2023 The 23rd International Conference on Control, Automation and Systems (ICCAS 2023)
Yeosu Venezia Hotel & Utop Marina Hotel, Yeosu, Korea, Oct. 17∼20, 2023

Learning-based Predictive Control for Vehicle Following Problems

Ziliang Wang[1] Shuyou Yu[1*] Yajing Zhang[1] Xiaohui Yu[1] Hong Chen[2]

[1]Department of Control Science and Engineering, Jilin University,
Changchun, 130012, China (shuyou@jlu.edu.cn)* Corresponding author
[2]College of Electronics and Information Engineering, Tongji University,
Shanghai, 201804, China (chenhong2019@tongji.edu.cn)

Abstract: Recent research shows that the combination of reinforcement learning (RL) with traditional control method can be an effective tool for designing near optimal feedback controller for dynamic systems. In this paper, a vehicle-following control based on reinforcement learning is proposed, in which pairs of the input-output of model predictive control (MPC) are chosen as offline-learning data. Through continuous iterations of actor-network and critic-network, the longitudinal vehicle-following controller can be obtained. Simulation results illustrate that proposed learning-based predictive control (LPC) can improve the computational efficiency, and obtain a better performance of policy optimization.

Keywords: Model predictive control, actor-critic, vehicle-following control, reinforcement learning

1. INTRODUCTION

Vehicle following is one of the important research objects in the field of autonomous driving, which can reduce driving burden and energy consumption, and has attracted extensive attention in different fields such as traffic engineering, statistical physics and psychology [1].

Traditional vehicle-following control methods need accurate state of the leading vehicle according to longitudinal vehicle dynamics model [2]. With the development of artificial intelligence (AI), high-precision vehicle motion datasets can be obtained. The stability and safety of the vehicle-following model is verified by considering the characteristics of multiple vehicles in [3]. A vehicle-following model with time-varying delay considering the time headway is constructed in [4]. A Nash optimality based distributed model predictive control scheme is proposed in [5]. A model predictive control method with the combination of Gaussian model is constructed for the solution of vehicle following problems [6].

Reinforcement learning (RL) is one of the paradigms and methodologies of machine learning based on Markov Decision Process (MDP). By interacting with the environment, RL can achieve the goal of obtaining the optimal solution of specific value function. The actor-critic algorithm can reduce the instability and fluctuation of the iteration, in which actor-network is responsible for generating actions, and critic-network is to evaluate the control input in next stage by updating the value function. A learning-based predictive control method is built in [7] which greatly reduces the computational time. A stochastic longitudinal leading model through Gaussian Process (GP) is established in [8], which optimizes the tracking performance. A decentralized transportable multi-agent actor-critic algorithm is designed in [9], where the trafficability of the vehicle is improved.

Learning-based predictive control strategy is proposed for vehicle-following control in this paper, in which only longitudinal dynamic is considered. The control input is calculated under the actor-critic scheme, and the results of model predictive control (MPC) are chosen as the offline-learning datasets. The objective of learning-based predictive control (LPC) is to optimize the control input and improve the computational efficiency.

The rest of the paper is as follows: Section II describes the establishment process of longitudinal vehicle model. Section III introduces the application of vehicle-vehicle (V2V) communication technology and the collection process of learning-datasets. Section IV introduces the modeling of the kernel-based actor-critic network. Section V shows the simulation results which verify the effectiveness. Section VI summarizes the whole paper.

This work is supported in part by the Natural Science Foundation of Jilin Province under Grant YDZJ202101ZYTS169, in part by the National Natural Science Foundation of China under Grant U1964202.

2. PROBLEM SETUP

This section starts with the longitudinal vehicle model, and all the vehicles should satisfy the premise of safety and consensus. Note that this paper focuses on the longitudinal control, i.e., all the vehicles in the platoon moves along the same straight lane.

2.1 Longitudinal vehicle model

The longitudinal characteristics of vehicle is represented by the following third-order equation [10]:

$$\begin{cases} \dot{s} = v \\ \dot{v} = a \\ \dot{a} = f(v,a) + g(v)\eta \end{cases} \tag{1}$$

where s represents the position of the vehicle, v and a respectively represent the velocity and acceleration, and η represents engine input, functions f and g can be respec-

tively expressed as:

$$\begin{cases} f(v,a) = \dfrac{-2C_D}{m}va - \dfrac{1}{\tau(v)}\left(a + \dfrac{C_D}{m}v^2 + \dfrac{d_m}{m}\right) \\ g(v) = \dfrac{1}{m\tau(v)} \end{cases} \tag{2}$$

where C_D is the aerodynamic coefficient, m the vehicle mass, τ the time constant of the engine, d_m the mechanical drag [11].

The engine input η is written as:

$$\eta = mu + C_D v^2 + d_m + 2\tau C_D va \tag{3}$$

in which the control input u is the deserved acceleration of the following vehicle.

Assume that the state of the leading vehicle and road information are known. The position error between vehicles is calculated as follows:

$$e_{s,i} = s_{i-1} - s_i - D_{des} \tag{4}$$

where s_i is the longitudinal position of the ith vehicle. D_{des} the expected distance between the leading vehicle and the following vehicle, i.e.,

$$D_{des} = D_0 + hv \tag{5}$$

where the constant h is the time headway, D_0 the minimum safety distance.

Similarly, the velocity error between vehicles is:

$$e_{v,i} = v_i - v_{i-1} \tag{6}$$

where v_i is the longitudinal velocity of the ith vehicle.

The goal of longitudinal vehicle-following control is to make the position error e_s and the velocity error e_v as close to zero as possible when $t \to \infty$.

Denote

$$x = \begin{bmatrix} e_s & e_v & a_i \end{bmatrix}^T, \quad u = a_{des}, \quad \omega = a_1 \tag{7}$$

Then the equivalent linear state-space representation of longitudinal vehicle model can be written as:

$$\dot{x}(t) = \overline{A}x(t) + \overline{B}_1 u(t) + \overline{B}_2 \omega(t) \tag{8}$$

where

$$\overline{A} = \begin{bmatrix} 0 & 1 & -h \\ 0 & 0 & -1 \\ 0 & 0 & -\frac{1}{\tau} \end{bmatrix} \overline{B}_1 = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{\tau} \end{bmatrix} \overline{B}_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tag{9}$$

Note that the position error e_s, the velocity error e_v and the actual acceleration of the following vehicle a_i are selected as state variables. The expected acceleration a_{des} is control input, and the acceleration of the leading vehicle can be treated as disturbances.

Define T_s as the sampling time, then the discrete-time state-space representation of the longitudinal vehicle model can be written as:

$$x(k+1) = Ax(k) + B_1 u(k) + B_2 a_1(k) \tag{10}$$

with

$$A = \begin{bmatrix} 1 & T_s & -T_s h \\ 0 & 1 & -T_s \\ 0 & 0 & 1 - T_s\tau^{-1} \end{bmatrix} B_1 = \begin{bmatrix} 0 \\ 0 \\ T_s\tau^{-1} \end{bmatrix} B_2 = \begin{bmatrix} 0 \\ T_s \\ 0 \end{bmatrix} \tag{11}$$

2.2 Vehicle-following control based on RL

In RL, the state-action value function in state s is defined as the expected and discounted total rewards when taking action a under the policy π:

$$Q^\pi(s,a) = E^\pi\left[\sum_{t=0}^{\infty}\gamma^t r_t \middle| s_0 = s, a_0 = a\right] \tag{12}$$

where $\gamma\in(0,1)$ is the discount factor, r_t is the reward at time-step t, and $E^\pi[\cdot]$ represents the expectation value under the policy π.

The optimal strategy $\pi^*(s)$ is obtained by maximizing the state-action value function (12) [12]:

$$\pi^*(s) = arg\ \underset{a}{max}\, Q^{\pi^*}(s,a) \tag{13}$$

Learning-based predictive control solves the vehicle following problems through actor-critic algorithm, which improves the computational efficiency and reduces fluctuations of system evolutions [13]. In this paper, the platoon includes one leading vehicle and three following vehicles, and the diagram of simplified vehicle following problem based on actor-critic scheme is as follows:

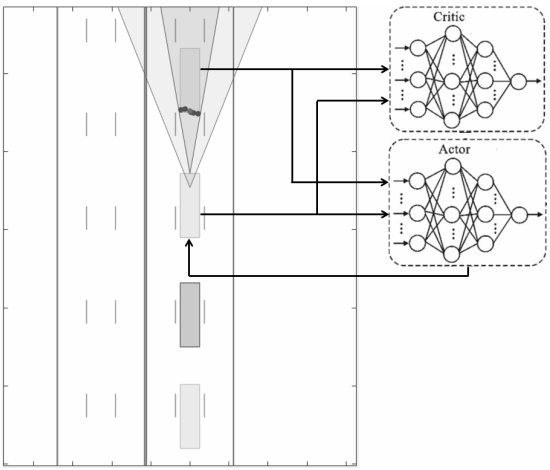


Fig. 1. The diagram of vehicle following

3. LEARNING DATA COLLECTION

Nowadays, with the development of wireless communication technology, V2V is used to exchange the state and control input between the leading vehicle and the following vehicles, which realize bidirectional communication between high-velocity vehicles [14]. As mentioned earlier, the calculation of the following vehicle control input u is related to the state variables of the leading vehicle. Thus in this paper, it is necessary to assume that the information of the leading vehicle can be known [15].

According to different learning styles, reinforcement learning can be divided into online-learning and offline-learning. The training datasets of online-learning is updated in real-time, and the computational time is shorter. However, due to insufficient initial datasets, online-learning may lead to the wrong direction and ultimately get a large residual error. By generating optimal state-control set from the past datasets, offline-learning can implement corresponding optimal strategy in the face of different states and can effectively utilize the existing datasets without adverse impact on the environment [16]. In this paper, offline-learning is taken.

Nash optimality iterative algorithm was proposed by Nash (1951) to solve the cooperative game problem. Vehicle following is a multi-vehicle cooperation problem, i.e., through the design of each local controller and the negotiation between local controllers, each vehicle can move in a given speed steadily. Nash optimality based MPC can reduce the computational burden through solving the problem jointly. The main procedure of Nash optimality based MPC is summarized as follows:

Algorithm 1 Nash optimality based MPC [5]

Step 1: At moment k, initialize the estimated value of control input of the following vehicle;
Step 2: Solve the predictive control problem and iterate the optimal solution;
Step 3: Judge whether the system meets the condition of iterative convergence, i.e., whether it meets the condition $\|U_{i,N}^{l+1}(k) - U_{i,N}^{l}(k)\| \leq \epsilon_i$
where $U_{i,N}^{l}(k)$ is the actual acceleration of the ith following vehicle at moment k. N is the prediction horizon length, and l is the number of iterations. If all vehicles meet this condition, the calculation ends and jumps to step 4; otherwise, returns to step 2;
Step 4: Apply the first control input of the sequence $u(k) = [I...0]U_{i,N}^{*}(k)$ at moment k to the vehicle;
Step 5: Set $k \rightarrow k+1$, go back to step 1.

One of current research fields of RL is to use the state-control sequence calculated by MPC as the learning samples of actor-critic weight update. Through simulation and verification, the convergence and stability of MPC can be confirmed, which is crucial for the selection of learning samples. Through learning-based predictive control, the fluctuation of control input and the computational time is further reduced.

4. KERNEL-BASED ACTOR-CRITIC NETWORK MODELING

The central idea of this section is using datasets of MPC as the initialization, where reinforcement learning is treated as the tool of policy updating [19]. The LPC algorithm is presented as a new class of closed-loop optimization method.

4.1 Learning-based predictive control scheme

Define the stage cost function as:

$$r(x(j), u_j(x(j))) = x^T(j)Qx(j) + u_j^T(x(j))Ru_j(x(j)) \tag{14}$$

where $Q \in R^{n\times n}$ and $R \in R^{m\times m}$ are the positively definite and symmetric weighting matrix. n is the dimension of the state x, and m is the dimenson of the control input u. Define N is the prediction horizon length, the cost function of finite horizon optimization problem in the prediction horizon $[k, k+N-1]$ at moment k is:

$$\underset{\|u_t(x(t))\|\leq\overline{U}}{minimize}\ J_t(x(t)) \tag{15}$$

with

$$J_t(x(t)) = E\big[r(t) + J_{t+1}(x(t+1))\big], t \in [k, k+N-1] \tag{16}$$

where $\overline{U}$ is the control quantity constraint, $E[\cdot]$ is the expectation operator, and $r(j) \equiv r(x(j), u_j(x(j)))$.

Assume that there exists an optimal control strategy u^* for the vehicle following control. According to Bellman's optimality principle, the optimal cost function $J_t^*(x(t))$ in $t \in [k, k+N-1]$ satisfies the following discrete-time Hamilton-Jacobi-Bellman (HJB) equation:

$$J_t^*(x(t)) = \min_{\|u_t(x(t))\|\leq\overline{U}} E\big[r(t) + J_{t+1}^*(x(t+1))\big] \tag{17}$$

The optimal control $u_t^*(t)$ satisfies:

$$u_t^*(t) = arg \min_{\|u_t(x(t))\|\leq\overline{U}} E[r(t) + J_{t+1}^*(x(t+1))] \tag{18}$$

In each prediction horizon time, it is actually quite difficult to directly solve the discrete HJB equation. Thus next section will introduce the specific method of solving the approximation problem.

4.2 Kernel-based actor-critic algorithm

This section describes how to apply the kernel-based actor-critic network on the LPC. In past several years, the method of kernel function is popular, especially in the field of reinforcement learning [20]. Define H as the feature space (Hilbert space), if there exists a mapping from the original space X to H:

$$\phi(x) : X \rightarrow H \tag{19}$$

then the kernel function $k(x, z)$ is:

$$k(x, z) = < \phi(x), \phi(z) > \tag{20}$$

where $x, z \in X$, $\phi(x)$ and $\phi(z)$ are the mapping function, $< \phi(x), \phi(z) >$ is the inner product of $\phi(x)$ and $\phi(z)$.

Kernel functions mainly include linear kernel function, polynomial kernel function and Gaussian kernel function [21]. In this paper, Gaussian kernel function is chosen, which basic form is as follows:

$$\begin{aligned} k(x, z) &= exp\Big(-\frac{\|x-z\|^2}{2\sigma^2}\Big) \\ &= exp(-\gamma\|x-z\|^2) \end{aligned} \tag{21}$$

where σ and γ are positive constants , $\gamma = \frac{1}{2\sigma^2}$.

In actor-critic algorithm, actor-network is responsible for updating the optimal control u_t^*, and critic-network is responsible for updating the evaluation indicators λ_t^*, which is the derivative of the cost function J_t^* with respect to x [22]. The schematic overview is as follows, where the dashed line indicates that critic is responsible for updating the actor and itself:

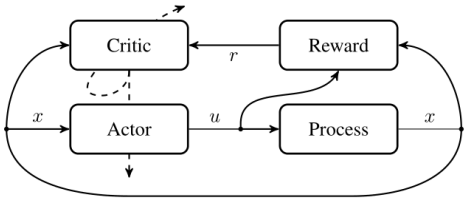


Fig. 2. Schematic overview of the actor-critic algorithm

To represent the influence of critic-network and actor-network in kernel-based actor-critic algorithm, the control input u is calculated as follows:

$$u_t^j(x(t)) = E\left[-\frac{1}{2R}\left(\frac{\partial x(t+1)}{\partial u_t^j(x(t))}\right)^T \lambda_{t+1}^i(x(t+1))\right] \tag{22}$$

where i represents the iteration number of critic-network, j represents the iteration number of actor-network, and $\lambda_{t+1}^i(x(t+1))$ is calculated by differentiating the state quantity $x(t)$:

$$\lambda_{t+1}^i(x(t+1)) = \frac{\partial J_{t+1}^i(x(t+1))}{\partial x(t)} \tag{23}$$

In each prediction horizon $[k, k+N-1]$, the structure of actor-network is written as:

$$\begin{aligned}\hat{u}(x(t)) =& \overline{U}\Gamma\left[\sum_l^{l=L} \omega_{aj}^{[l]}(t)\Psi_t^{[l]}(x(t))\right] \\ =& \overline{U}\Gamma(\omega_{aj}^T(t)\Psi_t(x(t)))\end{aligned} \tag{24}$$

where $\Gamma(\cdot)$ is the monotonic odd function, $\|\Gamma(\cdot)\| \leq 1$. Assume that the first-order derivative of function $\Gamma(\cdot)$ is bounded.

The structure of critic-network is written as:

$$\begin{aligned}\lambda_t^i(x(t)) =& \sum_l^{l=L} \omega_{ci}^{[l]}(t)\Phi_t^{[l]}(x(t)) \\ =& \omega_{ci}^T(t)\Phi_t(x(t))\end{aligned} \tag{25}$$

where ω_a and ω_c represent the weight vectors of actor-network and critic-network, and $\Psi_t(x(t))$ and $\Phi_t(x(t))$ are the feature vectors.

In general, the basis vectors $\Psi_t(x(t))$ and $\Phi_t(x(t))$ can be designed with manually selected parameters. The update process of actor-weight is as follows [7]:

$$\begin{aligned}\omega_a^{j+1}(t) =& (\Psi_t(x(t))\Psi_t^T(x(t)))^{-1} \\ & \times \Psi_t(x(t))D_a^{jT}(x(t), \omega_a^j(t), \omega_{ci}(t+1))\end{aligned} \tag{26}$$

with

$$\begin{aligned}& d_a^j(x(t), \omega_a^j(t), \omega_{ci}(t+1)) \\ &= E\left[-\frac{1}{2R}\left(\frac{\partial x(t+1)}{\partial u_t^j(x(t))}\right)\omega_{ci}^T(t+1)\Phi_{t+1}(x(t+1))\right]\end{aligned} \tag{27}$$

and

$$\begin{aligned}& D_a^j(x(t), \omega_a^j(t), \omega_{ci}(t+1)) \\ &= \Gamma^{-1}(\overline{U}^{-1} d_a^j(x(t), \omega_a^j(t), \omega_{ci}(t+1)))\end{aligned} \tag{28}$$

For the critic-network, the update process of critic-weight coefficient is as follows:

$$\begin{aligned}\omega_{ci+1}(t) = (\Phi_t(x(t))\Phi_t^T(x(t)))^{-1}\times \\ \Phi_t(x(t))D_{ci}^T(x(t), \omega_a^j(t), \omega_{ci}(t+1))\end{aligned} \tag{29}$$

with

$$\begin{aligned}& D_{ci}(x(t), \omega_a^j(t), \omega_{ci}(t+1)) = \\ & E\left[2Qx(t) + \left(\frac{\partial x(t+1)}{\partial (x(t))}\right)^T \omega_{ci}^T(t+1)\Phi_{t+1}(x(t+1))\right]\end{aligned} \tag{30}$$

Based on the above actor-critic update network scheme, the optimal longitudinal vehicle-following control at each moment can be calculated. According to the datasets S_t, the feature vectors $\Phi_t(x)$ and $\Psi_t(x)$ can be calculated. The computational complexity of LPC algorithm is $O(n^2N)$, and the computational complexity of traditional model predictive control is $O(n^{3.5}N^2)$ [7].

5. SIMULATION RESULTS

The simulation results of vehicle-following control in two different driving conditions are shown in this section. Set one vehicle as the leader, while the number of the following vehicles is three. In order to reflect the feasibility and generalization ability of the LPC algorithm, two different driving conditions are designed in this paper, i.e., the acceleration of the leading vehicle gradually decreases from $1.5m/s^2$ to 0, and gradually increases from 0 to $1.5m/s^2$ [18]:

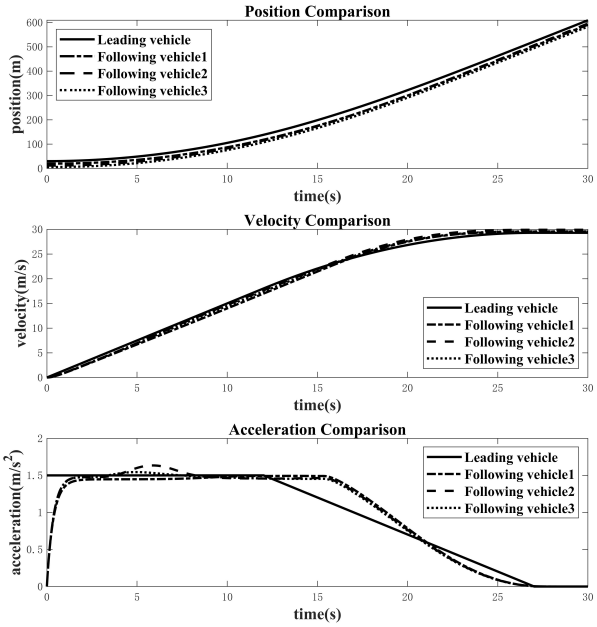
$$a_1 = \begin{cases} 1.5 & t \in [0, 12) \\ 1.5 - 0.1t & t \in [12, 27) \\ 0 & t \in [27, +\infty) \end{cases}$$

and

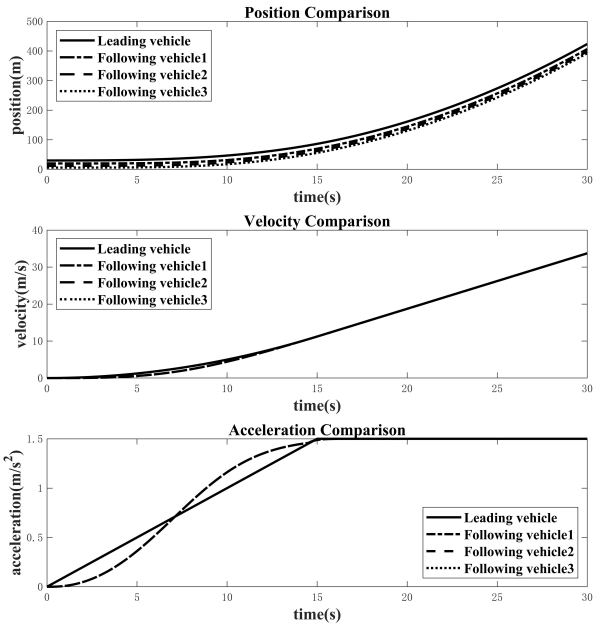
$$a_1 = \begin{cases} 0 & t = 0 \\ 0.1t & t \in (0, 15) \\ 1.5 & t \in [15, +\infty) \end{cases}$$

Based on the convergence properties of the learning process, maximum iterative number i_{max} and j_{max} are usually set between 20-40. The feature vectors are constructed according to MPC. To improve the network accuracy, $\Delta\omega_c$ and $\Delta\omega_a$ are chosen as 0.01. In this paper, i_{max} and j_{max} are chosen as 20, the prediction horizon N is 20, and the time headway h is chosen as 0.8.

Simulation results in two different driving conditions are as follows:



(a) Driving condition 1



(b) Driving condition 2

Fig. 3. Comparision in different conditions

All the simulations in this section are run under Matlab 2022a. CPU is Intel Core i7-8700 CPU 3.20GHz and GPU is NVIDIA GeForce GT 1030. Fig.3 (a) and (b) represent the comparison of position, velocity and acceleration in the longitudinal driving condition 1 and driving condition 2. The simulation results show that the vehicle-following control strategy based on LPC can achieve the goal of tracking the leading vehicle. When the driving condition changes, the following vehicle can still track the leading vehicle. The result verifies LPC has adaptation of environmental changes [24].

The comparison of computational time in two differenlt driving conditions is as follows:

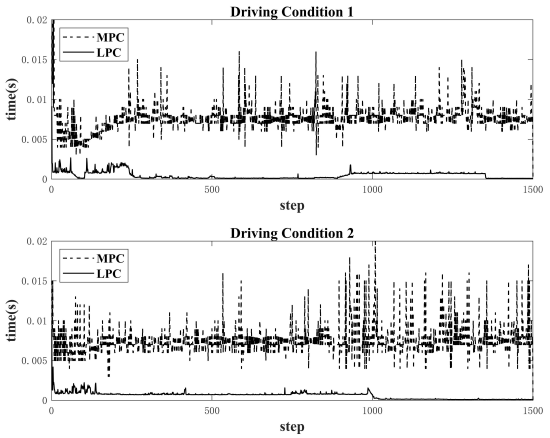


Fig. 4. Comparison of computational time

The simulation results show that total computational time of MPC in condition 1 is 11.487s,while LPC is 0.666s. In condition 2, total computational time of MPC is 11.328s, while LPC is 1.019s. These two sets of simulation experiments verify that the computational efficiency of LPC is much higher than MPC in the same driving condition.

In order to characterize the superiority of LPC compared with MPC, define the deviation E between actual and expected acceleration as:

$$E = \frac{\sum_{i=1}^{i_{max}} \left(\sum_{k=1}^{k_{max}} \left(a_{i,k} - a_{des,k} \right) \right)}{i_{max}} \tag{31}$$

where i_{max} is the number of the following vehicles, l_{max} is total time of the simulation. The results of acceleration error in two different driving conditions are as follows:

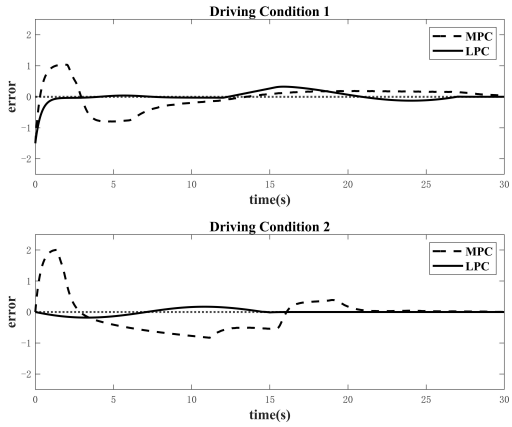


Fig. 5. Comparison of acceleration error

As shown in Fig.5, the tracking performance is improved, and the acceleration of the following vehicle based on LPC is more smoother.

6. CONCLUSION

In this paper, a learning-based predictive control was proposed for vehicle following problems. The feature vectors of actor-critic were established based on MPC

for vehicle following problems. Through continuous iterations of the kernel-based actor-network and critic-network, the LPC scheme was adopted to calculate the longitudinal control strategies in different driving conditions. Simulation results showed that LPC can obtain a better performance of policy optimality on the premise of the safety, and improve the computational efficiency.

REFERENCES

[1] Wang D, Yu R, Yan X, Hu J, Zhu Y, Cui S, "A review of the development of vehicle-following model," *Journal of Shandong University of Technology (Natural Science Edition)* (005), pp.036-042, 2022

[2] He Z, Xu R, Xie D, Zong F, "Review of data-driven following models," *Transportation systems engineering and Information*, pp.102-113, 2021.

[3] Wu B, Wang We, Li L, Liu Y, "Longitudinal control model of intelligent networked vehicle influenced by multiple vehicles," *Journal of Traffic and Transportation Engineering*, 20(2), pp.184-194, 2020.

[4] Du W, Li Y, Zhang J, Yu J, "Stability control for a class of vehicle- following model with time varying delay." *Control Theory and Applications*, 37(7), pp.1481-1490, 2020

[5] Yu, S. , Chen, H. , Feng, Y. , Zhang, Y. , Chen, H. , "Nash optimality based distributed model predictive control for vehicle platoon," *IFAC-PapersOnLine*, 53(2), pp.6610-6615, 2020

[6] Dominik, M. , Waschl, H. , Kirchsteiger, H. , Schmied, R. , Re, L. D. "Cooperative adaptive cruise control applying stochastic linear model predictive control strategies," *2015 European Control Conference (ECC)*, pp.3383-3388, 2015

[7] Xu, X. , Chen, H. , Lian, C. , Li, D. , "Learning-based predictive control for discrete-time nonlinear systems with stochastic disturbances," *IEEE Transactions on Neural Networks & Learning Systems*, pp.001-012, 2018

[8] Zhu Bing, Jiang Yuande, Zhao Jian, Chen Hong, Deng Weiwen. , "Vehicle following control based on deep reinforcement learning," *China Journal of Highway and Transport*, 32(6), pp.053-060, 2019

[9] Chu, T. , Wang, J. , L Codec, Li, Z. , "Multi-agent deep reinforcement learning for large-scale traffic signal control," *arXiv e-prints*, 2019

[10] Liu Fuchun, He Yun, Chen Yifeng. , "Collaborative Control algorithm and simulation for autonomous vehicles with time-delay mpc," *Computer Engineering and Applications*, 55(23), pp.222-227, 2019

[11] Stankovic, S. , Stanojevic, M. , Siljak, D. , "Decentralized overlapping control of a platoon of vehicles," *IEEE Transactions on Control Systems Technology*, 8(5), pp.816-832, 2000

[12] Zhu, M. , Wang, X. , Wang, Y. , "Human-like autonomous car-following model with deep reinforcement learning," *Transportation Research Part C: Emerging Technologies*, 97, pp.348-368, 2018

[13] Grondman, I. , Busoniu, L. , Lopes, G. , Babuska, R. , "A survey of actor-critic reinforcement learning: standard and natural policy gradients," *IEEE Transactions on Systems Man & Cybernetics Part C*, 42(6), pp.1291-1307, 2012

[14] Fu S, Jiang Z, & Zhang S. , "Optimization of vehicle formation Driving performance based on c-v2x Direct communication," *ZTE Communication Technology*, 26(1), pp.031-034, 2020

[15] Ma Y, Huang J, & Zhao H. , "Design of vehicle formation control Method based on Workshop communication," *Journal of Jilin University: Engineering and Technology Edition*, 50(2), pp.711-718, 2020

[16] Fadhloun, K. , & Rakha, H. , "A novel vehicle dynamics and human behavior car-following model: model development and preliminary testing", pp.014-028, 2019

[17] Chen Hao, "Distributed Predictive Control Strategy for Longitudinal Vehicle Queuing System," *Master dissertation, Jilin University*, 2021.

[18] Zhao J, Song D, Zhu B, Liu B, Chen Z, & Zhang P. , "Intelligent vehicle following control strategy based on self-learning and supervised learning hybrid drive," *China Journal of Highway and Transport*, 35(3), pp.055-065, 2022

[19] Shi, T., Ai, Y., Elsamadisy, O., & Abdulhai, B., "Bilateral deep reinforcement learning approach for better-than-human car following model," *arXiv e-prints:2203.04749*, 2022

[20] Hofmann, T. , SchoLkopf, B. , & Smola, A. J. , "Kernel methods in machine learning," *Annals of Statistics*, 36(3), pp.1171-1220, 2008

[21] Muller, K., Mika, S., Ratsch, G., Tsuda, K., & Scholkopf, B., "An introduction to kernel-based learning algorithms," *IEEE Transactions on Neural Networks*, 12(2), pp.181-201, 2001

[22] Grondman, I. , Busoniu, L. , Lopes, G. , & Babuska, R., "A survey of actor-critic reinforcement learning: standard and natural policy gradients," *IEEE Transactions on Systems Man & Cybernetics Part C*, 42(6), pp.1291-1307, 2012

[23] Peng, B., Mu, Y., Guan, Y., Li, S. E., Yin, Y., & Chen, J., "Model-based actor-critic with chance constraint for stochastic system", *2021 60th IEEE Conference on Decision and Control (CDC)*, 2020

[24] Wei H, Xu S, & Song W. , "Generalization Theory and generalization method of neural networks," *Acta Automatica Sinica*, 27(6), pp.806-815, 2001