2024 IEEE Intelligent Vehicles Symposium (IV)
June 2-5, 2024. Jeju Island, Korea

Online Physical Enhanced Residual Learning for Connected Autonomous Vehicles Platoon Centralized Control

Hang Zhou, Heye Huang*, Peng Zhang, Haotian Shi, Keke Long, Xiaopeng Li*

2024 IEEE Intelligent Vehicle Symposium (IV) | 979-8-3503-4881-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/IV55156.2024.10588534

***Abstract*—This paper introduces a novel physical enhanced residual learning (PERL) framework for Connected Autonomous Vehicles (CAVs) platoon, aimed at addressing the challenges posed by the dynamic and unpredictable nature of traffic environments. The proposed framework synergistically combines a physical model, represented by Model Predictive Control (MPC), with data-driven online Q-learning. The MPC controller, enhanced for centralized CAV platoons, employs vehicle velocity as a control input and focuses on multi-objective cooperative optimization. The learning-based residual controller enriches the MPC with prior knowledge and corrects residuals caused by traffic disturbances. The PERL framework not only retains the interpretability and transparency of physics-based models but also significantly improves computational efficiency and control accuracy in real-world scenarios. The experimental results present that the online Q-learning PERL controller, in comparison to the MPC controller and PERL controller with a neural network, exhibits significantly reduced position and velocity errors. Specifically, the PERL's cumulative absolute position and velocity errors are, on average, 86.73% and 55.28% lower than the MPC's, and 12.82% and 18.83% lower than the neural network-based PERL's, in four tests with different reference trajectories and errors. The results demonstrate our advanced framework's superior accuracy and quick convergence capabilities, proving its effectiveness in maintaining platoon stability under diverse conditions.**

I. Introduction

Connected autonomous vehicles (CAVs) platoon with inter-vehicle communication permits vehicles to travel close together, enhancing road capacity and traffic safety [1], [2]. The vehicle platoon system integrates a complex network of interconnected agents, involving multiple vehicles. The system is characterized by the non-linearity and coupling of individual vehicle dynamics models, the interactions between system agents, model uncertainties, and external disturbances, all of which can pose significant challenges to the performance of platoon control [3], [4], as illustrated in Figure 1. Considering real-world scenarios where vehicle platoons interact with dynamic traffic under unpredictable conditions, including extreme weather like rain or snow, maintaining steady-state control is crucial for safe destination arrival. Therefore, a safe and precise motion controller is essential for achieving the stability of CAVs formation, enhancing operability, and ensuring robustness against interferences.

*Research supported by the U.S. National Science Foundation under Grant No.2313578. (corresponding author: Heye Huang, Xiaopeng Li).

Hang Zhou, Heye Huang, Peng Zhang, Haotian Shi, Keke Long, and Xiaopeng Li, are now with the Department of Civil and Environmental Engineering, Univ. of Wisconsin-Madison, Madison, WI, USA. (e-mails: hhuang468@wisc.edu; xli2485@wisc.edu).

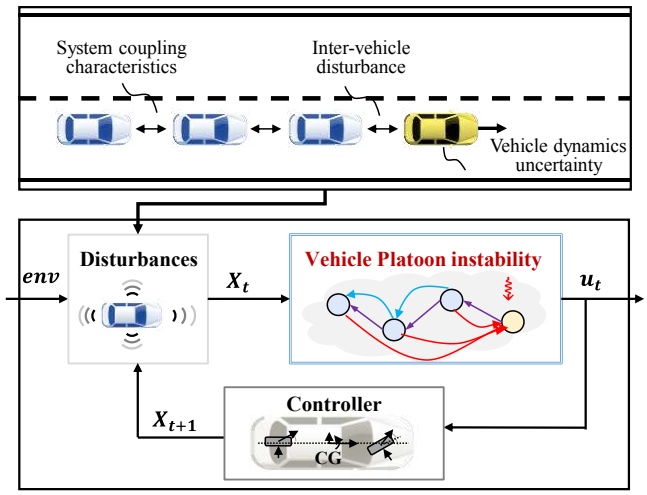

Figure 1. Safety control for vehicle toon under dynamic disturbances.

Existing methods for CAVs platoon control can be categorized into two categories [5], [6]: model-based methods (e.g. rule-driven, optimization methods), and learning-based methods (e.g. multi-agent collaborative control based on reinforcement learning and deep learning). Many platoon control designs have focused on classic control approaches based on physics models [7]. Leveraging a robust theoretical foundation, these methods provide a solid understanding and control of vehicle dynamics, offering universally applicable modeling, control, and analytical solutions for autonomous vehicle platoon control. Specifically, static linear controllers represent one of the most thoroughly investigated methods [8], [9]. They are convenient for application and facilitate the establishment of closed-loop system models for theoretical analysis of various system performances. However, this type of controller struggles to support constrained optimization frameworks with multiple explicit objectives and constraints. Furthermore, methods like Sliding Mode Control (SMC) [10], adaptive control, and Model Predictive Control (MPC) [11] have also been developed. Particularly, MPC has garnered attention due to its ability to explicitly handle safety constraints, integrate optimization of collaborative control objectives, and its clear potential for distributed application and robustness [4], [12]. These traditional methods often linearize complex systems to facilitate theoretical study, which tends to overly simplify complex dynamics and overlook the dynamic variability of platoon systems. This leads to model-based networked autonomous vehicle control methods facing challenges in effective application within real traffic.

Recently, data-driven machine learning methods [13]-[15]including Deep Learning (DL) and Reinforcement Learning (RL), have emerged as formidable tools [16], [17]. DL-based or data-driven model-free control methods, independent of dynamical models, can leverage driving trajectory data to directly design control strategies for networked autonomous vehicles. These methods are capable of capturing complex nonlinear relationships within data, enabling effective handling of various driving scenarios. RL is a commonly used model-free control method in the control of intelligent networked vehicles in mixed traffic. Various training algorithms, such as Deep Q-Networks (DQN) [18], [19] and Deep Deterministic Policy Gradients (DDPG) [20], [21], have been widely applied. Sallab et al. [22] employed deep reinforcement learning to implement lane-keeping control in the open racing car simulator (TORCS), comparing the discrete-space DQN method with the continuous action space DDAC method, demonstrating the latter's ability to achieve excellent control effects and smooth trajectories. Shi et al. [17] introduced a DRL-based cooperative CAV longitudinal control strategy for mixed traffic settings, segmenting the mixed platoon into multiple subsystems for efficient centralized cooperative control. Furthermore, multi-agent reinforcement learning, a concept explored by Busoniu et al. [23], has been widely adopted in networked CAVs platoon control [24], [25]. Compared to other machine learning algorithms, the reinforcement Q-learning method in cooperative control scenarios enables direct and simple output of Q-values in the current state to choose the best action sequence [26]. With sufficient samples or observational data, Q-learning learns the optimal state-action pairs. In practice, it has been proven to converge to the optimal state-action value with a probability of one. The Q-network reinforcement learning technique in [27] determines the optimal locations for base stations to provide enhanced platoon features to CAVs. Given the high dimensionality, continuous state and action spaces, and non-linearity of networked autonomous vehicle control problems, RL-based control strategies can learn complex control models through continuous exploration of the environment [27]. However, they often lack interpretability and transparency, making understanding the control processes and dynamic mechanisms of multi-vehicle autonomous driving challenging. Additionally, the data collection process for the required training data is inherently risky, with models only being usable post-pretraining. To summarize, both physical models and learning methods alone are inadequate for the dynamic and complex platooning control problem.

To address this gap, this paper introduces an online learning physical enhanced residual learning (PERL) controller for centralized CAV platoons. This framework integrates physical models with data-driven RL techniques. The vehicle dynamics' physical model, represented by MPC, provides foundational knowledge and constraints, while Q-learning, employed as an online residual learning method, focuses on capturing additional errors, such as those stemming from incorrect calibration of the physical model, thereby refining the model's output. In this manner, the controller bridges the gap between theoretical foundations and the dynamic, complex realities of autonomous driving scenarios. Our contributions are as follows:

- We develop a novel PERL framework for CAVs platoon control, maintaining the inherent interpretability and transparency of physics-based models. This framework simplifies troubleshooting and fine-tuning and theoretically demonstrates the control accuracy and stability.
- An enhanced MPC controller is formalized for centralized AV platoons. The velocity is employed as the control input for CAVs and considers multi-objective cooperative optimization, improving actual computational efficiency.
- We integrate a physical model with a data-driven online residual learning model. The physical MPC imbues PERL with a priori knowledge and the Q-learning composite residuals of the physical model. Experimental results indicate its commendable accuracy and rapid convergence.

The rest of this paper is organized as follows. Section 2 introduces our proposed PERL control method from the aspect of the framework, physical MPC controller, and the residual learning component Q-learning. In Section 3, we describe the experiment condition setting and present quantitative experiment results. Conclusions are given in Section 4.

II. Methodology

A. Model Framework

The framework of the proposed PERL controller is illustrated in **Figure 2**, consisting of two modules: 1) Fundamental physical-based control. We employ the MPC-based controller, which considers platoon constraints for centralized control. 2) Learning-based residual control. A residual feedback module is integrated into the traditional physical model. A reinforcement learning method Q-learning is utilized for online learning, fitting, and compensation of system model errors and external disturbances. This allows for appropriate driving speeds and real-time control output adjustments, enabling vehicles to minimize deviation from the target trajectory and maximize stability.

To help readers understand the relationship between these two modules, we introduce the workflow of the platoon control under the PERL controller before introducing the two modules. Consider an environment with discrete time steps. As illustrated in **Figure 2**, at the beginning of time step k, the platoon detects the information of the current state X_k. Then, the physical model is applied to obtain the optimal desired control input $u_k^p = f^{\mathrm{MPC}}(X_k)$. After that, the residual learning component runs with the inputs u_k^p to obtain the desired control input with residual $u_k^r = f^{\mathrm{RL}}(u_k^p)$. Finally, the controller applies u_k^r to the platoon. The state X_k transfers to the next stage X_{k+1} based on the real action with the system model errors and external disturbances u_k^a. In the next two sections, we will discuss the details of the physical MPC controller and the online Q-learning method.

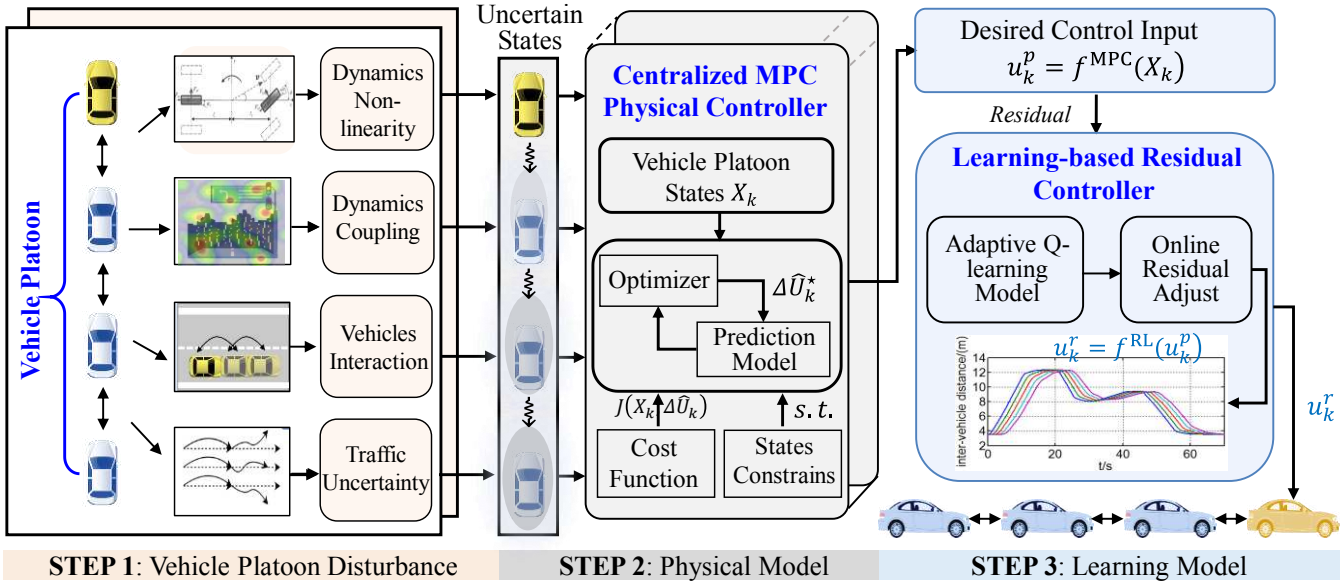

Figure 2. Illustration of the integrated CAVs platoon control framework. The inputs to the controller are uncertain vehicle platoon states with multiple disturbances. The output is the control action with residual compensation for all vehicles in the platoon.

.B. Centralized Vehicle Platoon Control with MPC

Centralized control necessitates that the central controller solves for the optimal control of each vehicle at every timestep. Addressing this type of centralized control presents challenges because: (1) the state and control of all vehicles are intertwined through the objective function and constraints; (2) a longer planning horizon requires forecasting future traffic dynamics, which can be impacted by the curse of dimensionality and disturbances.

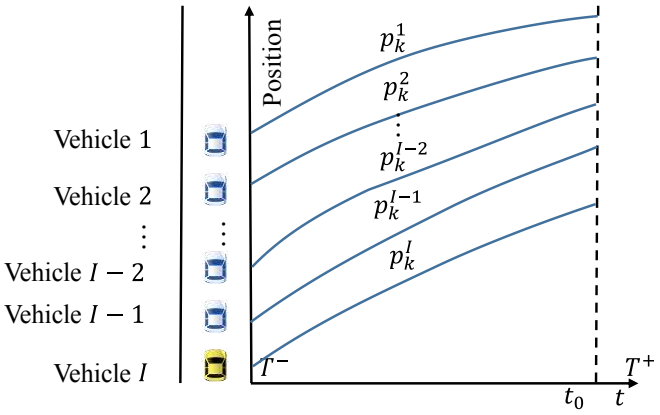

Figure 3. The centralized connected autonomous vehicles platoon.

The platoon as shown in **Figure 3**, comprises $I+1$ homogeneous vehicles, including a leading vehicle (the leader) and I following vehicles (the followers). Each vehicle in the platoon is an integral part of the control system. Operating under a predefined reference trajectory, the goal of the MPC controller is not just to approximate the real trajectory to this reference, but to do so with a level of precision that maximizes overall system efficiency, which ensures the smooth and safe operation of the entire platoon.

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:

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

where p_k^i denotes the position, v_k^i denotes the speed, a_k^i denotes the acceleration, u_k^i is the control input or desired speed of vehicle i at time step k, and τ^i is the inertial delay of vehicle longitudinal dynamics.

Based on this, we can obtain the state space model for vehicle i at time k:

$$x_{k+1}^i = A^i x_k^i + B^i u_k^i \tag{2}$$

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

$$A^i = \begin{bmatrix} 1 & \Delta t & \frac{\Delta t^2}{2} \\ 0 & 1 & \Delta t \\ 0 & -\frac{\Delta t}{\tau^i} & 0 \end{bmatrix}, B^i = \begin{bmatrix} 0 \\ 0 \\ \frac{\Delta t}{\tau^i} \end{bmatrix} \tag{3}$$

We assume that all vehicles in the platoon are homogeneous ($A^i = A, B^i = B, \tau^i = \tau$). Then we define:

$$X_k = [p_k^1, \dots, p_k^I, v_k^1, \dots, v_k^I, a_k^1, \dots, a_k^I]^\top \tag{4}$$

$$U_k = [u_k^1, \dots, u_k^I]^\top \tag{5}$$

such that the platoon dynamics are:

$$X_{k+1} = A_I X_k + B_I U_k \tag{6}$$

where $A_I = A \otimes E_I, B_I = B \otimes E_I, \otimes$ is the Kronecker operator, and E_I is the I dimensional elementary matrix.

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

$$\Delta U_k = U_k - U_{k-1} = [\Delta u_k^1, \dots, \Delta u_k^I]^\top \tag{7}$$

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

To predict the state value of the platoon for the next N steps, we introduce 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 using the model denoted as $\hat{X}_{k+n|k}$, with the prediction window defined as:

$$\mathcal{X}_k = \left[\hat{X}_{k+1|k}^\top, \dots, \hat{X}_{k+N|k}^\top\right]^\top \tag{8}$$

for the predicted value of the platoon controller as:

$$\Delta\widehat{U}_k = \left[\Delta\widehat{U}_{k|k}^\top, \dots, \Delta\widehat{U}_{k+N-1|k}^\top\right]^\top \tag{9}$$

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

$$\widehat{U}_{k+n|k} = \widehat{U}_{k+n-1|k} + \Delta\widehat{U}_{k+n|k} \tag{10}$$

$$\Delta\widehat{U}_{k+n|k} = \left[\Delta\hat{u}_{k+n|k}^1, \dots, \Delta\hat{u}_{k+n|k}^I\right]^\top \tag{11}$$

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 $\Delta\widehat{U}_k$:

$$\mathcal{X}_k = \Phi X_k + \lambda U_{k-1} + \Gamma\Delta\widehat{U}_k \tag{12}$$

where

$$\Phi = \begin{bmatrix} A_I \\ \vdots \\ A_I^N \end{bmatrix}, \lambda = \begin{bmatrix} A_I^0 B_I & & \\ \vdots & & \\ (A_I^{N-1} + \cdots + A_I^0)B_I & & \end{bmatrix} \tag{13}$$

$$\Gamma = \begin{bmatrix} B_I & \cdots & 0 \\ \vdots & \ddots & \vdots \\ (A_I^{N-1} + \cdots + A_I^0)B_I & \cdots & B_I \end{bmatrix} \tag{14}$$

Denote the reference state as $p_k^{i\star}, v_k^{i\star}, a_k^{i\star}$, and

$$X_k^\star = [p_k^{1\star}, \dots, p_k^{I\star}, v_k^{1\star}, \dots, v_k^{I\star}, a_k^{1\star}, \dots, a_k^{I\star}]^\top \tag{15}$$

$$\mathcal{X}_k^\star = [(X_{k+1}^\star)^\top, \dots, (X_{k+N}^\star)^\top]^\top \tag{16}$$

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

$$\begin{cases} \tilde{p}_k^i = p_k^i - p_k^{i*} \\ \tilde{v}_k^i = v_k^i - v_k^{i*} \\ \tilde{a}_k^i = a_k^i - a_k^{i*} \end{cases} \tag{17}$$

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

$$X_k - X_k^* = [\tilde{p}_k^1, \dots, \tilde{p}_k^I, \tilde{v}_k^1, \dots, \tilde{v}_k^I, \tilde{a}_k^1, \dots, \tilde{a}_k^I]^\top \tag{18}$$

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.

The formulation for the MPC controller with a finite prediction horizon of N steps is:

$$\begin{aligned} &J(k,N) \\ &= \min \sum_{n=0}^{N-1} \left[\sum_{i=1}^{I} q_1 \left(\hat{p}_{k+n|k}^i\right)^2 + q_2 \left(\hat{v}_{k+n|k}^i\right)^2 \right. \\ &\left. + q_3 \left(\hat{a}_{k+n|k}^i\right)^2 + q_4 \left(\Delta\hat{u}_{k+n|k}^i\right)^2 \right] \end{aligned} \tag{19}$$

$s.t.$:

$$d_{\min} \le p^{i-1} - p^i \le d_{\max}, \quad \forall i \in \mathcal{I} \tag{20}$$

$$v_{\min} \le v^i \le v_{\max}, \quad \forall i \in \mathcal{I} \tag{21}$$

$$a_{\min} \le a^i \le a_{\max}, \quad \forall i \in \mathcal{I} \tag{22}$$

where q_1, q_2, q_3, q_4 are the penalty on absolute position error, velocity error, acceleration error, and the control inputs, respectively. $d_{\max}, d_{\min}, a_{\max}, a_{\min}, v_{\max}, v_{\min}$ are the maximum and minimum values of the space, acceleration, and velocity. Constraints (20) represent the safety and maximum distance of the platoon. Constraints (21) represent the road speed limit. Constraints (22) represent the acceleration limit based on the engine and the braking systems of the vehicles.

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

$$\begin{aligned} &J\left(X_k, \Delta\widehat{U}_k\right) = \Delta\widehat{U}_k^\top(\Psi + \Gamma^\top\Omega\Gamma)\Delta\widehat{U}_k + 2(\Phi X_k + \\ &\lambda U_{k-1} - \mathcal{X}_k^\star)^\top\Omega\Gamma\Delta\widehat{U}_k \end{aligned} \tag{23}$$

$s.t.$:

$$\bar{G}\Gamma\Delta\widehat{U}_k \le -\bar{G}(\Phi X_k + \lambda U_{k-1}) - \bar{g} \tag{24}$$

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

$$Q = \begin{bmatrix} q_1 E_I & 0 & 0 \\ 0 & q_2 E_I & 0 \\ 0 & 0 & q_3 E_I \end{bmatrix}, R_\Delta = q_4 E_I \tag{25}$$

$$\bar{G} = \mathrm{diag}\left[\check{G}, \dots, \check{G}\right], \bar{g}^\top = [g^\top, \dots, g^\top], \tag{26}$$

$$\check{G} = \begin{bmatrix} \mathfrak{T}_I & 0 & 0 \\ -\mathfrak{T}_I & 0 & 0 \\ 0 & -E_I & 0 \\ 0 & E_I & 0 \\ 0 & 0 & -E_I \\ 0 & 0 & E_I \end{bmatrix}, g = \begin{bmatrix} 1_{I-1} d_{\min} \\ -1_{I-1} d_{\max} \\ 1_I v_{\min} \\ -1_I v_{\max} \\ 1_I a_{\min} \\ -1_I a_{\max} \end{bmatrix}, \tag{27}$$

$\mathfrak{T}_I$ is a size $(I-1) * I$ Toeplitz matrix with -1 on the diagonal and 1 on the first upper diagonal, and 1_{I-1} and 1_I are column vectors of ones of size $(I-1)$ and I, respectively. State-of-the-art solvers can quickly solve this quadratic optimization problem.

The optimal platoon control action is the change in control that minimizes the constrained finite horizon cost function

$$\Delta\widehat{U}_k^\star = \underset{\Delta\widehat{U}_k}{\arg\min}\, J\left(X_k, \Delta\widehat{U}_k\right) \tag{28}$$

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

C. Online Resudial Learning

Residual learning aims to approximate the residual term or "gap" between the predicted and actual system states and adjust the control output obtained by the physical model.

In residual learning, the key aspect is to measure the difference between the desired input speed set by the MPC and the actual output speed of the vehicles. This difference guides the adjustment of the vehicle's Direct Control Variable (DCV), which varies with the experimental platform. It's important to note that the input for the DCV changes based on the experimental platform used, which means the output of residual learning must be adapted to fit the actual platform. For instance, in a full-sized AV, the DCV is throttle/brake, whereas in a reduced-scale robot car, it is the motor's RPM. Our residual learning is expected to capture the disturbance caused by different DCVs.

This research utilizes Q-learning as the residual learning method. In the design of Q-learning, defining states and actions appropriately is crucial. Considering the structure of the residual learning previously outlined. The state and the action in the Q-learning are defined by the control output u_k^p obtained from the MPC and the changing rate η of the control output. Since both the state and action in this scenario are continuous variables, the resulting Q-table would be infinitely large, making it impractical to train. To address this challenge, the study employs two approaches for handling these continuous variables.

For actions, given that the vehicle acceleration is restricted within a limited range $[a_{\min}, a_{\max}]$, the output DCV also has a corresponding range. Therefore, the DCV can be discretized directly. The discretization interval Δ is a pre-determined parameter. Regarding states, the speed control error is transformed within the range of $[-\sigma, +\sigma]$ using a sigmoid function. Then, fuzzy logic is applied to discretize the continuous state space. This is achieved through the definition of several state membership functions, which help to categorize the continuous states into discrete groups.

In the training phase, the Q-learning agent chooses actions based on predicted maximum rewards, defined by the discrepancy between the vehicle's actual and input speeds; smaller errors yield higher rewards. The Q-table is updated continuously through exploration and interaction until convergence, signifying minimal changes in the table values.

III. Data Experiment and Results

In this section, the proposed PERL controller model's performance is evaluated in a simulation environment through a comparative analysis with two baseline models.

A. Experiment Condition Setting

In this simulation scenario, we set up a platoon consisting of 5 vehicles ($I = 4$) over $T = 15$ seconds, with a time step of $\Delta t = 0.1$ seconds. Two types of reference trajectories are considered. The first scenario involves uniform motion, where all vehicles maintain a constant speed of 15 m/s. The second scenario involves variable speed, divided into four phases: initially, vehicles proceed at a constant velocity of 15 m/s for 2 seconds, followed by a deceleration at -2 m/s² for 5.5 seconds, then accelerate at 2 m/s² for 5.5 seconds, and finally move at a constant velocity for the remaining 2 seconds. In both scenarios, the initial space between the five vehicles is set at 20 m. The input parameters are $d_{\min} = 15\text{m}, d_{\max} = 25\text{m}, a_{\max} = 3\text{m/s}^2, a_{\min} = -3\text{m/s}^2, v_{\max} = 20\text{m/s}, v_{\min} = 10\text{m/s}$. To simulate the error that happens during the transfer of the RPM and the velocity, we consider two types of error for the actual control output u_k^a. The first one is an affine error $u_k^a = 1.1u_k + 0.1 + x\ (m/s^2), x \sim N(0, 0.3)$, where u_k is the control input. The second one is quadratic error $u_k^a = 0.01u_k^2 + u_k + 0.1 + x\ (m/s^2), x \sim N(0, 0.3)$. For the online learning procedure, the residual learning is updated each 20 time steps (i.e., 2s) using the collected data during the experiments.

B. Baseline Models

To make a comprehensive comparison with current prediction models, we compare the performance of three different methods tested: 1. using only MPC, 2. using neural network for residual learning, and 3. using Q-learning for residual learning. All three methods will be evaluated in the two scenarios with two types of errors, i.e., in total four tests. A Multi-Layer Perceptron model with ReLU as an activation function is used as the neural network baseline model. The input and output of the network are the same as described above, i.e., the control output obtained from the MPC and the adjusted control output for the vehicles. Before the online learning, the NN model will be initially trained to make the input the same as the output.

C. Experiment Results

The time-space diagrams of the three methods in Scenario 2 are illustrated in **Figure 4**. The local magnification of the trajectory reveals that under the MPC controller, there is a considerable deviation between the actual trajectory and the reference trajectory. In contrast, with the online PERL method, the actual trajectory closely aligns with the reference trajectory, demonstrating the control accuracy. To further compare the performance of the three control methods, particularly to distinguish between the use of Q-learning and a neural network as the residual learning module in online PERL, we conduct an additional comparative analysis in **Table 1** and **Figure 5**. Given that the primary control output is velocity, we recorded both velocity and position errors to evaluate the performance of these methods. Notice that all the units for position are meters, and for velocity are meters per second.

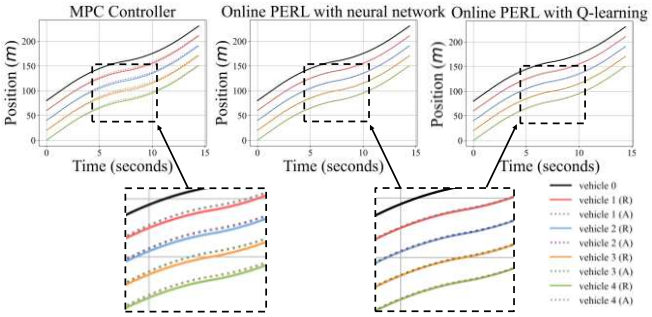

Figure 4. The time-space diagram for the three methods in Scenario 2.

Table 1 details the cumulative and maximum absolute velocity errors for the three methods across four tests. The results indicate that both variations of the PERL method, which utilize the Q-learning algorithm and a neural network, outperform using the MPC alone in four tests. Specifically, the average cumulative absolute velocity and position errors for the MPC combined with Q-learning are 55.28% and 86.73% smaller compared to using MPC alone, showing that the PERL method performance is much better than the MPC controller. When comparing the MPC combined with a neural network, the MPC combined with Q-learning has lower cumulative absolute velocity and position errors in all tests, averaging 12.82% and 18.83%, respectively. From the perspective of maximum absolute error, both the error gaps for position and velocity between the MPC combined with Q-learning and using MPC alone are still large in scenario 2, which are 88.33% and 71.24%, respectively. However, in scenario 1, the error gap of all three methods is smaller than 5%. The similarity arises from limited training data in early online learning stages, where residual learning doesn't adjust control output. In the complex environment of scenario 2, larger error gaps between the other methods and the MPC-Q-learning combination highlight the need for precise control in complex settings.

The variations of the velocity error for the whole trajectories are illustrated in **Figure 5,** which explains the results in **Table 1**. From **Figure 5**, it is evident that in the four tests, all three methods initially exhibit similar trends of the

velocity error, ranging between 0.2-0.3. As training progresses, residual learning learns the error, reducing the velocity error below zero, thereby compensating for the initial position error caused by the positive velocity error at the beginning. Moreover, **Figure 5** also demonstrates that, in all four tests, both PERL controllers eventually stabilize near zero, confirming their robustness and effectiveness. During the uniform motion in scenario 1, it is observed that the velocity error convergence for the MPC combined with Q-learning is noticeably smoother, achieving a minimum error of around 0.1, whereas, for the MPC combined with a neural network, the minimum error reaches approximately 0.2. Consequently, Q-learning, as an online residual learning approach, demonstrates a faster convergence rate, making it preferable for online PERL methods.

However, the current simulations, using simplified control environments and error forms, may not mirror real-world conditions accurately, thus necessitating further real-world testing to validate the PERL methods' performance.

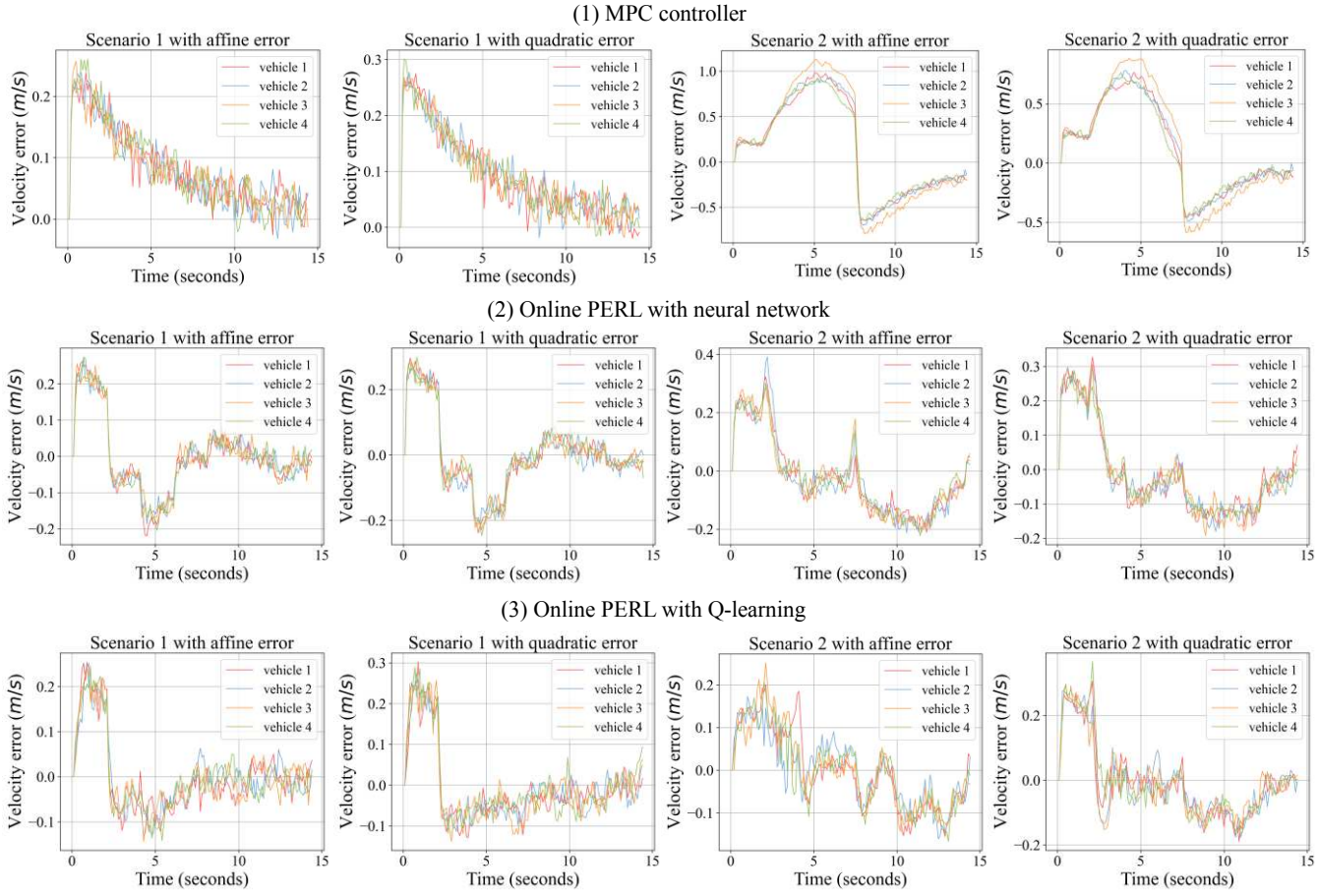

Figure 5. The comparison of experimental results for the three methods.

TABLE 1 Simulation results for the three control models in four tests.

Test results	Scenario 1 with Affine Error			Scenario 1 with Quadratic Error			Scenario 2 with Affine Error			Scenario 2 with Quadratic Error		
	M	**M+N**	**M+Q**	**M**	**M+N**	**M+Q**	**M**	**M+N**	**M+Q**	**M**	**M+N**	**M+Q**
$\boldsymbol{CAE_p}$	475.5	68.7	**62.2**	542.5	86.9	**79.2**	1388.2	199.9	**161.0**	1137.0	181.5	**157.0**
$\boldsymbol{GapCAE_p}$	86.92	9.44	**0.00**	85.40	8.88	**0.00**	88.41	19.46	**0.00**	86.19	13.51	**0.00**
$\boldsymbol{CAE_v}$	47.9	35.0	**32.9**	53.9	47.9	**38.6**	288.0	63.3	**41.7**	201.1	57.4	**48.4**
$\boldsymbol{GapCAE_v}$	31.32	6.02	**0.00**	28.33	19.38	**0.00**	85.53	34.19	**0.00**	75.94	15.71	**0.00**
$\boldsymbol{MAE_p}$	1.129	0.421	**0.370**	1.356	0.493	**0.417**	5.220	0.616	**0.493**	3.978	0.618	**0.552**
$\boldsymbol{GapMAE_p}$	68.83	11.92	**0.00**	69.26	15.44	**0.00**	90.55	19.96	**0.00**	86.11	10.63	**0.00**
$\boldsymbol{MAE_v}$	0.260	0.267	**0.254**	0.301	0.300	**0.302**	1.132	0.391	**0.251**	0.884	0.327	**0.312**
$\boldsymbol{GapMAE_v}$	2.32	4.88	**0.00**	-0.41	-0.58	**0.00**	77.80	35.73	**0.00**	64.68	4.68	**0.00**

Notations: CAE and MAE represent the cumulative absolute error and the maximum absolute error. Gap represents the error gap compared with the errors obtained by MPC combined with Q-learning. M, M+N, and M+Q to represent using MPC alone, MPC combined with neural network, and MPC with Q-learning.

IV. Conclusion

This paper proposes a novel online PERL controller that incorporates the physical model and residual learning to enhance centralized control of CAVs platoon. To mitigate the disturbance caused during the transition of the control output, the PERL controller applies Q-learning as a residual learning module to adjust the control output of the physical model, i.e., the MPC controller. This integrated model exhibits high precision in predicting control residuals and demonstrates exceptional adaptability. In the MPC controller, the vehicle dynamics' physical model with the inertial delay is incorporated. By setting velocity as the control output, multi-objective optimization under multiple constraints is achieved. For online residual learning, Q-learning is applied to learn the disturbance caused by the complex environment and vehicles' dynamics. The experiments demonstrate that the trajectories by the online Q-learning PERL controller exhibit significantly reduced errors, with cumulative absolute position and velocity errors averaging 86.73% and 55.28% lower than those of the MPC controller, and 12.82% and 18.83% lower compared to the PERL controller utilizing a neural network. The variation in velocity errors highlights the faster convergence speed of the Q-learning in the online learning process.

In the future, we will further explore the suitable physical control model and the residual learning method for the CAV control. Besides, considering that the manual affine and quadric errors cannot demonstrate real-world disturbance, we plan to apply our PERL controller in the real vehicle platform to evaluate the effectiveness and robustness of the controller.

References

[1] U. Montanaro *et al.*, "Towards connected autonomous driving: review of use-cases," *Veh. Syst. Dyn.*, vol. 57, no. 6, pp. 779–814, Jun. 2019.

[2] Z. Huang, S. Chen, Y. Pian, Z. Sheng, S. Ahn, and D. A. Noyce, "CV2X-LOCA: Roadside Unit-Enabled Cooperative Localization Framework for Autonomous Vehicles," arXiv, arXiv:2304.00676, Apr. 2023.

[3] A. A. Zafar, "Consensus, Control and Message Passing in Complex Control Systems," phd, Aston University, 2021. Accessed: Jan. 30, 2024. [Online]. Available: https://publications.aston.ac.uk/id/eprint/43993/

[4] J. M. Kennedy, J. Heinovski, D. E. Quevedo, and F. Dressler, "Centralized Model Predictive Control With Human-Driver Interaction for Platooning," *IEEE Trans. Veh. Technol.*, vol. 72, no. 10, pp. 12664–12680, Oct. 2023.

[5] Q. Li, Z. Chen, and X. Li, "A Review of Connected and Automated Vehicle Platoon Merging and Splitting Operations," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 12, pp. 22790–22806, Dec. 2022.

[6] S. E. Li, Y. Zheng, K. Li, L.-Y. Wang, and H. Zhang, "Platoon Control of Connected Vehicles from a Networked Control Perspective: Literature Review, Component Modeling, and Controller Synthesis," *IEEE Trans. Veh. Technol.*, pp. 1–1, 2018.

[7] S. Badnava *et al.*, "Platoon Transitional Maneuver Control System: A Review," *IEEE Access*, vol. 9, pp. 88327–88347, 2021.

[8] J. Rubió-Massegú, F. Palacios-Quiñonero, J. M. Rossell, and H. R. Karimi, "Static Output-Feedback Control for Vehicle Suspensions: A Single-Step Linear Matrix Inequality Approach," *Math. Probl. Eng.*, vol. 2013, p. e907056, Dec. 2013.

[9] S. E. Li, F. Gao, D. Cao, and K. Li, "Multiple-Model Switching Control of Vehicle Longitudinal Dynamics for Platoon-Level Automation," *IEEE Trans. Veh. Technol.*, vol. 65, no. 6, pp. 4480–4492, Jun. 2016.

[10] F. Mohd Zaihidee, S. Mekhilef, and M. Mubin, "Robust Speed Control of PMSM Using Sliding Mode Control (SMC)—A Review," *Energies*, vol. 12, no. 9, Art. no. 9, Jan. 2019.

[11] C. F. Caruntu, C. Braescu, A. Maxim, R. C. Rafaila, and A. Tiganasu, "Distributed model predictive control for vehicle platooning: A brief survey," in *2016 20th International Conference on System Theory, Control and Computing (ICSTCC)*, Sinaia, Romania: IEEE, Oct. 2016, pp. 644–650.

[12] R. Kianfar, P. Falcone, and J. Fredriksson, "A control matching model predictive control approach to string stable vehicle platooning," *Control Eng. Pract.*, vol. 45, pp. 163–173, Dec. 2015.

[13] H. Shi, 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," *Knowl.-Based Syst.*, vol. 269, p. 110485, Jun. 2023.

[14] P. Wu, Z. Huang, Y. Pian, L. Xu, J. Li, and K. Chen, "A Combined Deep Learning Method with Attention-Based LSTM Model for Short-Term Traffic Speed Forecasting," *J. Adv. Transp.*, vol. 2020, p. e8863724, Nov. 2020.

[15] Z. Huang, Z. Sheng, C. Ma, and S. Chen, "Human as AI Mentor: Enhanced Human-in-the-loop Reinforcement Learning for Safe and Efficient Autonomous Driving." arXiv, Jan. 09, 2024. Accessed: Jan. 23, 2024. [Online]. Available: http://arxiv.org/abs/2401.03160

[16] B. Liu, Z. Ding, and C. Lv, "Platoon control of connected autonomous vehicles: A distributed reinforcement learning method by consensus," *IFAC-Pap.*, vol. 53, no. 2, pp. 15241–15246, Jan. 2020.

[17] H. Shi, 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," *Transp. Res. Part C Emerg. Technol.*, vol. 148, p. 104019, Mar. 2023.

[18] H. Wang, 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," *J. Intell. Transp. Syst.*, vol. 27, no. 3, pp. 314–334, May 2023.

[19] J. Dong, S. Chen, P. Y. J. Ha, Y. Li, and S. Labi, "A DRL-based Multiagent Cooperative Control Framework for CAV Networks: a Graphic Convolution Q Network," arXiv, arXiv:2010.05437, Oct. 2020.

[20] R. Yan, 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 Trans. Autom. Sci. Eng.*, vol. 19, no. 4, pp. 2816–2824, Oct. 2022.

[21] X. Luo, T. Chen, M. Li, and S. Li, "Platoon Control of Automatic Vehicles Based on Deep Deterministic Policy Gradient," in *2021 40th Chinese Control Conference (CCC)*, Shanghai, China: IEEE, Jul. 2021, pp. 6154–6159.

[22] A. E. Sallab, M. Abdou, E. Perot, and S. Yogamani, "Deep Reinforcement Learning framework for Autonomous Driving," *Electron. Imaging*, vol. 2017, no. 19, pp. 70–76, 2017.

[23] L. Busoniu, R. Babuska, and B. De Schutter, "Multi-Agent Reinforcement Learning: A Survey," in *2006 9th International Conference on Control, Automation, Robotics and Vision*, Singapore: IEEE, 2006, pp. 1–6.

[24] H. Ge, Y. Song, C. Wu, J. Ren, and G. Tan, "Cooperative Deep Q-Learning With Q-Value Transfer for Multi-Intersection Signal Control," *IEEE Access*, vol. 7, pp. 40797–40809, 2019.

[25] L. Roveda, A. Testa, A. A. Shahid, F. Braghin, and D. Piga, "Q-Learning-based model predictive variable impedance control for physical human-robot collaboration," *Artif. Intell.*, vol. 312, p. 103771, Nov. 2022.

[26] M. Hamadouche, C. Dezan, D. Espes, and K. Branco, "Comparison of Value Iteration, Policy Iteration and Q-Learning for solving Decision-Making problems," in *2021 International Conference on Unmanned Aircraft Systems (ICUAS)*, Athens, Greece: IEEE, Jun. 2021, pp. 101–110.

[27] S. B. Prathiba, G. Raja, K. Dev, N. Kumar, and M. Guizani, "A Hybrid Deep Reinforcement Learning For Autonomous Vehicles Smart-Platooning," *IEEE Trans. Veh. Technol.*, vol. 70, no. 12, pp. 13340–13350, Dec. 2021.

[28] L. Lei, T. Liu, K. Zheng, and L. Hanzo, "Deep Reinforcement Learning Aided Platoon Control Relying on V2X Information," *IEEE Trans. Veh. Technol.*, vol. 71, no. 6, pp. 5811–5826, Jun. 2022.

[29] I. Draganjac, D. Miklic, Z. Kovacic, G. Vasiljevic, and S. Bogdan, "Decentralized Control of Multi-AGV Systems in Autonomous Warehousing Applications," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 4, pp. 1433–1447, Oct. 2016.