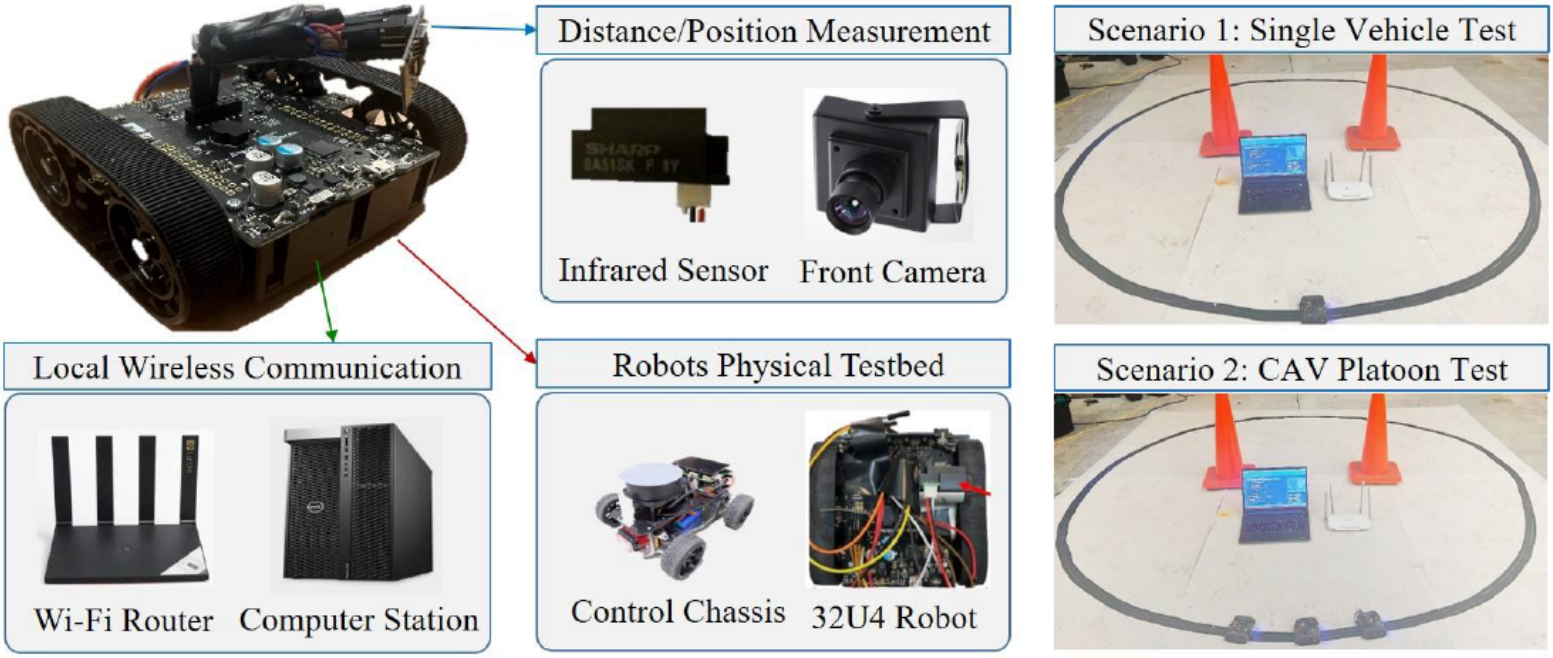
CENTER FOR CONNECTED AND AUTOMATED TRANSPORTATION

Final Report CCAT-2025-527
August 2025



Traffic Control based on CARMA Platform for Maximal Traffic Mobility and Safety

Xiaopeng (Shaw) Li, PhD
David Noyce, PhD
Heye Huang, PhD

WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON

CENTER FOR CONNECTED AND AUTOMATED TRANSPORTATION

Report No. CCAT-2025-527 **August 2025**

Project Start Date: January 1, 2024

Project End Date: July 31, 2025

Traffic Control based on CARMA Platform for Maximal Traffic Mobility and Safety

by
Xiaopeng (Shaw) Li, Professor
David Noyce, Professor
University of Wisconsin–Madison

CENTER FOR CONNECTED AND AUTOMATED TRANSPORTATION

DISCLAIMER

Funding for this research was provided by the Center for Connected and Automated Transportation under Grant No. 69A3551747105 of the U.S. Department of Transportation, Office of the Assistant Secretary for Research and Technology (OST-R), University Transportation Centers Program. The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated under the sponsorship of the Department of Transportation, University Transportation Centers Program, in the interest of information exchange. The U.S. Government assumes no liability for the contents or use thereof.

Suggested APA Format Citation:

Li, X., Noyce, D., Huang, H. (2025). Traffic Control based on CARMA Platform for Maximal Traffic Mobility and Safety. Report No. CCAT-2025-527.

Contacts

For more information:

Xiaopeng (Shaw) Li
University of Wisconsin–Madison
1415 Engineering Drive Madison, WI 53706
Phone: 608-890-0104
Email: xli2485@wisc.edu

David Noyce
University of Wisconsin–Madison
1415 Engineering Drive Madison, WI 53706
Phone: 608-265-1882
Email:danoyce@wisc.edu

CCAT
University of Michigan Transportation Research Institute
2901 Baxter Road
Ann Arbor, MI 48152
uumtri-ccat@umich.edu
(734) 763-2498

CENTER FOR CONNECTED AND AUTOMATED TRANSPORTATION

Technical Report Documentation Page

1. Report No. CCAT-2025-527	**2. Government Accession No.**	**3. Recipient's Catalog No.**
4. Title and Subtitle Traffic Control based on CARMA Platform for Maximal Traffic Mobility and Safety		**5. Report Date** August 2025 **6. Performing Organization Code** N/A
7. Author(s) Xiaopeng (Shaw) Li, Ph.D.: https://orcid.org/0000-0002-5264-3775 David Noyce, Ph.D.: https://orcid.org/0000-0001-5887-8391 Heye Huang, Ph.D.: https://orcid.org/0000-0002-8964-8764		**8. Performing Organization Report No.** Enter any/all unique alphanumeric report numbers assigned by the performing organization, if applicable.
9. Performing Organization Name and Address Center for Connected and Automated Transportation University of Michigan Transportation Research Institute 2901 Baxter Road Ann Arbor, MI 48109 ADD YOUR ORGANIZATION'S NAME AND ADDRESS HERE.		**10. Work Unit No.** **11. Contract or Grant No.** Contract No. 69A3551747105
12. Sponsoring Agency Name and Address U.S. Department of Transportation Office of the Assistant Secretary for Research and Technology 1200 New Jersey Avenue, SE Washington, DC 20590		**13. Type of Report and Period Covered** Final Report (January 2023 – July 2025) **14. Sponsoring Agency Code** OST-R
15. Supplementary Notes Conducted under the U.S. DOT Office of the Assistant Secretary for Research and Technology's (OST-R) University Transportation Centers (UTC) program.		

16. Abstract

This project developed and evaluated an online adaptive platoon control framework for connected and automated vehicles (CAVs) that simultaneously enhances mobility and safety through integration with digital infrastructure based on the CARMA platform. The proposed Physics Enhanced Residual Learning (PERL) framework combines a physics-based centralized controller, which models vehicle dynamics to ensure stability, with a neural network-based residual learning module that adaptively corrects unmodeled dynamics in real time. PERL can contribute to CARMA's platooning plugin as a tactical-level cooperative longitudinal control component, enabling adaptive gap regulation and disturbance mitigation. High-fidelity simulations and scaled robot car experiments were conducted to assess performance under diverse traffic and disturbance scenarios. Results show that the PERL framework significantly improves position and speed tracking accuracy, achieves rapid convergence following external disturbances, and maintains robust platoon stability compared to purely physics-based or purely learning-based approaches. These findings demonstrate that the PERL framework can reduce the conventional safety–mobility trade-off in CAV platooning and support deployment within Transportation System Management and Operations (TSMO) strategies. Transportation agencies and system developers may apply this approach to improve cooperative driving efficiency, enhance roadway throughput, and inform future standards for adaptive platoon control systems.

UMTRI | The University of Akron | Central State University | Illinois Center for Transportation | University of Minnesota | Northwestern | Purdue University | Washtenaw Community College | Wisconsin – University of Wisconsin–Madison

CENTER FOR CONNECTED AND AUTOMATED TRANSPORTATION

17. Key Words	18. Distribution Statement		
Connected and automated vehicles; Vehicle platoons; Adaptive control; Cooperative driving; Traffic mobility	No restrictions.		
19. Security Classif. (of this report)	**20. Security Classif. (of this page)**	**21. No. of Pages**	**22. Price**
Unclassified	Unclassified	34	

Form DOT F 1700.7 (8-72) Reproduction of completed page authorized

UMTRI | The University of Akron | Central State University | Illinois Center for Transportation | University of Minnesota | Northwestern | Purdue University | Washtenaw Community College | Wisconsin University of Wisconsin–Madison

Table of Contents

List of Figures

Project abstract

This project developed and evaluated an online adaptive platoon control framework for connected and automated vehicles (CAVs) to improve mobility and safety in dynamic traffic environments. The proposed Physics Enhanced Residual Learning (PERL) framework integrates a physics-based centralized controller, which models vehicle dynamics to ensure stability and interpretability, with a neural network residual module that adaptively corrects unmodeled dynamics and disturbances in real time. Our work can contribute to the U.S. DOT CARMA Platform by serving as a tactical-level cooperative longitudinal control component within the platooning plugin. In this role, PERL has the potential to enhance CARMA's mobility operations layer by enabling real-time gap regulation, adaptive disturbance mitigation, and improved string stability under heterogeneous and mixed traffic scenarios. By providing a flexible yet interpretable control approach, PERL could support CARMA's cooperative driving automation objectives and facilitate smoother integration of platooning strategies into broader transportation system management and operations. High-fidelity simulations and scaled platform experiments demonstrated that PERL outperforms both pure physics-based and pure learning-based controllers, reducing cumulative position and speed errors by more than 50% in simulation and up to 99% in scaled platform tests. These results suggest that the proposed framework could strengthen CARMA's cooperative driving capabilities, improve the resilience of platoon control, and support the deployment of safe, efficient, and adaptive CAV platooning in real-world operations.

1. Introduction

Connected and automated vehicles (CAVs) have the potential to improve roadway throughput, reduce congestion, and enhance traffic safety by enabling close, coordinated vehicle following. Among the various cooperative driving applications, vehicle platooning has emerged as a promising strategy that leverages vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communications to achieve string stability and maintain short headways without compromising safety. Despite its advantages, the practical deployment of CAV platooning faces significant challenges. Real-world traffic environments are dynamic and uncertain, characterized by fluctuating traffic densities, heterogeneous vehicle capabilities, environmental disturbances, and communication imperfections. These factors can degrade platoon performance, creating a persistent trade-off between safety and mobility.

Traditional platoon control methods can be broadly divided into three categories: physical model-based (e.g., rule-driven, optimization), learning-based (e.g., reinforcement learning), and hybrid approaches combining learning with physical models. Physical model-based control strategies for CAV platoons predominantly rely on classical control theories, leveraging well-established physical rules of vehicle dynamics and behavior (*1*). These methods that explicitly model dynamics and ensure interpretable, stable control. For instance, linear control theories, including proportional-integral-derivative (PID) control (*2*) and linear quadratic regulators (LQR) (*3*), provide simple and computationally efficient solutions but are limited in handling nonlinearities. Nonlinear control methods, like sliding mode control (SMC) (*4*) offer robustness against model uncertainties and external disturbances. While model predictive control (MPC) (*1*), optimize control inputs over prediction horizons and can incorporate system constraints; extensions such as distributed MPC (*5*) and hybrid MPC for intersections (*6*) enhance scalability and coordination. Robust control techniques, such as tube-based MPC (*7*), are designed to maintain stability and performance under bounded disturbances and modeling uncertainties, ensuring string stability across heterogeneous platoons. The proposed robust control strategy significantly enhances disturbance rejection performance and substantially reduces the online computation burden(*8*). Although these methods are grounded in strong theoretical foundations and maintain interpretability, they tend to oversimplify real-world nonlinearities, particularly in heterogeneous platoon scenarios. This can lead to significant control errors, causing the system's output to deviate notably from ideal control and prediction scenarios (*9*, *10*).

Learning-based control methods enable online adaptations to dynamic environments through data-driven approaches (*11*, *12*). This group includes data-driven adaptive and policy learning approaches using machine learning and reinforcement learning, enabling to capture characteristic nonlinear relationships and to evolve dynamics in urban traffic (*13*, *14*). These methods excel in environments where the vehicle dynamics and interactions are too complex for traditional models to handle accurately. For instance, deep reinforcement learning (DRL) and neural networks (NN) have been utilized to optimize platoon strategies in real-time, adapting to changes in traffic flow and signal timings without relying on predefined models (*15*, *16*). Shi et al. (*17*) proposed a cooperative control strategy for CAVs in mixed traffic environments, utilizing DRL to

segment the platoon into multiple subsystems for centralized management. Li et al. developed an adaptive platoon-based intersection control model, named INTEL-PLT, which utilizes learning-based control to optimize multiple dynamic objectives (*18*). Multi-agent reinforcement learning (MARL) further expands these capabilities, facilitating distributed decision-making within platoons. Representative works by Busoniu et al. (*19*), Xu et al. (*20*) have shown that MARL can effectively capture inter-vehicle dependencies, leading to scalable and flexible traffic coordination. Multi-agent systems extend these capabilities by enabling decentralized decision-making in CAV platoons, using multi-agent NNs to support scalable, flexible traffic management (*21*, *22*). Despite their adaptability, learning-based controllers often suffer from low interpretability and transparency, especially in safety-critical environments where decision traceability is crucial. Moreover, their dependency on large, high-quality datasets can hinder performance in rare-event scenarios and environments with limited real-world training data.

To address the limitations of pure physics-based or purely learning-based approaches, recent studies have explored Physics-Informed Machine Learning (PIML) approaches that integrate physical principles into data-driven models (*23*, *24*). For instance, Physics-Informed Neural Networks (PINNs) encode physical laws, such as partial differential equations (PDEs), as soft constraints within neural network training, enabling solutions that inherently respect dynamical principles (*25*, *26*). While PINNs have shown success in modeling dynamical systems, their reliance on end-to-end offline training significantly limits real-time adaptability, especially in response to disturbances such as abrupt braking or sensor degradation. This integration enhances the learning efficiency and generalizability of the models under limited data conditions typical in novel traffic scenarios. For instance, we aim to leverage a physical model as the foundation to achieve approximately 90% of the desired control performance from scratch, and then employ neural networks and other learning-based methods to capture and refine the remaining 10% induced by dynamic factors. These hybrid methods aim to achieve high performance by leveraging physical priors while compensating for unmodeled dynamics through machine learning (*27*). Despite recent advances, existing hybrid control methods face key limitations in dynamic, safety-critical applications such as CAV platoons. Most PIML approaches rely on offline retraining when encountering novel disturbances, limiting real-time adaptability. Their tight integration of physics and learning also reduces interpretability, complicating fault diagnosis and certification (*17*). Moreover, frameworks like PINNs often model full dynamics from scratch, requiring large-scale data and overlooking the value of existing physical models.

Fig. 1 compares the PERL model with three existing models: the fixed physics model, the NN pure learning model, and the PINN model. The fixed physics model, reliant on offline control, incurs increased positional errors. The NN model improves control through data but lacks interpretability and is heavily data-dependent. The PINN model, merging physical and data-driven features, offers adaptability but underperforms with limited data. Furthermore, PINNs are typically trained offline and are not designed for real-time adaptation. Our proposed PERL framework innovatively combines physical models with data-driven NN methods. It utilizes prior knowledge from the physical model and employs residual learning to correct physical model predictions, enhancing control precision in scenarios with platoon dynamics disturbances. Unlike PINNs,

which treat physical constraints as training regularizes, PERL maintains the physical model as a fixed, reliable base and learns residual errors separately. This design enables both interpretability and online adaptability, allowing PERL to achieve high-precision control under limited data and dynamic disturbances.

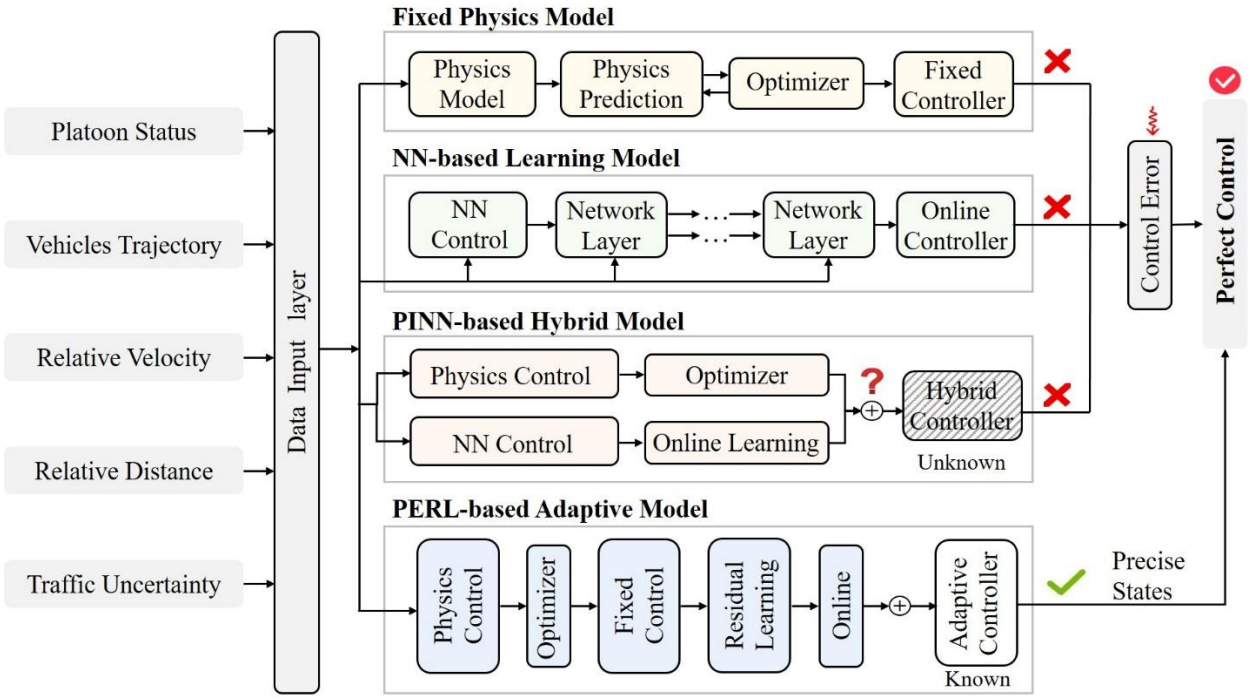


Fig. 1 Comparison of different model structures.

Fig. 1 compares four control model structures for connected and automated vehicle platoon control. (a) Fixed physics model: physics model, physics prediction, and optimizer produce a fixed controller, with high control error. (b) Neural network (NN)-based learning model: NN control with multiple network layers and an online controller, with high control error. (c) Physics-informed neural network (PINN)-based hybrid model: physics control and NN control with optimizer and online learning form a hybrid controller, adaptability unknown, with notable control error. (d) Physics Enhanced Residual Learning (PERL)-based adaptive model: physics control, optimizer, residual learning, and adaptive controller achieve precise states with minimal control error.

Inputs for all models include platoon status, vehicle trajectory, relative velocity, relative distance, and traffic uncertainty.This project addresses these challenges by developing an online adaptive platoon control framework. The proposed Physics Enhanced Residual Learning (PERL) framework embeds a physics-based centralized platoon controller within a neural network residual learning module that continuously corrects

unmodeled dynamics in real time. This hybrid approach maintains the stability and interpretability of physics-based control while leveraging the adaptability of data-driven methods to respond to unpredictable operating conditions. By enabling rapid adaptation to disturbances and maintaining safe inter-vehicle gaps, the PERL framework aims to reduce the traditional safety–mobility trade-off, supporting Transportation System Management and Operations (TSMO) goals and paving the way for large-scale, cooperative CAV deployment. Our work can contribute to the U.S. DOT CARMA Platform by serving as a tactical-level cooperative longitudinal control component within the platooning plugin, enhancing its mobility operations layer through adaptive gap regulation and disturbance mitigation. In future implementations, the PERL framework could be integrated into CARMA's tactical control modules to strengthen cooperative driving automation capabilities in both simulated and real-world environments.

2. Framework

As shown in **Fig. 2**, the proposed vehicle platoon control system consists of three main modules: vehicle platoon disturbance, inherent physical model, and residual learning model. The controller's input consists of uncertain vehicle platoon states with multiple disturbances, while the output includes control actions with residual compensation for all vehicles. (1) Vehicle platoon disturbance module: this module accounts for disturbances such as non-linearity, dynamic coupling, vehicle interactions, and external friction. These disturbances are captured and utilized in the control model to understand and adjust for deviations in platoon behavior. (2) Inherent physical model module: This module generates the reference trajectory based on vehicle dynamics, reference states, and control inputs. As the controller's core, it ensures path adherence by modeling motion from physical parameters and dynamics. (3) Residual learning model module: this module compensates for residual errors not addressed by the physical model. A NN is trained with a small amount of dynamic data to learn and adjust control signals in real time, compensating for discrepancies between the ideal and actual vehicle behaviors, thereby enhancing accuracy and system robustness. Through this adaptive learning process, the controller allows for appropriate driving speeds and real-time control adjustments, minimizing deviations from the reference trajectory while maximizing stability. Together, these modules enable the controller to effectively handle dynamic changes and external disturbances, ensuring precise and stable control for all vehicles in the platoon.

Specifically, PERL fundamentally differs from PINN in principle, as illustrated in **Fig. 1** (b) and (c). The physical model is not merely a guide for the learning process but a core component of the PERL output. By incorporating prior knowledge, the residual learning model focuses on predicting the residuals as corrections. PERL learns residuals to correct the physical model, offering key advantages: (1) lower bias risk due to the physical model; (2) efficient learning by reducing dimensions and concentrating on residuals; (3) lower data requirements; (4) improved interpretability and transparency, vital for autonomous systems. These benefits improve platoon control under uncertainty.

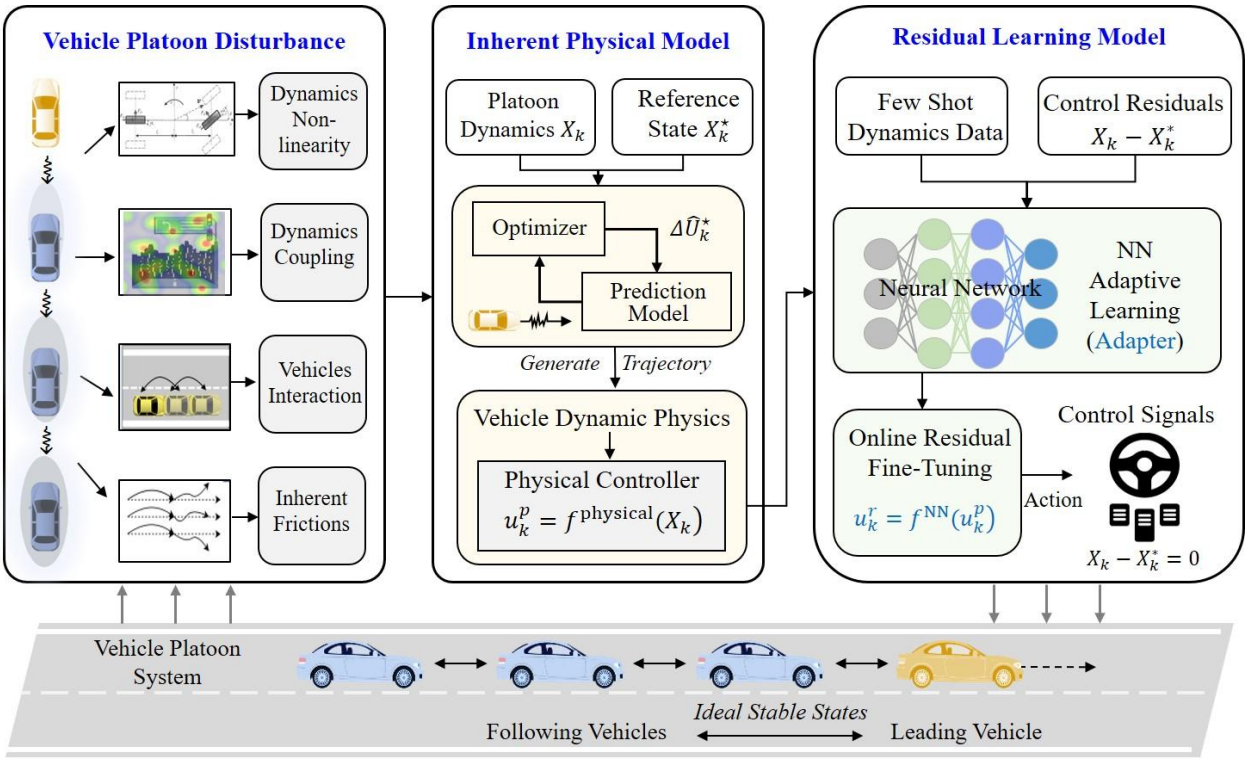


Fig. 2 The integrated CAVs platoon control framework.

Fig. 2 presents the integrated connected and automated vehicle (CAV) platoon control framework. It includes three modules: (1) Vehicle platoon disturbance, covers dynamics non-linearity, dynamics coupling, vehicle interaction, and inherent frictions; (2) Inherent physical model, uses platoon dynamics, reference state, optimizer, prediction model, and physical controller to generate control commands; (3) Residual learning model, uses few-shot dynamics data and neural network adaptive learning to produce control residuals, with online residual fine-tuning for improved accuracy. The framework regulates following and leading vehicles to maintain ideal stable states.

3. Adaptive platoon control

Connected and automated vehicles (CAVs) have the potential to improve roadway throughput, reduce congestion, and enhance traffic safety by enabling close, coordinated vehicle following. Among the various cooperative driving applications, vehicle platooning has emerged as a promising strategy that leverages vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communications to achieve string stability and maintain short headways without compromising safety. Despite its advantages, the practical deployment of CAV platooning faces significant challenges. Real-world traffic environments are dynamic and uncertain, characterized by fluctuating traffic densities, heterogeneous vehicle capabilities, environmental disturbances, and communication imperfections. These factors can degrade platoon performance, creating a persistent

trade-off between safety and mobility.

The PERL controller framework streamlines into two core components: (1) The foundational physical-based controller that accounts for platoon dynamics in centralized control. (2) A supplementary learning module that embeds a residual feedback mechanism into the physical model, employing NN for adaptive online correction of model inaccuracies and disturbances. This dual system facilitates precise speed regulation and dynamic control responses, ensuring vehicles adhere closely to their trajectories for optimal platoon stability.

3.1 Vehicle platoon system dynamics

The platoon, consisting of $I+1$ homogeneous vehicles including a leader and I followers, is a crucial component of the control system. The main task of the platoon control is to use V2V communication to form a flexible mixed platoon from individual vehicles on the road, achieving vehicle coordination through mutual cooperation. According to the hierarchical structure, platoon control is divided into centralized control and distributed control. Centralized control relies on a central controller to collect and process data from all vehicles, generating control commands to coordinate the movement of each vehicle. This approach can optimize and coordinate the entire system. In this study, we employ the centralized control for real vehicle platoon system.

According to the vehicle longitudinal dynamics, the linear model for a single vehicle $i \in \mathcal{I} = \{1,2,\dots,I\}$ at time step $k \in \mathbb{Z}$ with constant sampling interval Δt:

$$\begin{cases} p_{k+1}^i = p_k^i + v_k^i \Delta t + \frac{1}{2} a_k^i \Delta t^2 \\ v_{k+1}^i = v_k^i + a_k^i \Delta t \\ a_{k+1}^i = -\frac{\Delta t}{\tau^i} v_k^i + \frac{\Delta t}{\tau^i} s_k^i = -\frac{\Delta t}{\tau^i} v_k^i + \frac{\alpha^i \Delta t}{\tau^i} u_k^i + \frac{\beta^i \Delta t}{\tau^i} \end{cases}$$

where p_k^i denotes the position, v_k^i denotes the speed, a_k^i denotes the acceleration, τ denotes the control cycle time interval, α denotes the proportional constant relating control input (e.g., PWM duty cycle or motor torque command) to linear acceleration in the physical model, β denotes the static resistance offset.

Then, we can get the state space model for vehicle i at time k:

$$x_{k+1}^i = A^i x_k^i + B^i u_k^i + C^i$$

where $x_k^i = [p_k^i, v_k^i, a_k^i]^\top$ is the state information for vehicle i,

$$A^i = \begin{bmatrix} 1 & \Delta t & \frac{\Delta t^2}{2} \\ 0 & 1 & \Delta t \\ 0 & -\frac{\Delta t}{\tau^i} & 0 \end{bmatrix}, B^i = \begin{bmatrix} 0 \\ 0 \\ \frac{\alpha^i \Delta t}{\tau^i} \end{bmatrix}, C^i = \begin{bmatrix} 0 \\ 0 \\ \frac{\beta^i \Delta t}{\tau^i} \end{bmatrix}$$

This study assume all vehicles in the platoon are homogeneous ($A^i = A, B^i = B, \tau^i = \tau$). We then define:

$$X_k = \left[p_k^1, \dots, p_k^I, v_k^1, \dots, v_k^I, a_k^1, \dots, a_k^I\right]^\top$$

$$U_k = \left[u_k^1, \dots, u_k^I\right]^\top$$

such that the platoon dynamics are:

$$X_{k+1} = A_I X_k + B_I U_k$$

where $A_I = A \otimes E_I, B_I = B \otimes E_I, \otimes$ is the Kronecker operator, which is used to extend the single-vehicle dynamics A and control matrix B to the entire platoon by replicating them across all I vehicles. E_I is the I dimensional elementary matrix.

Define the change in control actions ΔU_k from the previous control action U_{k-1}:

$$\Delta U_k = U_k - U_{k-1} = \left[\Delta u_k^1, \dots, \Delta u_k^I\right]^\top$$

where Δu_k^i is the change in control action for vehicle i.

To predict the platoon's state for the next N steps, this study introduces the predicted state value of the platoon at time $k+n$ for $n \in \mathcal{N} = \{1, \dots, N\}$ from the measured state value at time k, denoted as $\hat{X}_{k+n|k}$, with the prediction window defined as:

$$\mathcal{X}_k = \left[\hat{X}_{k+1|k}^\top, \dots, \hat{X}_{k+N|k}^\top\right]^\top$$

For the predicted value of the platoon controller as:

$$\Delta\hat{U}_k = \left[\Delta\hat{U}_{k|k}^\top, \dots, \Delta\hat{U}_{k+N-1|k}^\top\right]^\top$$

where from the measurement at time k the predicted applied control at time $k+n$ is:

$$\hat{U}_{k+n|k} = \hat{U}_{k+n-1|k} + \Delta\hat{U}_{k+n|k}$$

$$\Delta\hat{U}_{k+n|k} = \left[\Delta\hat{u}_{k+n|k}^1, \dots, \Delta\hat{u}_{k+n|k}^I\right]^\top$$

The state prediction of the platoon $\mathcal{X}_k$ can be written as a linear combination of the current state X_k, the previously applied control U_{k-1} and the predicted change in control $\Delta\hat{U}_k$:

$$\mathcal{X}_k = \Phi X_k + \lambda U_{k-1} + \Gamma\Delta\hat{U}_k$$

Where

$$\Phi = \begin{bmatrix} A_I \\ \vdots \\ A_I^N \end{bmatrix}, \lambda = \begin{bmatrix} A_I^0 B_I \\ \vdots \\ (A_I^{N-1} + \cdots + A_I^0) B_I \end{bmatrix}$$

$$\Gamma = \begin{bmatrix} B_I & \cdots & 0 \\ \vdots & \ddots & \vdots \\ (A_I^{N-1} + \cdots + A_I^0) B_I & \cdots & B_I \end{bmatrix}$$

where Φ is constructed via a Kronecker product to capture the temporal evolution of the platoon state over the prediction horizon N, λ accounts for the influence of previously applied control inputs U_{k-1}, Γ represents the mapping from predicted control input changes $\Delta \widehat{U}_k$ to the future state vector.

This study defines the reference state of each vehicle $i \in \mathcal{I}$ at time step k as $p_k^{i*}, v_k^{i*}, a_k^{i*}$, the joint reference state for the platoon is denoted as:

$$X_k^* = \left[p_k^{1*}, \ldots, p_k^{I*}, v_k^{1*}, \ldots, v_k^{I*}, a_k^{1*}, \ldots, a_k^{I*}\right]^\top$$

$$\mathcal{X}_k^* = [(X_{k+1}^*)^\top, \ldots, (X_{k+N}^*)^\top]^\top$$

where $*$ denotes the desired or nominal value obtained from the first-step prediction of the physical model, considering vehicle dynamics and inter-vehicle constraints. At each control step, the model solves an optimization problem to generate a finite-horizon trajectory, from which the first predicted state is extracted as the reference for residual learning.

Consider for each vehicle $i \in \mathcal{I}$, the absolute position, velocity, and acceleration errors between the actual state and the reference state is computed as:

$$\begin{cases} \tilde{p}_k^i = p_k^i - p_k^{i*} \\ \tilde{v}_k^i = v_k^i - v_k^{i*} \\ \tilde{a}_k^i = a_k^i - a_k^{i*} \end{cases}$$

For the entire platoon, these errors can be written as:

$$X_k - X_k^* = \left[\tilde{p}_k^1, \ldots, \tilde{p}_k^I, \tilde{v}_k^1, \ldots, \tilde{v}_k^I, \tilde{a}_k^1, \ldots, \tilde{a}_k^I\right]^\top$$

Then denote $\hat{p}_{k+n|k}^i, \hat{v}_{k+n|k}^i, \hat{a}_{k+n|k}^i$ as the prediction error where the subscript indicates the state prediction at time $k+n$ given the state at time k.

3.2 Optimized MPC for platoon control

Centralized control requires a central controller to optimize vehicle control at each timestep, facing challenges from intertwined vehicle states and controls, and the need to forecast complex, long-term traffic dynamics due to dimensionality and disturbances. MPC, a closed-loop method, predicts system behavior for a defined

future period and optimizes control laws within constraints, applying only the first-step control. Repeated each timestep, MPC effectively adjusts to control errors and uncertainties, making it well-suited for CAV Platoon control. The formulation of the MPC controller, featuring a finite prediction horizon of N steps, is as follows:

$$J(k,N) = \min \sum_{n=0}^{N-1} \left[\sum_{i=1}^{I} q_1 \left(\hat{p}^i_{k+n|k}\right)^2 + q_2 \left(\hat{v}^i_{k+n|k}\right)^2 + q_3 \left(\hat{a}^i_{k+n|k}\right)^2 + q_4 \left(\Delta \hat{u}^i_{k+n|k}\right)^2 \right]$$

$$s.t.:$$

$$d_{\min} \le p^{i-1} - p^i \le d_{\max}, \qquad \forall i \in \mathcal{I}$$

$$v_{\min} \le v^i \le v_{\max}, \qquad \forall i \in \mathcal{I}$$

$$a_{\min} \le a^i \le a_{\max}, \qquad \forall i \in \mathcal{I}$$

where q_1, q_2, q_3, q_4 correspond to errors in position, velocity, acceleration, and control inputs. $d_{\max}, d_{\min}, a_{\max}, a_{\min}, v_{\max}, v_{\min}$ are the maximum and minimum limits for space, acceleration, and velocity. Constraint (20) ensures platoon safety and distance limits; constraint (21) enforces road speed limits; constraint (22) sets acceleration limits based on vehicle engine and braking capacities.

The problem can be written in the form of a quadratic program:

$$J\left(X_k, \Delta \hat{U}_k\right) = \Delta \hat{U}_k^\top (\Psi + \Gamma^\top \Omega \Gamma) \Delta \hat{U}_k + 2(\Phi X_k + \lambda U_{k-1} - \mathcal{X}_k^\star)^\top \Omega \Gamma \Delta \hat{U}_k$$

$$s.t.:$$

$$\bar{G} \Gamma \Delta \hat{U}_k \le -\bar{G}(\Phi X_k + \lambda U_{k-1}) - \bar{g}$$

where $\Omega = \text{diag}\{Q, \dots, Q, 0\}, \Psi = \text{diag}\{R_\Delta, \dots, R_\Delta\}$ are block diagonal matrices,

$$Q = \begin{bmatrix} q_1 E_I & 0 & 0 \\ 0 & q_2 E_I & 0 \\ 0 & 0 & q_3 E_I \end{bmatrix}, R_\Delta = q_4 E_I$$

$$\bar{G} = \text{diag}\left[\check{G}, \dots, \check{G}\right], \bar{g}^\top = [g^\top, \dots, g^\top],$$

$$\check{G} = \begin{bmatrix} \mathfrak{T}_I & 0 & 0 \\ -\mathfrak{T}_I & 0 & 0 \\ 0 & -E_I & 0 \\ 0 & E_I & 0 \\ 0 & 0 & -E_I \\ 0 & 0 & E_I \end{bmatrix}, g = \begin{bmatrix} 1_{I-1} d_{\min} \\ -1_{I-1} d_{\max} \\ 1_I v_{\min} \\ -1_I v_{\max} \\ 1_I a_{\min} \\ -1_I a_{\max} \end{bmatrix},$$

where $\mathfrak{T}_I$ is a size $(I-1) * I$ Toeplitz matrix with -1 on the diagonal and 1 on the first upper diagonal.

Vectors 1_{I-1} and 1_I are columns of ones of size $(I-1)$ and I, respectively. Advanced solvers can rapidly solve this quadratic optimization problem. The optimal platoon control action minimizes the constrained finite horizon cost function.

$$\Delta\widehat{U}_k^{\star} = \underset{\Delta\widehat{U}_k}{\operatorname{argmin}}\, J\left(X_k, \Delta\widehat{U}_k\right)$$

where the first item $\Delta\widehat{U}_{k|k}^{\star}$ will be the output control u_k^p of the physical model.

3.3 The inherent vehicle physical model

For vehicle platoon system, to effectively model the nonlinear characteristics of platoon dynamics within processing constraints and without incurring substantial computational expenses, this study utilizes a model based on physical principles that captures the essence of these dynamics efficiently. The control command, expressed in terms of revolutions per minute (RPM), ensures the smooth and safe operation of the entire platoon by directly adjusting the duty cycle of the pulse width modulation (PWM), which ranges from 0% to 100%. This PWM duty cycle, in turn, linearly dictates the motor voltage, thereby determining the motor speed and allowing for precise regulation of the robotic system's movement. As shown in Fig. 3, under ideal conditions, the motor voltage and RPM should exhibit a linear relationship.

Therefore, the input to the physical model consists of the initial vehicle states and the predicted trajectory over a finite horizon, as illustrated in Fig. 4b.

$$S^{\text{Phy}} = [p_{i0}, v_{i0}, a_{i0}]_{\forall i\in\mathcal{I}}, \qquad [\hat{p}_{ik}, \hat{v}_{ik}, \hat{a}_{ik}]_{\forall k\in\mathcal{K}, i\in\mathcal{I}}$$

The output is a set of control commands:

$$f^{\text{Phy}}: \left(S^{\text{Phy}} \middle| \theta^{\text{Phy}}\right) \rightarrow [u_i]_{\forall i\in\mathcal{I}} = \text{linear}\left(\text{MPC}\left(S^{\text{Phy}}\right)\right)$$

where $S^{\text{Phy}}$is the model inputs for the physics model; u_i is the control command of CAV $i \in \mathcal{I}$, i.e., the revolutions per minute (RPM), f^{Phy} is the physics model; θ^{Phy} is the model parameters for the physics model.

However, experimental observations reveal systematic deviations due to nonlinearities introduced by motor friction, drivetrain losses, internal resistance, and battery voltage drops. These unmodeled physical effects lead to residual errors between expected and actual system responses. an unmodeled residual capturing low-speed/high-speed distortions, including friction, back EMF, and battery variation. As illustrated in Fig. 4b, given the difficulty of modeling residual errors analytically, this study employs a data-driven residual learner (see Section 3.4) to capture and correct this term in real time. This hybrid strategy ensures that the physical model remains interpretable and analytically valid, while the PERL framework compensates for its systematic bias under real-world conditions.

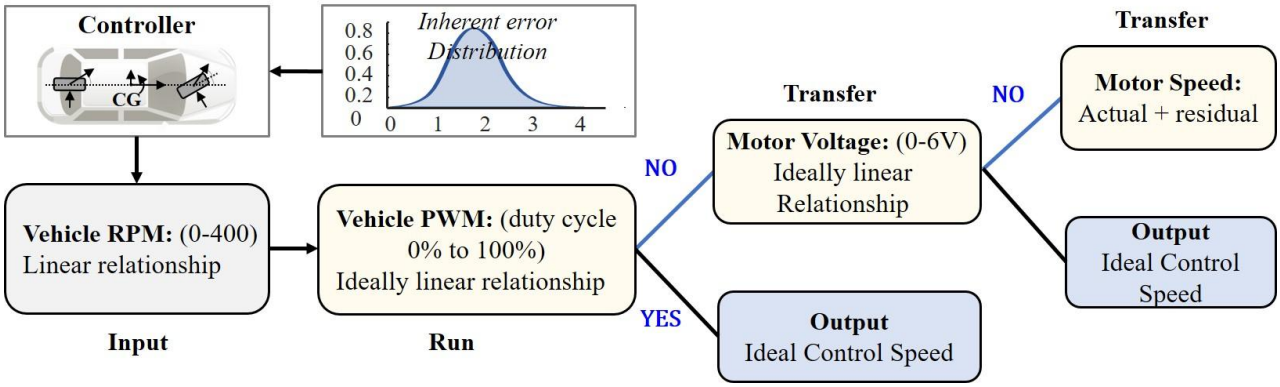


Fig. 3 The control command for vehicle platoon system.

Fig. 3 describes the control command process for a vehicle platoon system. Input: controller outputs vehicle RPM (0–400) with a linear relationship. Run: vehicle PWM duty cycle (0%–100%) ideally has a linear relationship with RPM. Transfer: motor voltage (0–6V) ideally linear to control speed; if not, output is ideal control speed. If voltage is linear but motor speed deviates, actual speed includes residual errors. Final output is ideal control speed when no residuals are present.

3.4 Online adaptation based on residual learning

While physics-based controllers provide reliable and interpretable nominal control, their simplified dynamics and offline parameterization often fail to capture real-world nonlinearities such as actuator saturation, voltage drops, and frictional effects. As illustrated in Fig. 4b, to address this, the PERL framework incorporates a residual learning module that continuously refines the output of the physical controller by learning and compensating for systematic deviations observed during execution. Specifically, residual learning refines control outputs by addressing the "gap" between predicted and actual system states. Simplified motion models often introduce residual dynamic errors between actual outputs and intended commands. To address this, the residual module employs a three-layer fully connected multilayer perceptron (MLP) with 64, 32, and 16 units, respectively, and ReLU activations. It is trained online using the Adam optimizer and outputs discrete adjustments to the control signal. It maps a set of observed and predicted variables to a control correction term $\bar{u}_i$ for each vehicle. The input to the residual model is:

$$S^{\mathrm{RL}} = [p_{i0}, v_{i0}, a_{i0}]_{\forall i \in \mathcal{I}},\ [\hat{p}_{ik}, \hat{v}_{ik}, \hat{a}_{ik}]_{\forall k \in \mathcal{K}, i \in \mathcal{I}},\ \ [e_i, r_i, b_i]_{\forall i \in \mathcal{I}}$$

The output is a set of control commands:

$$f^{\mathrm{RL}}: \left(S^{\mathrm{RL}} \middle| \theta^{\mathrm{RL}}\right) \rightarrow [\bar{u}_i]_{\forall i \in \mathcal{I}} = \mathrm{MLP}\left(\mathrm{MPC}\left(S^{\mathrm{RL}}\right), S^{\mathrm{RL}}\right)$$

where S^{RL} is the model inputs for the residual learning; $\bar{u}_i$ is the residual of the RPM of CAV $i \in \mathcal{I}$; f^{RL} is

the residual learning model; θ^{RL} is model parameters for the residual learning.

Therefore, the physical controller outputs a baseline control u_i, and the NN predicts a residual term $\bar{u}_i$, which is then combined to produce the final control command for PERL:

$$f^{\mathrm{PERL}}\left(S^{\mathrm{PERL}}\middle|\theta^{\mathrm{PERL}}\right) = f^{\mathrm{Phy}}:\left(S^{\mathrm{Phy}}\middle|\theta^{\mathrm{Phy}}\right) + f^{\mathrm{RL}}:\left(S^{\mathrm{RL}}\middle|\theta^{\mathrm{RL}}\right) \rightarrow [u_i + \bar{u}_i]_{\forall i \in \mathcal{I}}$$

where f^{PERL} is the PERL model; S^{PERL} is the model inputs for the PERL; θ^{PERL} is the model parameters for the PERL.

To ensure stable deployment across various platforms, the residual policy is trained over a discretized action space within the allowable acceleration range. The NN-based residual learner is decoupled from the physical controller, allowing independent adaptation without retraining the entire model pipeline. This decoupled architecture differs fundamentally from monolithic models like Physics-Informed Neural Networks (PINNs), where physical constraints are embedded directly in the loss function and training is typically offline. In contrast, PERL treats the physics-based model as a stable, fixed prior and applies residual learning only to compensate for unmodeled effects.

Illustration of control deviation and residual compensation in PERL. (a) Control errors from actuator imperfections and external disturbances cause actual trajectories (red) to deviate from planned ones (black dashed), undermining platoon performance. (b) The PERL framework mitigates this by combining a physical controller with a neural residual module: the former generates nominal control u_i, while the latter refines it using features like turning rate r_i, battery state b_i, and past trajectory errors to generate a residual correction $\bar{u}_i$. (c) The final control (green) improves tracking accuracy and platoon stability.

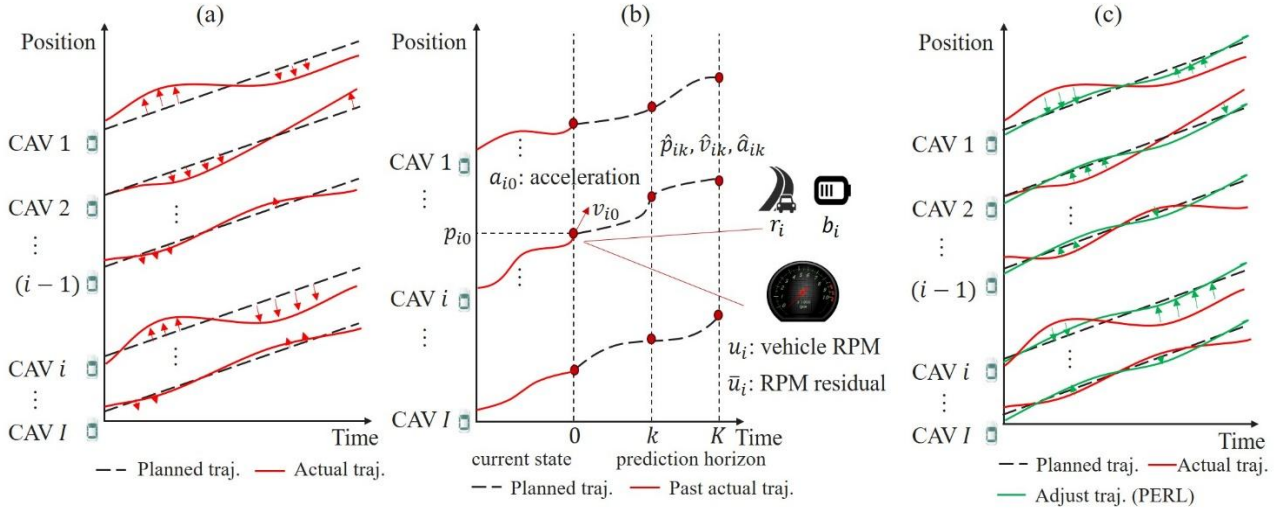


Fig. 4 Illustration of control deviation and residual compensation in PERL.

Fig. 4 is the illustration of control deviation and residual compensation in the PERL framework. (a) Control errors from actuator imperfections and external disturbances cause actual trajectories (red) to deviate from planned ones (black dashed), undermining platoon performance. (b) The PERL framework mitigates this by combining a physical controller with a neural residual module: the former generates nominal control u_i, while the latter refines it using features like turning rate r_i, battery state b_i, and past trajectory errors to generate a residual correction $\bar{u}_i$. (c) Adjusted trajectories (green) closely match planned trajectories, improving tracking accuracy and platoon stability.

Fig. 4 illustrates the motivation and core idea behind the proposed PERL framework. As shown in subfigure (a), conventional model-based controllers, generate planned trajectories based on simplified vehicle dynamics. However, real-world actuator imperfections (e.g., motor nonlinearity, delay) and external disturbances (e.g., wind, road friction) cause actual trajectories to diverge from the plan, leading to safety risks and reduced control performance. To mitigate this, PERL introduces high-level physical controller and a residual learning module, as shown in (b). The physical model outputs a nominal control command u_i, while the residual learner observes tracking deviation and platform features such as turning rate r_i, battery state b_i, and past trajectory errors to generate a residual correction $\bar{u}_i$. As a result, the adjusted control leads to trajectories (green curves in (c)) that closely follow the intended path, thereby reducing accumulated error, improving control fidelity, and enabling tighter platoon spacing. This integration enables real-time adaptability without sacrificing interpretability.

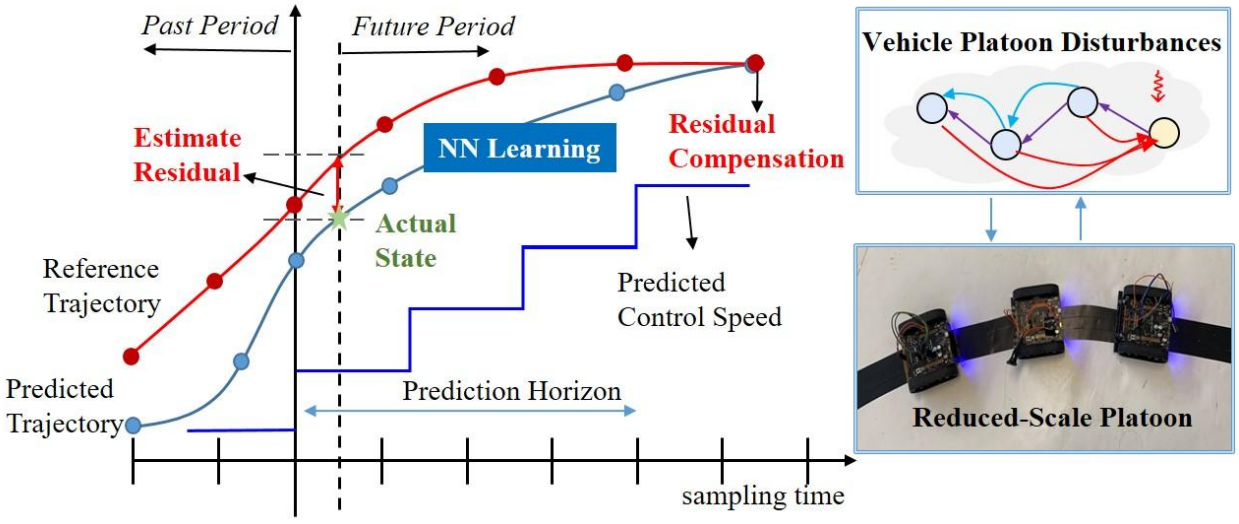


Fig. 5 The dynamic residuals between reference and predicted trajertory control in CAV platoon.

Fig. 5 is the dynamic residuals between reference and predicted trajectory control in a CAV platoon. The reference trajectory (red) and predicted trajectory (blue) differ due to vehicle platoon disturbances. Residuals are estimated in the past period, corrected through neural network learning, and compensated in the future period to match the actual state. Predicted control speed follows the corrected trajectory. Insets show conceptual vehicle platoon disturbances and a reduced-scale robot platoon on a circular track.

Fig. 5 illustrates the dynamic residuals between the reference and predicted trajectory control in a CAV platoon system. It outlines the key phases of trajectory prediction and control, divided into past and future periods. The red line represents the reference trajectory, while the blue line depicts the predicted trajectory. Based on the outputs of the baseline physical model, residuals inevitably arise, which are mitigated through residual compensation using neural network (NN) learning. The PERL framework achieves 90% of the control effect via the baseline model and further incorporates NN-driven residual learning and compensation to reduce the remaining 10% of residual errors and disturbance impacts.

4. Simulation experiment and results

In this section, we assess the performance of the proposed PERL controller model within a simulation environment by conducting a comparative analysis against two baseline models. The simulation platform utilized is Python 3.8.18. The quadratic programming problem is solved using the Sequential Least Squares Programming (SLSQP) method, implemented through the "scipy.optimize" package in Python.

4.1 Experiment setting and baselines

In this simulation, we set a platoon of 5 vehicles for 30 seconds, with a $\Delta t = 0.1$ seconds. To test the effectiveness of the PERL controller under different scenarios, we designed two types of test trajectories. (1) Real-world platooning trajectories, sourced from the OpenACC dataset (*28*). These trajectories are derived from actual AV-following scenarios on highways and provide empirical benchmarks that reflect realistic driving dynamics, including inter-vehicle spacing, acceleration patterns, and natural human-like perturbations. By requiring controllers to track these observed trajectories during testing, the evaluation process ensures a realistic and representative benchmark for assessing algorithmic performance. (2) Synthetic trajectories from car-following models. Specifically, the intelligent driver model (IDM) (*29*) is applied to obtain the longitudinal trajectories. The models are particularly versatile due to its capacity to adjust headway parameters, thus allowing for the simulation of various traffic densities and driving behaviors. The formulation of the IDM is shown as follows:

$$a_k = a_{max}\left[1 - \left(\frac{v_k}{v_0}\right)^{\delta} - \left(\frac{s^*(v_k, \Delta v_k)}{s_k}\right)^2\right]$$

where the definitions and values of the parameters are shown in **Table 1** and the desired minimum gap $s^*(v_k, \Delta v_k)$ is calculated as:

$$s^*(v_k, \Delta v_k) = s_0 + s_1\sqrt{\frac{v_k}{v_0}} + Tv_k + \frac{v_k \Delta v_k}{2\sqrt{ab}}$$

For the first type of trajectories, we randomly extracted one trajectory as Scenario 1. For the second type of trajectory, another one trajectory is generated as Scenario 2. By employing both types of trajectories, the study aims to cover a comprehensive range of driving conditions, enhancing the robustness and applicability of the developed algorithms. Diverse disturbance scenarios have been extensively tested in our previous work (*30*), so this study focuses on two representative reference trajectories to validate online adaptability.

The parameters in the MPC constraints are set as follows: $d_{\min} = 5\text{m}, d_{\max} = 80\text{m}, a_{\max} = 5\text{m/s}^2, a_{\min} = -5\text{m/s}^2, v_{\max} = 50\text{m/s}, v_{\min} = 5\text{m/s}$. To account for control output errors during RPM and velocity transfer, two error models were used: affine error$u_k^a = 1.1u_k - 3 + x\,, x\sim N(0,1)$,and quadratic error $u_k^q = 0.01u_k^2 + u_k - 3 + x\,, x\sim N(0,1)$. The online learning updated the NN every 20 time steps (2 seconds) using experimental data. To evaluate the effectiveness of the PERL method, the results are compared with the MPC with physical model and the PERL controller. These methods were tested across two scenarios and two error types, totaling four evaluations.

Table 1 Parameters of the IDM model used for generating reference trajectories.

Parameters	Typical value
Desired speed v_0	33.3 m/s
Safety time headway T	1.6s
Maximum acceleration a_{max}	0.73 m/s^2
Desired deceleration b	1.67 m/s^2
Acceleration exponent δ	4
Jam distance s_0	2 m
Jam distance s_1	0 m

4.2 Simulation results and analysis

Table 2 details the cumulative and maximum absolute velocity errors for the MPC with physical model and the online PERL method across four tests. This study uses four metrics to measure the performance of the two methods: the cumulative absolute error and the maximum absolute error for position and velocity, denoted as CAE_p, MAE_p, CAE_v, and MAE_v, respectively. The 'Gap' represents the difference in errors when compared to those obtained by the PERL method. The results demonstrate that the PERL method consistently outperforms the MPC with the physical model and the MPC with NN across all four tests. Specifically, the average cumulative absolute position errors for the PERL method are 58.5% lower compared to the MPC with the physical model and 58.4% lower compared to the MPC with NN. Furthermore, the average gap in cumulative absolute velocity errors between the PERL method and the other two methods is 40.1% and

47.7%, respectively, indicating that the PERL method significantly outperforms the alternatives. In terms of maximum absolute error, the error gaps for position between the PERL method and the other methods are substantial, at 53.3% and 57.7%, respectively. However, the gap in maximum absolute velocity error is relatively smaller, averaging 2.1% and 17.4%, respectively.

The variations in velocity error across all trajectories are depicted in **Table 2**, providing an explanation for the results. The results confirm that the online PERL method significantly outperforms the MPC with the physical model and NN. Additionally, **Table 2** demonstrates that, in all four tests, the errors for PERL eventually stabilize near zero. This trend is particularly evident in Scenario 2 with quadratic error, where PERL initially exhibits a larger absolute velocity error than the physics-based controller. This result, however, does not indicate a fundamental limitation of PERL. Rather, it reflects dynamic factors such as transient phase sensitivity and nonlinearity in velocity-position coupling, which can amplify momentary error magnitudes before convergence.

Table 2 Simulation results for the three control models in four tests.

Test	**Scenario 1 (affine error)**			**Scenario 1 (quadratic error)**		
Method	Physics	NN	PERL	Physics	NN	PERL
CAE_p	3689	2409	1884	5503	9194	2892
$GapCAE_p$	48.9	21.8	0.0	47.4	68.5	0.0
CAE_v	268.6	200.5	154.0	367.6	631.3	273.8
$GapCAE_v$	42.6	23.2	0.0	25.5	56.6	0.0
MAE_p	7.0	5.0	3.7	8.1	16.2	5.4
$GapMAE_p$	47.2	26.4	0.0	33.4	66.9	0.0
MAE_v	0.6	0.5	0.5	0.8	1.1	0.6
$GapMAE_v$	19.5	7.7	0.0	21.3	41.3	0.0
Test	**Scenario 2 (affine error)**			**Scenario 2 (quadratic error)**		
Method	Physics	NN	PERL	Physics	NN	PERL
CAE_p	12953	5089	1985	3752	10016	1769
$GapCAE_p$	84.7	61.0	0.0	52.9	82.3	0.0
CAE_v	838.5	345.1	155.1	310.6	629.0	277.0
$GapCAE_v$	81.5	55.1	0.0	10.8	56.0	0.0
MAE_p	21.4	9.0	3.7	6.9	16.3	3.4
$GapMAE_p$	82.8	58.8	0.0	49.8	78.9	0.0
MAE_v	1.1	1.3	0.6	0.9	1.3	1.7
$GapMAE_v$	48.3	56.2	0.0	-80.6	-35.6	0.0

In this work, we have already conducted multiple independent training runs and selected the result that offers the most balanced performance across all evaluation metrics. For the online PERL controller, the initial lack of sufficient training data results in a large negative error. However, as training progresses, the residual learning component effectively learns and corrects the error, reducing the velocity error below zero and compensating for the initial position error caused by the negative velocity error at the outset. This

phenomenon explains the maximum absolute error observed for the PERL method in **Table 2**. After training, the error stabilizes within the ±0.3 range, validating the effectiveness of online learning. In contrast, the MPC with NN approach, lacking the support of a physical model, continues to exhibit significant error even after training, indicating that achieving results comparable to those of PERL would require a larger volume of training data.

It is also important to note that these simulations are conducted under simplified conditions, which may not fully capture the complexities of real-world vehicle dynamics, thus necessitating further real-world testing to validate the PERL methods' performance.

5. Reduced-scale platform test

For the experiment, this study utilizes a reduced-scale robot car platform shown in **Fig. 6** to evaluate the proposed PERL. These robot cars provide full controllability and minimize safety risks, making them effective proxies for validating models before full-scale deployment. The platform includes Pololu Zumo 32U4 robots, infrared sensors for distance measurement, a PC (Intel i5, 8 GB RAM, macOS) for computation, Wi-Fi modules for communication, and a circular test track. This setup enables the extraction of velocity and acceleration to assess PERL's performance.

5.1 Reduced-scale platform

As shown in **Fig. 6**， in this reduced-scale platform, this study divides it into 4 parts, including: 1) Robots: The experimental setup featured a Pololu Zumo 32U4 robot, equipped with a 75:1 HP gear ratio and advanced line sensors for precise line tracking. Key components include an Arduino microcontroller for ease of programming, additional serial port for communication upgrades, and dual motors with encoders for accurate position and speed monitoring. 2) Distance/Position Measurement: An infrared sensor, GP2Y0A51SK0F, mounted on the robot provides rapid and accurate distance measurements by converting analog signals to digital. This facilitates effective collision avoidance controlled by a calibrated proportional-derivative (PD) controller. 3) Wireless Communication: Each robot uses an ESP8266 Wi-Fi chip for low-power, IP/TCP-based communication with the PC via a Wi-Fi router, enabling real-time exchange of speed data and trajectories. Packet loss is mitigated through retransmission with timeout handling, and robots are synchronized via periodic polling and timestamp alignment. Measured latency remains stable at 15–20 ms and is accounted for in controller tuning. 4) Test Track: The test track is a ring-shaped path with a 110 cm inner radius and 5 cm width, designed to optimize line tracking via sensor-calibrated reflectivity contrast between the track materials.

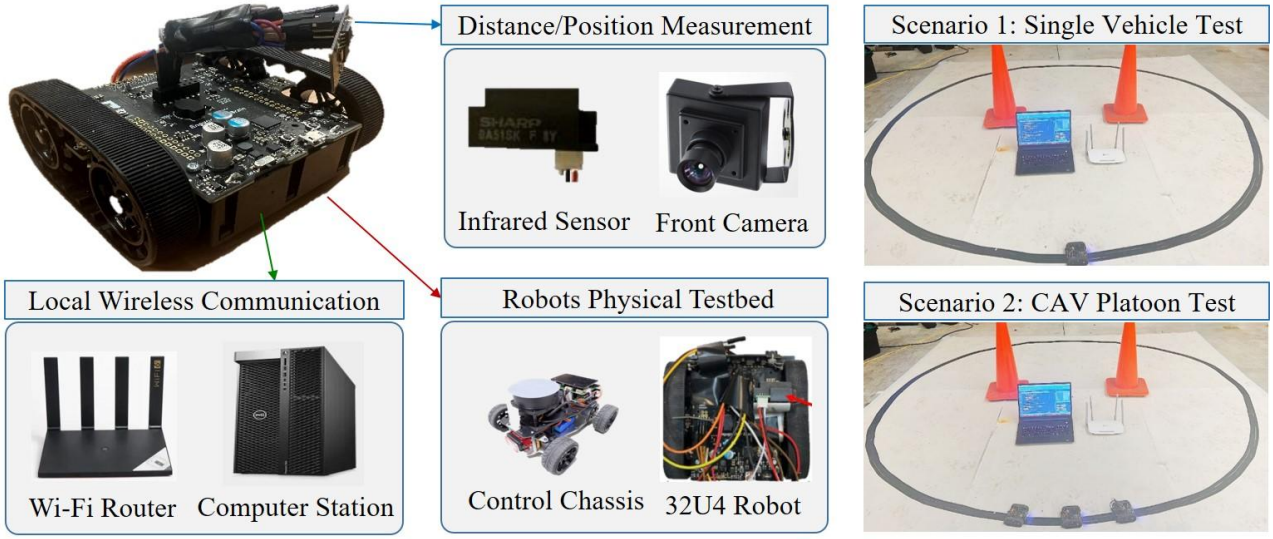


Fig. 6 Reduced-scale platform and the test track.

Fig. 6 introduces the reduced-scale platform and test track for CAV platoon experiments. Components include distance/position measurement using an infrared sensor and front camera, local wireless communication via Wi-Fi router and computer station, and robots physical testbed with control chassis and 32U4 robot. Scenario 1 shows a single vehicle test on a circular track with cones; Scenario 2 shows a CAV platoon test with multiple robots on the same track.

5.2 Experiment setting

Table 3 details the parameter settings for the online adaptation algorithm, where both physical model and online learning update parameters every 80 milliseconds. A count threshold (Cth) of 5 allows for parameter updates every 0.4 seconds, independent of road condition changes.

Table 3 Parameters of online adaptation.

Parameters	Optimal	Parameters	Optimal
Layer	64	Input dim	3
Epochs	100	Validation split	0.2
Verbose	1	Loss	MSE
Activation	relu	Optimizer	ADAM

To assess the control accuracy of various algorithms, this study deploys three Pololu Zumo robots, each equipped with consistent dynamic models, to form a multi-robot platoon that autonomously navigated a predetermined circular track (as shown in **Fig. 7**). During the experiment, the robots' states, such as position

and speed, were continuously recorded. **Fig. 7**(a) illustrates the platoon system, where the robots, under dynamic disturbances, followed the circular trajectory, testing the platoon's ability to maintain stable control and precise tracking. **Fig. 7**(b) shows the X-Y position trajectories collected from 10 independent runs of the circular platoon experiment. Each colored line represents the trajectory from one run, illustrating how the actual paths vary slightly from the planned trajectory due to model inaccuracies and external disturbances. The platoon's performance will be validated under the PERL and baseline models (physical, NN) through multiple trajectory control trials. **Fig. 7**(c) demonstrates the reduced-scale platform testing, which includes a series of evaluations designed to optimize the platoon's stability and trajectory control for the reduced-scale robot models, offering valuable insights into the scalability of the proposed control strategies for more complex systems.

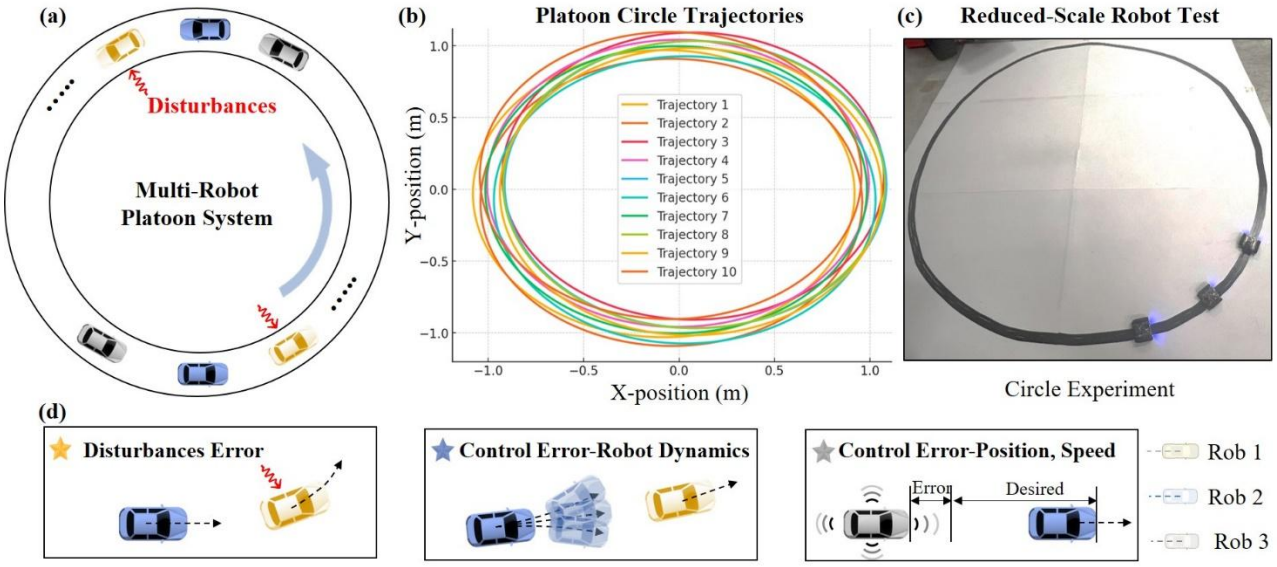


Fig. 7 Robot platoon circle trajectories for 10 laps with a target velocity.

Fig. 7 presents the robot platoon circle trajectories for 10 laps with a target velocity. (a) Schematic of a multi-robot platoon navigating a circular trajectory under external disturbances. (b) Circular trajectory plots for 10 control trials. Each colored line represents a full-lap trajectory from one trial, used to evaluate tracking consistency, inter-robot coordination, and control repeatability. (c) Real-world reduced-scale experiment setup with three Zumo robots. (d) Examples of disturbance error, control error from robot dynamics, and control error in position and speed for robots 1, 2, and 3.

The evaluations are divided into two stages to comprehensively assess the performance of the algorithms: 1) Preliminary testing, assessing stability control through pure trajectory tracking in both single and multi-robot platoons, focusing on responsiveness and coordination; 2) Introducing disturbances, where dynamic challenges are introduced to the circle scenario in order to test trajectory control under real-world conditions with external disruptions. **Fig. 7**(d) highlights various control errors, including disturbance errors, robot dynamics-related control errors, and position and speed control errors, to evaluate the overall performance

and robustness of the algorithms. The errors offer insights into the system's ability to recover from perturbations and maintain the desired trajectory.

5.3 Experiment results and analysis

Fig. 8 presents the experimental results of speed and position control for a single robot under three models: the Fixed Physical Model, the Online Neural Network (NN) Model, and the Online PERL Framework. Each graph shows the robot's speed (desired speed as a dashed line, actual speed as a solid line) and position (desired position as a dashed line, actual position as a solid line) over time steps. The Fixed Physical Model exhibits significant fluctuations, especially between time steps 300 and 400, indicating slow response and imprecise adjustments. In contrast, both the Online NN and PERL models follow the desired speed more closely, with PERL offering smoother variations, highlighting its superior accuracy and responsiveness.

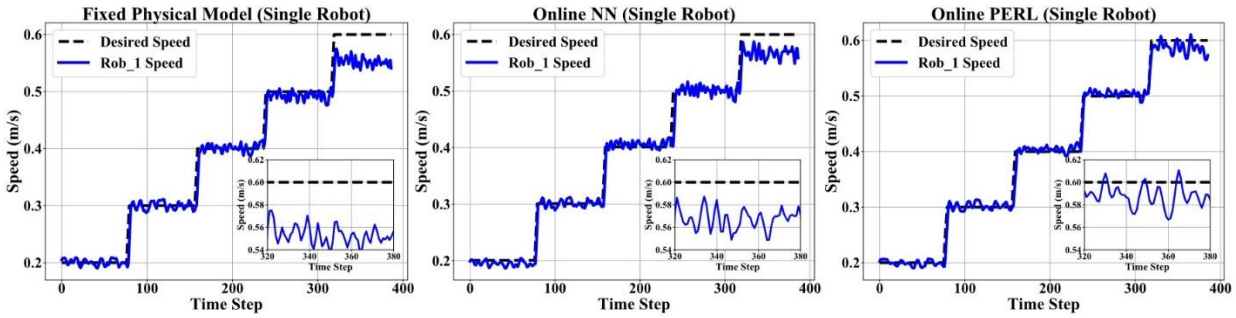


Fig. 8 Speed comparison in single robort control.

Fig. 8 compares speed in single robot control. It shows desired speed (black dashed) and actual robot speed (blue) for a single robot under three control methods: fixed physical model, online neural network (NN), and online PERL. Insets display zoomed-in speed variations. The PERL method achieves smoother and more accurate speed tracking compared to the other methods.

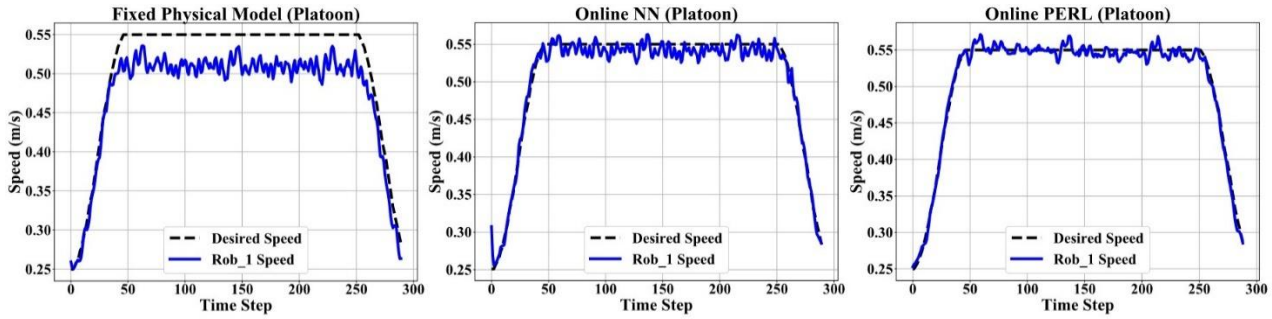


Fig. 9 Speed comparison for robot 1 in platoon control.

Fig. 9 compares speed tracking for robot 1 in platoon control under three methods: fixed physical model,

online neural network (NN), and online PERL. Desired speed is shown in black dashed lines and actual robot speed in blue. The online PERL framework closely follows the desired speed with minimal fluctuations, outperforming the other methods in precision, stability, and responsiveness for multi-robot platoon control.

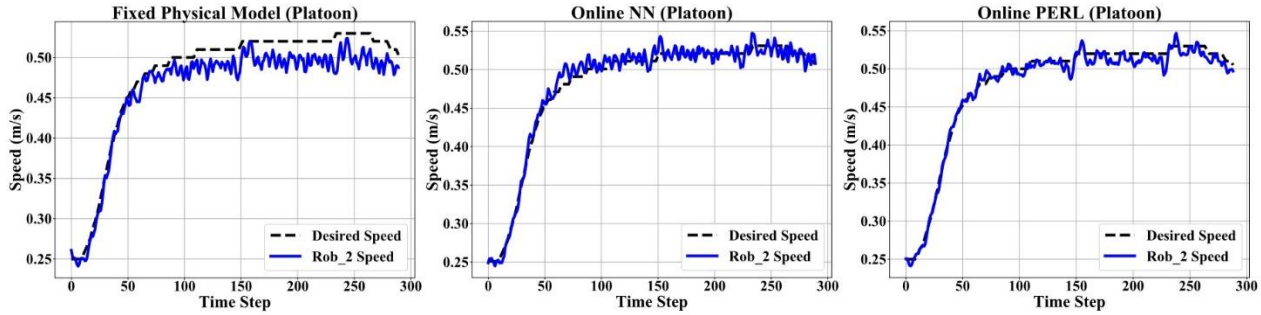


Fig. 10 Speed comparison for robot 2 in platoon control.

Fig. 10 compares speed tracking for robot 2 in platoon control under three methods: fixed physical model, online neural network (NN), and online PERL. Desired speed is shown as black dashed lines and actual robot speed as blue lines. The online PERL framework maintains close alignment with the desired speed and minimal fluctuations, outperforming the other methods in precision, stability, and responsiveness for multi-robot platoon control.

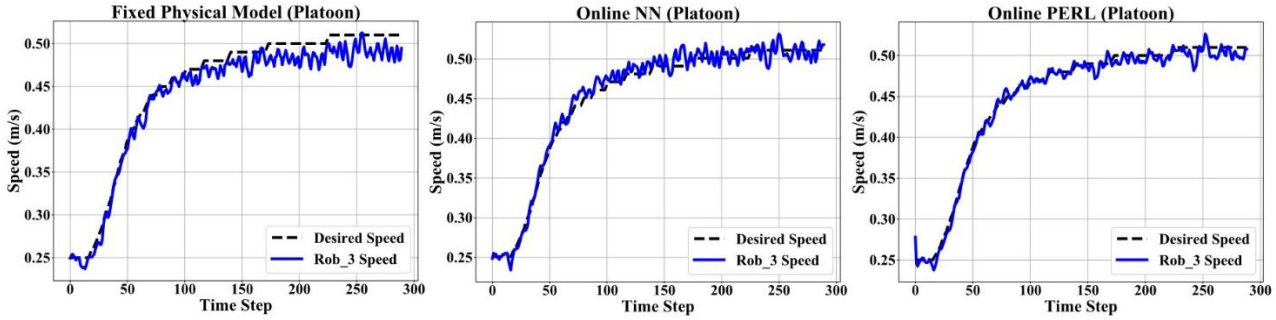


Fig. 11 Speed comparison for robot 3 in platoon control.

Fig. 11 compares speed tracking for robot 3 in platoon control under three methods: fixed physical model, online neural network (NN), and online PERL. Desired speed is shown as black dashed lines and actual robot speed as blue lines. The online PERL framework closely follows the desired speed with minimal fluctuations, outperforming the other methods in precision, stability, and responsiveness for multi-robot platoon control.

Regarding position control, the Fixed Physical Model keeps the robot's position near the desired trajectory, with slight deviations later on. The Online NN and PERL models show marked improvements, particularly PERL, where the robot consistently follows the desired path, emphasizing its real-time adaptability and

precision. Overall, the PERL model outperforms the Fixed Physical and NN models in speed and position control, particularly in dynamic adjustments and precise tracking.

In **Figs. 9-11**, the first row illustrates the speed tracking performance of three robots under three different models. In the fixed physical model, the robot speed fluctuates significantly around the desired speed, demonstrating lower stability and control precision. The online NN model shows improvement with smaller fluctuations, though some deviations persist. The online PERL model excels, with the robot speed closely following the desired trajectory and minimal fluctuations, showcasing superior control accuracy and stability. The second row of subplots presents the corresponding position tracking data. In the fixed physical model, there are significant deviations between the actual and desired positions, especially in the later stages (time steps 250-300), highlighting clear position control errors. The online NN model, through real-time adjustments, improves position control across all three robots. The online PERL model, however, nearly perfectly tracks the desired position with minimal error, demonstrating its powerful capability in dynamic platoon control.

The comparisons in **Figs. 9-11** indicate that the online PERL framework significantly outperforms both the fixed physical model and the online NN model in both speed and position control, showcasing exceptional performance in multi-robot platoons. The PERL model offers higher precision, stability, and responsiveness, proving its immense advantage in tasks requiring precise control.

Table 4 Control performance with different models in the field experiments.

Test	Single Robot			Platoon (Robot 1, 2, 3)								
				Robot 1			Robot 2			Robot 3		
Metrics	Physic	NN	PERL	Physic	NN	PERL	Physic	NN	PERL	Physic	NN	PERL
S. MSE (m^2/s^2)	0.0007	0.0005	<u>0.0003</u>	0.0014	0.0002	<u>0.0002</u>	0.0005	0.0013	<u>0.0002</u>	0.0003	0.0002	<u>0.0002</u>
P. MSE (m^2)	0.0796	0.0436	<u>0.0017</u>	1.2525	0.0202	<u>0.0019</u>	0.2818	0.0156	<u>0.0090</u>	0.1075	0.0211	<u>0.0047</u>

Table 4 summarizes the control performance of different models in single-robot and robot-platoon field tests, reporting mean squared error (MSE) values for speed (S.MSE) and position (P.MSE). The PERL model consistently achieved the lowest MSE values, demonstrating superior accuracy and performance. In single-robot tests, the PERL model recorded an S.MSE of 0.0003 and a P.MSE of 0.0019, reducing speed and position errors by 57.14% and 97.86% compared to the physical model, and by 40.00% and 96.10% compared to the NN model. This trend persisted in the platoon tests, where the PERL model outperformed its counterparts. For robots 1, 2, and 3, the PERL model also outperformed others, with average S.MSE and P.MSE values of 0.0002 and 0.0052 across three robots, reducing errors by 72.73% and 99.05% against the physical model, and by 64.71% and 72.58% against the NN model. These results highlight the exceptional performance of the PERL framework, offering lower control errors and higher precision compared to the fixed physical and NN models in both single robot and platoon control.

6. Findings and Conclusions

This study introduces an online adaptive PERL controller that combines a physical model with residual learning, showcasing exceptional capability in addressing the challenges of dynamic and unpredictable CAV platoon systems. The PERL controller, using NN-learning to adjust the physical controller's output, minimizes transitional disturbances. It integrates high-precision residual predictions and adaptability, incorporating a physical model with inertial delay and using velocity as the control output for multi-objective optimization. Online residual learning effectively addresses disturbances from complex environments and vehicle dynamics. Simulation results reveal that trajectories generated by the online PERL controller have significantly lower errors compared to the physical model, with average cumulative absolute position and speed errors reduced by 58.5% and 40.1%, respectively, and by 58.4% and 47.7% compared to NN models. Tests on a reduced-scale robot car platform further confirm the superiority of the adaptive PERL controller, with position and speed errors reduced by 72.73% and 99.05% compared to the physical model and by 64.71% and 72.58% compared to NN models. PERL's online parameter updates in response to external disturbances lead to marked improvements in vehicle control, with the system demonstrating rapid convergence and high accuracy, ensuring platoon stability under various conditions.

Despite encouraging results, this work remains limited by small-scale validation under simplified conditions, which cannot fully reflect real-world CAV platoon complexity. Future work will optimize the physical model and enhance residual learning to better capture unmodeled dynamics and communication delays. Large-scale deployment at facilities such as Mcity will assess performance under realistic traffic and dense V2X interactions. The proposed PERL controller could also be applied within the U.S. DOT CARMA Platform as a tactical-level cooperative longitudinal control component in the *platooning plugin*, enhancing adaptive gap regulation and disturbance mitigation for coordinated driving automation.

The outputs, outcomes, and impacts of this study are summarized as follows:

7. Research Outputs

- Publication:

Zhang, P., Huang, H., Zhou, H., Shi, H., Long, K., & Li, X. (2025). Online adaptive platoon control for connected and automated vehicles via physics enhanced residual learning. Transportation Research Part C: Emerging Technologies, 178, 105242.

Zhou, H., Huang, H., Zhang, P., Shi, H., Long, K., & Li, X. (2024, June). Online Physical Enhanced Residual Learning for Connected Autonomous Vehicles Platoon Centralized Control. In 2024 IEEE Intelligent Vehicles Symposium (IV) (pp. 16-22). IEEE.

- Poster:

Zhou, H., Huang, H., Zhang, P., Shi, H., Long, K. and Li, X. Online Physical Enhanced Residual Learning for Connected Autonomous Vehicles Platoon Centralized Control. TRB Annual Meeting, Presented at the 104th TRB Annual Meeting, Washington D.C., January 2025.

8. Research Outcomes

- Enhanced Platoon Control Performance: The developed Physics Enhanced Residual Learning (PERL) framework significantly improves CAV platoon stability and mobility, reducing position and speed errors by over 50% in simulation and up to 99% in scaled platform tests under dynamic disturbances.
- Real-Time Adaptive Capability: The framework enables online adjustment of control parameters in response to changing traffic conditions and disturbances, providing a robust and scalable solution for real-world connected and automated vehicle platoon operations.

9. Research Impacts

- Traffic Flow Stability: The PERL framework enhances longitudinal stability in CAV platoons, reducing speed oscillations and shockwave propagation, which can contribute to smoother traffic flow and reduced congestion.
- Safety Enhancement: By enabling rapid adaptation to disturbances and maintaining safe inter-vehicle gaps, the framework reduces collision risks in dynamic and uncertain traffic environments.
- Operational Efficiency: The adaptive control strategy allows transportation agencies and fleet operators to maintain high throughput while minimizing unnecessary accelerations and decelerations, reducing fuel consumption and improving overall operational efficiency.
- CARMA Platform Relevance: The framework's capabilities align with the U.S. DOT CARMA Platform, where it could serve as a tactical-level cooperative longitudinal control component in the platooning plugin, strengthening coordinated driving automation.

References

1. Caruntu, C. F., C. Braescu, A. Maxim, R. C. Rafaila, and A. Tiganasu. Distributed Model Predictive Control for Vehicle Platooning: A Brief Survey. Presented at the 2016 20th International Conference on System Theory, Control and Computing (ICSTCC), Sinaia, Romania, 2016.
2. Nandhini, M., and M. M. Rabik. Comparative Analysis of PID and LQR Control for Autonomous Vehicle Platoons. Singapore, 2025.
3. Li, L., C. Wang, J. Gan, Y. Zhao, X. Qu, and B. Ran. Optimizing Platoon Safety through Key Node Selection in Pinning Control Strategy. *Physica A: Statistical Mechanics and its Applications*, Vol. 643, 2024, p. 129830. https://doi.org/10.1016/j.physa.2024.129830.
4. Mohd Zaihidee, F., S. Mekhilef, and M. Mubin. Robust Speed Control of PMSM Using Sliding Mode Control (SMC)—A Review. *Energies*, Vol. 12, No. 9, 2019, p. 1669. https://doi.org/10.3390/en12091669.
5. Wang, B., and R. Su. A Distributed Platoon Control Framework for Connected Automated Vehicles in an Urban Traffic Network. *IEEE Transactions on Control of Network Systems*, Vol. 9, No. 4, 2022, pp. 1717–1730. https://doi.org/10.1109/TCNS.2022.3181522.
6. Zhang, H., and L. Du. Platoon-Centered Control for Eco-Driving at Signalized Intersection Built upon Hybrid MPC System, Online Learning and Distributed Optimization Part I: Modeling and Solution Algorithm Design. *Transportation Research Part B: Methodological*, Vol. 172, 2023, pp. 174–198. https://doi.org/10.1016/j.trb.2023.02.006.
7. Mao, X., P. Li, Z. Weng, and J. Zhao. Distributed Tube Model Predictive Control for String Stability of Heterogeneous Vehicle Platoons. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, Vol. 237, 2023, p. 095965182311737. https://doi.org/10.1177/09596518231173763.
8. Hu, M., J. Li, Y. Bian, J. Wang, B. Xu, and Y. Zhu. Distributed Coordinated Brake Control for Longitudinal Collision Avoidance of Multiple Connected Automated Vehicles. *IEEE Transactions on Intelligent Vehicles*, Vol. 8, No. 1, 2023, pp. 745–755. https://doi.org/10.1109/TIV.2022.3197951.
9. Kennedy, J. M., J. Heinovski, D. E. Quevedo, and F. Dressler. Centralized Model Predictive Control With Human-Driver Interaction for Platooning. *IEEE Transactions on Vehicular Technology*, Vol. 72, No. 10, 2023, pp. 12664–12680. https://doi.org/10.1109/TVT.2023.3277451.
10. Kianfar, R., P. Falcone, and J. Fredriksson. A Control Matching Model Predictive Control Approach to String Stable Vehicle Platooning. *Control Engineering Practice*, Vol. 45, 2015, pp. 163–173. https://doi.org/10.1016/j.conengprac.2015.09.011.

11. Shi, H., Y. Zhou, K. Wu, S. Chen, B. Ran, and Q. Nie. Physics-Informed Deep Reinforcement Learning-Based Integrated Two-Dimensional Car-Following Control Strategy for Connected Automated Vehicles. *Knowledge-Based Systems*, Vol. 269, 2023, p. 110485. https://doi.org/10.1016/j.knosys.2023.110485.

12. Huang, Z., Z. Sheng, C. Ma, and S. Chen. Human as AI Mentor: Enhanced Human-in-the-Loop Reinforcement Learning for Safe and Efficient Autonomous Driving. http://arxiv.org/abs/2401.03160. Accessed Jan. 23, 2024.

13. Liu, B., Z. Ding, and C. Lv. Platoon Control of Connected Autonomous Vehicles: A Distributed Reinforcement Learning Method by Consensus. *IFAC-PapersOnLine*, Vol. 53, No. 2, 2020, pp. 15241–15246. https://doi.org/10.1016/j.ifacol.2020.12.2310.

14. Wang, J., Y. V. Pant, and Z. Jiang. Learning-Based Modeling of Human-Autonomous Vehicle Interaction for Improved Safety in Mixed-Vehicle Platooning Control. *Transportation Research Part C: Emerging Technologies*, Vol. 162, 2024, p. 104600. https://doi.org/10.1016/j.trc.2024.104600.

15. Wang, H., Y. Yuan, X. T. Yang, T. Zhao, and Y. Liu. Deep Q Learning-Based Traffic Signal Control Algorithms: Model Development and Evaluation with Field Data. *Journal of Intelligent Transportation Systems*, Vol. 27, No. 3, 2023, pp. 314–334. https://doi.org/10.1080/15472450.2021.2023016.

16. Gao, H., C.-C. Yen, and M. Zhang. DRL Based Platooning Control with Traffic Signal Synchronization for Delay and Fuel Optimization. *Transportation Research Part C: Emerging Technologies*, Vol. 163, 2024, p. 104655. https://doi.org/10.1016/j.trc.2024.104655.

17. Shi, H., D. Chen, N. Zheng, X. Wang, Y. Zhou, and B. Ran. A Deep Reinforcement Learning Based Distributed Control Strategy for Connected Automated Vehicles in Mixed Traffic Platoon. *Transportation Research Part C: Emerging Technologies*, Vol. 148, 2023, p. 104019. https://doi.org/10.1016/j.trc.2023.104019.

18. Li, D., J. Wu, F. Zhu, T. Chen, and Y. D. Wong. Modeling Adaptive Platoon and Reservation-Based Intersection Control for Connected and Autonomous Vehicles Employing Deep Reinforcement Learning. *Computer-Aided Civil and Infrastructure Engineering*, Vol. 38, No. 10, 2023, pp. 1346–1364. https://doi.org/10.1111/mice.12956.

19. Busoniu, L., R. Babuska, and B. De Schutter. Multi-Agent Reinforcement Learning: A Survey. Presented at the 2006 9th International Conference on Control, Automation, Robotics and Vision, Singapore, 2006.

20. Xu, Y., Y. Shi, X. Tong, S. Chen, and Y. Ge. A Multi-Agent Reinforcement Learning Based Control Method for Connected and Autonomous Vehicles in A Mixed Platoon. *IEEE Transactions on Vehicular Technology*, 2024, pp. 1–14. https://doi.org/10.1109/TVT.2024.3415660.

21. Yan, R., R. Jiang, B. Jia, J. Huang, and D. Yang. Hybrid Car-Following Strategy Based on Deep Deterministic Policy Gradient and Cooperative Adaptive Cruise Control. *IEEE Transactions on*

Automation Science and Engineering, Vol. 19, No. 4, 2022, pp. 2816–2824. https://doi.org/10.1109/TASE.2021.3100709.

22. Luo, X., T. Chen, M. Li, and S. Li. Platoon Control of Automatic Vehicles Based on Deep Deterministic Policy Gradient. Presented at the 2021 40th Chinese Control Conference (CCC), Shanghai, China, 2021.
23. Jiang, S., and M. Keyvan-Ekbatani. Hybrid Perimeter Control with Real-Time Partitions in Heterogeneous Urban Networks: An Integration of Deep Learning and MPC. *Transportation Research Part C: Emerging Technologies*, Vol. 154, 2023, p. 104240. https://doi.org/10.1016/j.trc.2023.104240.
24. Bahavarnia, M., J. Ji, A. F. Taha, and D. B. Work. On the Constrained CAV Platoon Control Problem. http://arxiv.org/abs/2401.13552. Accessed Jul. 31, 2024.
25. Latrach, A., M. L. Malki, M. Morales, M. Mehana, and M. Rabiei. A Critical Review of Physics-Informed Machine Learning Applications in Subsurface Energy Systems. *Geoenergy Science and Engineering*, Vol. 239, 2024, p. 212938. https://doi.org/10.1016/j.geoen.2024.212938.
26. Jang, M., J. Hyun, T. A. Nguyen, and J.-W. Lee. PIGD-TL: Physics-Informed Generative Dynamics with Transfer Learning. Presented at the 2023 23rd International Conference on Control, Automation and Systems (ICCAS), Yeosu, Korea, Republic of, 2023.
27. Huang, H., J. Wang, C. Fei, X. Zheng, Y. Yang, J. Liu, X. Wu, and Q. Xu. A Probabilistic Risk Assessment Framework Considering Lane-Changing Behavior Interaction. *SCIENCE CHINA Information Sciences*, Vol. 63, No. 9, 2020, p. 190203. https://doi.org/10.1007/s11432-019-2983-0.
28. Makridis, M., K. Mattas, A. Anesiadou, and B. Ciuffo. OpenACC. An Open Database of Car-Following Experiments to Study the Properties of Commercial ACC Systems. *Transportation Research Part C: Emerging Technologies*, Vol. 125, 2021, p. 103047. https://doi.org/10.1016/j.trc.2021.103047.
29. Treiber, M., A. Hennecke, and D. Helbing. Congested Traffic States in Empirical Observations and Microscopic Simulations. *Physical Review E*, Vol. 62, No. 2, 2000, pp. 1805–1824. https://doi.org/10.1103/PhysRevE.62.1805.
30. Zhou, H., H. Huang, P. Zhang, H. Shi, K. Long, and X. Li. Online Physical Enhanced Residual Learning for Connected Autonomous Vehicles Platoon Centralized Control. 2024.