{long}} \in \{1, 2, 3, 4\}$. As shown in Fig. 7b, the best configuration in terms of AIC is $\eta = 10$, $\Gamma = 14$, $n_{v,\text{long}} = 2$, $n_{y,\text{long}} = 4$. Using Eq. (15), this results in 384 learnable parameters.

C. PERFORMANCE ANALYSIS

This section evaluates the performance of our MS-NN-full and the benchmarks. Although MS-NN-full jointly models the coupled lateral-longitudinal dynamics, we present the results separately for its lateral (Sec. IV-C.1) and longitudinal (Sec. IV-C.5) components to highlight their individual contributions.

1) COMBINED LATERAL DYNAMICS

Fig. 8a compares the yaw rate prediction errors of the proposed MS-NN-full and the benchmarks G-NN and MS-NN-bench from [15], as described in Sec. IV-A.2. Our MS-NN-full achieves the lowest median error and reduces the occurrence of outliers. This is confirmed by Fig. 8b–8c, which show the predicted yaw rate profiles on parts of the training and validation datasets. The MS-NN-bench from [15] tends to overestimate the yaw rate peaks and underestimates the valleys, while the G-NN is closer to ours but still fails to accurately capture the peaks and zero-crossings, especially on the unseen validation data.

Table 1 offers a quantitative comparison using the Root Mean Square Error (RMSE) and the Fraction of Variance Unexplained[4] (FVU), which determines the proportion of variance in the data not captured by the model (the lower the FVU, the better the model). With the nominal *large* training set (around 2 minutes of driving data), our MS-NN-full outperforms the benchmarks G-NN and MS-NN-bench in terms of RMSE and FVU on the training and validation datasets. Specifically, the improvement on the unseen validation data is 23 – 29% in RMSE and 41 – 48% in FVU.

2) GENERALIZATION WITH SMALLER TRAINING SETS

We now examine how the performance varies with the training dataset size. Two reduced training sets are considered: a medium set (80% of the original, ≈ 1.5 min) and a small set (60%, ≈ 1 min), while the validation set remains unchanged. As shown in Table 1, our MS-NN-full consistently outperforms the benchmarks (G-NN, MS-NN-bench) across all sizes. The performance gap widens on unseen data as the training set shrinks: with the medium and small sets, our MS-NN-full achieves 24 – 38% better RMSE and 52 – 66% better FVU. Notably, the accuracy of the general-purpose G-NN drops sharply with less data, indicating poor generalization. In contrast, our MS-NN-full learns the underlying vehicle dynamics rather than specific data patterns, leading to better generalization with limited training data.

3) ABLATION STUDY

Table 1 analyzes the performance of the first two ablation models introduced in Sec. IV-A.3. Specifically, MS-NN-Ablation 1 removes the longitudinal acceleration a_x from the transient dynamics part of MS-NN-lat (Fig. 1b), while Ablation 2 further omits a_x from the quasi-steady-state component. These modifications decrease the number of trainable

[4]The FVU is the complement to 1 of the coefficient of determination R^2, and is defined as $FVU = 1 - R^2 = \text{var}(y - \hat{y})/\text{var}(y)$, where y is the actual output, $\hat{y}$ is the model prediction, and var(·) is the variance operation.

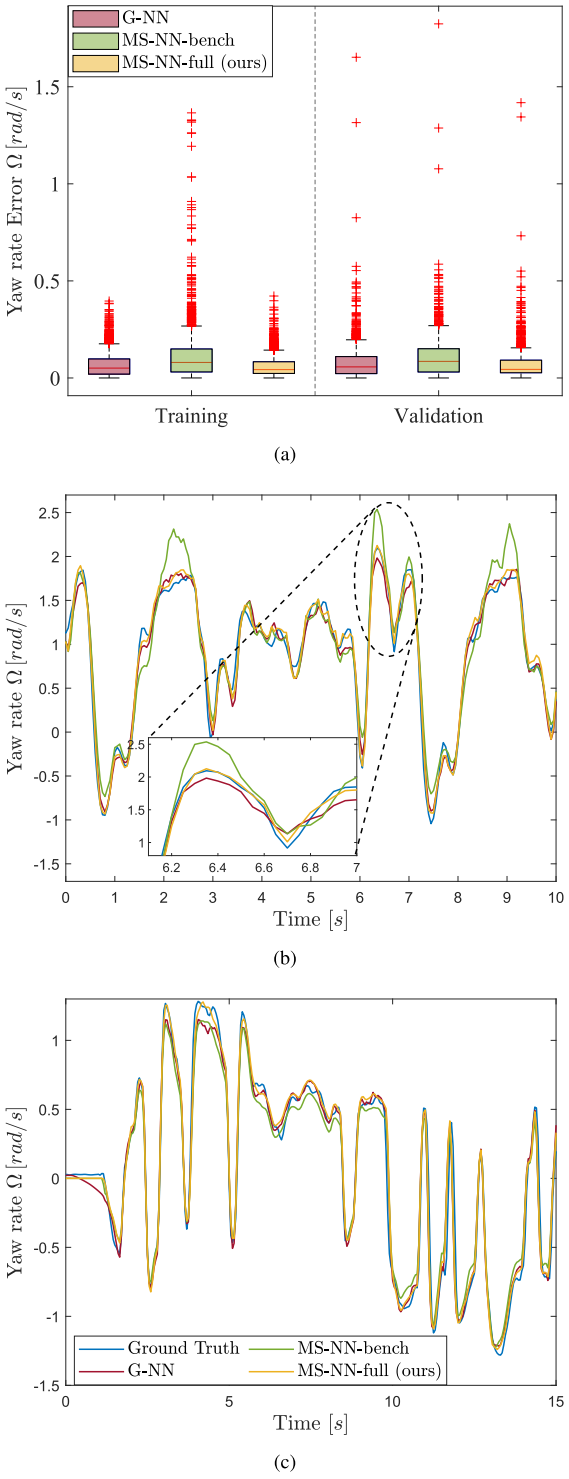


FIGURE 8. Prediction of the combined lateral dynamics (yaw rate Ω) on the RoboRacer vehicle: our MS-NN-full against the benchmarks G-NN and MS-NN-bench [15]. (a) Box plot of the Ω error; results on part of the training (b) and validation (c) sets.

parameters: 120 for Ablation 1 and 70 for Ablation 2, compared to our full model.

As shown in Table 1, the two ablations still outperform the benchmarks G-NN and MS-NN-bench. With the small training set, Ablation 2 even surpasses MS-NN-full on the validation data, indicating that this simpler model generalizes well from very limited data. With larger training sets, MS-NN-full outperforms Ablation 1 by 4 – 11% in RMSE and Ablation 2 by 6 – 9%, on the unseen validation set. This suggests that learning the effect of the longitudinal acceleration on the lateral dynamics enhances the model accuracy, particularly when sufficient data is available.

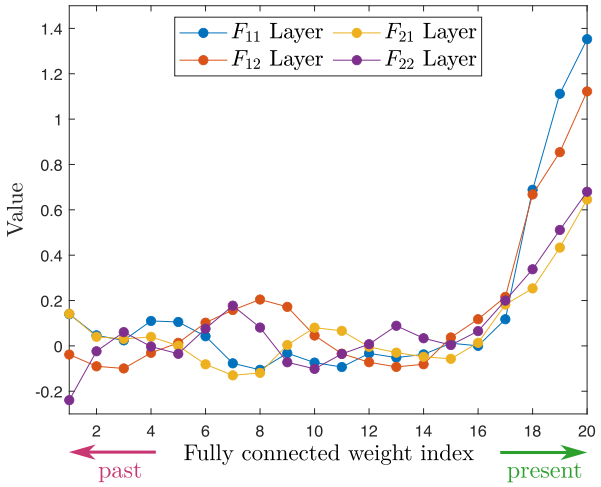


FIGURE 9. Interpreting the learned weights of the F_{lp} fully connected layers in our MS-NN-lat (purple blocks in Fig. 1b and Eq. (5)), $l \in \{1, 2\}$, $p \in \{1, 2\}$. Their increasing trend from past to present inputs indicates that more recent inputs have a higher influence on the output, and captures the transient response of the lateral vehicle dynamics.

4) INTERPRETABILITY

In this section, we provide some hints on interpreting the learned parameters of the proposed MS-NN-full architecture. As an example, we focus on the learned weights of the fully connected layers $\boldsymbol{F}_{lp}$ in MS-NN-lat (purple blocks in Fig. 1b and Eq. (5)). These weights learn the transient lateral dynamics of the vehicle in response to past inputs over a time horizon of $(q+1) \cdot T_s = 20 \cdot 0.05$ s $= 1$ s, where $q+1$ is the tuned number of past samples (see Sec. IV-B). Fig. 9 plots how the learned weights $\boldsymbol{F}_{lp}$ vary across the time horizon and for different local models, with $l \in \{1, n_{v,\text{lat}} = 2\}$ and $p \in \{1, n_{x,\text{lat}} = 2\}$ defining the local models in different ranges of longitudinal velocity v_x and acceleration a_x. As shown in Fig. 9, present inputs (closer to the current time step k) have larger weights, indicating a stronger influence on the output, while past inputs (further back in time) have smaller weights, indicating a weaker influence. Indeed, the $\boldsymbol{F}_{lp}$ weights represent the influence of past inputs on the current output, similar to the coefficients of a Finite Impulse Response (FIR) filter that captures the system's transient response.

The possibility to interpret the learned parameters in physical terms is a key advantage of the proposed model-structured architecture, which combines data-driven learning with physical insights.

5) COMBINED LONGITUDINAL DYNAMICS

This section evaluates the prediction results for the combined longitudinal dynamics, focusing on the longitudinal

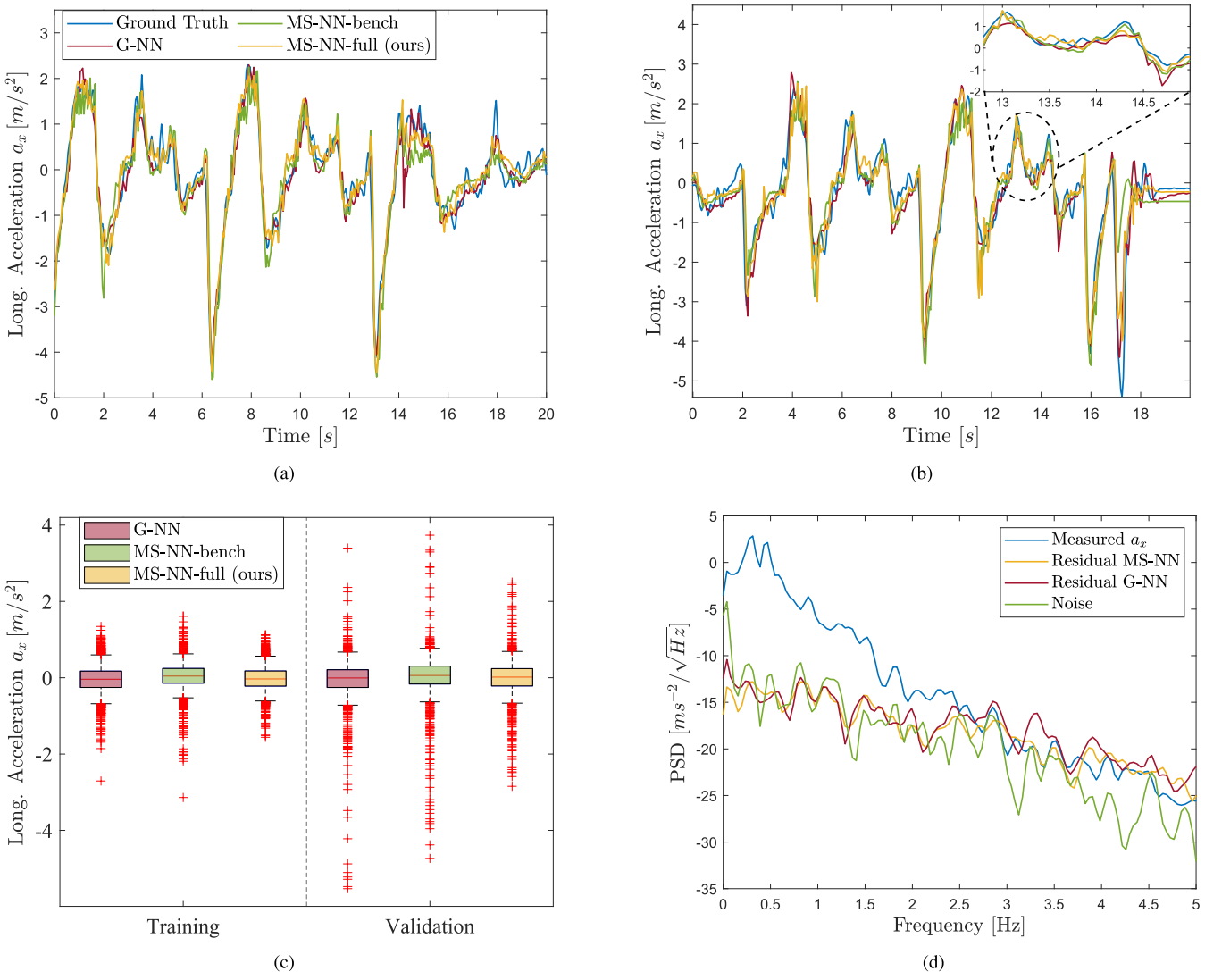


FIGURE 10. Prediction of the combined longitudinal dynamics (longitudinal acceleration a_x) on the RoboRacer vehicle: our MS-NN-full against the benchmarks G-NN and MS-NN-bench [14]. Results on parts of the training (a) and validation (b) sets; box plots of the a_x error (c); power spectral densities (d).

TABLE 2. Prediction of combined longitudinal dynamics (longitudinal acceleration a_x): comparison of the proposed MS-NN-full with the benchmarks (G-NN, MS-NN-bench) and an ablation.Bold: best; Underlined: second best).

Model	Metrics	Training	Validation
G-NN	RMSE ↓ [m/s^2]	<u>0.348</u>	<u>0.530</u>
	FVU ↓ [-]	<u>0.127</u>	<u>0.289</u>
MS-NN-bench [14]	RMSE ↓ [m/s^2]	0.492	0.615
	FVU ↓ [-]	0.331	0.464
MS-NN-Ablation 3	RMSE ↓ [m/s^2]	0.398	0.607
	FVU ↓ [-]	0.161	0.453
MS-NN-full (ours)	RMSE ↓ [m/s^2]	**0.328**	**0.521**
	FVU ↓ [-]	**0.106**	**0.281**

acceleration a_x. Table 2 and Fig. 10 compare our MS-NN-full with the benchmarks G-NN (Sec. IV-A.1) and MS-NN-bench [14] (Sec. IV-A.2). As shown in Fig. 10a, 10b, 10c, our MS-NN-full achieves lower median errors and reduces the occurrence of outliers compared to the benchmarks. Specifically, our MS-NN-full reduces the RMSE by 6 – 33% in training and 2 – 15% in validation (unseen data), and the FVU by 17 – 68% in training and 3 – 39% in validation. The performance improvement is larger in comparison with MS-NN-bench, which does not model the coupling with the lateral dynamics. On the other hand, the G-NN benchmark still performs relatively well. Indeed, part of the modeling complexity lies in the relation between the motor current input and the generated wheel torques (actuation model in Fig. 3), which is hard to capture with physics-based models and first principles, and requires data-driven learning. For this reason, the G-NN benchmark, which directly learns the input-output mapping, can achieve satisfactory performance in this case.

Moreover, the measured acceleration a_x signal in Fig. 10a–10b is noisier compared to the yaw rate Ω in Fig. 8b–8c. This poses a challenge for the models, as

they must learn the underlying vehicle dynamics without overfitting noise. Fig. 10d shows the Power Spectral Densities (PSD) of the measured a_x signal, along with the estimated noise[5] and the residuals of our MS-NN-full and the G-NN benchmark. Up to 3 Hz, the PSD of both models' residuals closely follows the noise level, indicating that they have effectively learned the informative content of the acceleration signal, with the remaining part corresponding to noise. Beyond 3 Hz, the measured signal's power is closer to the noise level, meaning that this frequency range contains negligible useful information for vehicle dynamics modeling. This spectral analysis explains how MS-NN-full avoids overfitting the noise while capturing the vehicle dynamics in the informative frequency range.

6) ABLATION STUDY

Table 2 compares our MS-NN-full against the MS-NN-Ablation 3 introduced in Sec. IV-A.3, which removes the coupling with the lateral dynamics in MS-NN-long (*i.e.,* we remove the lateral force model in Fig. 3). This ablation worsens the RMSE by 14% and the FVU by 38% on the unseen validation data, highlighting the impact of the lateral-longitudinal dynamics coupling.

D. GENERALIZATION TO NEW VEHICLE SETUPS

This section evaluates the ability of the different models to generalize and rapidly adapt to new vehicle setups. After training all models with the nominal RoboRacer vehicle, we perform the following changes:

- Different tire types: we test two new tire configurations with varying grip levels (tire sets 1 and 2 in Fig. 11a).
- Increased vehicle mass: we increase the total vehicle mass by 10%.

We evaluate the models in a zero-shot deployment scenario (without any further training) and after a **rapid adaptation** phase via fine-tuning on a small dataset.

1) FINE-TUNING SETTINGS

For each new setup, we use 20 seconds of autonomous driving data[6] in the new vehicle conditions, and we fine-tune all models for 60 epochs, starting from the pre-trained weights obtained with the nominal vehicle. In the case of the benchmarks G-NN and MS-NN-bench, we fine-tune all learnable parameters. In contrast, for our MS-NN-full, we selectively fine-tune only the parameters most affected by the changes in vehicle setup, exploiting the modular and physically interpretable structure of our architecture.

For the combined lateral dynamics, fine-tuning is restricted to the transient dynamics part of our MS-NN-lat (Fig. 1b), as tire variations and vehicle mass changes were found to primarily impact the transient vehicle response. Hence, we fine-tune $(q+1)\cdot n_{v,\text{lat}}\cdot n_{x,\text{lat}} = 20\cdot 2\cdot 2 = 80$ parameters.

[5]Noise is estimated with the vehicle driving straight at constant speed. All PSDs are estimated using the Welch's method, employing a Hamming window of 256 samples with 50% overlap and a 512-point FFT.

[6]The new data is collected in the way described in Sec. III-B.

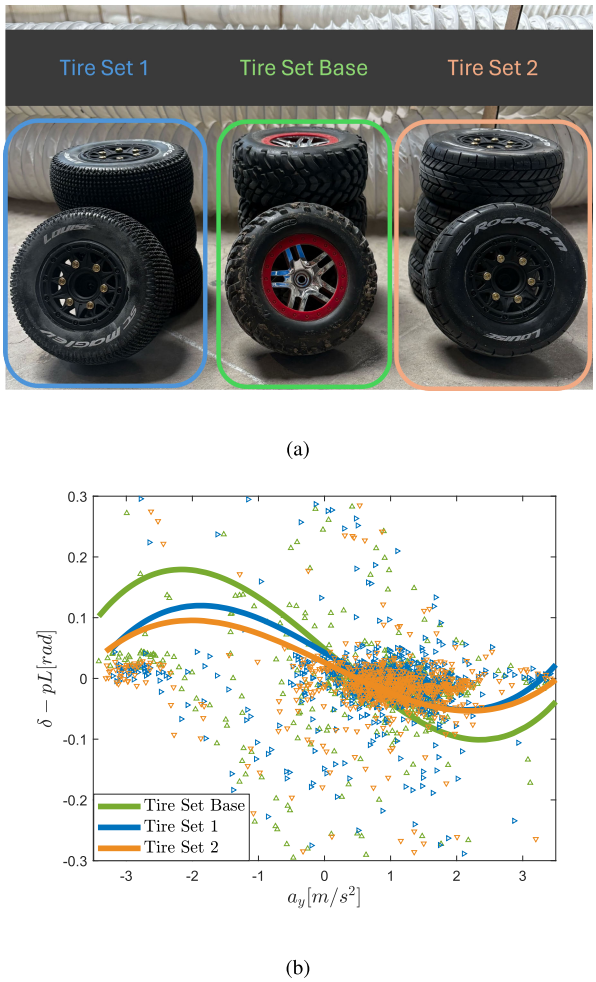


FIGURE 11. (a) Different tire types used in our generalization and adaptation analyses: nominal tire set (green) and two new tire sets (blue and orange). (b) Experimental handling diagrams of the RoboRacer vehicle with the three tire setups, and polynomial fits of the diagrams (solid lines).

For the combined longitudinal dynamics, fine-tuning is applied to the actuation model in Fig. 3, which captures the relation between the motor current input and the generated wheel torques, and to the vehicle mass m appearing in Eq. (12c). Thus, we fine-tune 341 parameters.

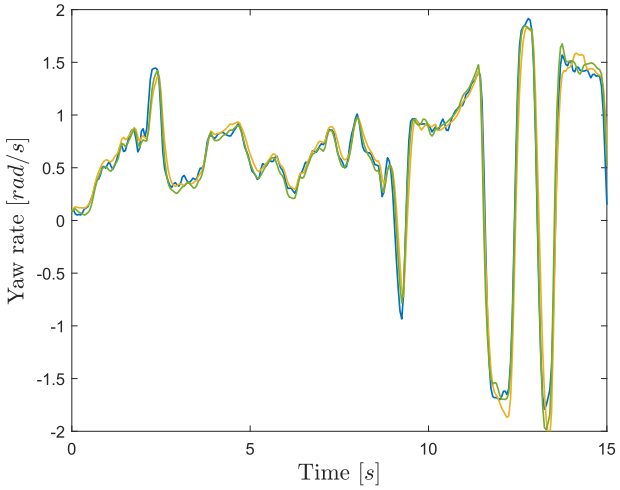
Our fine-tuning strategy aims to minimize the computational load and data requirements, paving the way for online adaptation.

2) DIFFERENT TIRE TYPES

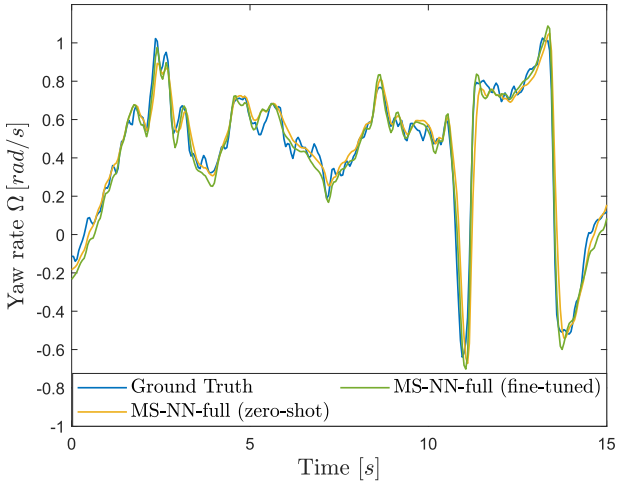
We now analyze the two new tire configurations. To assess their impact on the vehicle dynamics, we perform sinusoidal steering maneuvers with our RoboRacer and plot the resulting Handling Diagrams (HDs) for the three tire setups (nominal, set 1, set 2). The HD characterizes a vehicle's under- or oversteering behavior [8], [20] by comparing the actual steering angle δ with the kinematic angle ρL, where L is the wheelbase, as a function of a_y. As shown in Fig. 11b, the different tire types affect the HD shape and thus the steering response: the vehicle becomes less oversteering with the new

TABLE 3. Zero-shot deployment and rapid adaptation via fine-tuning to new vehicle setups: comparison of our MS-NN-full with the benchmarks (G-NN, MS-NN-bench) across three test cases—two new tire types and a vehicle mass increase. FVU values are shown in parentheses. (Bold: best; Underlined: second best).

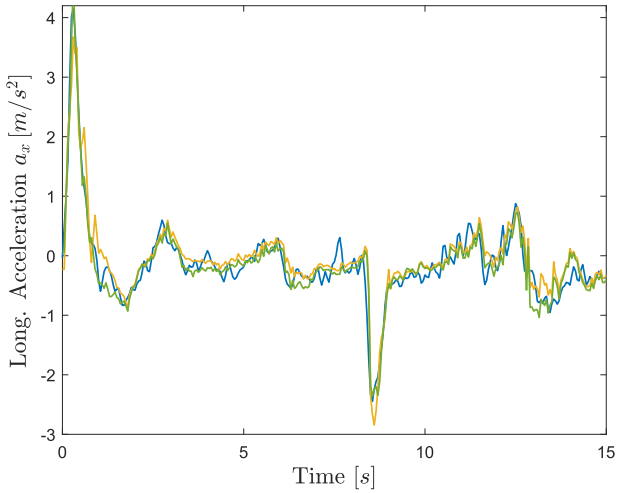
Test Case	Models	Yaw Rate Ω: RMSE (FVU) Zero-shot $\longrightarrow$ Fine-tuning	Long. Accelerat. a_x: RMSE (FVU) Zero-shot $\longrightarrow$ Fine-tuning
Tire set 1	G-NN	0.147 (0.050) $\longrightarrow$ 0.107 (0.027)	0.614 (0.485) $\longrightarrow$ **0.380 (0.185)**
	MS-NN-bench	0.102 (0.024) $\longrightarrow$ 0.096 (0.022)	0.462 (0.275) $\longrightarrow$ 0.425 (0.233)
	MS-NN-full (ours)	0.172 (0.068) $\longrightarrow$ **0.085 (0.017)**	0.531 (0.363) $\longrightarrow$ 0.395 (0.201)
Tire set 2	G-NN	0.110 (0.067) $\longrightarrow$ 0.070 (0.027)	0.624 (0.537) $\longrightarrow$ 0.368 (0.187)
	MS-NN-bench	0.079 (0.035) $\longrightarrow$ 0.059 (0.019)	0.373 (0.191) $\longrightarrow$ 0.347 (0.166)
	MS-NN-full (ours)	0.129 (0.093) $\longrightarrow$ **0.053 (0.015)**	0.468 (0.300) $\longrightarrow$ **0.334 (0.153)**
+10% Mass	G-NN	0.088 (0.061) $\longrightarrow$ 0.046 (0.017)	0.489 (0.974) $\longrightarrow$ 0.329 (0.441)
	MS-NN-bench	0.072 (0.042) $\longrightarrow$ 0.053 (0.023)	0.396 (0.639) $\longrightarrow$ 0.375 (0.572)
	MS-NN-full (ours)	0.075 (0.046) $\longrightarrow$ **0.045 (0.016)**	0.398 (0.648) $\longrightarrow$ **0.290 (0.344)**



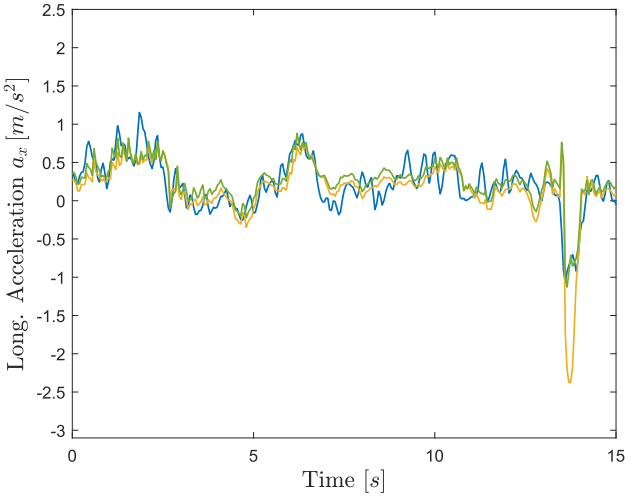
(a) Tire Set 2.



(b) +10% Mass.



(c) Tire Set 2.



(d) +10% Mass.

FIGURE 12. Prediction of the yaw rate Ω (a, b) and longitudinal acceleration a_x (c, d) under new vehicle setups, with zero-shot deployment (yellow lines) and after rapid adaptation via fine-tuning (green lines): tire set n.2 (a, c) and + 10% mass (b, d).

tire sets. This highlights the need for model adatation in the new conditions.

Table 3 summarizes the results for tire sets 1 and 2. Under zero-shot generalization (training on nominal tires and testing

on new ones without fine-tuning), the best performance is achieved by MS-NN-bench [14], [15]. This is due to its lower parameter count, which reduces the tendency to overspecialize for the nominal setup. After rapid adaptation (20 s of new data and 60 fine-tuning epochs), our MS-NN-full outperforms the benchmarks. For example, with tire set 2, MS-NN-full improves the predictions of the yaw rate (10 – 24% better RMSE, 21 – 44% better FVU) and longitudinal acceleration (4 – 9% better RMSE, 8 – 18% better FVU), compared to MS-NN-bench and G-NN. For tire set 1, the a_x prediction after fine-tuning is comparable between MS-NN-full and G-NN (the latter being slightly better by 3%): as discussed in Sec. IV-C.5, the longitudinal dynamics strongly depend on the data-driven actuation model, reducing the relative advantage of MS-NN-full.

Fig. 12a–12c plot the predictions of our MS-NN-full with tire set 2, showing that fine-tuning improves the accuracy especially close to the main peaks.

These results demonstrate the effectiveness of our MS-NN-full model in rapidly adapting to new friction conditions with minimal data and computational effort.

3) VEHICLE MASS VARIATION

The results for the + 10% vehicle mass scenario are reported in Table 3. In zero-shot deployment, our MS-NN-full and MS-NN-bench achieve comparable performance (4% difference in RMSE for Ω, 0.5% for a_x) and yield better predictions than G-NN. After rapid adaptation via fine-tuning, our MS-NN-full outperforms the benchmarks in both outputs (by 2 – 23% in RMSE and 6 – 40% in FVU).

Fig. 12b–12d show the predictions of our MS-NN-full before and after fine-tuning. The longitunal dynamics are significantly affected by the mass increase, as the same motor current input produces lower acceleration: fine-tuning effectively captures this change, improving the prediction accuracy.

V. CONCLUSION AND FUTURE WORK

In this paper, we introduce a new model-structured neural network (MS-NN-full) to learn the coupled lateral-longitudinal vehicle dynamics. Our architecture embeds physical insights directly into the NN design. To learn the combined lateral dynamics, we use a multi-model structure with dedicated neuro-fuzzy modules that capture the quasi-steady-state and transient responses. Their formulations draw from physical vehicle diagrams and principles, such as the dependence of the transient behavior on past inputs, the influence of the longitudinal speed and acceleration. We also analyze the interpretability of the learned parameters. The longitudinal dynamics combine a Newtonian physics-based module, a data-driven actuation block to learn the unknown motor-torque map, and a structured neural module that learns the coupling with the lateral dynamics. These components form a unified closed-loop model that predicts the yaw rate and longitudinal acceleration from past windows of requested steering angle, motor current, and vehicle speed.

We validate MS-NN-full on a 1:10-scale autonomous electric vehicle. Trained on just 2 minutes of driving data, MS-NN-full outperforms two existing MS-NN baselines and a general-purpose NN (G-NN) in both accuracy and generalization. On unseen data, MS-NN-full reduces the yaw rate RMSE and FVU by 23 – 29% and 41 – 48%, and reduces longitudinal acceleration RMSE and FVU by 2–15% and 3 – 39%, respectively. The relative advantage of our model grows as the training set shrinks, indicating improved generalization by learning the underlying dynamics rather than maneuver patterns. An ablation study highlights the impact of explicitly modeling the lateral-longitudinal dynamics coupling.

We further show that MS-NN-full rapidly adapts to new vehicle configurations with different tires and increased mass. With only 20 seconds of fine-tuning data, and by re-training only a subset of parameters, it outperforms fine-tuned baselines by 10 – 24% in yaw-rate RMSE and 4 – 23% in longitudinal-acceleration RMSE.

Our open-source implementation and datasets are expected to favor further research and developments. Future work will incorporate the proposed MS-NN-full into a model predictive controller for autonomous racing, and will explore the applicability of our architecture to full-scale autonomous race cars.

CONTRIBUTIONS OF THE AUTHORS

Aniello Mungiello: code implementation, experiments, and first-draft writing. Felix Jahncke: hardware setup support and paper review. Stefania Santini: paper review and funding acquisition. Johannes Betz: paper review and funding acquisition. Gastone Pietro Rosati Papini: development of the general concept of MS-NNs, `nnodely` software library, and paper review. Mattia Piccinini: project lead, conceptualization, methodology, supervision, and paper writing across all stages.

REFERENCES

[1] T. Zhang, Y. Sun, Y. Wang, B. Li, Y. Tian, and F.-Y. Wang, "A survey of vehicle dynamics modeling methods for autonomous racing: Theoretical models, physical/virtual platforms, and perspectives," *IEEE Trans. Intell. Vehicles*, vol. 9, no. 3, pp. 4312–4334, Mar. 2024.

[2] J. Betz et al., "Autonomous vehicles on the edge: A survey on autonomous vehicle racing," *IEEE Open J. Intell. Transp. Syst.*, vol. 3, pp. 458–488, 2022.

[3] M. Piccinini, S. Taddei, J. Betz, and F. Biral, "Kineto-dynamical planning and accurate execution of minimum-time maneuvers on three-dimensional circuits," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2025, pp. 1–7.

[4] 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.

[5] M. Abe, *Vehicle Handling Dynamics: Theory and Application*. Amsterdam, The Netherlands: Elsevier, 2009.

[6] H. Pacejka, *Tire and Vehicle Dynamics*, 3rd ed., Amsterdam, The Netherlands: Elsevier, 2012.

[7] D. Stocco, F. Biral, and E. Bertolazzi, "A physical tire model for real-time simulations," *Math. Comput. Simul.*, vol. 223, pp. 654–676, Sep. 2024.

[8] M. Guiggiani, *The Science of Vehicle Dynamics: Handling, Braking, and Ride of Road and Race Cars*. Cham, Switzerland: Springer, 2018.

[9] L. Hermansdorfer, R. Trauth, J. Betz, and M. Lienkamp, "End-to-end neural network for vehicle dynamics modeling," in *Proc. 6th IEEE Congr. Inf. Sci. Technol.*, Jul. 2020, pp. 407–412.

[10] S. Kuutti, R. Bowden, Y. Jin, P. Barber, and S. Fallah, "A survey of deep learning applications to autonomous vehicle control," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 2, pp. 712–733, Feb. 2021.

[11] G. P. R. Papini. (2025). *Nnodely: Model-Structured Neural Networks for Modeling, Control and Estimation of Physical Systems*. [Online]. Available: https://github.com/tonegas/nnodely

[12] M. Piccinini, M. Zumerle, J. Betz, and G. Pietro Rosati Papini, "A road friction-aware anti-lock braking system based on model-structured neural networks," *IEEE Open J. Intell. Transp. Syst.*, vol. 6, pp. 522–536, 2025.

[13] M. Piccinini, A. Mungiello, G. Jank, G. P. R. Papini, F. Biral, and J. Betz, "Model-structured neural networks to control the steering dynamics of autonomous race cars," in *Proc. IEEE 28th Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2025, pp. 4129–4136.

[14] M. Da Lio, D. Bortoluzzi, and G. P. Rosati Papini, "Modelling longitudinal vehicle dynamics with neural networks," *Vehicle Syst. Dyn.*, vol. 58, no. 11, pp. 1675–1693, Nov. 2020.

[15] M. Da Lio, R. Dona, G. P. R. Papini, F. Biral, and H. Svensson, "A mental simulation approach for learning neural-network predictive control (in self-driving Cars)," *IEEE Access*, vol. 8, pp. 192041–192064, 2020.

[16] N. Baumann et al., "ForzaETH race stack-scaled autonomous head-to-head racing on fully commercial off-the-shelf hardware," *J. Field Robot.*, vol. 42, no. 4, pp. 1037–1079, 2025.

[17] P. Karle et al., "EDGAR: An autonomous driving research platform {-} from feature development to real-world application," 2023, *arXiv:2309.15492*.

[18] M. Althoff, M. Koschi, and S. Manzinger, "CommonRoad: Composable benchmarks for motion planning on roads," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2017, pp. 719–726.

[19] J. K. Subosits and J. C. Gerdes, "From the racetrack to the road: Real-time trajectory replanning for autonomous driving," *IEEE Trans. Intell. Vehicles*, vol. 4, no. 2, pp. 309–320, Jun. 2019.

[20] E. Pagot, M. Piccinini, and F. Biral, "Real-time optimal control of an autonomous RC car with minimum-time maneuvers and a novel kineto-dynamical model," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 2390–2396.

[21] B. Li, Y. Ouyang, X. Li, D. Cao, T. Zhang, and Y. Wang, "Mixed-integer and conditional trajectory planning for an autonomous mining truck in loading/dumping scenarios: A global optimization approach," *IEEE Trans. Intell. Vehicles*, vol. 8, no. 2, pp. 1512–1522, Feb. 2023.

[22] J. L. Vázquez, M. Brühlmeier, A. Liniger, A. Rupenyan, and J. Lygeros, "Optimization-based hierarchical motion planning for autonomous racing," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 2397–2403.

[23] B. Zarrouki, C. Wang, and J. Betz, "A stochastic nonlinear model predictive control with an uncertainty propagation horizon for autonomous vehicle motion control," in *Proc. Amer. Control Conf. (ACC)*, Jul. 2024, pp. 5466–5473.

[24] A. Raji et al., "Motion planning and control for multi vehicle autonomous racing at high speeds," in *Proc. IEEE 25th Int. Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2022, pp. 2775–2782.

[25] F. Jahncke et al., "Differentiable weights-varying nonlinear MPC via gradient-based policy learning: An autonomous vehicle guidance example," *IEEE Robot. Autom. Lett.*, vol. 11, no. 3, pp. 3724–3731, Mar. 2026.

[26] C. Voser, R. Y. Hindiyeh, and J. C. Gerdes, "Analysis and control of high sideslip manoeuvres," *Vehicle Syst. Dyn.*, vol. 48, no. sup1, pp. 317–336, Dec. 2010.

[27] J. K. Subosits and J. C. Gerdes, "Impacts of model fidelity on trajectory optimization for autonomous vehicles in extreme maneuvers," *IEEE Trans. Intell. Vehicles*, vol. 6, no. 3, pp. 546–558, Sep. 2021.

[28] G. Corradini, D. Stocco, E. Bertolazzi, and F. Biral, "Estimation of tire forces and torques via nonlinear suspension models and optimal control," *J. Comput. Nonlinear Dyn.*, vol. 21, no. 4, Apr. 2026, Art. no. 041002.

[29] D. Weihmayr, C. Birkner, H. Marzbani, and R. N. Jazar, "Data-driven vehicle dynamics: Leveraging SINDy for optimization-based vehicular motion planning," *IEEE Access*, vol. 13, pp. 136584–136597, 2025.

[30] G. Devineau, P. Polack, F. Altché, and F. Moutarde, "Coupled longitudinal and lateral control of a vehicle using deep learning," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2018, pp. 642–649.

[31] M. Piccinini, M. Larcher, E. Pagot, D. Piscini, L. Pasquato, and F. Biral, "A predictive neural hierarchical framework for on-line time-optimal motion planning and control of black-box vehicle models," *Vehicle Syst. Dyn.*, vol. 61, no. 1, pp. 83–110, Jan. 2023.

[32] 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, p. 1975, Mar. 2019.

[33] Z. Gao, W. Wen, Y. Xing, and A. Tsourdos, "An integrated framework for autonomous driving planning and tracking based on NNMPC considering road surface variations," *IEEE Trans. Intell. Vehicles*, vol. 10, no. 2, pp. 848–862, Feb. 2025.

[34] 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.

[35] S. Gottschalk, M. Gerdts, and M. Piccinini, "Reinforcement learning and optimal control: A hybrid collision avoidance approach," in *Proc. 10th Int. Conf. Vehicle Technol. Intell. Transp. Syst.*, 2024, pp. 76–87.

[36] T. Weiss and M. Behl, "DeepRacing: A framework for autonomous racing," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2020, pp. 1163–1168.

[37] W. Xiao, H. Xue, T. Tao, D. Kalaria, J. M. Dolan, and G. Shi, "AnyCar to anywhere: Learning universal dynamics model for agile and adaptive mobility," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2025, pp. 8819–8825.

[38] Y. Tsuchiya, T. Balch, P. Drews, and G. Rosman, "Online adaptation of learned vehicle dynamics model with meta-learning approach," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2024, pp. 802–809.

[39] O. Dikici et al., "Learning-based on-track system identification for scaled autonomous racing in under a minute," *IEEE Robot. Autom. Lett.*, vol. 10, no. 2, pp. 1984–1991, Feb. 2025.

[40] J. Miao et al., "Residual learning towards high-fidelity vehicle dynamics modeling with transformer," *IEEE Robot. Autom. Lett.*, vol. 10, no. 7, pp. 7404–7411, Jul. 2025.

[41] H. Xue, E. L. Zhu, J. M. Dolan, and F. Borrelli, "Learning model predictive control with error dynamics regression for autonomous racing," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2024, pp. 13250–13256.

[42] J. Kabzan, L. Hewing, A. Liniger, and M. N. Zeilinger, "Learning-based model predictive control for autonomous racing," *IEEE Robot. Autom. Lett.*, vol. 4, no. 4, pp. 3363–3370, Oct. 2019.

[43] Y. Zhang, P. Tino, A. Leonardis, and K. Tang, "A survey on neural network interpretability," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 5, no. 5, pp. 726–742, Oct. 2021.

[44] J. Drgona et al., "Safe physics-informed machine learning for dynamics and control," in *Proc. Amer. Control Conf. (ACC)*, Jul. 2025, pp. 591–606.

[45] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *J. Comput. Phys.*, vol. 378, pp. 686–707, Feb. 2019.

[46] C. Meng, S. Griesemer, D. Cao, S. Seo, and Y. Liu, "When physics meets machine learning: A survey of physics-informed machine learning," *Mach. Learn. for Comput. Sci. Eng.*, vol. 1, no. 1, p. 20, May 2025.

[47] H. Lim, J. Won Lee, J. Boyack, and J. Brad Choi, "EV-PINN: A physics-informed neural network for predicting electric vehicle dynamics," 2024, *arXiv:2411.14691*.

[48] H. Zeipel, T. Habich, T. Seel, and S. F. G. Ehlers, "Fast and accurate prediction of vehicle dynamics using physics-informed neural networks," to be published.

[49] S. Liang, Z. Shi, H. Chen, H. Ren, and L. Meng, "Survey on physics-informed neural networks for vehicle dynamics modeling in autonomous driving," in *Proc. 25th Int. Conf. Softw. Qual., Rel., Secur. Companion (QRS-C)*, Jul. 2025, pp. 311–319.

[50] M. Piccinini, S. Taddei, M. Larcher, M. Piazza, and F. Biral, "A physics-driven artificial agent for online time-optimal vehicle motion planning and control," *IEEE Access*, vol. 11, pp. 46344–46372, 2023.

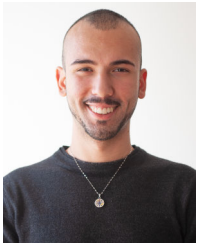
[51] J. Chrosniak, J. Ning, and M. Behl, "Deep dynamics: Vehicle dynamics modeling with a physics-constrained neural network for autonomous racing," *IEEE Robot. Autom. Lett.*, vol. 9, no. 6, pp. 5292–5297, Jun. 2024.

[52] S. Fang and K. Yu, "Fine-tuning hybrid dynamics with physics-informed neural networks for vehicle dynamics estimation," *Int. J. Intell. Robot. Appl.*, vol. 9, no. 4, pp. 1–17, Dec. 2025.

[53] J. Wegrzynowski, G. Czechmanowski, P. Kicki, and K. Walas, "Learning dynamics models for velocity estimation in autonomous racing," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2024, pp. 972–979.

[54] Z. Zhou, Y. Wang, G. Zhou, X. Liu, M. Wu, and K. Dai, "Vehicle lateral dynamics-inspired hybrid model using neural network for parameter identification and error characterization," *IEEE Trans. Veh. Technol.*, vol. 73, no. 11, pp. 16173–16186, Nov. 2024.

[55] T. Kim, H. Lee, and W. Lee, "Physics embedded neural network vehicle model and applications in risk-aware autonomous driving using latent features," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2022, pp. 4182–4189.
[56] E. Pagot, M. Piccinini, E. Bertolazzi, and F. Biral, "Fast planning and tracking of complex autonomous parking maneuvers with optimal control and pseudo-neural networks," *IEEE Access*, vol. 11, pp. 124163–124180, 2023.
[57] M. Piccinini, S. Gottschalk, M. Gerdts, and F. Biral, "Computationally efficient minimum-time motion primitives for vehicle trajectory planning," *IEEE Open J. Intell. Transp. Syst.*, vol. 5, pp. 642–655, 2024.
[58] M. D. Lio, M. Piccinini, and F. Biral, "Robust and sample-efficient estimation of vehicle lateral velocity using neural networks with explainable structure informed by kinematic principles," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 12, pp. 13670–13684, Dec. 2023.
[59] S. A. Faroughi et al., "Physics-guided, physics-informed, and physics-encoded neural networks and operators in scientific computing: Fluid and solid mechanics," *J. Comput. Inf. Sci. Eng.*, vol. 24, no. 4, Apr. 2024, Art. no. 040802.
[60] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent.*, 2014.
[61] H. Bozdogan, "Model selection and Akaike's information criterion (AIC): The general theory and its analytical extensions," *Psychometrika*, vol. 52, no. 3, pp. 345–370, Sep. 1987.



ANIELLO MUNGIELLO (Graduate Student Member, IEEE) received the M.S. degree in autonomous vehicle engineering from the University of Naples Federico II in 2022, with a thesis focused on the design and experimental validation of an autonomous driving control system for a race car. He is currently pursuing the Ph.D. degree in information technology and electrical engineering. During his Ph.D., he spent six months as a Visiting Researcher at the Technical University of Munich (TUM). His research interests include optimal control strategies for vehicle dynamics and cooperative driving.

FELIX JAHNCKE (Graduate Student Member, IEEE) received the bachelor's degree in mechanical engineering and the master's degree in automotive engineering from the Technical University of Munich (TUM), where he is currently pursuing the Ph.D. degree, focusing on the transition from traditional software architectures toward end-to-end differentiable learning methods. During his master's studies, he completed research visits at Delft University of Technology, The Netherlands, and the University of Pennsylvania, USA.

STEFANIA SANTINI is currently a Professor with the Department of Electrical Engineering and Information Technologies (DIETI), University of Naples Federico II, Naples, Italy, where she leads the Distributed Automation Systems Laboratory. She is involved in many projects with industry, including small- and medium-sized enterprises, also operating in the transportation field. Her research interests include nonlinear control of nonlinear and cyber-physical systems, networked control with applications to energy, automotive engineering, and transportation technologies. She is a member of IEEE TC on Smart Cities (TC-SC). She is the Vice Chair of IEEE ITSS-Italian Chapter. She is a Senior Editor of IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS.

JOHANNES BETZ (Member, IEEE) received the B.Eng. degree from Coburg University of Applied Sciences in 2011, the M.Sc. degree from the University of Bayreuth in 2012, and the Ph.D. and M.A. degrees in philosophy from the Technical University of Munich (TUM) in 2019 and 2021, respectively. He is currently an Assistant Professor with the Department of Mobility Systems Engineering, TUM, where he is leading the Autonomous Vehicle Systems (AVS) Laboratory. He is one of the founders of the TUM Autonomous Motorsport Team. His research focuses on developing adaptive dynamic path planning and control algorithms, decision-making algorithms that work under high uncertainty in multi-agent environments, and validating the algorithms on real-world robotic systems.

GASTONE PIETRO ROSATI PAPINI (Member, IEEE) received the M.Sc. degree in automation engineering from the University of Pisa in 2012 and the Ph.D. degree in emerging digital technologies from the Sant'Anna School of Advanced Studies in 2016. He is currently an Associate Professor with the Department of Industrial Engineering, University of Trento. He is the Co-Founder of Cheros s.r.l., a spin-off company from the University of Pisa. His research focuses on advanced control and learning methods for autonomous systems, particularly in autonomous driving and ADAS, integrating model-based and data-driven approaches. In December 2024, he was awarded Italian FIS2 Grant for the Project "Neu4mes: Structured Neural Network Framework for Modeling and Control of Autonomous Systems." For more information please visit his webpage: https://tonegas.com/

MATTIA PICCININI (Member, IEEE) received the M.Sc. degree (cum laude) in mechatronics engineering and the Ph.D. degree (cum laude) in autonomous systems from the University of Trento, Italy, in 2019 and 2024, respectively. He was a Visiting Researcher at the Universität der Bundeswehr München and Eindhoven University of Technology. He is currently a Humboldt Post-Doctoral Fellow with the Autonomous Vehicle Systems (AVS) Laboratory, Technical University of Munich (TUM). His research focuses on physics-guided motion generation and the control of mobile ground robots. He received the TUM Global Post-Doctoral Fellowship in 2024, the ITSS Best Dissertation Award in 2025, and the Humboldt Post-Doctoral Fellowship in 2025. He serves as an Associate Editor for IEEE IROS and IEEE ITSC conferences.