Multitimescale Control and Communications With Deep Reinforcement Learning—Part II: Control-Aware Radio Resource Allocation

Lei Lei, *Senior Member, IEEE*, Tong Liu, *Graduate Student Member, IEEE*, Kan Zheng, *Fellow, IEEE*, and Xuemin Shen, *Fellow, IEEE*

***Abstract*— In Part I of this two-part paper (Multitimescale Control and Communications with deep reinforcement learning (DRL)—Part I: Communication-Aware Vehicle Control), we decomposed the multitimescale control and communications (MTCCs) problem in cellular vehicle-to-everything (C-V2X) system into a communication-aware DRL-based platoon control (PC) subproblem and a control-aware DRL-based radio resource allocation (RRA) subproblem. We focused on the PC subproblem and proposed the MTCC-PC algorithm to learn an optimal PC policy given an RRA policy. In this article (Part II), we first focus on the RRA subproblem in MTCC assuming a PC policy is given, and propose the MTCC-RRA algorithm to learn the RRA policy. Specifically, we incorporate the PC advantage function in the RRA reward function, which quantifies the amount of PC performance degradation caused by observation delay. Moreover, we augment the state space of RRA with PC action history for a more well-informed RRA policy. In addition, we utilize reward shaping and reward backpropagation prioritized experience replay (RBPER) techniques to efficiently tackle the multiagent and sparse reward problems, respectively. Finally, a sample- and computational-efficient training approach is proposed to jointly learn the PC and RRA policies in an iterative process. In order to verify the effectiveness of the proposed MTCC algorithm, we performed experiments using real driving data for the leading vehicle, where the performance of MTCC is compared with those of the baseline DRL algorithms.**

***Index Terms*—Deep reinforcement learning (DRL), multi-timescale decision-making, radio resource allocation (RRA).**

I. Introduction

IN PART I of this two-part paper, we introduce the problem of multitimescale control and communications (MTCCs) in cellular vehicle-to-everything (C-V2X) systems, where two types of decisions need to be made at different timescales: 1) platoon control (PC) decisions that are made with a coarse time grid of every T ms control interval and 2) radio resource allocation (RRA) decisions that are made with a fine time grid of every 1 ms communication interval. PC aims to determine the control inputs for following vehicles so that all vehicles move at the same speed while maintaining the desired distance between each pair of preceding vehicle (i.e., predecessor) and following vehicle (i.e., follower) [1]. The C-V2X communications enable information exchange between vehicles so that more informed PC decisions can be made to reduce the intervehicle distance while guaranteeing string stability. In the C-V2X system, vehicle-to-vehicle (V2V) links coexist with vehicle-to-infrastructure (V2I) links. A V2I link connects a vehicle to the base station (BS) and is used for high-throughput services, and a V2V link connects a pair of predecessor and follower for periodic transmission of collaborative adaptive message (CAM) according to the predecessors following (PF) information typology (IFT) [2]. The PC decisions are made at each follower with the target of optimizing its PC performance, which is affected by the observation delay that depends on the RRA decisions of the C-V2X system. The RRA decisions, including subchannel allocation and power control, are made at each predecessor with the targets of 1) maximizing the V2I throughput and 2) minimizing the PC performance degradation due to delayed observation. The two targets are contradictory with each other and an optimal tradeoff should be struck. The tradeoff heavily depends on the impact of observation delay on PC performance, which in turn is affected by the PC decisions.

The interplay between RRA and PC necessitates the collaborative design of communications and control functions. RRA in C-V2X systems should be control-aware, taking into account the control performance degradation due to the delay or packet loss for control-related information delivery. Meanwhile, PC should be communication-aware, considering the statistical properties of random delay and packet loss in C-V2X communications. Both PC and RRA are Sequential stochastic decision problem (SSDP), where a sequence of decisions have to be made over a specific time horizon for a dynamic system whose states evolve in the face of uncertainty. We believe that tackling the SSDP under a unified deep reinforcement learning (DRL) framework will better reveal the interdependency between control and communications, and thus facilitate the joint optimization task [3].

Since employing the full-space approach to solve the multitimescale SSDP yields formidable computation complexity,

Manuscript received 23 September 2023; revised 16 November 2023; accepted 21 December 2023. Date of publication 29 December 2023; date of current version 25 April 2024. This work was supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada under Discovery Grant. *(Corresponding author: Lei Lei.)*

Lei Lei is with the School of Engineering, University of Guelph, Guelph, ON N1G 2W1, Canada (e-mail: leil@uoguelph.ca).

Tong Liu is with the Intelligent Computing and Communication Lab, Key Laboratory of Universal Wireless Communications, Ministry of Education, Beijing University of Posts and Telecommunications, Beijing 100876, China.

Kan Zheng is with the College of Electrical Engineering and Computer Sciences, Ningbo University, Ningbo 315211, China.

Xuemin Shen is with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo ON N2L 3G1, Canada.

Digital Object Identifier 10.1109/JIOT.2023.3348594

we decompose the MTCC problem into two subproblems, i.e., 1) communication-aware DRL-based PC and 2) control-aware DRL-based RRA. In Part I of this two-part paper, we have studied the communication-aware DRL-based PC assuming an RRA policy is given, and proposed the MTCC-PC algorithm to learn the PC policy. The PC problem with observation delay is essentially a Random Delay Decentralized Partially Observable Markov decision process (MDP). Each follower is a PC agent, which makes a local and delayed observation at each control interval and decides on its local actions to maximize its expected cumulative individual reward. We augment the PC state space with the observation delay and PC action history, and define the reward function for the augmented state to construct an augmented state MDP. It is proved that the optimal policy for the MDP is optimal for the PC problem with delayed observation. We show by experiments that the proposed MTCC-PC algorithm outperforms the state-of-the-art communication-aware control by training in a delayed environment generated by fine-grained embedded simulation of C-V2X communications rather than by a simple stochastic delay model.

This article is Part II of the two-part paper, in which we first focus on the control-aware DRL-based RRA problem assuming a PC policy is available, and propose the MTCC-RRA algorithm to learn the RRA policy. Different from most existing works on RRA in C-V2X systems [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], the RRA decisions are optimized taking into account the impact of observation delay on PC performance. The main contributions of this article (Part II) are summarized as follows.

1) *DRL-Based Value of Information (VoI) Estimation and Optimization:* A systematic method is proposed for the DRL-based PC agent of each follower to estimate the VoI of any given information that are to be shared by its predecessor, where the VoI is leveraged by the DRL-based RRA agents of the predecessors to make optimal RRA decisions. First, the PC performance degradation due to observation delay for each follower is quantified by the performance difference between the de-facto PC policy with observation delay and the optimal policy with undelayed observation. Then, the optimization objective of RRA is formulated by incorporating the performance degradation term. To solve such an optimization problem, it is rigorously proved that the advantage function of the optimal PC policy should be included in the RRA reward function. Moreover, the PC action history has to be included in the RRA state space to make the RRA policy more well-informed. One important advantage of this design is that the RRA decisions are made based on "VoI per control interval," i.e., the advantage function of the optimal PC policy, which provides a finer-grained VoI compared with the existing VoI calculation methods.
2) *Efficient DRL Solution Addressing Multiagent and Sparse Reward Problems:* Since the RRA agents cooperatively optimize the global expected cumulative reward, and it is hard for each agent to deduce its own contribution to the global reward, the simple independent learner (IL) approach used in PC is not suitable for the RRA problem falling into the multiagent domain. To solve this problem, we apply the reward shaping technique to design an individual RRA reward for each RRA agent. In addition, since the RRA rewards that reflect the PC performance degradation are only provided at the last communication interval of each control interval, it is difficult to learn an efficient RRA policy due to the sparse reward problem coupled with long-horizon planning. To solve this problem, we use the reward backpropagation prioritized experience replay (RBPER) technique [15] to improve training efficiency.
3) *Sample- and Computational-Efficient Training Approach:* Since it is impossible to learn the optimal PC policy or RRA policy first without knowing the other optimal policy due to the interactions between PC and RRA, we design a sample- and computational-efficient training approach to jointly train MTCC-PC and MTCC-RRA in an iterative process to learn the optimal PC and RRA policies.

The remainder of this article is organized as follows. We recall the system model for the MTCC problem and the MTCC-PC algorithm in Section II. Subsequently, Sections III and IV introduce the control-aware DRL model for RRA and the MTCC-RRA algorithm. Then, a joint optimization algorithm for MTCC is presented in Section V. Section VI conducts the performance experiments, demonstrating that our proposed algorithm is superior to the baseline algorithms. Finally, Section VII concludes this article.

II. System Model and Previous Results

A. Multitimescale Decision-Making Framework

Consider a platoon with a number of $N > 2$ vehicles, i.e., $\mathcal{V} = \{0, 1, \ldots, N-1\}$. All vehicles communicate with one another using C-V2X communications. The PC problem is considered within a finite time horizon, which is discretized into K equal-length control intervals indexed by $k \in \mathcal{K} = \{0, 1, \ldots, K-1\}$. The duration of each control interval is T milliseconds (ms). At each control interval k, the PC module of each follower $i \in \mathcal{V}\backslash\{0\}$ determines the vehicle control input $a_{i,k}^{\mathrm{CL}}$ based on the observations of the system state. The vehicle driving status is sampled at time kT. We adopt the PF IFT, where the CAM $c_{i-1,k} = \{p_{i-1,k}, v_{i-1,k}, acc_{i-1,k}\}$ of the predecessor $i-1 \in \mathcal{V}\backslash\{N-1\}$ are transmitted to the follower i. For this purpose, each control interval k is further divided into T communication intervals of every 1 ms indexed by $t \in \mathcal{T} = \{0, 1, \ldots, T-1\}$ on a faster timescale. Dynamic scheduling is considered, where the C-V2X communication module makes RRA decisions to transmit CAM at each communication interval (k, t).

Since the PC decisions are made with a coarse time grid of every T ms, while the RRA decisions are made with a fine time grid of every 1 ms, we have a multitimescale decision-making problem.

B. Platoon Control Module and C-V2X Communications Module

For the PC module, each vehicle $i \in \mathcal{V}$ obeys the dynamics model described by a first-order system, and the dynamics model is derived in discrete time on the basis of forward Euler discretization

$$p_{i,k+1} = p_{i,k} + Tv_{i,k} \tag{1}$$

$$v_{i,k+1} = v_{i,k} + \mathrm{Tacc}_{i,k} \tag{2}$$

$$\mathrm{acc}_{i,k+1} = \left(1 - \frac{T}{\tau_i}\right)\mathrm{acc}_{i,k} + \frac{T}{\tau_i}a_{i,k}^{\mathrm{CL}} \tag{3}$$

where τ_i is a time constant representing driveline dynamics. The tracking errors, i.e., gap-keeping error $e_{pi,k}$ and velocity error $e_{vi,k}$ of follower i are defined as

$$e_{pi,k} = d_{i,k} - d_{r,i,k} \tag{4}$$

$$e_{vi,k} = v_{i-1,k} - v_{i,k} \tag{5}$$

where $d_{i,k} = p_{i-1,k} - p_{i,k} - L_{i-1}$ is the headway of follower i at control interval k, and $d_{r,i,k} = r_i + h_i v_{i,k}$ is the desired headway.

For C-V2X communications module, we consider a typical urban C-V2X network, where M V2I links (uplink considered) coexist with $N-1$ V2V links. A V2I link $m \in \mathcal{M} = \{0, \ldots, M-1\}$ connects a vehicle to the BS and is used for high-throughput services. According to the PF IFT, a V2V link $i \in \mathcal{V}\backslash\{N-1\}$ connects a pair of predecessor i and follower $i+1$ for periodic transmission of CAM. In order to enhance spectrum utilization, the V2V links reuse the subchannels of the V2I links for CAM transmission. We use the binary allocation indicator $\theta_{i,m,(k,t)} \in \{0, 1\}$ to indicate whether V2V link i occupies subchannel m at communication interval (k, t) or not. In each communication interval (k, t), each vehicle i transmits the data in its queue according to the local RRA decisions. Assume the instantaneous channel gain of V2V link i over subchannel m (occupied by V2I link m) at communication interval (k, t) is denoted by $G_{i,m,(k,t)}$. Similarly, let $G_{m,(k,t)}$ denote the channel gain of the V2I link m; $G_{i,B,m,(k,t)}$ the interference channel gain from V2V link i transmitter to V2I link m receiver; $G_{B,i,m,(k,t)}$ the interference channel gain from V2I link m transmitter to V2V link i receiver; and $G_{j,i,m,(k,t)}$ the interference channel gain from the V2V link j transmitter to the V2V link i receiver over the subchannel m. The transmit power of V2V link i is $P^{\mathrm{V}}_{i,m,(k,t)}$ and the constant transmit power of V2I link m is P^{I}_m. The SINR $\gamma_{m,(k,t)}$ of V2I link m and the SINR $\gamma_{i,m,(k,t)}$ of V2V link i on subchannel m at communication interval (k, t) are derived by

$$\gamma_{m,(k,t)} = \frac{P^{\mathrm{I}}_m G_{m,(k,t)}}{\sigma^2 + \sum_{i\in\mathcal{V}\backslash\{N-1\}} \theta_{i,m,(k,t)} P^{\mathrm{V}}_{i,m,(k,t)} G_{i,B,m,(k,t)}} \tag{6}$$

and

$$\gamma_{i,m,(k,t)} = \frac{P^{\mathrm{V}}_{i,m,(k,t)} G_{i,m,(k,t)}}{\sigma^2 + I_{i,m,(k,t)}} \tag{7}$$

respectively, where σ^2 is the power of channel noise which satisfies the independent Gaussian distribution with a zero mean value. $I_{i,m,(k,t)}$ is the total interference power received by V2V link i over subchannel m, where

$$I_{i,m,(k,t)} = P^{\mathrm{I}}_m G_{B,i,m,(k,t)} + \sum_{j\in\mathcal{V}\backslash\{i,N-1\}} \theta_{j,m,(k,t)} P^{\mathrm{V}}_{j,m,(k,t)} G_{j,i,m,(k,t)}.$$

Then, the instantaneous data rate in terms of CAM of V2V link i and the instantaneous data rate of V2I link m at communication interval (k, t) are, respectively, derived as

$$r^{\mathrm{CAM}}_{i,(k,t)} = \frac{\sum_{m=0}^{M-1} W \log_2\left(1 + \gamma_{i,m,(k,t)}\right)}{N_c} \tag{8}$$

and

$$r_{m,(k,t)} = W \log_2\left(1 + \gamma_{m,(k,t)}\right) \tag{9}$$

where W is the bandwidth of a subchannel, and N_c is the constant CAM size.

Let $q^{\mathrm{CAM}}_{i,(k,t)}$ denote the CAM queue length of vehicle i at communication interval (k, t). If the queue length $q^{\mathrm{CAM}}_{i,(k,t)}$ reaches the buffer capacity N_Q, the subsequent arriving data will be dropped. The queue process evolves as

$$q^{\mathrm{CAM}}_{i,(k,t+1)} = \begin{cases} \min\left[N_Q, \max\left[0, q^{\mathrm{CAM}}_{i,(k,t)} - 10^{-3}\times r^{\mathrm{CAM}}_{i,(k,t)}\right] + 1\right], & \text{if} \quad t = 0 \\ \max\left[0, q^{\mathrm{CAM}}_{i,(k,t)} - 10^{-3} \times r^{\mathrm{CAM}}_{i,(k,t)}\right], & \text{otherwise.} \end{cases} \tag{10}$$

The CAM that is not transmitted during control interval k will continue to be transmitted in the next control interval $k+1$.

C. Correlation Between Platoon Control Decisions and Radio Resource Allocation Decisions

Let $\tau_{i,k}$ be the observation delay of follower i at control interval k. Thus, $c_{i-1,k-\tau_{i,k}} = \{p_{i-1,k-\tau_{i,k}}, v_{i-1,k-\tau_{i,k}}, \mathrm{acc}_{i-1,k-\tau_{i,k}}\}$ is the most recent available delayed CAM at follower i, which correspond to the position, velocity, and acceleration sampled at predecessor $i-1$ in control interval $k - \tau_{i,k}$. Therefore, the observed driving status of vehicle i is defined as

$$x_{i,k-\tau_{i,k}} = \{e_{pi,k-\tau_{i,k}}, e_{vi,k-\tau_{i,k}}, \mathrm{acc}_{i,k-\tau_{i,k}}, \mathrm{acc}_{i-1,k-\tau_{i,k}}\} \tag{11}$$

where $e_{pi,k-\tau_{i,k}} = p_{i-1,k-\tau_{i,k}} - p_{i,k-\tau_{i,k}} - L_{i-1} - d_{r,i,k}$ and $e_{vi,k-\tau_{i,k}} = v_{i-1,k-\tau_{i,k}} - v_{i,k-\tau_{i,k}}$.

The observation delay $\tau_{i,k}$ depends on the transmission delay of CAM over V2V link $i-1$, which can be derived from $q^{\mathrm{CAM}}_{i-1,(k-1,T)}$ or $q^{\mathrm{CAM}}_{i-1,(k,0)}$ as

$$\tau_{i,k} = \left\lceil q^{\mathrm{CAM}}_{i-1,(k-1,T)} \right\rceil + 1 = \left\lceil q^{\mathrm{CAM}}_{i-1,(k,0)} \right\rceil + 1. \tag{12}$$

A detailed analysis of the correlation between PC and RRA decisions can be found in Part I of this two-part paper.

D. Communication-Aware PC and the Corresponding DRL Solution

We assume that the RRA policy π^{CM} is available and focus on learning the PC policy π^{CL}_i, $i \in \mathcal{V}\backslash\{0\}$.

1) Communication-Aware PC Model:

a) PC state: The state for each PC agent i at control interval k is defined as

$$S_{i,k}^{\mathrm{CL}} = \left\{ x_{i,k-\tau_{i,k}}, \left\{ a_{i,k'}^{\mathrm{CL}} \right\}_{k'=k-\tau_{\max}}^{k-1}, \tau_{i,k} \right\}. \tag{13}$$

b) PC action: The control input, $a_{i,k}^{\mathrm{CL}}$ of the PC agent i is regarded as its PC action at control interval k.

c) PC reward function: The individual reward for each PC agent i is given by

$$R_{i,k}^{\mathrm{CL}}\left(x_{i,k}, a_{i,k}^{\mathrm{CL}}\right) = -\left\{ \left| \frac{e_{pi,k}}{\hat{e}_{p,\max}} \right| + \alpha_1 \left| \frac{e_{vi,k}}{\hat{e}_{v,\max}} \right| + \alpha_2 \left| \frac{a_{i,k}^{\mathrm{CL}}}{a_{\max}^{\mathrm{CL}}} \right| + \alpha_3 \left| \frac{j_{i,k}}{2acc_{\max}/T} \right| \right\} \tag{14}$$

where $j_{i,k} = ([\mathrm{acc}_{i,k+1} - \mathrm{acc}_{i,k}]/T) = -(1/[\tau_i])\mathrm{acc}_{i,k} + (1/\tau_i)a_{i,k}^{\mathrm{CL}}$ is the jerk.

The objective of the PC problem is for each PC agent i to find the optimal policy $\pi_i^{\mathrm{CL}*}$ under delayed observation that maximizes its individual expected return J_i^{CL}, i.e.,

$$\pi_i^{\mathrm{CL}*} = \arg\max_{\pi_i^{\mathrm{CL}*}} J_i^{\mathrm{CL}} \quad \forall i \in \mathcal{V}\backslash\{0\} \tag{15}$$

where

$$J_i^{\mathrm{CL}} = \mathrm{E}_{\pi^{\mathrm{CM}}} \mathrm{E}_{\pi_i^{\mathrm{CL}}} \left[\sum_{k=0}^{K-1} \gamma^k R_{i,k}^{\mathrm{CL}} \right], \ 0 \le \gamma \le 1. \tag{16}$$

Note that γ is the PC reward discount factor.

2) DRL Solution for Communication-Aware PC Model:

a) Multiagent problem in DRL-based PC: The PC problem corresponds to a Dec-POMDP and lies in the multiagent domain. We adopt the IL approach where each agent learns independently since the nonstationary environment issue for IL is trivial in the PC problem and the credit assignment issue in the multiagent problem does not exist for our PC model.

b) Random observation delay problem in DRL-based PC: We approximately consider that the undelayed driving status $x_{i,k}$ at PC agent i is Markov, ignoring the impact of the predecessors' actions on $x_{i,k+1}$. However, each PC agent i can only observe the delayed driving status $x_{i,k-\tau_{i,k}}$ instead of $x_{i,k}$. It is proved in Theorem 1 of Part I that $S_{i,k}^{\mathrm{CL}}$ becomes a Markov state by augmenting the delayed observation of driving status with action history.

We construct an MDP $\tilde{\mathcal{M}}_i = (S_{i,k}^{\mathrm{CL}}, a_{i,k}^{\mathrm{CL}}, \tilde{R}_{i,k}^{\mathrm{CL}}, p, \gamma)$ for the delayed observations, where the reward function $\tilde{R}_{i,k}^{\mathrm{CL}}(S_{i,k}^{\mathrm{CL}}, a_{i,k}^{\mathrm{CL}})$ for the augmented state of each follower $i \in \mathcal{V}\backslash\{0\}$ is defined as $\tilde{R}_{i,k}^{\mathrm{CL}}(S_{i,k}^{\mathrm{CL}}, a_{i,k}^{\mathrm{CL}}) = \mathrm{E}_{x_{i,k}}[R_{i,k}^{\mathrm{CL}}(x_{i,k}, a_{i,k}^{\mathrm{CL}})|S_{i,k}^{\mathrm{CL}}]$. It is proved in Theorem 2 of Part I that the optimal policy $\tilde{\pi}_i^{\mathrm{CL}*}$ for the augmented state MDP $\tilde{\mathcal{M}}_i$ is the same as the optimal policy $\pi_i^{\mathrm{CL}*}$ in (15) for our PC problem under delayed observation.

c) MTCC-PC algorithm: Based on the above discussion, the DDPG algorithm [16] is utilized to solve the PC problem. The deterministic policy gradient (DPG) Theorem can be directly applied since $\tilde{\mathcal{M}}_i$ is an MDP, i.e., the DPG for the augmented state MDP $\tilde{\mathcal{M}}_i$ is

$$\nabla_{\theta_i^\mu} J_i^{\mathrm{CL}}\left(\mu_i^{\mathrm{CL}}\right) = \mathrm{E}\Big[\nabla_{\theta_i^\mu} \mu_i^{\mathrm{CL}}\left(S_{i,k}^{\mathrm{CL}}|\theta_i^\mu\right) \nabla_a Q_i^{\mathrm{CL}}\left(S_{i,k}^{\mathrm{CL}}, a|\theta_i^Q\right)|_{a=\mu_i^{\mathrm{CL}}\left(S_{i,k}^{\mathrm{CL}}|\theta_i^\mu\right)} \Big]. \tag{17}$$

In order to sample the DPG in (17), we need to evaluate the action-value function $Q_{\mu_{\theta_i}}^{\mathrm{CL}}(S_{i,k}^{\mathrm{CL}}, a_{i,k}^{\mathrm{CL}})$ of the augmented state MDP $\tilde{\mathcal{M}}_i$. Based on the Bellman equation, the expected cumulative discounted reward from control interval k is defined as

$$Q_{\mu_{\theta_i}}^{\mathrm{CL}}\left(S_{i,k}^{\mathrm{CL}}, a_{i,k}^{\mathrm{CL}}\right) = \mathrm{E}_{x_{i,k}}\left[R_{i,k}^{\mathrm{CL}}\left(x_{i,k}, a_{i,k}^{\mathrm{CL}}\right)|S_{i,k}^{\mathrm{CL}}\right] + \gamma \mathrm{E}_{S_{i,k+1}^{\mathrm{CL}}}\left[Q_{\mu_{\theta_i}}^{\mathrm{CL}}\left(S_{i,k+1}^{\mathrm{CL}}, \mu_{\theta_i}(S_{i,k+1}^{\mathrm{CL}})\right)|S_{i,k}^{\mathrm{CL}}, a_{i,k}^{\mathrm{CL}}\right]. \tag{18}$$

During training, the sampled DPG ascent on $Q_i^{\mathrm{CL}}(S_{i,k}, \mu_i^{\mathrm{CL}}(S_{i,k}|\theta_i^\mu)|\theta_i^Q)$ with regard to θ_i^μ is used to train the actor network, and the critic network is trained by minimizing the root mean square error (RMSE) $L_{i,k} = y_{i,k} - Q_i^{\mathrm{CL}}(S_{i,k}^{\mathrm{CL}}, a_{i,k}^{\mathrm{CL}}|\theta_i^Q)$ using the sampled gradient descent with respect to θ_i^Q, where the TD target $y_{i,k}$ is calculated by

$$y_{i,k} = R_{i,k}^{\mathrm{CL}}\left(x_{i,k}, a_{i,k}^{\mathrm{CL}}\right) + \gamma Q_{\mu_{\theta_i}}^{\mathrm{CL}}\left(S_{i,k+1}^{\mathrm{CL}}, \mu_{\theta_i}(S_{i,k+1}^{\mathrm{CL}})\right). \tag{19}$$

Note that in the last control interval $K-1$ of the control episode, we still calculate $y_{i,k}$ using (19) instead of only the immediate reward. In this way, we convert the finite horizon SSDP to an infinite horizon one, since DDPG and most DRL algorithms are designed for infinite or indefinite horizon problems [17].

It is assumed that the PC agent i can sample the undelayed reward $R_{i,k}^{\mathrm{CL}}(x_{i,k}, a_{i,k}^{\mathrm{CL}})$ during training. This is possible since learning can be performed in a simulator or a laboratory in which the undelayed reward is available.

An important characteristic of the proposed MTCC-PC algorithm is that it is trained in a delayed environment generated by the simulation of C-V2X communications with de facto RRA policy rather than by a coarse-grained stochastic delay model. This is to ensure the delay distribution in the training environment is the same as that in the execution environment.

III. Control-Aware DRL-Based Radio Resource Allocation

We assume that the PC policy is available and focus on learning the RRA policy in this section.

The RRA problem is essentially a Dec-POMDP. Each predecessor $i \in \mathcal{V}\backslash\{N-1\}$ is an RRA agent, which makes a local observation at each communication interval (k, t), and decides on its local actions to maximize the expected cumulative *global* reward. Since one of the objectives of RRA is to minimize the PC performance degradation due to delayed observation, the main challenges in formulating the DRL model are the design of the reward function and state space.

A. PC Performance Degradation Due to Delayed Observation

We quantify the PC performance degradation for each follower $i \in \mathcal{V}\backslash\{0\}$ by the performance difference between $\pi_i^{\mathrm{CL}}(S_{i,k}^{\mathrm{CL}})$ and $\hat{\pi}_i^{\mathrm{CL}*}(x_{i,k})$, where the former is the policy in Section II when the PC agent only has access to the augmented state $S_{i,k}^{\mathrm{CL}}$ with delayed observation; while the latter is the optimal policy when the PC agent has oracle access to the current driving status $x_{i,k}$ at each control interval k. Therefore, the total PC performance degradation ΔJ^{CL} over all the followers is

$$\Delta J^{\mathrm{CL}} = \sum_{i=0}^{N-2}\left(J_{i+1}^{\mathrm{CL}} - \hat{J}_{i+1}^{\mathrm{CL}*}\right) \tag{20}$$

where J_{i+1}^{CL} is the expected return of policy $\pi_{i+1}^{\mathrm{CL}}(S_{i+1,k}^{\mathrm{CL}})$ given in (16), and

$$\hat{J}_{i+1}^{\mathrm{CL}*} = \mathrm{E}_{\hat{\pi}_{i+1}^{\mathrm{CL}*}}\left[\sum_{k=0}^{K-1}\gamma^k R_{i+1,k}^{\mathrm{CL}}\right] \tag{21}$$

is the expected return of optimal policy $\hat{\pi}_i^{\mathrm{CL}*}(x_{i+1,k})$, which is not affected by π^{CM} since it is assumed that the PC agent can observe the current driving status $x_{i+1,k}$ without delay.

B. RRA Reward Modeling

The optimization objective of RRA is formulated as

$$J^{\mathrm{CM}} = \kappa_1 \sum_{m=0}^{M-1}\mathrm{E}_{\pi^{\mathrm{CM}}}\left[\sum_{k=0}^{K-1}\sum_{t=0}^{T-1}\eta^{kT+t} r_{m,(k,t)}\right] + \kappa_2 \Delta J^{\mathrm{CL}} \tag{22}$$

where η is the RRA reward discount factor. The first term corresponds to the discounted sum throughput of all V2I links $m \in \mathcal{M}$ over all the communication intervals of a control episode, and the second term corresponds to the PC performance degradation given in (20). The weight factors κ_1 and κ_2 indicate the relative importance of minimizing PC performance degradation versus maximizing the V2I throughput.

The optimization objective J^{CM} should correspond to the expected return of the DRL model, i.e.,

$$J^{\mathrm{CM}} = \mathrm{E}_{\pi^{\mathrm{CM}}}\mathrm{E}_{\pi^{\mathrm{CL}}}\left[\sum_{k=0}^{K-1}\sum_{t=0}^{T-1}\eta^{kT+t} R_{(k,t)}^{\mathrm{CM}}\right], 0 \le \eta \le 1 \tag{23}$$

where the expectation is taken with respect to the probability distribution of the state-action trajectories when the RRA policy is π^{CM} and PC policy is π^{CL}. Since both RRA and PC are multiagent problems, the RRA (resp. PC) policy corresponds to the set of policies of all the RRA (resp. PC) agents, i.e., $\pi^{\mathrm{CM}} = \{\pi_i^{\mathrm{CM}}\}_{i\in\mathcal{V}\backslash\{N-1\}}$ and $\pi^{\mathrm{CL}} = \{\pi_i^{\mathrm{CL}}\}_{i\in\mathcal{V}\backslash\{0\}}$.

We will try to derive the RRA reward function $R_{(k,t)}^{\mathrm{CM}}$ from (22) and (23). Specifically, we have

$$R_{(k,t)}^{\mathrm{CM}} = \begin{cases} \kappa_1 R_{\mathrm{I},(k,t)}, & 0 \le t < T-1 \\ \kappa_1 R_{\mathrm{I},(k,t)} + \kappa_2 R_{\mathrm{V},(k,T)}, & t = T-1 \end{cases} \tag{24}$$

where

$$R_{\mathrm{I},(k,t)} = \sum_{m=0}^{M-1} r_{m,(k,t)} \tag{25}$$

is the component related to V2I throughput, and $R_{\mathrm{V},(k,T)}$ is the component related to the PC performance degradation.

At first glance, $R_{\mathrm{V},(k,T)}$ seems to be the difference between the two PC rewards $R_{i+1,k+1}^{\mathrm{CL}}(x_{i+1,k+1}, \pi_{i+1}^{\mathrm{CL}}(S_{i+1,k+1}^{\mathrm{CL}}))$ and $R_{i+1,k+1}^{\mathrm{CL}}(x_{i+1,k+1}, \hat{\pi}_{i+1}^{\mathrm{CL}*}(x_{i+1,k+1}))$ based on (20), (16), and (21). However, this is not correct since the expectations in (16) and (21) are with respect to different trajectory distributions. In order to derive $R_{\mathrm{V},(k,T)}$, the following lemma is used.

Lemma 1: The PC performance degradation due to observation delay for follower i equals to the expected cumulative advantage of $\pi_i^{\mathrm{CL}}(S_{i,k}^{\mathrm{CL}})$ over $\hat{\pi}_i^{\mathrm{CL}*}(x_{i,k})$, i.e.,

$$J_i^{\mathrm{CL}} - \hat{J}_i^{\mathrm{CL}*} = \mathrm{E}_{\pi^{\mathrm{CM}}}\mathrm{E}_{\pi_i^{\mathrm{CL}}}\left[\sum_{k=0}^{K-1}\gamma^k A_{\hat{\pi}_i^{\mathrm{CL}*}}\left(x_{i,k}, \pi_i^{\mathrm{CL}}\left(S_{i,k}^{\mathrm{CL}}\right)\right)\right] \quad 0 \le \gamma \le 1 \tag{26}$$

where

$$A_{\hat{\pi}_i^{\mathrm{CL}*}}\left(x_{i,k}, \pi_i^{\mathrm{CL}}\left(S_{i,k}^{\mathrm{CL}}\right)\right) = Q_{\hat{\pi}_i^{\mathrm{CL}*}}\left(x_{i,k}, \pi_i^{\mathrm{CL}}\left(S_{i,k}^{\mathrm{CL}}\right)\right) - V_{\hat{\pi}_i^{\mathrm{CL}*}}\left(x_{i,k}\right) \tag{27}$$

is the advantage function of policy $\hat{\pi}_i^{\mathrm{CL}*}(x_{i,k})$.

The proof of Lemma 1 is given in Appendix A.

Based on Lemma 1, $R_{\mathrm{V},(k,T)}$ is defined as

$$R_{\mathrm{V},(k,T)} = \sum_{i=0}^{N-2} A_{\hat{\pi}_{i+1}^{\mathrm{CL}*}}\left(x_{i+1,k+1}, \pi_{i+1}^{\mathrm{CL}}\left(S_{i+1,k+1}^{\mathrm{CL}}\right)\right). \tag{28}$$

Since the advantage function $A_{\hat{\pi}_{i+1}^{\mathrm{CL}*}}(x_{i+1,k}, \pi_{i+1}^{\mathrm{CL}}(S_{i+1,k}^{\mathrm{CL}}))$ belongs to the optimal policy $\hat{\pi}_{i+1}^{\mathrm{CL}*}(x_{i+1,k})$ without observation delay, the PC agent $i+1$ needs to learn $\hat{\pi}_{i+1}^{\mathrm{CL}*}(x_{i+1,k})$ and the corresponding advantage function in addition to the acting policy $\pi_{i+1}^{\mathrm{CL}}(S_{i+1,k}^{\mathrm{CL}})$.

Finally, the following theorem formally justifies the RRA reward function defined above.

Theorem 1: Given the reward function defined in (24), (25), and (28), the expected return as calculated in (23) is the same as the optimization objective in (22), where the relationship between the PC reward discount factor and RRA reward discount factor is $\eta = \gamma^{(1/T)}$.

The proof of Theorem 1 is given in Appendix B.

C. RRA State Definition

The state of each RRA agent i is defined as

$$S_{i,(k,t)}^{\mathrm{CM}} = \left\{\boldsymbol{G}_{i,(k,t)}, q_{i,(k,t)}^{\mathrm{CAM}}, \left\{a_{i,k'}^{\mathrm{CL}}\right\}_{k'=k-\tau_{\max}+1}^{k}, t, \epsilon\right\} \tag{29}$$

which includes four types of information as discussed below.

a) Channel state information: $\boldsymbol{G}_{i,(k,t)}$ is the local observation of CSI at vehicle i in communication interval (k,t), i.e.,

$$\boldsymbol{G}_{i,(k,t)} = \left\{G_{i,m,(k,t)}, G_{j,i,m,(k,t)}, G_{B,i,m,(k,t)} G_{i,B,m,(k,t)} G_{m,(k,t)}\right\}_{m\in\mathcal{M}}. \tag{30}$$

Specifically, $\boldsymbol{G}_{i,(k,t)}$ contains the following information: 1) the channel gain of V2V link i over all subchannels

$m \in \mathcal{M}$, $\{G_{i,m,(k,t)}\}_{m\in\mathcal{M}}$; 2) the interference channel gain from all V2I link m, $\{G_{B,i,m,(k,t)}\}_{m\in\mathcal{M}}$, and other V2V link j, $\{G_{j,i,m,(k,t)}\}_{j\neq i,m\in\mathcal{M}}$; 3) the interference channel gain from V2V link i to the BS over all subchannel $m \in \mathcal{M}$, $\{G_{i,B,m,(k,t)}\}_{m\in\mathcal{M}}$; and 4) the channel gain of all V2I link m, $\{G_{m,(k,t)}\}m \in \mathcal{M}$. Channel gains 1) and 2) can be accurately estimated by the receiver of V2V link i at the beginning of each communication interval (k, t) [18], while 3) and 4) are estimated at the BS in each communication interval (k, t) and then broadcast to all vehicles in its coverage, incurring a small signaling overhead.

b) Queue state information: $S^{\mathrm{CM}}_{i,(k,t)}$ includes the queue length $q^{\mathrm{CAM}}_{i,(k,t)}$ at vehicle i, whose value at communication interval $(k, 0) = (k-1, T)$ reflects the observation delay of vehicle $i+1$ at control interval k according to (12).

c) PC information: Since the RRA policy aims at minimizing PC performance degradation due to observation delay, it is our hypothesis, that including some PC information in the RRA state, could help the RRA agent make more informed decisions. However, it is not a trivial task to determine what information should be included. Ignoring useful information is obviously undesirable, while including useless information should also be avoided due to the curse-of-dimensionality issue.

Based on the following lemmas, Theorem 2 reveals the PC information that should be included in the RRA state.

Lemma 2: The optimal PC policy $\pi^{\mathrm{CL}*}_i(S^{\mathrm{CL}}_{i,k})$ based on delayed observation $S^{\mathrm{CL}}_{i,k}$ is equivalent to the optimal PC policy $\hat{\pi}^{\mathrm{CL}*}_i(x_{i,k})$ based on current driving status if $x_{i,k}$ can be fully determined by the delayed observation $S^{\mathrm{CL}}_{i,k}$, i.e., $x_{i,k} = f(S^{\mathrm{CL}}_{i,k})$ where $f(S^{\mathrm{CL}}_{i,k})$ is a deterministic function of $S^{\mathrm{CL}}_{i,k}$.

The proof of Lemma 2 is given in Appendix C.

Lemma 3: The current driving status $x_{i,k}$ can be fully determined by the delayed observation $S^{\mathrm{CL}}_{i,k}$ if the sequence of actions of vehicle $i-1$ within the time window of $[k-\tau_{i,k}, k-1]$ is available to vehicle i, i.e., $x_{i,k} = f(S^{\mathrm{CL}}_{i,k}, \{a^{\mathrm{CL}}_{i-1,k'}\}^{k-1}_{k'=k-\tau_{i,k}})$.

The proof of Lemma 3 is given in Appendix D.

Theorem 2: The impact of observation delay $\tau_{i,k}$ to the PC performance of vehicle i depends on the variation of vehicle $i-1$'s control action $a^{\mathrm{CL}}_{i-1,k'}$ within the period $k' \in [k-\tau_{\max}+1, k]$.

The proof of Theorem 2 is given in Appendix E.

According to Theorem 2, $\{a^{\mathrm{CL}}_{i,k'}\}^{k}_{k'=k-\tau_{\max}+1}$ is included in $S^{\mathrm{CM}}_{i,(k,t)}$ since it determines how sensitive the PC performance is to the observation delay for a specific control interval k. In other words, the characteristics of $a^{\mathrm{CL}}_{i-1,k'}$ within the period $k' \in [k-\tau_{\max}+1, k]$ determines the VoI of the received message at vehicle i at control interval k.

Remark 1 (VoI Per Control Interval): In the context of NCS or task-oriented scheduling, Age of Information (AoI) and VoI are two important metrics. AoI captures the importance of information by measuring its timeliness attribute [19], while VoI measures how much the recipient of the information can reduce the uncertainty of the stochastic processes related to decision-making [20], [21], [22]. It is shown in [23] that VoI is related to AoI while providing more accurate guidance to the scheduler design. However, VoI is normally derived based on mutual information as a single value for the whole control task, ignoring the variation in the significance of transmitted data during the control episode. In this article, we design the RRA reward function to capture VoI based on the PC performance degradation due to the observation delay of shared information in each control interval. Moreover, we include the PC information in the RRA state, so that the RRA decisions are made based on the *VoI per control interval* that captures VoI at a finer time granularity.

d) Other information: In order to stabilize experience replay for multiagent DRL, we augment $S^{\mathrm{CM}}_{i,(k,t)}$ with the exploration rate ϵ [24], i.e., the probability of random selecting action in ϵ-greedy policy for deep Q networks (DQNs).

Finally, the index of communication interval t is a part of the RRA state since it relates to the reward function in (24).

D. RRA Action

The RRA action of each vehicle $i \in \mathcal{V}\backslash\{N-1\}$ at communication interval (k, t) includes the decisions on the subchannel allocation $\{\theta_{i,m,(k,t)}\}_{m\in\mathcal{M}}$ and transmit power $\{P^{\mathrm{V}}_{i,m,(k,t)}\}_{m\in\mathcal{M}}$ for V2V link i. Since we consider that at most one subchannel is allocated to a V2V link at any communication interval (k, t), i.e., $\sum_{m=0}^{M-1}\theta_{i,m,(k,t)} \leq 1$, we can simplify the subchannel allocation action from $\{\theta_{i,m,(k,t)}\}_{m\in\mathcal{M}}$ to $m_i \in \mathcal{M}\cup\{-1\}$, where $m_i \in \mathcal{M}$ means that subchannel m_i is selected and $m_i = -1$ means that no subchannel is allocated to V2V link i.

We discretize the transmit power to four levels for ease of learning and practical circuit restriction [9], [11], i.e., $P^{\mathrm{V}}_{i,m,(k,t)} \in \mathcal{A}_P = \{23, 15, 5, -100\}$ dBm. Noted that -100 dBm can be considered as zero transmit power.

Therefore, the RRA action of each vehicle $i \in \mathcal{V}\backslash\{N-1\}$ at communication interval (k, t) is defined as

$$a^{\mathrm{CM}}_{i,(k,t)} = \left\{m_i, P^{\mathrm{V}}_{i,m,(k,t)} | m_i \in \mathcal{M}\cup\{-1\}, P^{\mathrm{V}}_{i,m,(k,t)} \in \mathcal{A}_P\right\}. \quad (31)$$

The objective of the RRA problem is to find the optimal policy $\pi^{\mathrm{CM}*}$ that maximizes the expected return J^{CM}, i.e.,

$$\pi^{\mathrm{CM}*} = \arg\max_{\boldsymbol{\pi}^{\mathrm{CM}}} J^{\mathrm{CM}}. \quad (32)$$

IV. DRL Solution for RRA

The proposed DRL algorithm to solve the RRA problem is based on double deep Q networks (DDQNs) [25], which is an improved version of the classical DQN algorithm, and has been widely adopted to deal with RL tasks with discrete action space. DQN combines Q-learning with a deep neural network (DNN) $Q^{\mathrm{CM}}_i(S^{\mathrm{CM}}_{i,(k,t)}, a^{\mathrm{CM}}_{i,(k,t)}|\theta_i)$, which acts as a function approximator and maps states to actions' value estimation [26]. Since DQN's max operator selects and evaluates action $a^{\mathrm{CM}}_{i,(k,t)}$ using the same value, the overestimated value is more likely to be chosen, leading to over-optimistic estimates. In order to avoid this, DDQN is proposed for decoupling the selection of actions and their evaluation in the target maximization [25]. Specifically, the action $a^{\mathrm{CM}}_{i,(k,t)}$ is chosen by the current Q networks $Q^{\mathrm{CM}}_i(S^{\mathrm{CM}}_{i,(k,t)}, a^{\mathrm{CM}}_{i,(k,t)}|\theta_i)$ with the maximum Q value, and the evaluation of the action $a^{\mathrm{CM}}_{i,(k,t)}$ is used by the target Q network $Q^{\mathrm{CM}'}_i(S^{\mathrm{CM}}_{i,(k,t)}, a^{\mathrm{CM}}_{i,(k,t)}|\theta'_i)$.

When tackling the RRA problem in this article, an effective policy cannot be learned by direct application of DDQN due to the following two challenges.

Multiagent Problem: Similar to PC, RRA is also a multiagent problem. However, the simple IL approach is not suitable for RRA since the agents cooperatively optimize the *global* expected cumulative reward, and it is hard for each agent to deduce its own contribution to the global reward. The challenge in distinguishing the agents' credit makes the learning of an effective policy nontrivial.

Long-Range Planning With Sparse Reward Problem: According to (22), the RRA problem involves optimization over a time horizon of KT communication intervals, where T consecutive communication intervals correspond to a control interval. In the reward function defined in (24), $R_{\mathrm{I},(k,t)}$ is provided to the agent in every communication interval (k,t), while $R_{\mathrm{V},(k,T)}$ is only available at the last communication interval $T-1$ of each control interval $k \in \mathcal{K}$. The sparse-reward tasks coupled with long-horizon planning become prohibitively difficult as highly specific action sequences must be executed prior to observing any nontrivial feedback.

In the following, we propose the MTCC-RRA algorithm based on DDQN, where we apply the *reward shaping* and *RBPER* [15] techniques to deal with these two challenges, respectively.

1) Reward Shaping: To overcome the multiagent challenge as described above, we design an individual RRA reward $R^{\mathrm{CM}}_{i,(k,t)}$ for each vehicle $i \in \mathcal{V}\backslash\{N-1\}$ as

$$R^{\mathrm{CM}}_{i,(k,t)} = \begin{cases} \kappa_1 \hat{R}_{\mathrm{I},i,(k,t)}, & t < T-1 \\ \kappa_1 \hat{R}_{\mathrm{I},i,(k,t)} + \\ \kappa_2 A_{\hat{\pi}^{\mathrm{CL}*}_{i+1}}\left(x_{i+1,k+1}, \pi^{\mathrm{CL}}_{i+1}\left(S^{\mathrm{CL}}_{i+1,k+1}\right)\right), & t = T-1 \end{cases} \tag{33}$$

where

$$\hat{R}_{\mathrm{I},i,(k,t)} = r_{m,(k,t)} - \bar{r}^{i}_{m,(k,t)} \tag{34}$$

is the difference reward that reflects the data rate loss of V2I link m resulting from sharing the subchannel with V2V link i, while the subchannel allocations of the other V2V links remain the same. $r_{m,(k,t)}$ is the actual data rate of V2I link m at communication interval (k,t), while $\bar{r}^{i}_{m,(k,t)}$ is the data rate of V2I link m if V2V link i does not occupy subchannel m. All the channel gains for the calculation of $\bar{r}^{i}_{m,(k,t)}$ are included in the state space. Since $\hat{R}_{\mathrm{I},i,(k,t)}$ is a negative value, maximizing its value over time is equivalent to minimizing the throughput loss of V2I links caused by interference from V2V link i.

Moreover, since the RRA decisions at vehicle i will only affect the observation delay and thus the PC performance of its follower $i+1$, the advantage function $A_{\hat{\pi}^{\mathrm{CL}*}_{i+1}}(x_{i+1,k+1}, \pi^{\mathrm{CL}}_{i+1}(S^{\mathrm{CL}}_{i+1,k+1}))$ is a part of the individual reward of vehicle i at each communication interval $(k, T-1)$ in (33).

Remark 2 [Obtaining $\hat{R}_{\mathrm{I},i,(k,t)}$ and $A_{\hat{\pi}^{\mathrm{CL}*}_{i+1}}$ in (33)]: We consider the centralized training decentralized execution paradigm [27] for MTCC-RRA, so that the global state and global action are available to the RRA agents during training. Therefore, both $r_{m,(k,t)}$ and $\bar{r}^{i}_{m,(k,t)}$ can be calculated by (6) and (9), where $\theta_{i,m,(k,t)}$ in (6) for V2V link i is set to 1 and 0, respectively. Then, $\hat{R}_{\mathrm{I},i,(k,t)}$ can be derived by (34). Moreover, the PC advantage function $A_{\hat{\pi}^{\mathrm{CL}*}_{i+1}}(x_{i+1,k+1}, \pi^{\mathrm{CL}}_{i+1}(S^{\mathrm{CL}}_{i+1,k+1}))$ of follower $i+1$ is available to its predecessor i due to the centralized training assumption. Note that each RRA agent trains a local critic instead of all the agents collectively training a centralized critic, since the global reward is decomposed into the individual rewards of the RRA agents.

2) Reward Backpropagation Prioritized Experience Replay: To resolve the difficulty of training for long-range planning with sparse reward, RBPER assigns a higher sampling priority β to the transition $(S^{\mathrm{CM}}_{i,(k,T-1)}, a^{\mathrm{CM}}_{i,(k,T-1)}, R^{\mathrm{CM}}_{i,(k,T-1)}, S^{\mathrm{CM}}_{i,(k,T)})$ of the last communication interval $T-1$ of each control interval k and then propagates the priority β back to the transition of the previous communication interval $T-2$ once it has been sampled for training and so on. The process will keep going until β is propagated to the transition $(S^{\mathrm{CM}}_{i,(k,0)}, a^{\mathrm{CM}}_{i,(k,0)}, R^{\mathrm{CM}}_{i,(k,0)}, S^{\mathrm{CM}}_{i,(k,1)})$ of the first communication interval 0 of each control interval k. Then priority β will be propagated back to the transition $(S^{\mathrm{CM}}_{i,(k,T-1)}, a^{\mathrm{CM}}_{i,(k,T-1)}, R^{\mathrm{CM}}_{i,(k,T-1)}, S^{\mathrm{CM}}_{i,(k,T)})$ and start a new round of propagation. The high priority will be decayed by factor $\zeta \in (0,1)$, once we start a new round until it goes down to normal priority 1. Thus, the target Q, $Q^{\mathrm{CM}'}_{i}(S^{\mathrm{CM}}_{i,(k,t+1)}, a^{\mathrm{CM}}_{i,(k,t+1)}|\theta'_i)$, of the previous communication interval t will be established on the basis of the trained Q, $Q^{\mathrm{CM}}_{i}(S^{\mathrm{CM}}_{i,(k,t+1)}, a^{\mathrm{CM}}_{i,(k,t+1)}|\theta_i)$, of the next communication interval $t+1$, improving the accuracy of Q value estimation and training efficiency.

V. Joint Optimization of Multitimescale Control and Communications

As mentioned above, it is assumed that the RRA (resp. PC) policy is available when training the PC (resp. RRA) policy. Due to the interactions between RRA and PC, it is not possible to learn the optimal RRA or PC policy first without knowing the other optimal policy. To solve this dilemma, the most straightforward approach is to train the proposed MTCC-PC and MTCC-RRA algorithms simultaneously. However, training both algorithms from scratch poses a great challenge in convergence, especially for RRA whose rewards depend on the PC. The rewards calculated from a PC model that has not reached convergence yet are misleading to RRA, resulting in substantial difficulty in learning an efficient RRA policy. Moreover, it is more challenging for MTCC-RRA algorithm to reach convergence than MTCC-PC algorithm under the multitimescale framework, since the number of communication intervals is T times that of the control intervals per episode. In this article, we propose to iteratively train MTCC-PC and MTCC-RRA algorithms, where each iteration $z \in \{1, \ldots, Z\}$ consists of the following two steps.

Step 1 The MTCC-PC is trained for E^{CL} episodes under the observation delay induced by the RRA policy learned in Step 2 of iteration $z-1$. The RRA transitions $(S^{\mathrm{CM}}_{i,(k,t)}, a^{\mathrm{CM}}_{i,(k,t)}, R^{\mathrm{CM}}_{i,(k,t)}, S^{\mathrm{CM}}_{i,(k,t+1)})$ are stored in an experience replay buffer D^{CM}_{i} for all episodes $E > E^{\mathrm{CL_{Th}}}$.

Algorithm 1 MTCC Algorithm

for follower $i \in \mathcal{V}\backslash\{0\}$ **do**
  Initialize PC actor network $\mu_i^{\mathrm{CL}}(s|\theta_i^{\mu})$ and critic network $Q_i^{\mathrm{CL}}(s,u|\theta_i^{Q})$ with random weights θ_i^{μ} and θ_i^{Q}
  Initialize PC target network weights with $\theta_i^{\mu'} = \theta_i^{\mu}$ and $\theta_i^{Q'} = \theta_i^{Q}$
  Initialize replay buffer D_i^{CL}
end for
for predecessor $i \in \mathcal{V}\backslash\{N-1\}$ **do**
  Initialize RRA Q networks $Q_i^{\mathrm{CM}}(s,a|\theta_i)$ with random weights θ_i
  Initialize RRA target network weights with $\theta_i' = \theta_i$
  Initialize replay buffer D_i^{CM}
end for
for Iteration $z = 1, \ldots, Z$ **do**
  Step 1:
  Function MTCC(PC_TRAIN $= 1$, RRA_TRAIN $= 0$, E^{CL}, ITER $= z$)
  Delete RRA experience for episodes $E \le E^{\mathrm{CL_{Th}}}$ from D_i^{CM}
  Step 2:
  Function MTCC(PC_TRAIN $= 0$, RRA_TRAIN $= 1$, E^{CM}, ITER $= z$)
  Delete PC experience for episodes $E \le E^{\mathrm{CM_{Th}}}$ from D_i^{CL}
end for

Step 2 The MTCC-RRA is trained for E^{CM} episodes, where the PC model learned in Step 1 of iteration z is used to calculate vehicle trajectories and RRA rewards. The PC transitions $(S_{i,k}^{\mathrm{CL}}, a_{i,k}^{\mathrm{CL}}, R_{i,k}^{\mathrm{CL}}, S_{i,k}^{\mathrm{CL}})$ are stored in an experience replay buffer D_i^{CL} for all episodes $E > E^{\mathrm{CM_{Th}}}$.

Note that it is essential to train the MTCC-PC first in Step 1, and the MTCC-RRA next in Step 2. This is because RRA training heavily depends on the PC advantage function to calculate reasonable reward signals as given in (28). The initial RRA policy randomly selects the RRA actions according to uniform distribution.

To improve the sample efficiency, the RRA (resp. PC) experience is stored in the experience replay buffer D_i^{CM} (resp. D_i^{CL}) when training the PC (resp. RRA) model. The stored experience can be leveraged at the initial phase of training when RRA (resp. PC) is trained in Step 2 (resp. Step 1 of the next iteration). To ensure that the experience are generated by relatively well-learned PC (resp. RRA) policies in the previous step, only the transitions at the later stage of PC (resp. RRA) training when $E > E^{\mathrm{CL_{Th}}}$ (resp. $E > E^{\mathrm{CM_{Th}}}$) are used.

Although it takes multiple iterations to obtain optimal results in MTCC, we will demonstrate in Section VI that satisfactory policies can be learned in only one iteration.

The proposed MTCC training algorithm is given in Algorithm 1, where the MTCC-PC and MTCC-RRA algorithms are iteratively trained for Z times. In each iteration, the function MTCC(PC_TRAIN $= 1$, RRA_TRAIN $= 0$, E^{CL}, ITER $= z$) corresponds to the MTCC-PC algorithm and the function MTCC(PC_TRAIN $= 0$, RRA_TRAIN $= 1$, E^{CM}, ITER $= z$) corresponds to the MTCC-RRA algorithm. To improve the sample efficiency, the transitions at the later stage of PC (resp. RRA) training when $E > E^{\mathrm{CL_{Th}}}$ (resp. $E > E^{\mathrm{CM_{Th}}}$) are used at the initial phase of training when RRA (resp. PC) is trained in Step 2 (resp. Step 1 of the next iteration). The pseudocode of the function MTCC is provided in function Algorithm 1. When the input argument PC_TRAIN $= 1$, the DDPG algorithm is used to train the PC agent. When the input argument RRA_TRAIN $= 1$, the DDQN-RBPER algorithm is used to train the RRA agent. The pseudocode of the DDPG algorithm and DQN-RBPER algorithm can be found in [15] and [16], respectively. The DDQN-RBPER algorithm extends DQN-RBPER with DDQN [25].

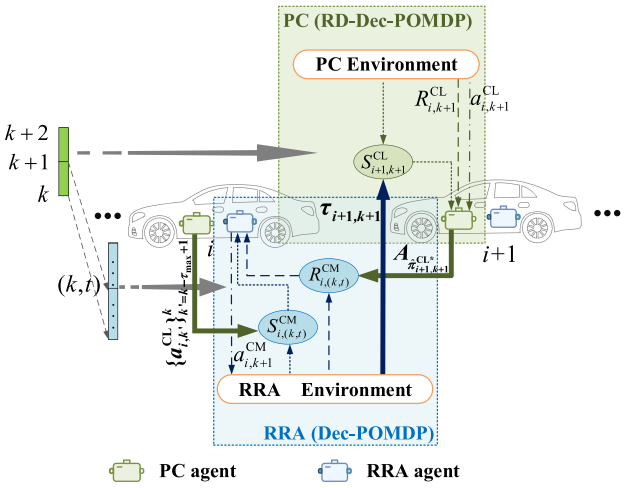

Fig. 1. Framework of MTCC algorithm.

The unified DRL training framework for MTCC is illustrated in Fig. 1. Each vehicle i contains a PC agent and an RRA agent, except for the leading vehicle 0 which only has an RRA agent and the last vehicle $N-1$ which only has a PC agent. The environment can also be divided into the PC environment, which provides driving status and PC rewards to the PC agents; and the RRA environment, which provides wireless channel and queue states as well as V2I-related rewards to the RRA agents. Meanwhile, the PC (resp. RRA) agents also interact with the RRA (resp. PC) environment/agents. For each RRA agent of vehicle $i \in \mathcal{V}\backslash\{N-1\}$, the PC advantage function $A_{\hat{\pi}_{i+1,k+1}^{\mathrm{CL}*}}$ of vehicle $i+1$ at control interval $k+1$ is used to calculate the PC-related portion of RRA reward $R_{i,(k,T-1)}^{\mathrm{CM}}$ at communication interval $(k, T-1)$. Moreover, the RRA state $S_{i,(k,t)}^{\mathrm{CM}}$ of vehicle i at communication interval (k,t) is augmented with its PC action history $\{a_{i,k'}^{\mathrm{CL}}\}_{k'=k-\tau_{\max}+1}^{k}$. Meanwhile, for each PC agent of vehicle $i+1 \in \mathcal{V}\backslash\{0\}$, its state $S_{i+1,k+1}^{\mathrm{CL}}$ at control interval $k+1$ depends on the observation delay $\tau_{i+1,k+1}$, which is provided by the RRA environment and determined by the queue state $q_{i,(k,T)}^{\mathrm{CAM}}$ of vehicle i at communication interval (k,T).

After the PC agents and RRA agents are trained in the centralized fashion, each vehicle makes PC and/or RRA decisions

Function 1 MTCC(PC_TRAIN, RRA_TRAIN, E, ITER)

for episode $e = 1, \ldots, E$ **do**
  for control interval $k = 0, \ldots, K-1$ **do**
    for communication interval $t = 0, \ldots, T-1$ **do**
      for predecessor $i \in \mathcal{V}\backslash\{N-1\}$ **do**
        Update the vehicle position and receive RRA state $S^{\text{CM}}_{i,(k,t)}$
        if RRA_TRAIN $= 1$ **then**
          Select RRA action $a^{\text{CM}}_{i,(k,t)}$ according to the ϵ-greedy policy with respect to $Q^{\text{CM}}_i(S^{\text{CM}}_{i,(k,t)}, a|\theta_i)$
        else
          if ITER $= 1$ **then**
            Select a random RRA action $a^{\text{CM}}_{i,(k,t)}$
          else
            Select RRA action $a^{\text{CM}}_{i,(k,t)}$ according to the greedy policy with respect to $Q^{\text{CM}}_i(S^{\text{CM}}_{i,(k,t)}, a|\theta_i)$
          end if
        end if
        Execute RRA action $a^{\text{CM}}_{i,(k,t)}$ and observe RRA reward $R^{\text{CM}}_{i,(k,t)}$ and next RRA state $S^{\text{CM}}_{i,(k,t+1)}$
        Store RRA transition $(S^{\text{CM}}_{i,(k,t)}, a^{\text{CM}}_{i,(k,t)}, R^{\text{CM}}_{i,(k,t)}, S^{\text{CM}}_{i,(k,t+1)})$ with priority p in D^{CM}_i
        if RRA_TRAIN $= 1$ **then**
          Update θ_i, θ'_i with DDQN-RBPER algorithm
        end if
      end for
    end for
    for follower $i \in \mathcal{V}\backslash\{0\}$ **do**
      Calculate observation delay $\tau_{i,k}$ and receive PC state $S^{\text{CL}}_{i,k}$
      Select PC action $a^{\text{CL}}_{i,k} = \mu^{\text{CL}}_i(S^{\text{CL}}_{i,k}|\theta^{\mu}_i)$
      if PC_TRAIN $= 1$ **then**
        Add exploration noise to PC action $a^{\text{CL}}_{i,k} = a^{\text{CL}}_{i,k} + \mathcal{N}_t$
      end if
      Store PC transition $(S^{\text{CL}}_{i,k}, a^{\text{CL}}_{i,k}, R^{\text{CL}}_{i,k}, S^{\text{CL}}_{i,k+1})$ in D^{CL}_i
      if PC_TRAIN $= 1$ **then**
        Update $\theta^{\mu}_i, \theta^{Q}_i, \theta^{\mu'}_i, \theta^{Q'}_i$ with DDPG algorithm
      end if
    end for
  end for
end for

in a decentralized manner based on its local observations of the PC and RRA states. The decisions can be made in real-time since only the forward propagation in the DNNs is involved during execution.

VI. Experimental Results

In this section, we design experiments to answer the following questions.

1) Can MTCC-RRA outperform the state-of-the-art control-aware communications by capturing "VoI per control interval" through its state space and reward design?
2) Can the reward shaping and RBPER mechanisms efficiently tackle the multiagent and sparse reward problems for MTCC-RRA?
3) Can the satisfactory policies for MTCC be learned in only one iteration? The experimental results of the proposed MTCC algorithm and the baseline DRL algorithms are presented and compared for the above purposes.

TABLE I
Hyperparameters of the DRL Algorithms for Training

Parameter	Value
Network size	$256, 64$
Activation function	relu, relu, relu, linear
Learning rate	0.0001
Exploration rate ϵ	$1 \rightarrow 0.05$
Higher sampling priority β	100
Decay factor ζ	0.2
Batch size N_b	64
Replay buffer size	200000
Reward discount factor γ	0.99979
Target update frequency of DDQN N^-	4
Final layer weights/biases initialization	Random uniform distribution $[-3 \times 10^{-3}, 3 \times 10^{-3}]$
Other layer weights/biases initialization	Random uniform distribution$[-\frac{1}{\sqrt{f}}, \frac{1}{\sqrt{f}}]$ (f is the fan-in of the layer)

A. Experimental Setup

We conduct experiments based on the PC and RRA environments provided in Part I of this two-part paper. In addition to the hyperparameters of DDPG for solving the communication-aware PC problem, we summarized the hyperparameters for DDQN solving the control-aware RRA problem in Table I. The values of all the hyperparameters were selected by performing a grid search as in [26], using the values reported in [25] as a reference. DDQN has two hidden layers with 256 and 64 nodes, respectively, where 128 of the nodes in the first hidden layer are modeled as long short-term memory (LSTM) for processing PC action history $\{a^{\text{CL}}_{i,k'}\}^{k}_{k'=k-\tau_{\max}+1}$ in $S^{\text{CM}}_{i,(k,t)}$ and the rest of the 128 nodes in the first hidden layer are used for the remaining dimensions of state $S^{\text{CM}}_{i,(k,t)}$. Then all the 256 nodes are connected to the second layer with 64 nodes. The target network of DDQN is updated every 4 communication intervals.

1) Baseline Algorithms: Several baseline algorithms are selected for performance comparison with the proposed algorithm.

1) *delay-aware RRA (Delay-RRA)*, which is an improved version of the DRL-based RRA algorithm for C-V2X communications in [9], [11], and [12]. Liang et al. [9] focused on RRA within one control interval and aims to minimize the latency and maximize the success probability of delivering the CAM generated at the beginning of the control interval. To deal with the sparse reward problem, the V2V transmission rate at each time step is included in the reward function to

TABLE II
PERFORMANCE AFTER TRAINING RRA IN ONE ITERATION. WE PRESENT RRA PERFORMANCE AND AVERAGE SUM V2I THROUGHPUT, AND SUM PC PERFORMANCE FOR MTCC-RRA, DELAY-RRA, AND AOI-RRA, MTCC_WO_RS, AND MTCC_WO_RBPER, RESPECTIVELY

	MTCC-RRA	**Delay-RRA**	**AoI-RRA**	**MTCC_wo_RS**	**MTCC_wo_RBPER**
RRA performance	68.2553	43.0803	43.8063	58.3390	-209.1657
Average sum V2I throughput (Mbps)	4.3982	3.6646	3.6432	4.0040	4.7523
Sum PC performance	-1.5390	-1.4231	-1.4231	-3.2671	-28.1860

expedite convergence. Since [9] does not address the credit assignment problem in MARL, we improve the algorithm by applying reward shaping similar to MTCC for a fair comparison. Moreover, to apply the algorithm in a longer time horizon with multiple control intervals, we assume that an agent replaces any old CAM that has not yet been fully delivered in the previous control interval with the newly generated CAM at the beginning of each control interval. Therefore, the reward function $R^{\text{IR}}_{i,(k,t)}$ of Delay-RRA is given as

$$R^{\text{IR}}_{i,(k,t)} = \lambda_1 \hat{R}_{\text{I},i,(k,t)} + \lambda_2 \left(r_{i,(k,t)}|_{q^{\text{CAM}}_{i,(k,t)}>0} + G|_{q^{\text{CAM}}_{i,(k,t)}=0} \right) \tag{35}$$

where $\lambda_1 = 0.001/W$ and $\lambda_2 = 0.1/W$. Notice that G is a tuned hyperparameter encouraging early completion of transmission for CAM whose value is greater than the largest $r_{i,(k,t)}$ ever obtained. We set $G = 10W$ in the experiments.

2) *AoI-aware RRA (AoI-RRA)*, which is similar to the DRL-based RRA in MTCC except that it aims to minimize cumulative AoI instead of maximizing cumulative "VoI per control interval." Specifically, the advantage function $A_{\hat{\pi}^{\text{CL}*}_{i+1}}\left(x_{i+1,k+1}, \pi^{\text{CL}}_{i+1}(S^{\text{CL}}_{i+1,k+1})\right)$ in (33) is replaced by $-\text{AoI}_{i+1,k}$, where $\text{AoI}_{i+1,k}$ is the AoI of vehicle $i+1$ that evolves according to

$$\text{AoI}_{i+1,k+1} = \begin{cases} T, & \text{if } q^{\text{CAM}}_{i,(k,T)} = 0 \\ \text{AoI}_{i+1,k} + T, & \text{if } q^{\text{CAM}}_{i,(k,T)} > 0. \end{cases} \tag{36}$$

For Delay-RRA and AoI-RRA, $q^{\text{CAM}}_{i,(k,0)}$ is always 1 and no longer equals to $q^{\text{CAM}}_{i,(k-1,T)}$.

3) *MTCC without reward shaping (MTCC_wo_RS)*, where the global reward in (24) is used in MTCC for training DRL-based RRA, instead of the individual reward in (33) after reward shaping.

4) *MTCC without RBPER (MTCC_wo_RBPER)*, where DDQN without RBPER is used in MTCC for training DRL-based RRA.

2) Experiment Design: To answer the three questions mentioned at the beginning of Section VI, we design the following three experiments. The proposed MTCC is trained in one iteration for Experiments 1 and 2.

1) *Experiment 1 (Performance Comparison of MTCC-RRA With Delay-RRA and AoI-RRA):* To answer Question (1), all three algorithms are trained for $E^{\text{CL}} = 100$ episodes. The trained PC policy by MTCC-PC is adopted during both the training and testing of the RRA algorithms. Therefore, the PC performance difference between Delay-RRA, AoI-RRA, and MTCC-RRA is only due to the difference in induced observation delay by the RRA algorithms.

2) *Experiment 2 (Performance Comparison of MTCC-RRA with MTCC_wo_RS and MTCC_wo_RBPER):* To answer Question (2), MTCC_wo_RS and MTCC_wo_RBPER are trained for $E^{\text{CL}} = 100$ episodes and the resulting performance is compared with that of MTCC-RRA.

3) *Experiment 3 (Performance Comparison of MTCC in One and Two Iterations):* To answer Question 3), we train MTCC for a second iteration and observe the resultant performance gain.

Note that the episode thresholds for storing experience are set to $E^{\text{CL}_{\text{Th}}} = (E^{\text{CL}}/5)$ and $E^{\text{CM}_{\text{Th}}} = (E^{\text{CM}}/5)$, respectively.

B. Performance Comparison of MTCC-RRA, Delay-RRA, and AoI-RRA.

1) Performance for Testing Data: The RRA performance are reported in Table II for MTCC-RRA, Delay-RRA, and AoI-RRA, respectively, which are obtained by averaging the RRA return over 100 testing episodes when training is completed. The RRA return is derived as the cumulative global RRA reward defined in (24) overall communication intervals in one control episode. Since the global RRA reward consists of two parts, i.e., the sum throughput of all V2I links and the sum PC performance degradation due to delayed observation for all followers, we also report the average sum V2I throughput and sum PC performance in Table II. The former is obtained by averaging the sum throughput for all V2I links over 100 testing episodes, while the latter is obtained by averaging the sum PC returns of all followers. Note that the sum PC performance and the sum PC performance degradation are essentially the same since their values are different by the same fixed amount for all the algorithms, where the difference corresponds to the optimal sum PC performance without observation delay.

As shown in Table II, MTCC-RRA outperforms AoI-RRA and Delay-RRA algorithms by 58.44% and 55.81%, respectively, in terms of RRA performance. The result is not surprising since MTCC-RRA aims at maximizing the RRA performance corresponding to the cumulative global RRA reward in (24), while Delay-RRA and AoI-RRA have different reward functions.

The reward function of MTCC-RRA differs from those of Delay-RRA and AoI-RRA in that MTCC-RRA aims at minimizing the PC performance degradation due to observation delay instead of minimizing the delay or AoI, while all the reward functions also target at maximizing the V2I throughput. Therefore, MTCC-RRA can achieve larger V2I throughput by allocating more radio resources to V2I links when larger

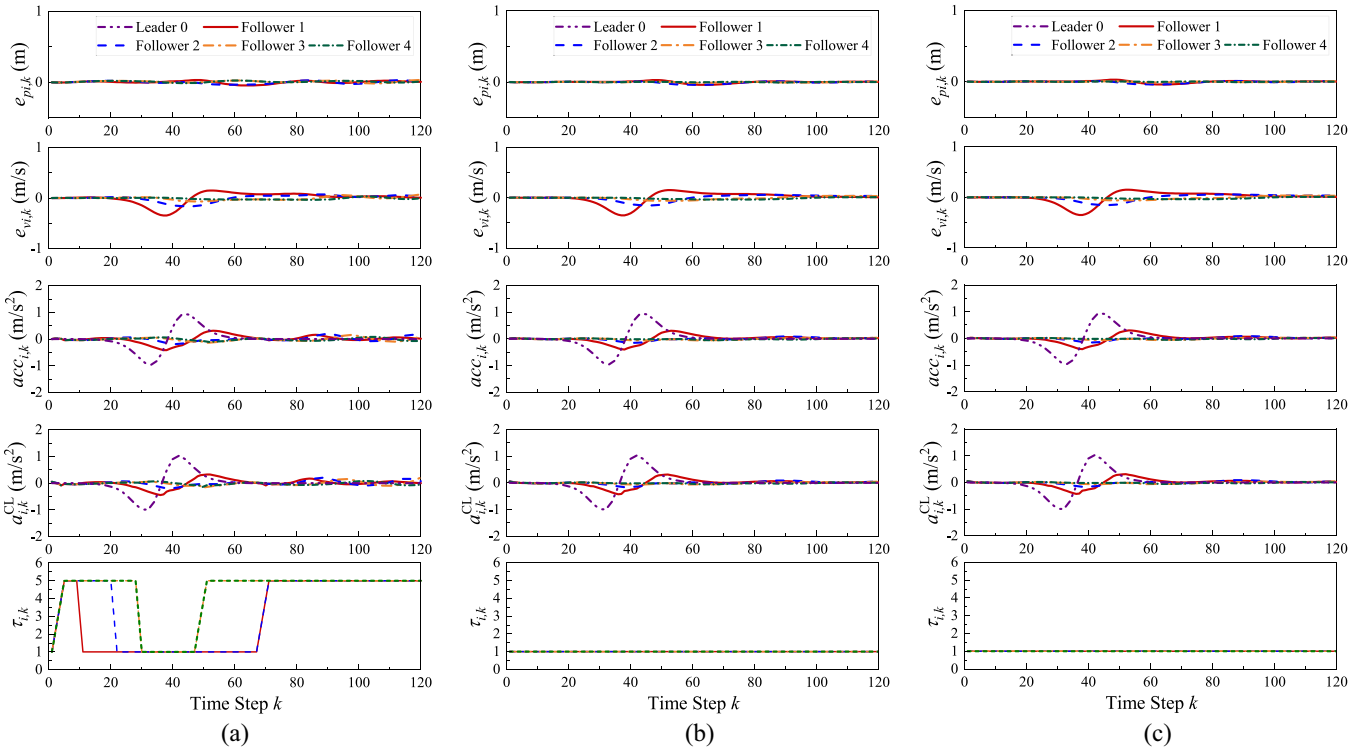

Fig. 2. Results of a specific test control episode. The tracking errors $e_{pi,k}$, $e_{vi,k}$, and $acc_{i,k}$ along with the control input $a^{\mathrm{CL}}_{i,k}$ and observation delay $\tau_{i,k}$ of each follower i are represented as different curves, respectively. (a) MTCC-RRA. (b) Delay-RRA. (c) AoI-RRA.

observation delays do not have significant impacts on the PC performance. This is further proved by a close examination of the average sum V2I throughput and sum PC performance in Table II. In terms of the average sum V2I throughput, MTCC-RRA outperforms AoI-RRA and Delay-RRA algorithms by 20.02% and 20.72%, respectively. Meanwhile, MTCC-RRA is slightly inferior to AoI-RRA and Delay-RRA in terms of sum PC performance by 8.14%. Therefore, MTCC-RRA can strike a better tradeoff between PC performance and V2I throughput compared with the other two RRA algorithms.

2) Testing Results of One Episode: The key reasons for the superior performance of MTCC-RRA are quantifying the "VoI per control interval" in the reward function, and including the control action history in the RRA state. To better understand the positive effects of these design factors, we zoom in to one control episode, where the tracking errors $e_{pi,k}$ and $e_{vi,k}$ of each follower $i \in \mathcal{V}\backslash\{0\}$ as well as the acceleration $acc_{i,k}$ and control input $a^{\mathrm{CL}}_{i,k}$ of each vehicle $i \in \mathcal{V}$ for all time steps $k \in \{1, 2, \ldots, 120\}$ are plotted in Fig. 2 for MTCC-RRA, Delay-RRA, and AoI-RRA. In addition, the observation delay, $\tau_{i,k}$, of each follower $i \in \mathcal{V}\backslash\{0\}$ for each time step k are also plotted.

It can be observed that the performance curves of Delay-RRA and AoI-RRA are the same since the minimum observation delay of 1 control interval is always achieved for both algorithms. Meanwhile, the overall shapes of $e_{pi,k}$, $e_{vi,k}$, $acc_{i,k}$, and $a^{\mathrm{CL}}_{i,k}$ curves for MTCC-RRA look very similar to those of Delay-RRA and AoI-RRA, while the $\tau_{i,k}$ curves are quite different. Instead of achieving a constant minimum delay at all time steps, it can be seen that the observation delay of MTCC-RRA for each follower $i \in \mathcal{V}\backslash\{0\}$ is minimum only when the control input $a^{\mathrm{CL}}_{i-1,k}$ of its predecessor $i-1$ changes significantly and rapidly. When $a^{\mathrm{CL}}_{i-1,k}$ remains relatively constant, the follower i generally has large observation delays, which means that the predecessor $i-1$ refrains from sharing the V2I subchannels for CAM transmission to support larger V2I throughput. This observation demonstrates that the MTCC-RRA agent has the ability to allocate radio resources based on the VoI per control interval.

As a result, the PC performance of MTCC-RRA is only slightly affected compared to those of Delay-RRA and AoI-RRA. Fig. 2 shows that the convergence speed to steady state for all algorithms are similar, and the PC performance differences of the algorithms are reflected in the oscillations of the tracking errors, acceleration, and control input. After the steady state is reached and the tracking errors are almost zero, the performance curves of MTCC-RRA have slightly larger oscillations compared with those in Delay-RRA and AoI-RRA. As the oscillations are negligible, MTCC-RRA can still ensure driving safety and comfort. Moreover, the amplitudes of the oscillations in $e_{pi,k}$, $e_{vi,k}$, and $acc_{i,k}$ for each follower $i \in \{1, 2\}$ are smaller than those of their respective predecessors $i-1$ for all algorithms, which means that the string stability of the platoons can be achieved by MTCC-RRA despite the larger observation delays.

3) Convergence Properties: The RRA performance of Delay-RRA, AoI-RRA, and MTCC-RRA are evaluated periodically during training by testing in an environment where the trained MTCC-PC policy is used to calculate vehicle trajectories and RRA rewards. Specifically, we run 1 test control episodes after every 1 training control episode and

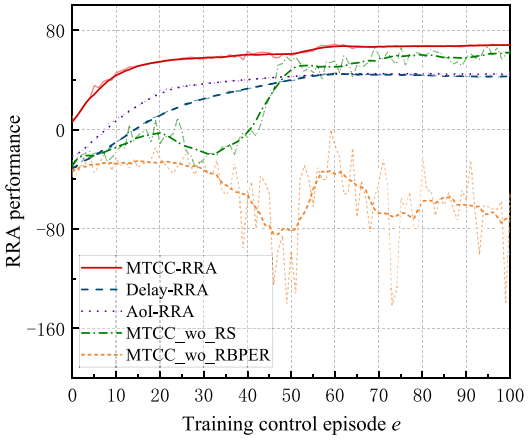

Fig. 3. RRA Performance during DRL-based RRA algorithms training in one iteration. The vertical axis corresponds to the returns over 1 test episode. The dark curves correspond to smoothed curves and the light color curves correspond to the original curves.

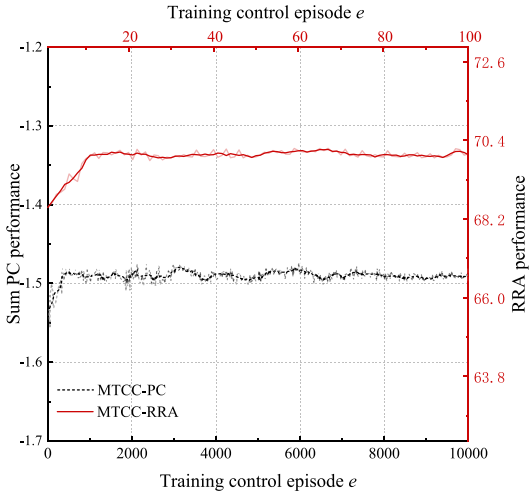

Fig. 4. Sum PC performance and RRA performance during training in the second iteration.

calculate the RRA return as the performance for the latest 1 training episode. The RRA performance as a function of the number of training episodes with MTCC-RRA, Delay-RRA, and AoI-RRA algorithms is plotted in Fig. 3. It is shown that the convergence rate for all three algorithms are similar, with a ranking of MTCC-RRA >AoI-RRA>Delay-RRA.

C. Performance Comparison of MTCC-RRA, MTCC_wo_RS, and MTCC_wo_RBPER

The RRA performance, average sum V2I throughput, and sum PC performance are reported in Table II for MTCC_wo_RS and MTCC_wo_RBPER, respectively. In terms of RRA performance, MTCC-RRA outperforms MTCC_wo_RS and MTCC_wo_RBPER algorithms by 16.95% and 132.63%, respectively. The superior performance of MTCC-RRA over MTCC_wo_RS demonstrates the effectiveness of the reward shaping technique in efficiently distinguishing the multiple agents' credit in the global reward. Moreover, the significant performance gain of MTCC-RRA over MTCC_wo_RBPER shows that the RBPER technique is essential in solving the long-range planning with sparse reward problem. The same conclusion can also be drawn from the convergence curves of MTCC_wo_RS and MTCC_wo_RBPER shown in Fig. 3. Both performance curves have large fluctuations, and that of MTCC_wo_RBPER is not convergent.

D. Performance Comparison of MTCC in one and Two Iterations

In the above experiments, MTCC is only trained in one iteration, i.e., $Z = 1$, in Algorithm 1. In the following, we train MTCC for one additional iteration to observe the resultant performance gain. For the second iteration, the performance curves for the PC policy, i.e., MTCC-PC, during training in Step 1 and for the RRA policy, i.e., MTCC-RRA, during training in Step 2 are given in Fig. 4, respectively. The performance curve of MTCC-PC is obtained in a similar way as that in Fig. 3 in Part I, except that the MTCC-PC policy is trained and tested under a delayed environment generated by the MTCC-RRA policy learned in Step 2 of iteration 1. Similarly, the performance curve of MTCC-RRA is obtained in a similar way as that in Fig. 3, except that the MTCC-RRA policy is trained and tested under the PC environment generated by the MTCC-PC policy learned in Step 1 of iteration 2. From Fig. 4, it can be observed that the sum PC performance and RRA performance improve with increasing training episodes in Step 1 and Step 2 of the second iteration, respectively. However, the performance gains are not significant.

Specifically, the sum PC performance of MTCC-PC is around -1.55 at the beginning of training and finally converges around -1.49 toward the end, with a performance gain of 3.87%. The performance gain in Step 1 is due to the fact that the MTCC-PC policy is trained under the observation delay generated by the de facto MTCC-RRA policy instead of the random RRA policy. Similarly, the RRA performance of MTCC-RRA is around 68.4 at the beginning of training and finally converges around 70.2 toward the end, with a performance gain of 2.63%. The performance gain in Step 2 is because the de facto MTCC-PC policy trained in iteration 2 is adopted instead of that trained in iteration 1 to simulate the PC environment, so that the reward function of MTCC-RRA calculated from the MTCC-PC advantage function can more accurately represent the PC performance degradation due to observation delay.

The above results show that satisfactory policies for MTCC can be learned in one iteration and a second iteration is merely a fine-tuning for small performance improvement. Moreover, it can be anticipated that the performance gain will keep decreasing when training in more iterations.

VII. Conclusion

In this pair of papers, we have studied how to solve the joint optimization of the MTCC problem in the C-V2X system. We have established a unified DRL framework for MTCC and proposed the MTCC-PC algorithm to solve the communication-aware PC subproblem in Part I of this

two-part paper. In this article, we have proposed the MTCC-RRA algorithm to solve the control-aware RRA subproblem. Specifically, the RRA decisions have been made based on the advantage function of the PC model, which provides a fine-grained VoI per control interval. The multiagent problem and the sparse reward problem in RRA have been tackled with the reward shaping and RBPER techniques, respectively. Finally, we have designed a sample- and computational-efficient training approach to jointly learn the PC and RRA policies iteratively. The experimental results have shown that the proposed MTCC-RRA algorithm outperforms the baseline DRL algorithms, striking a good tradeoff between minimizing PC performance degradation due to observation delay and maximizing V2I throughput. Moreover, it is demonstrated that the joint training approach can learn satisfactory PC and RRA policies in only one iteration. In our future work, we will apply the proposed MTCC framework to other autonomous driving tasks, such as lane changing and merging.

APPENDIX

A. Proof of Lemma 1

As explained in Section II-D, we convert the finite-horizon PC problem to an infinite-horizon one by calculating the TD target $y_{i,K-1}$ using (19) for the last control interval $K-1$ instead of setting $y_{i,K-1}$ to be the immediate reward. Therefore, in the following proof, we replace K with ∞ on the RHS of (26):

$$\begin{aligned}
&\mathrm{E}_{\pi^{\mathrm{CM}}}\mathrm{E}_{\pi_i^{\mathrm{CL}}}\left[\sum_{k=0}^{\infty}\gamma^k A_{\hat{\pi}_i^{\mathrm{CL}*}}\left(x_{i,k},\pi_i^{\mathrm{CL}}(S_{i,k}^{\mathrm{CL}})\right)\right]\\
&\overset{(a)}{=}\mathrm{E}_{\pi^{\mathrm{CM}}}\mathrm{E}_{\pi_i^{\mathrm{CL}}}\left[\sum_{k=0}^{\infty}\gamma^k\left(Q_{\hat{\pi}_i^{\mathrm{CL}*}}(x_{i,k},\pi_i^{\mathrm{CL}}(S_{i,k}^{\mathrm{CL}}))\right.\right.\\
&\qquad\left.\left.-V_{\hat{\pi}_i^{\mathrm{CL}*}}(x_{i,k})\right)\right]\\
&\overset{(b)}{=}\mathrm{E}_{\pi^{\mathrm{CM}}}\mathrm{E}_{\pi_i^{\mathrm{CL}}}\left[\sum_{k=0}^{\infty}\gamma^k(\mathrm{E}_{x_{i,k+1}}[R_{i,k}^{\mathrm{CL}}+\gamma V_{\hat{\pi}_i^{\mathrm{CL}*}}(x_{i,k+1})]\right.\\
&\quad\left.-V_{\hat{\pi}_i^{\mathrm{CL}*}}(x_{i,k}))\right]\\
&\overset{(c)}{=}\mathrm{E}_{\pi^{\mathrm{CM}}}\mathrm{E}_{\pi_i^{\mathrm{CL}}}\left[\sum_{k=0}^{\infty}\gamma^k(R_{i,k}^{\mathrm{CL}}+\gamma V_{\hat{\pi}_i^{\mathrm{CL}*}}(x_{i,k+1})\right.\\
&\quad\left.-V_{\hat{\pi}_i^{\mathrm{CL}*}}(x_{i,k}))\right]\\
&=\mathrm{E}_{\pi^{\mathrm{CM}}}\mathrm{E}_{\pi_i^{\mathrm{CL}}}\left[\sum_{k=0}^{\infty}\gamma^k R_{i,k}^{\mathrm{CL}}-V_{\hat{\pi}_i^{\mathrm{CL}*}}(x_{i,0})\right]\\
&=\mathrm{E}_{\pi^{\mathrm{CM}}}\mathrm{E}_{\pi_i^{\mathrm{CL}}}\left[\sum_{k=0}^{\infty}\gamma^k R_{i,k}^{\mathrm{CL}}\right]-\mathrm{E}_{x_{i,0}}\left[V_{\hat{\pi}_i^{\mathrm{CL}*}}(x_{i,0})\right]\\
&\overset{(d)}{=}J_i^{\mathrm{CL}}-\hat{J}_i^{\mathrm{CL}*}
\end{aligned}\tag{37}$$

where (a) follows from (27); (b) follows from the Bellman equation; (c) follows by merging the expectation $\mathrm{E}_{x_{i,k+1}}$ into the expectation $\mathrm{E}_{\pi^{\mathrm{CM}}}\mathrm{E}_{\pi_i^{\mathrm{CL}}}$; and (d) follows from (16), (21) and the definition of the value function.

B. Proof of Theorem 1

$$\begin{aligned}
J^{\mathrm{CM}}&=\mathrm{E}_{\pi^{\mathrm{CM}}}\mathrm{E}_{\pi^{\mathrm{CL}}}\left[\sum_{k=0}^{K-1}\sum_{t=0}^{T-1}\eta^{kT+t}R_{(k,t)}^{\mathrm{CM}}\right]\\
&\overset{(a)}{=}\kappa_1\sum_{m=0}^{M-1}\mathrm{E}_{\pi^{\mathrm{CM}}}\left[\sum_{k=0}^{K-1}\sum_{t=0}^{T-1}\eta^{kT+t}r_{m,(k,t)}\right]\\
&\quad+\kappa_2\sum_{i=0}^{N-2}\left(\mathrm{E}_{\pi^{\mathrm{CM}}}\right.\\
&\left.\mathrm{E}_{\pi_{i+1}^{\mathrm{CL}}}\left[\sum_{k=0}^{K-1}\eta^{kT+T-1}A_{\hat{\pi}_{i+1}^{\mathrm{CL}*}}(x_{i+1,k+1},\pi_{i+1}^{\mathrm{CL}}(S_{i+1,k+1}^{\mathrm{CL}}))\right]\right)\\
&\overset{(b)}{=}\kappa_1\sum_{m=0}^{M-1}\mathrm{E}_{\pi^{\mathrm{CM}}}\left[\sum_{k=0}^{K-1}\sum_{t=0}^{T-1}\eta^{kT+t}r_{m,(k,t)}\right]\\
&+\kappa_2\sum_{i=0}^{N-2}\left(\mathrm{E}_{\pi^{\mathrm{CM}}}.\right.\\
&\left.\mathrm{E}_{\pi_{i+1}^{\mathrm{CL}}}\left[\sum_{k=0}^{K-1}\gamma^k A_{\hat{\pi}_{i+1}^{\mathrm{CL}*}}(x_{i+1,k+1},\pi_{i+1}^{\mathrm{CL}}(S_{i+1,k+1}^{\mathrm{CL}}))\right]\right)\\
&\overset{(c)}{=}\kappa_1\sum_{m=0}^{M-1}\mathrm{E}_{\pi^{\mathrm{CM}}}\left[\sum_{k=0}^{K-1}\sum_{t=0}^{T-1}\eta^{kT+t}r_{m,(k,t)}\right]\\
&+\kappa_2\sum_{i=0}^{N-2}\left(J_{i+1}^{\mathrm{CL}}-\hat{J}_{i+1}^{\mathrm{CL}*}\right)\\
&\overset{(d)}{=}\kappa_1\sum_{m=0}^{M-1}\mathrm{E}_{\pi^{\mathrm{CM}}}\left[\sum_{k=0}^{K-1}\sum_{t=0}^{T-1}\eta^{kT+t}r_{m,(k,t)}\right]+\kappa_2\Delta J^{\mathrm{CL}}
\end{aligned}\tag{38}$$

where (a) follows from (24) and (25); (b) holds if $\eta^{kT+T-1}=\gamma^k$; (c) follows from Lemma 1; and (d) follows from (20). Thus, it is proved that the expected return as calculated in (23) is the same as the optimization objective in (22).

In the above derivation, we assume that $\eta^{kT+T-1}=\gamma^k$. Since the PC problem is essentially an infinite horizon problem as discussed in Section II-D, we have

$$\eta=\gamma^{\frac{k}{(k+1)T-1}}=\lim_{k\to+\infty}\gamma^{\frac{k}{(k+1)T-1}}=\gamma^{\frac{1}{T}}.\tag{39}$$

C. Proof of Lemma 2

If $x_{i,k}$ can be fully determined by $S_{i,k}^{\mathrm{CL}}$ with $x_{i,k}=f(S_{i,k}^{\mathrm{CL}})$, the relationship between $\hat{\pi}_i^{\mathrm{CL}*}(x_{i,k})$ and $\pi_i^{\mathrm{CL}*}(S_{i,k}^{\mathrm{CL}})$ can be established as

$$a_{i,k}^{\mathrm{CL}*}=\hat{\pi}_i^{\mathrm{CL}*}(x_{i,k})=\hat{\pi}_i^{\mathrm{CL}*}\left(f(S_{i,k}^{\mathrm{CL}})\right)=\pi_i^{\mathrm{CL}*}\left(S_{i,k}^{\mathrm{CL}}\right).\tag{40}$$

D. Proof of Lemma 3

The system dynamics evolve in discrete time on the basis of forward Euler discretization is

$$x_{i,k+1}=g\left(x_{i,k},a_{i,k}^{\mathrm{CL}},a_{i-1,k}^{\mathrm{CL}}\right)=Ax_{i,k}+Ba_{i,k}^{\mathrm{CL}}+Ca_{i-1,k}^{\mathrm{CL}}\tag{41}$$

where

$$A=\begin{bmatrix}1 & T & -h_iT & 0\\ 0 & 1 & -T & T\\ 0 & 0 & 1-\frac{T}{\tau} & 0\\ 0 & 0 & 0 & 1-\frac{T}{\tau}\end{bmatrix}, B=\begin{bmatrix}0\\ 0\\ \frac{T}{\tau}\\ 0\end{bmatrix}, C=\begin{bmatrix}0\\ 0\\ 0\\ \frac{T}{\tau}\end{bmatrix}. \quad (42)$$

Then

$$\begin{aligned} x_{i,k} &= g\Big(x_{i,k-1}, a^{\mathrm{CL}}_{i,k-1}, a^{\mathrm{CL}}_{i-1,k-1}\Big)\\ &= g\Big(g(x_{i,k-2}, a^{\mathrm{CL}}_{i,k-2}, a^{\mathrm{CL}}_{i-1,k-2}), a^{\mathrm{CL}}_{i,k-1}, a^{\mathrm{CL}}_{i-1,k-1}\Big)\\ &\quad \cdots\\ &= f\Big(x_{i,k-\tau_{i,k}}, \{a^{\mathrm{CL}}_{i,k'}\}^{k-1}_{k'=k-\tau_{i,k}}, \tau_{i,k}, \{a^{\mathrm{CL}}_{i-1,k'}\}^{k-1}_{k'=k-\tau_{i,k}}\Big)\\ &= f\Big(S^{\mathrm{CL}}_{i,k}, \{a^{\mathrm{CL}}_{i,k'}\}^{k-1}_{k'=k-\tau_{i,k}}\Big). \end{aligned} \quad (43)$$

Therefore, if the sequence of actions of vehicle $i-1$ within the time window of $[k-\tau_{i,k}+1, k]$ is available to vehicle i, the current driving status $x_{i,k}$ can be fully determined by the delayed observation $S^{\mathrm{CL}}_{i,k}$.

We would like to point out that the state space model in (41) ignores the random perturbations caused by mechanical noise due to practical system limitations. Therefore, the practical driving status $x_{i,k}$ might slightly deviate from the derived values by (43). However, the sequence of PC actions $\{a^{\mathrm{CL}}_{i,k'}\}^{k-1}_{k'=k-\tau_{i,k}}$ of vehicle $i-1$ still play the most important role in predicting the current driving status $x_{i,k}$ based on the delayed observation $S^{\mathrm{CL}}_{i,k}$.

E. Proof of Theorem 2

From Lemmas 2 and 3, we can derive that if vehicle $i-1$'s control actions $a^{\mathrm{CL}}_{i-1,k'}$ within the period $k' \in [k-\tau_{\max}, k-1]$ is available to vehicle i at control interval k, the optimal PC policy $\pi^{\mathrm{CL}*}_i(S^{\mathrm{CL}}_{i,k})$ based on delayed observation $S^{\mathrm{CL}}_{i,k}$ performs as well as the optimal policy $\hat{\pi}^{\mathrm{CL}*}_i(x_{i,k})$ based on the current driving status $x_{i,k}$ at vehicle i. In practice, only the control action $a^{\mathrm{CL}}_{i-1,k-\tau_{i,k}}$ of vehicle $i-1$ is available to vehicle i at control interval k. Consider the extreme case when $a^{\mathrm{CL}}_{i-1,k'}$ is a constant value within period $k' \in [k-\tau_{\max}, k-1]$, which means that $a^{\mathrm{CL}}_{i-1,k-\tau_{i,k}}$ can accurately represent all of vehicle $i-1$'s control actions within the time window. In this case, $x_{i,k}$ can be fully determined by $S^{\mathrm{CL}}_{i,k}$ according to Lemma 3. Thus, $\hat{\pi}^{\mathrm{CL}*}_i(x_{i,k})$ and $\pi^{\mathrm{CL}*}_i(S^{\mathrm{CL}}_{i,k})$ are equivalent from Lemma 2. In other words, if the variation of $a^{\mathrm{CL}}_{i-1,k'}$ is zero within period $k' \in [k-\tau_{\max}, k-1]$, the PC performance at vehicle i is not impacted by the observation delay. Extending to the general case, the smaller the variation of $a^{\mathrm{CL}}_{i-1,k'}$ within the period $k' \in [k-\tau_{\max}, k-1]$, the more accurate $a^{\mathrm{CL}}_{i-1,k-\tau_{i,k}}$ can be used to represent all the control actions of vehicle $i-1$ within the time window, and the smaller the impact of observation delay on PC performance of vehicle i.

REFERENCES

[1] S. E. Li et al., "Dynamical modeling and distributed control of connected and automated vehicles: Challenges and opportunities," *IEEE Intell. Transp. Syst. Mag.*, vol. 9, no. 3, pp. 46–58, Jul. 2017.

[2] 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.

[3] L. Lei, Y. Tan, K. Zheng, S. Liu, K. Zhang, and X. Shen, "Deep reinforcement learning for autonomous Internet of Things: Model, applications and challenges," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 3, pp. 1722–1760, 3rd Quart., 2020.

[4] S. Omidshafiei, J. Pazis, C. Amato, J. P. How, and J. Vian, "Deep decentralized multi-task multi-agent reinforcement learning under partial observability," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 2681–2690.

[5] H. Ye, G. Y. Li, and B.-H. F. Juang, "Deep reinforcement learning based resource allocation for V2V communications," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3163–3173, Apr. 2019.

[6] H. Yang, X. Xie, and M. Kadoch, "Intelligent resource management based on reinforcement learning for ultra-reliable and low-latency IoV communication networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 4157–4169, May 2019.

[7] K. Zia, N. Javed, M. N. Sial, S. Ahmed, A. A. Pirzada, and F. Pervez, "A distributed multi-agent RL-based autonomous spectrum allocation scheme in D2D enabled multi-tier HetNets," *IEEE Access*, vol. 7, pp. 6733–6745, 2019.

[8] Z. Nan, Y. Jia, Z. Ren, Z. Chen, and L. Liang, "Delay-aware content delivery with deep reinforcement learning in Internet of Vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 8918–8929, Jul. 2021.

[9] L. Liang, H. Ye, and G. Y. Li, "Spectrum sharing in vehicular networks based on multi-agent reinforcement learning," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2282–2292, Oct. 2019.

[10] H. V. Vu, Z. Liu, D. H. Nguyen, R. Morawski, and T. Le-Ngoc, "Multi-agent reinforcement learning for joint channel assignment and power allocation in platoon-based C-V2X systems," 2020, *arXiv:2011.04555*.

[11] P. Xiang, H. Shan, M. Wang, Z. Xiang, and Z. Zhu, "Multi-agent RL enables decentralized spectrum access in vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 70, no. 10, pp. 10750–10762, Oct. 2021.

[12] H. Zhang et al., "Mean-field-aided multiagent reinforcement learning for resource allocation in vehicular networks," *IEEE Internet Things J.*, vol. 10, no. 3, pp. 2667–2679, Feb. 2023.

[13] M. Parvini, M. R. Javan, N. Mokari, B. Abbasi, and E. A. Jorswieck, "AoI-aware resource allocation for platoon-based C-V2X networks via multi-agent multi-task reinforcement learning," *IEEE Trans. Veh. Technol.*, vol. 72, no. 8, pp. 9880–9896, Aug. 2023.

[14] Y. Xu, K. Zhu, H. Xu, and J. Ji, "Deep reinforcement learning for multi-objective resource allocation in multi-platoon cooperative vehicular networks," *IEEE Trans. Wireless Commun.*, vol. 22, no. 9, pp. 6185–6198, Sep. 2023.

[15] Y. Zhong, B. Wang, and Y. Wang, *Reward Backpropagation Prioritized Experience Replay*, Stanford Univ., Palo Alto, CA, USA, 2017.

[16] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.

[17] T. Liu, L. Lei, K. Zheng, and K. Zhang, "Autonomous platoon control with integrated deep reinforcement learning and dynamic programming," *IEEE Internet Things J.*, vol. 10, no. 6, pp. 5476–5489, Mar. 2023.

[18] Y. S. Nasir and D. Guo, "Deep reinforcement learning for distributed dynamic power allocation in wireless networks," 2018, *arXiv:1808.00490*.

[19] S. Kaul, R. Yates, and M. Gruteser, "Real-time status: How often should one update?" in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, 2012, pp. 2731–2735.

[20] A. Kosta, N. Pappas, A. Ephremides, and V. Angelakis, "Age and value of information: Non-linear age case," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, 2017, pp. 326–330.

[21] Z. Wang, M.-A. Badiu, and J. P. Coon, "A value of information framework for latent variable models," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2020, pp. 1–6.

[22] Z. Wang, M.-A. Badiu, and J. P. Coon, "A framework for characterizing the value of information in hidden Markov models," *IEEE Trans. Inf. Theory*, vol. 68, no. 8, pp. 5203–5216, Aug. 2022.

[23] O. Ayan, M. Vilgelm, M. Klügel, S. Hirche, and W. Kellerer, "Age-of-information vs. value-of-information scheduling for cellular networked control systems," in *Proc. 10th ACM/IEEE Int. Conf. Cyber-Physical Syst.*, 2019, pp. 109–117.

[24] J. Foerster et al., "Stabilising experience replay for deep multi-agent reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1146–1155.

[25] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double *Q*-learning," in *Proc. 30th AAAI Conf. Artif. Intell.*, 2016, pp. 2094–2100.

[26] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
[27] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6382–6393.

Lei Lei (Senior Member, IEEE) received the B.S. and Ph.D. degrees in telecommunications engineering from Beijing University of Posts and Telecommunications, Beijing, China, in 2001 and 2006, respectively.

She is currently an Associate Professor with the College of Engineering and Physical Sciences, University of Guelph, Guelph, ON, Canada. Her research interests mainly lie in machine learning/deep reinforcement learning, Internet of Things/Internet of Vehicles, mobile edge computing, and smart grid.

Tong Liu (Graduate Student Member, IEEE) received the B.S. degree from Nanjing University of Posts and Telecommunications, Nanjing, China, in 2018. He is currently pursuing the Ph.D. degree with Beijing University of Posts and Telecommunications, Beijing, China.

His current research interests include deep reinforcement learning, modern control theory, and their application in Internet of Vehicles.

Kan Zheng (Fellow, IEEE) received the B.S., M.S., and Ph.D. degrees from Beijing University of Posts and Telecommunications, Beijing, China, in 1996, 2000, and 2005, respectively.

He is currently a Full Professor with Ningbo University, Ningbo, Zhejiang, China. He has rich experiences in research and standardization of new emerging technologies toward 6G. He has authored over 200 journal articles and conference papers in the field of wireless communications, vehicular networks, IoT, and security.

Prof. Zheng holds editorial board positions with several journals. He has also served in the organizing/TPC committees for more than ten conferences.

Xuemin (Sherman) Shen (Fellow, IEEE) received the Ph.D. degree in electrical engineering from Rutgers University, New Brunswick, NJ, USA, in 1990.

He is a University Professor with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. His research focuses on network resource management, wireless network security, Internet of Things, 5G and beyond, and vehicular networks.

Dr. Shen received the Canadian Award for Telecommunications Research from the Canadian Society of Information Theory in 2021, the R.A. Fessenden Award in 2019 from IEEE, Canada, the Award of Merit from the Federation of Chinese Canadian Professionals (Ontario) in 2019, the James Evans Avant Garde Award in 2018 from the IEEE Vehicular Technology Society, the Joseph LoCicero Award in 2015 and Education Award in 2017 from the IEEE Communications Society (ComSoc), and the Technical Recognition Award from Wireless Communications Technical Committee in 2019 and AHSN Technical Committee in 2013. He has also received the Excellent Graduate Supervision Award in 2006 from the University of Waterloo and the Premier's Research Excellence Award in 2003 from the Province of Ontario, Canada. He served as the Technical Program Committee Chair/Co-Chair for IEEE Globecom'16, IEEE Infocom'14, IEEE VTC'10 Fall, IEEE Globecom'07, and the Chair for the IEEE ComSoc Technical Committee on Wireless Communications. He is the President of the IEEE ComSoc. He was the Vice President for Technical and Educational Activities, the Vice President for Publications, the Member-at-Large on the Board of Governors, the Chair of the Distinguished Lecturer Selection Committee, and the member of IEEE Fellow Selection Committee of the ComSoc. He served as the Editor-in-Chief for the IEEE INTERNET OF THINGS JOURNAL, IEEE NETWORK, and *IET Communications*. He is a Registered Professional Engineer of Ontario, Canada, an Engineering Institute of Canada Fellow, a Canadian Academy of Engineering Fellow, a Royal Society of Canada Fellow, a Chinese Academy of Engineering Foreign Member, and a Distinguished Lecturer of the IEEE Vehicular Technology Society and Communications Society.