MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Model-based deep reinforcement learning for CACC in mixed-autonomy vehicle platoons

Chu, T.; Kalabic, U.

TR2019-142 December 11, 2019

Abstract

This paper proposes a model-based deep reinforcement learning (DRL) algorithm for cooperative adaptive cruise control (CACC) of connected vehicles. Differing from most existing CACC works, we consider a platoon consisting of both human-driven and autonomous vehicles. The humandriven vehicles are heterogeneous and connected via vehicleto-vehicle (V2V) communication and the autonomous vehicles are controlled by a cloud-based centralized DRL controller via vehicle-to-cloud (V2C) communication. To overcome the safety and robustness issues of RL, the algorithm informs lowerlevel controllers of desired headway signals instead of directly controlling vehicle accelerations. The lower-level behavior is modeled according to the optimal velocity model (OVM), which determines vehicle acceleration according to a headway input. Numerical experiments show that the model-based DRL algorithm outperforms its model-free version in both safety and stability of CACC. Furthermore, we study the impact of different penetration ratios of autonomous vehicles on the safety, stability, and optimality of the CACC policy.

IEEE Conference on Decision and Control (CDC)

Model-based deep reinforcement learning for CACC in mixed-autonomy vehicle platoons

(Invited Paper)

Tianshu Chu Uroš Kalabić

***Abstract*— This paper proposes a model-based deep reinforcement learning (DRL) algorithm for cooperative adaptive cruise control (CACC) of connected vehicles. Differing from most existing CACC works, we consider a platoon consisting of both human-driven and autonomous vehicles. The human-driven vehicles are heterogeneous and connected via vehicle-to-vehicle (V2V) communication and the autonomous vehicles are controlled by a cloud-based centralized DRL controller via vehicle-to-cloud (V2C) communication. To overcome the safety and robustness issues of RL, the algorithm informs lower-level controllers of desired headway signals instead of directly controlling vehicle accelerations. The lower-level behavior is modeled according to the optimal velocity model (OVM), which determines vehicle acceleration according to a headway input. Numerical experiments show that the model-based DRL algorithm outperforms its model-free version in both safety and stability of CACC. Furthermore, we study the impact of different penetration ratios of autonomous vehicles on the safety, stability, and optimality of the CACC policy.**

I. Introduction

Vehicle platooning, or driving in grouped rows of vehicles, has been recognized as providing a significant social and environmental benefit as it can both significantly decrease fuel economy and increase road capacity. For this reason, effort has been devoted in both academia and industry to improving the autonomy, adaptivity, safety, and reliability of vehicular platoon control. Platoon control relies on cooperation between vehicles and, for this reason, research has focused on the development of cooperative adaptive cruise control (CACC) [1]. Relying on vehicle-specific wireless communication protocols, such as vehicle-to-vehicle (V2V) and vehicle-to-cloud (V2C), which allow for the real-time information sharing and control, CACC has the potential to improve traffic throughput and reduce incidences of collision [2], [3].

The majority of work on CACC focuses on developing robust and safe controllers, assuming all vehicles of the platoon are autonomous and connected. Some work considers a predecessor-following model [4], designing CACC for a two-vehicle system. Other work considers CACC for connected, multi-vehicle systems with more global, V2V communication structures [5], [6]. However, in the near future, it is expected that most road traffic will be a mixture of both autonomous and human-driven vehicles and it is therefore desirable to design CACC for mixed-autonomy, multi-vehicle system. Examples of such work can be found in [7]–[9], but most of the proposed approaches require solving optimal control problems online, which may not be efficient and scalable for real-time application. To address this challenge, in this paper we propose a purely data-driven reinforcement learning (RL) based approach.

As the joint area of machine learning and artificial intelligence, RL has had rapid and significant progress in recent years. RL was originally proposed in the control domain for optimal stochastic controls under uncertainties, within the framework of Markov decision processes (MDPs) [10]. Unlike traditional model-based optimization approaches, RL directly fits a parametric model to learn the optimal control policy, based on its experience interacting with the control system. Recently, deep neural networks (DNNs) have been successfully applied to enhance the learning capacity of RL, and the resulting deep reinforcement learning (DRL) algorithms have demonstrated breakthrough human-level performance on complex tasks like playing Go [11]. The common way to deploy DNN models is to maintain them on the cloud and provide output signals as API services [12], in order to protect the trained model and allocate sufficient resource for model inference. With the coming roll-out of 5G, the communication latency between cloud and edge devices will be reduced significantly and, considering this communication structure, this paper proposes a new CACC architecture where a centralized cloud-based DRL controller exchanges information with all connected autonomous vehicles via V2C communication.

Recent studies of RL-based CACC include [13], which was the first to apply a policy gradient method for CACC but results in oscillatory behavior due to the discrete longitudinal control; [14], which applies a policy iteration method to learn parameters of a classical proportional-integral (PI) controller instead of direct longitudinal control; [15], which proposes an informative reward design to ensure safety and robustness of the Q-learning method; and [16], which applies deep-deterministic policy gradient (DDPG) to learn the continuous longitudinal control with predicted leading vehicle trajectories. Most of these works consider a predecessor-following problem in a two-vehicle system rather than a multi-vehicle setting. Furthermore, they do not guarantee constraints and rely heavily on the correctness of input signals, which may lead to safety and robustness concerns.

In this work, we design a model-based DRL scheme for CACC. The DRL controller, instead of directly controlling

This work was supported by Mitsubishi Electric Research Laboratories.

T. Chu is with Stanford University, Palo Alto, CA 94305, USA. Email: cts198859@hotmail.com

U. Kalabić is with Mitsubishi Electric Research Laboratories, Cambridge, MA 02139, USA. Email: kalabic@merl.com

autonomous vehicle accelerations, informs the parameters of an optimal velocity model (OVM) [17], which in turn is used to determine the required acceleration to reach the optimal velocity according to the model. Our choice is informed by results found in [15], that controlling acceleration does not allow RL to explore a safe policy; in contrast, learning an input to a model, which by itself provides some safety guarantees, removes the need for us to deliver the same guarantees using RL. This idea is similar to that found in [14], where the authors use RL to determine desired velocity and parameters of a PI controller. Our DRL controller uses a DDPG approach to learn the desired control. Numerical simulations are performed comparing our model-based DDPG approach to a model-free DDPG one. We show that the model-based approach is superior, providing convergence to a locally optimal control policy as well as stability of the platoon.

The rest of the paper is organized as follows. In Section II, we present the system model. In Section III, we formulate the CACC problem. In Section IV, we introduce our model-based DRL approach. In Section V, we provide numerical results. Section VI is the conclusion.

II. System model

We consider a platoon or set of connected vehicles $\mathcal{V}$, traveling on a straight road. The platoon consists of both human-driven and autonomous vehicles so that $\mathcal{V} = \mathcal{V}_h \cup \mathcal{V}_a$ where $\mathcal{V}_h$ is the set of human-driven vehicles and $\mathcal{V}_a$ is the set of autonomous vehicles. Autonomous vehicles are controlled by a cloud-based DRL controller through V2C communication, while human-driven vehicles only transmit state information via V2V communication.

A. Vehicle dynamics

Given a vehicle $i \in \mathcal{V}$, we denote its headway, *i.e.,* bumper-to-bumper distance between i and its leading vehicle $i-1$, by h_i, its velocity by v_i, and its acceleration by u_i. The vehicle dynamics are given by,

$$\begin{aligned} \dot{h}_i &= v_{i-1} - v_i, \\ \dot{v}_i &= u_i, \end{aligned}$$

which we discretize as,

$$h_{i,t+1} = h_{i,t} + \int_t^{t+\Delta t} (v_{i-1,\tau} - v_{i,\tau}) d\tau, \tag{1a}$$

$$v_{i,t+1} = v_{i,t} + u_{i,t}\Delta t, \tag{1b}$$

with sampling time Δt. To ensure comfort and safety, the following constraints are applied to each vehicle,

$$h_{\min} \leq h_{i,t}, \tag{2a}$$

$$0 \leq v_{i,t} \leq v_{\max}, \tag{2b}$$

$$u_{\min} \leq u_{i,t} \leq u_{\max}, \tag{2c}$$

$h_{\min}$ is the minimum safe headway, $v_{\max}$ is the speed limit, and $u_{\min} < 0$ and $u_{\max} > 0$ are deceleration and acceleration limits, respectively.

The car-following behavior of each human-driven vehicle $i \in \mathcal{V}_h$ is modeled using the OVM,

$$u_{i,t} = \alpha_i \left(v^o(h_{i,t}; h^s, h^g) - v_{i,t} \right) + \beta_i (v_{i-1,t} - v_{i,t}), \tag{3}$$

where α_i and β_i are headway gain and relative velocity gain for each human driver and v^o is a headway-based velocity policy:

$$v^o(h) = \begin{cases} 0 & \text{if } h \leq h^s, \\ \frac{1}{2} v_{\max} \left(1 - \cos\left(\pi \frac{h - h^s}{h^g - h^s} \right) \right) & \text{if } h^s < h < h^g, \\ v_{\max} & \text{if } h \geq h^g, \end{cases} \tag{4}$$

where h^s is the stop headway and h^g is the full-speed headway.

B. Communication between vehicles

We assume that each human-driven vehicle $i \in \mathcal{V}_h$ is able to send its headway h_i, velocity v_i, and acceleration u_i, to any nearby vehicle within range D. We define $\hat{\mathcal{V}}_{h,t} \subset \mathcal{V}_h$ as a group of human-driven vehicles whose states are accessible by an autonomous vehicle, based on the vehicle position x at time t,

$$\hat{\mathcal{V}}_{h,t} = \cup_{i \in \mathcal{V}_a} \{ j \in \mathcal{V}_h : |x_{j,t} - x_{i,t}| \leq D \}. \tag{5}$$

We also assume that all autonomous vehicles are able to send their state and receive control signals from the central, cloud-based controller.

Let $s_{i,t}$, $i \in \mathcal{V}$, denote each vehicle's state,

$$s_{i,t} = \begin{bmatrix} h_{i,t} \\ v_{i,t} \\ u_{i,t} \end{bmatrix}.$$

We design a control policy μ to determine the acceleration inputs $u_{i,t}$ of all autonomous vehicles $i \in \mathcal{V}_a$. The control inputs are given by,

$$\mathbf{u}_{a,t} = \mu(\hat{\mathbf{s}}_t), \tag{6}$$

where $\mathbf{u}_{a,t}$ is the vector of all autonomous vehicles states,

$$\mathbf{u}_{a,t} = \left[u_{i,t} \right]_{i \in \mathcal{V}_a},$$

and $\hat{\mathbf{s}}_t$ is the vector of all vehicle states,

$$\hat{\mathbf{s}}_t = \left[\hat{s}_{i,t} \right]_{i \in \mathcal{V}},$$

with $\hat{s}_{i,t} = s_{i,t}$ if $i \in \mathcal{V}_a \cup \hat{\mathcal{V}}_{h,t}$ and 0 otherwise. Note that, since the RL algorithm must have a fixed dimension (in this case $3|\mathcal{V}|$), we must choose a value for the set the states of out-of-range vehicles $\mathcal{V}_h \setminus \hat{\mathcal{V}}_{h,t}$.

In Fig. 1 we present an illustration of the communication between vehicles and the cloud-based DRL controller. In the illustration, we show that each autonomous vehicle collects states of nearby human-driven vehicle within V2V communication range D, that the cloud-based DRL controller collects states from all autonomous vehicles and that it outputs a control action to the same vehicles, and that each human-driven or autonomous vehicle performs a longitudinal control based on either the OVM or the received command, respectively.

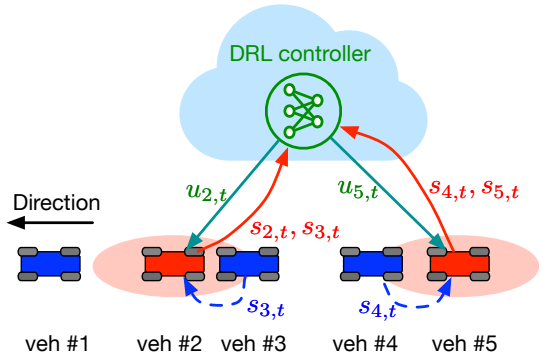

Fig. 1. Illustration showing a platoon of connected vehicles; blue vehicles are human-driven and red vehicles are autonomous; the V2V communication range D is illustrated by a red ellipse; V2V and V2C communications are shown as dashed and solid arrows, respectively, inheriting their color from the sender

III. CACC

The common goal of CACC is to ensure plant- and string-stability of the controlled platoon for a desired headway. A platoon is plant stable if all vehicles approach the constant velocity of their leading vehicle; it is string stable if disturbances in velocity are attenuated along the entire platoon.

To achieve this, we set the objective as the minimization of the mean-squared errors of velocity and headway with respect to a desired velocity v_t^* and headway h^* with a quadratic penalty on control, *i.e.,* we maximize,

$$\bar{V}(\mu) = \frac{1}{T}\sum_{t=1}^{T} \bar{r}_t, \tag{7}$$

subject to (1)-(3), (6), for all $t = 1, 2, \ldots, T$, where T is the planning horizon and,

$$\bar{r}_t = -\frac{1}{|\mathcal{V}|}\sum_{i\in\mathcal{V}} \bar{c}_{i,t}, \tag{8}$$

with $\bar{c}_{i,t} = \left((h_{i,t+1} - h^*)^2 + a(v_{i,t+1} - v_t^*)^2 + bu_{i,t}^2\right)$, is the step-wise reward, where a and b are weights corresponding to plant and string stability, respectively.

Since the dynamics of human-driven vehicles are nonlinear and depend on unknown parameters, such as α_i and β_i, $i \in \mathcal{V}_h$, it is impossibly difficult to determine μ analytically. For this reason, we pursue an RL approach, which attempts to determine the optimal policy statistically, from explored experience of underlying system dynamics.

RL is formulated as an MDP with the definition of state, action, transition, and reward. The state $s_t \in \mathcal{S}$ contains all accessible information for decision-making $\hat{\mathbf{s}}_t$. In model-free RL, the action $a_t \in \mathcal{A}$ is simply the centralized longitudinal control $\mathbf{u}_{a,t}$. However, learning a safe and robust longitudinal control is challenging due to the data-driven nature of RL. We therefore apply RL to learn adaptive high-level car-following strategies, while the actual acceleration is still controlled by OVM. Specifically, the action is the full-speed headway recommendation $a_t := [h_{i,t}^g]_{i\in\mathcal{V}_a}$, and the longitudinal control of each autonomous vehicle $i \in \mathcal{V}_a$ is,

$$u_{i,t} = \bar{\alpha}\left(v^o(h_{i,t}; h^s, h_{i,t}^g) - v_{i,t}\right) + \bar{\beta}(v_{i-1,t} - v_{i,t}), \tag{9}$$

where $\bar{\alpha}$ and $\bar{\beta}$ are design parameters. To improve exploration efficiency, we bound the action space to $\mathcal{A} = [h_{\min}^g, h_{\max}^g]^{|\mathcal{V}_a|}$, where $h_{\min}^g < h_{\max}^g$ are minimum and maximum full-speed headways.

The system transition is defined by (1) and (3) and the control (9) and it is Markovian. Control constraints are directly applied to each control $u_{i,t}$. Headway constraints (2a) may easily become violated due to human behavior or unsafe RL policy and therefore, to make RL become aware of safety constraints, we follow [15] to represent these as part of the reward according to,

$$r_t = \bar{r}_t + \frac{c}{|\mathcal{V}|}\sum_{i\in\mathcal{V}} \left(\min\{h_{i,t+1} - h_s, 0\}\right)^2, \tag{10}$$

where c is a weighting parameter. The above holds as long as $h_{i,t} \geq h_{\min}$ for all $i \in \mathcal{V}$. If the headway constraint is violated for any $i \in \mathcal{V}$ in training, we set $r_t = -G$ for some large G and terminate training early.

RL learns the optimal stationary policy $\pi_\theta : \mathcal{S} \to \mathcal{A}$ to maximize the discounted return,

$$V(\pi_\theta) = \sum_{t=1}^{T} \gamma^{t-1} r_t, \ 0 < \gamma < 1. \tag{11}$$

We use θ to denote a trainable parametric model such as a DNN.

IV. DRL for CACC

In this section, we introduce and describe details of the model-based DRL algorithm applied to CACC and training strategies.

A. DRL algorithm

To learn the CACC policy, we use DDPG as it has been widely applied to robotic continuous control and its effectiveness has been verified in various applications. DDPG trains both an actor network to learn the optimal policy π_θ and a critic network to learn the corresponding return (Q-value) estimate $Q_w : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. As many off-policy RL algorithms, DDPG maintains an experience replay buffer to store recently explored experience for sample-efficient training. Instead of random exploration like an $\epsilon-$greedy, it adopts the Ornstein-Uhlenbeck (OU) process to achieve temporally correlated exploration [18]. To ensure that the explored action remains inside the action space, for every $i \in \mathcal{V}_a$, we saturate the action between $h_{\min}^g$ and $h_{\max}^g$,

$$h_{i,t}^g = \text{clip}\{h_{\min}^g, \pi_\theta(s_t)[i] + \epsilon^o(t; \theta^o, \sigma^o), h_{\max}^g\}, \tag{12}$$

where the noise ϵ^o is sampled from the OU process with parameters θ^o and σ^o.

We provide the algorithm pseudo-code in Algorithm 1. To differentiate our algorithm from the model-free DDPG-based CACC algorithm, we refer to it as DDPG-OVM. At each step, a full-speed headway recommendation is first explored based on the current state (line 6), then the corresponding

Algorithm 1: DDPG-OVM based CACC

1 **initialize** DDPG model θ, w and replay buffer $\mathcal{D}$;
2 **for** *each training episode* **do**
3 **initialize** $s_1 = s$, $t = 1$;
4 **while** $t \leq T$ **do**
5 **observe** state s_t;
  `/* OU exploration */`
6 **explore** $a_t = [h^g_{i,t}]_{i\in\mathcal{V}_a}$ by (12);
  `/* vehicle control */`
7 **get** $u_{i,t}$ by (9) for all $i \in \mathcal{V}_a$;
8 **get** $u_{i,t}$ by (3) for all $i \in \mathcal{V}_h$;
9 **perform** for all $i \in \mathcal{V}$ constrain $u_{i,t}$ by (2c);
  `/* experience collection */`
10 **observe** updated state s_{t+1} by (1);
11 **compute** compute r_t by (10);
  `/* model update */`
12 **update** $\mathcal{D} \leftarrow \mathcal{D} \cup \{e_t := (s_t, a_t, s_{t+1}, r_t)\}$;
13 **sample** minibatch $\mathcal{D}_m := \{e_\tau\}$ from $\mathcal{D}$;
14 **update** θ, w based on gradient from $\mathcal{D}_m$;
15 **if** $r_t = -G$ **then**
16 **break**;
17 **end**
18 **update** $t \leftarrow t + 1$;
19 **end**
20 **end**

constrained longitudinal control is performed by each vehicle (lines 7-9). Next, the system transfers to another state and the corresponding reward signal is collected (lines 10 and 11). Note the human parameters α_i and β_i, $i \in \mathcal{V}_h$ are known to the system but not provided to the agent and therefore not included in the state. Finally, the agent attaches this new experience to the replay buffer and samples a minibatch of experiences from it to update the model parameters (lines 12-14). The gradient update procedure we use can be found in [19]. Note that (2a) may be violated during the exploration and that the corresponding experience is accumulated for the agent to learn how to avoid headway constraint violation. If violation occurs, the episode is terminated (line 15-17).

B. DRL training

1) DNN settings: We adopt the DNN structure proposed in [19] and illustrated in Fig. 2. The DNN contains two hidden layers of rectified non-linearity with 400 and 300 units respectively. The final layer of the critic network is linear with a scalar output of the Q-value, and the final layer of the actor network is `tanh` with a $|\mathcal{V}_a|-$dimensional vector output of bounded headway recommendations. The actions are not included until the second hidden layer of the critic network. Default hyper-parameters are used for training DNN weights: the learning rates are 10^{-4} and 10^{-3} for actor and critic networks, respectively, and the critic network has a 10^{-2}-weighted L^2-norm weight regularization. Finally, the global gradient norm is clipped at 40 for stabilizing the update.

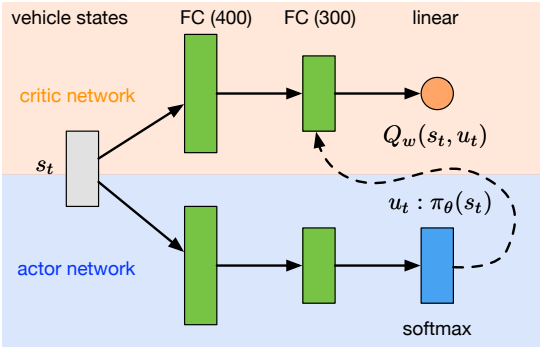

Fig. 2. DDPG actor and critic DNN structures, with different layer types in different colors

State normalization is important in DNN-training as the scale of input signal is maintained when it is passed through a DNN. To provide reward-related information after normalization, acceleration is normalized according to $-u/u_{\min}$ if $u < 0$ and $u/u_{\max}$ otherwise, velocity is normalized according to $3(v - v_t^*)/v_t^*$, and headway is normalized according to $(h - h^*)/h^*$. All normalized states are clipped between -2 and 2 to prevent outliers.

2) Training settings: Default hyper-parameters are used for DDPG training, *i.e.,* the discount factor $\gamma = 0.99$ and the OU process $\theta^o = 0.15, \sigma^o = 0.2$. The reward coefficients are $a = 1$, $b = 0.1$, $c = 5$, and $G = 1000$. Considering ordinary V2V communication [20], whose range and sampling time are $D = 40$m and $\Delta t = 0.2$s, respectively. The DDPG model is trained over 10^6 steps, and each episode simulates the vehicle dynamics of a CACC platoon for 2 minutes, *i.e.,* $T = 600$ steps. We use the following constraint parameters [4], [7]: $u_{\min} = -2.5\text{m/s}^2$, $u_{\max} = 2.5\text{m/s}^2$, $v_{\max} = 30\text{m/s}$, $h_{\min} = 2\text{m}$, $h_s = 5\text{m}$, $h_g = 35\text{m}$. The output of the actor network is scaled so that $\pi_\theta(s_t) \in [10\text{m}, 60\text{m}]$, *i.e.,* $h^g_{\min} = 10\text{m}$ and $h^g_{\max} = 60\text{m}$.

V. Numerical Simulations

A. Setup

Noting that recent demonstrations of CACC [5]–[7] perform simulation of between five and ten connected vehicles, we simulate a platoon of 8 vehicles, excluding the leading vehicle, with half of them set to be autonomous vehicles running at odd positions, *i.e.,* we choose $\mathcal{V}_a = \{1, 3, 5, 7\}$ and $\mathcal{V}_h = \{2, 4, 6, 8\}$. For human-driven vehicles, we set differing OVM parameters $\alpha_2 = 0.4$, $\beta_2 = 0.4$, $\alpha_4 = 0.3$, $\beta_4 = 0.5$, $\alpha_6 = 0.3$, $\beta_6 = 0.4$, $\alpha_8 = 0.5$, and $\beta_8 = 0.5$; for autonomous vehicles the design OVM parameters are chosen as $\bar{\alpha} = \bar{\beta} = 0.4$.

As has been done in [7], we investigate how CACC helps the platoon catch up to the head vehicle. Specifically, the desired velocity profile is $v_t^* = v^* = 15\text{m/s}$ and the desired headway is $h^* = 20\text{m}$. In the initial state, all vehicles are already driving at v^* and h^*, except for the first vehicle of the platoon whose headway is $h_{1,1} = 80\text{m}$. Therefore the CACC task is to control the platoon to transit to another steady state

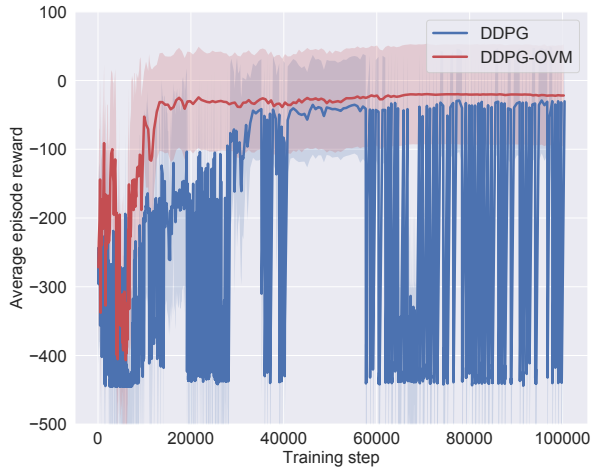

Fig. 3. Learning performance of DDPG and DDPG-OVM over 10^6 training steps; curve and shade indicate the average and standard deviation of rewards per training episode, respectively

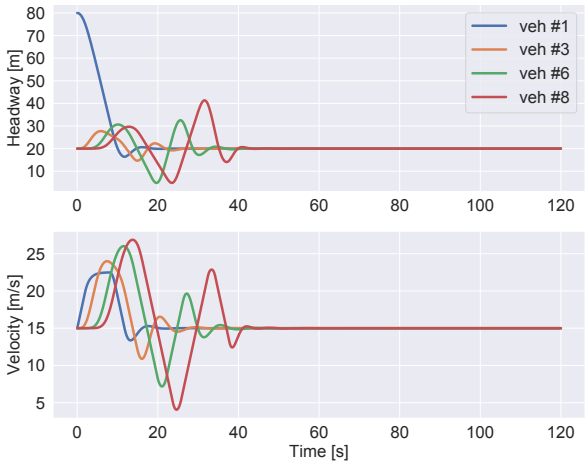

Fig. 4. Headway and velocity trajectories of selected vehicles, where all vehicles are controlled by OVM

where it is closer to the head vehicle, *i.e.,* $h_{1,t} = h^*$, while ensuring plant and string stability during transition.

B. Learning performance

We compare the learning performance of the DDPG-OVM algorithm against a model-free DDPG algorithm that directly learns the longitudinal control of autonomous vehicles, *i.e.,* $\pi_\theta(s_t) = \mathbf{u}_t$. Model-free DDPG was demonstrated to have good performance for car-following in a two-vehicle system recently [16], so we are interested in how well it will perform in a complex multi-vehicle system with unknown nonlinear dynamics of human vehicles. Both DDPG and DDPG-OVM are trained with identical settings, and Fig. 3 plots the average reward per training episode $\frac{1}{T}\sum_{t=1}^{T} r_t$ against the training step. Note that, even though this is the reward of the behavior policy (12), it is correlated to that of the target policy π_θ. In a successful learning process, the training reward curve first increases as RL improves an initial random policy from explored experiences, then it becomes stable as RL converges to a local optimal policy. The results show that DDPG-OVM learns more efficiently and converges before $2 \cdot 10^5$ steps, while DDPG takes a longer time to achieve a similar performance at around $4 \cdot 10^5$ steps and is not able to converge at the end. This implies that it is challenging to learn stable longitudinal control directly in complex systems, and a model-based DRL may help by restricting the action space.

C. Evaluation

Now we evaluate the inference performance of the trained DDPG-OVM model. We are interested in the performance of a baseline CACC system where autonomous vehicles are also controlled by OVM using (3) instead of (9). Alternatively, this system can be considered as a platoon of purely human-driven vehicles with enabled OVM-based ACC. Fig. 4 shows the headway and velocity trajectories of selected vehicles of the OVM-controlled system, with the average reward -32.09. We can see it takes about 40s for the platoon to catch up to the head vehicle and reach the steady state h^* and v^*. Plant stability is achieved as all vehicles eventually drive at the desired velocity v^*, despite accelerations and decelerations in the transition. However, string stability is violated, since whenever the first vehicle changes its velocity, the fluctuation is amplified though the platoon. Specifically, the velocity of the last vehicle ($v_{8,t}$) varies between 5m/s to 25 m/s, and its headway ($h_{8,t}$) drops below 5m, leading to safety and stability concerns.

We evaluate the same CACC system with autonomous vehicles controlled by the trained DDPG-OVM model. Fig. 4 shows the headway and velocity trajectories from the same vehicles, with average reward -20.59. We can see the platoon reaches h^* and v^* much faster, within 20s. Furthermore, string stability is achieved since all other vehicles follow nearly the same velocity profile as that of the first vehicle. This can also be verified in the plot of headway trajectories, where all other vehicles are able to maintain a headway close to h^* during transition. Interestingly, not only autonomous vehicles but also human-driven vehicles $\{6, 8\}$ are able to achieve a safe and stable longitudinal control. This implies that the DDPG-OVM model is able to learn the behavior of unknown human drivers and optimize CACC over the entire platoon, based on its experience interacting with such a system.

We perform a final simulation to evaluate the CACC system where all vehicles are controlled by DDPG-OVM. With the settings above, the agent fails to learn due to the challenges in exploring a both safe and optimal policy. To address this issue, we reduce the action space size so that the agent is restricted to explore more within the safe OVM policy. Specifically, we set $h^g_{\min} = 20$m and $h^g_{\max} = 50$m. Fig. 6 shows the headway and velocity trajectories under the learned policy. We can see similar plant and string stabilities are achieved in this system, though a higher diversity is observed among velocity trajectories of different vehicles.

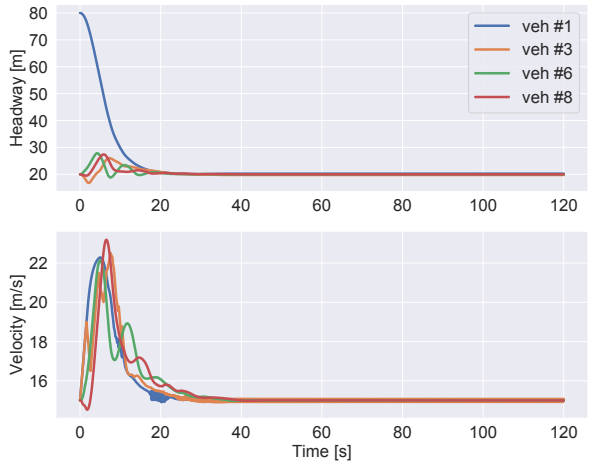

Fig. 5. Headway and velocity trajectories of selected vehicles, where autonomous vehicles are controlled by DDPG-OVM

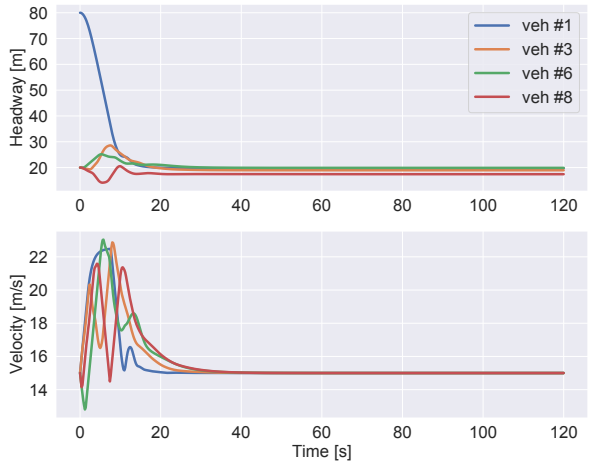

Fig. 6. Headway and velocity trajectories of selected vehicles when all vehicles are controlled using DDPG-OVM

Experiments in this catch-up scenario imply that CACC-enabled autonomous vehicles can improve the plant and string stabilities of a platoon. However, beyond a certain threshold, further increase in the penetration ratio of autonomous vehicles may not steadily improve performance; it may instead introduce safety concerns from the learning-based CACC policies.

VI. Conclusion

In this paper, we presented a model-based DRL approach to the CACC of vehicle platoons consisting of both human-driven and autonomous vehicles. We designed a DRL controller based on a DDPG approach which, instead of directly controlling vehicle accelerations, determines headway parameters of an OVM model. Numerical simulation results show that the model-based approach is superior to a model-free approach, with the former exhibiting convergence to a local optimum as well as stability.

References

[1] L. Güvenç, I. M. C. Uygan, K. Kahraman, R. Karaahmetoglu, I. Altay, M. Sentürk, M. T. Emirler, A. E. H. Karci, B. A. Güvenç, E. Altug, M. C. Turan, O. S. Tas, E. Bozkurt, U. Özgüner, K. Redmill, A. Kurt, and B. Efendioglu, "Cooperative adaptive cruise control implementation of Team Mekar at the Grand Cooperative Driving Challenge," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 3, pp. 1062–1074, 2012.

[2] B. van Arem, C. J. G. van Driel, and R. Visser, "The impact of cooperative adaptive cruise control on traffic-flow characteristics," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 4, pp. 429–436, 2006.

[3] K. C. Dey, L. Yan, X. Wang, Y. Wang, H. Shen, M. Chowdhury, L. Yu, C. Qiu, and V. Soundararaj, "A review of communication, driver characteristics, and controls aspects of cooperative adaptive cruise control (CACC)," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 2, pp. 491–509, 2016.

[4] C. Massera Filho, M. H. Terra, and D. F. Wolf, "Safe optimization of highway traffic with robust model predictive control-based cooperative adaptive cruise control," *IEEE Trans. Intell. Trasnp. Syst.*, vol. 18, no. 11, pp. 3193–3203, 2017.

[5] S. Gong, A. Zhou, J. Wang, T. Li, and S. Peeta, "Cooperative adaptive cruise control for a platoon of connected and autonomous vehicles considering dynamic information flow topology." arXiv:1807.02224, 2018.

[6] A. M. H. Al-Jhayyish and K. W. Schmidt, "Feedforward strategies for cooperative adaptive cruise control in heterogeneous vehicle strings," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 1, pp. 113–122, 2018.

[7] W. Gao, Z.-P. Jiang, and K. Ozbay, "Data-driven adaptive optimal control of connected vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 5, pp. 1122–1133, 2017.

[8] I. G. Jin and G. Orosz, "Dynamics of connected vehicle systems with delayed acceleration feedback," *Transport Res. C: Emer.*, vol. 46, pp. 46–64, 2014.

[9] C. Wu, A. M. Bayen, and A. Mehta, "Stabilizing traffic with autonomous vehicles," in *Proc. IEEE Int. Conf. Robotics and Automation*, (Brisbane, Australia), pp. 6012–6018, 2018.

[10] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 2nd ed., 2018.

[11] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[12] B. Kehoe, S. Patil, P. Abbeel, and K. Goldberg, "A survey of research on cloud robotics and automation," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 2, pp. 398–409, 2015.

[13] C. Desjardins and B. Chaib-Draa, "Cooperative adaptive cruise control: A reinforcement learning approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 4, pp. 1248–1260, 2011.

[14] J. Wang, X. Xu, D. Liu, Z. Sun, and Q. Chen, "Self-learning cruise control using kernel-based least squares policy iteration," *IEEE Trans. Control Syst. Technol.*, vol. 22, no. 3, pp. 1078–1087, 2014.

[15] Z. Li, T. Chu, I. V. Kolmanovsky, and X. Yin, "Training drift counteraction optimal control policies using reinforcement learning: An adaptive cruise control example," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 9, pp. 2903–2912, 2018.

[16] M. Buechel and A. Knoll, "Deep reinforcement learning for predictive longitudinal control of automated vehicles," in *Proc. Int. Conf. Intell. Transp. Syst.*, (Maui, HI), pp. 2391–2397, 2018.

[17] M. Bando, K. Hasebe, A. Nakayama, A. Shibata, and Y. Sugiyama, "Dynamical model of traffic congestion and numerical simulation," *Phys. Rev. E*, vol. 51, no. 2, pp. 1035–1042, 1995.

[18] G. E. Uhlenbeck and L. S. Ornstein, "On the theory of the brownian motion," *Physical review*, vol. 36, no. 5, p. 823, 1930.

[19] T. Lillicrap, J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning." arXiv:1509.02971, 2015.

[20] A. Rauch, F. Klanner, and K. Dietmayer, "Analysis of V2X communication parameters for the development of a fusion architecture for cooperative perception systems," in *Proc. IEEE Intell. Vehicles Symp.*, (Baden-Baden, Germany), pp. 685–690, 2011.