

# Deep Multi-Agent Reinforcement Learning for Highway On-Ramp Merging in Mixed Traffic

Dong Chen<sup>ID</sup>, Graduate Student Member, IEEE, Mohammad R. Hajidavalloo<sup>ID</sup>, Zhaojian Li<sup>ID</sup>, Senior Member, IEEE, Kaian Chen<sup>ID</sup>, Yongqiang Wang<sup>ID</sup>, Senior Member, IEEE, Longsheng Jiang<sup>ID</sup>, and Yue Wang<sup>ID</sup>, Senior Member, IEEE

**Abstract**—On-ramp merging is a challenging task for autonomous vehicles (AVs), especially in mixed traffic where AVs coexist with human-driven vehicles (HDVs). In this paper, we formulate the mixed-traffic highway on-ramp merging problem as a multi-agent reinforcement learning (MARL) problem, where the AVs (on both merge lane and through lane) collaboratively learn a policy to adapt to HDVs to maximize the traffic throughput. We develop an efficient and scalable MARL framework that can be used in dynamic traffic where the communication topology could be time-varying. Parameter sharing and local rewards are exploited to foster inter-agent cooperation while achieving great scalability. An action masking scheme is employed to improve learning efficiency by filtering out invalid/unsafe actions at each step. In addition, a novel priority-based safety supervisor is developed to significantly reduce collision rate and greatly expedite the training process. A gym-like simulation environment is developed and open-sourced with three different levels of traffic densities. We exploit curriculum learning to efficiently learn harder tasks from trained models under simpler settings. Comprehensive experimental results show the proposed MARL framework consistently outperforms several state-of-the-art benchmarks.

**Index Terms**—Multi-agent deep reinforcement learning, connected autonomous vehicles, safety enhancement, on-ramp merging.

## I. INTRODUCTION

AUTONOMOUS vehicle (AV) technologies, such as Tesla Autopilot [1] and Baidu Apollo [2], have already been deployed in (semi-)autonomous vehicles on real-world roads. Despite the great advances over the past decade that have made this possible, the number of traffic accidents involving AVs are increasing in recent years [3], [4]. The accidents are often caused by the inability of AVs to timely react to the dynamic

Manuscript received 7 May 2021; revised 18 January 2022, 12 June 2022, 28 March 2023, and 26 May 2023; accepted 8 June 2023. This work was supported in part by the National Science Foundation under Project CNS-2219488. The Associate Editor for this article was Y. Wang. (Corresponding author: Zhaojian Li.)

Dong Chen, Mohammad R. Hajidavalloo, Zhaojian Li, and Kaian Chen are with the Department of Mechanical Engineering, Michigan State University, Lansing, MI 48824 USA (e-mail: chendon9@msu.edu; hajidava@msu.edu; lizhaoj1@msu.edu; chenka@msu.edu).

Yongqiang Wang is with the Department of Electrical and Computer Engineering, Clemson University, Clemson, SC 29630 USA (e-mail: yongqiw@clemson.edu).

Longsheng Jiang and Yue Wang are with the Department of Mechanical Engineering, Clemson University, Clemson, SC 29634 USA (e-mail: longshj@g.clemson.edu; yue6@g.clemson.edu).

Digital Object Identifier 10.1109/TITS.2023.3285442

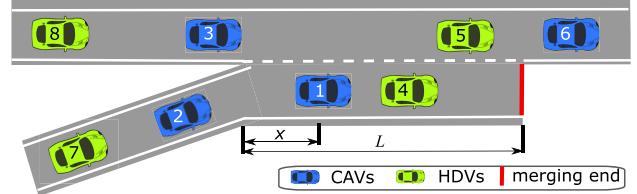


Fig. 1. Illustration of the considered on-ramp merging traffic scenario. CAVs (blue) and HDVs (green) coexist on both ramp and through lanes.

driving environment, especially in mixed traffic with both AVs and human-driven vehicles (HDVs); the AVs need not only to react to road objects but also to attend to the behaviors of HDVs. Among the many challenging driving scenarios, highway on-ramp merging is one of the most difficult tasks for AVs [5], [6], which is the topic of this paper.

The considered on-ramp merging scenario is illustrated in Fig. 1, where we consider a general setup that AVs and HDVs coexist on both merge lane and through lane. On-ramp vehicles need to efficiently merge onto the through lane without collision. In an ideal cooperative setting, the vehicles on the through lane should proactively decelerate or accelerate to make adequate space for on-ramp vehicles to safely merge whereas the on-ramp vehicles also adjust speed and promptly cut in when it is safe, to avoid deadlock situations [7], [8]. It is clear that coordination between the vehicles is a crucial enabler for safe and efficient merging maneuvers. While this is relatively easy to achieve in a full-AV scenario, AV coordination in the presence of HDVs is a significantly more challenging task.

Rule-based and optimization-based methods have been proposed to tackle the automated merging problem [9], [10], [11], [12]. In particular, rule-based approaches employ heuristics and hard-coded rules to guide the AVs [10], [11]. While this is feasible for simple traffic scenarios, these methods quickly become impractical in more complex merging scenarios [13]. In an optimal control setting, vehicle interactions are modeled as a dynamic system with actions from controlled vehicles as inputs. For example, a model predictive control (MPC) approach is developed to control an AV to merge in a parallel-type ramp [13]. While promising results are demonstrated, the MPC-based methods rely on accurate dynamic merging models (including human driving models) and are typically computationally-involved as online

optimizations are needed at each time step [14]. Extensive surveys on model-based control strategies for on-ramp merging are presented in [15], [16], and [17]. However, those approaches only consider fully automated vehicles, which are not applicable to the considered mixed-traffic setting. The theory of gap acceptance for merging behavior is also explored in [18] and [19] with complex modeling characteristics of all traffic participants, which is not practical when the HDVs behave differently and there are multiple co-existed CAVs [20].

On the other hand, data-driven methods such as reinforcement learning (RL) have received increased attention and been explored for AV highway merging [12], [21]. Specifically, a multi-objective reward function for safety and jerk minimization is designed for AV merging and the Deep Deterministic Policy Gradient (DDPG) algorithm [22] is exploited to solve the RL problem in [12]. In [21], RL and MPC are integrated to promote learning efficiency, which achieves a good trade-off between passenger comfort, efficiency, crash rate, and robustness. However, those approaches are only designed for a single AV, treating all other vehicles as part of the environment.

In this paper, we treat a general setup (see Fig. 1) where multiple AVs learn to adapt to HDVs and cooperatively accomplish merging tasks to maximize traffic throughput safely. As a result, it is natural to extend the single-agent RL to a multi-agent reinforcement learning (MARL) framework where the AVs collaboratively learn control policies to achieve the aforementioned goal (see Section II-B for a review of state-of-the-art MARL algorithms). However, this is a challenging task due to dynamic connectivity topology, complex motion patterns involving AV-coupled dynamics, and intricate decision makings. This complexity is even more pronounced when human drivers are involved.

While several MARL approaches have been developed for connected and autonomous vehicles (CAVs) in car-following and lane overtaking scenarios [23], [24], [25], [26], [27], [28], [29], to the best of our knowledge, no MARL algorithm has been developed for the considered highway on-ramp merging scenario. In this work, we develop a novel decentralized MARL framework to enable AVs to efficiently learn a safe and efficient policy in the highway on-ramp merging scenario where a general policy is learned for vehicles on both lanes. A priority-based safety supervisor is designed to enhance safety and improve learning efficiency, through sequential and multi-step predictions. Parameter sharing and local rewards are exploited to foster inter-agent cooperation while achieving great scalability. The main contributions and the technical advancements of this paper are summarized as follows.

- 1) We formulate the mixed-traffic on-ramp merging problem (with AVs and HDVs coexisting on both ramp and through lanes) as a decentralized MARL problem. The formulation can allow for a dynamic environment with a time-varying connectivity topology. A corresponding gym-like simulation platform with three different levels of traffic density is developed and open-sourced.<sup>1</sup>

- 2) We develop a novel, efficient, and scalable MARL algorithm, featuring a parameter-sharing mechanism, effective reward function design, and action masking. Furthermore, a priority-based safety supervisor is developed, which significantly reduces collision rates in training and subsequently improves learning efficiency.
- 3) We employ curriculum learning to speed up the learning for harder tasks by building upon trained models from less complex traffic scenarios.
- 4) We conduct comprehensive experiments, and the results show that the proposed approach consistently outperforms several state-of-the-art algorithms in terms of driving safety and efficiency.

The remainder of the paper is organized as follows. Section II briefly introduces RL and MARL and reviews state-of-the-art algorithms. The problem formulation and the proposed MARL framework are described in Section III whereas the priority-based safety supervisor is detailed in Section IV. Experiments, results, and discussions are presented in Section V. We conclude the paper and discuss future works in Section VI.

## II. BACKGROUND

In this section, we review the preliminaries of RL and introduce several state-of-the-art MARL algorithms to put our proposed work in the proper context. Readers who are familiar with the RL and MARL literature can skip this section and jump to Section III directly.

### A. Preliminaries of Reinforcement Learning (RL)

In an RL setting, at each time step  $t$ , the agent observes the state  $s_t \in \mathcal{S} \subseteq \mathbb{R}^n$ , takes an action  $a_t \in \mathcal{A} \subseteq \mathbb{R}^m$ , and subsequently receives a reward signal  $r_t \in \mathbb{R}$  and an updated state  $s_{t+1}$  at time  $t + 1$  from the environment. The goal of the RL agent is to learn an optimal policy  $\pi^* : \mathcal{S} \rightarrow \mathcal{A}$ , a mapping from state to action, that maximizes the accumulated reward  $R_t = \sum_{k=0}^T \gamma^k r_{t+k}$ , where  $r_{t+k}$  is the reward at time step  $t+k$  and  $\gamma \in (0, 1]$  is the discount factor that quantifies the relative importance one wants to place on future rewards.

The state-action value function (or Q-function) under policy  $\pi$ , denoted by  $Q^\pi(s_t, a_t)$ , is an estimation of the expected return (accumulated reward in an infinite horizon) if starting from state  $s_t$ , taking an immediate action  $a_t$ , and then following policy  $\pi$  afterwards. The optimal Q-function can be characterized by the following Bellman equation,  $Q^*(s_t, a_t) = E[r(s_t, a_t) + \gamma \sum_{s_{t+1}} P(s_{t+1}|s_t, a_t) \max_{a_{t+1}} Q^*(s_{t+1}, a_{t+1})]$ , where the next state  $s_{t+1}$  is sampled from the environment's transition rules  $P(s_{t+1}|s_t, a_t)$ . The state value function of a state  $s_t$  under policy  $\pi$ ,  $V^\pi(s_t)$ , is defined as the expected return if starting from  $s_t$  and immediately following policy  $\pi$ , i.e.,  $V^\pi(s_t) = E_\pi[R_t|s_t = s]$ . Often the agent's policy is parameterized by some parameters  $\theta$  and the goal is to learn appropriate  $\theta$  to achieve desired system behavior. In actor-critic (A2C) algorithms [30], two networks are employed: a critic network parameterized by  $\phi$  to learn the value function  $V_\phi^{\pi_\theta}(s_t)$  and an actor network  $\pi_\theta(a_t|s_t)$  parameterized by  $\theta$ .

<sup>1</sup>See [https://github.com/DongChen06/MARL\\_CAVs](https://github.com/DongChen06/MARL_CAVs)

The policy network is updated by maximizing the following objective function:

$$J^{\pi_\theta} = E_{\pi_\theta} [\log \pi_\theta(a_t | s_t) A_t], \quad (1)$$

where  $A_t = Q^{\pi_\theta}(s_t, a_t) - V_\phi(s_t)$  is the advantage function that characterizes the improvement on the reward if taking action  $a_t$  over the average reward of all possible actions taken at state  $s_t$  [30]. The value function parameter  $\phi$  is updated by minimizing the following loss function:

$$J^{V_\phi} = \min_{\phi} E_{\mathcal{D}} \left( R_t + \gamma V_{\phi'}(s_{t+1}) - V_\phi(s_t) \right)^2, \quad (2)$$

where  $\mathcal{D}$  denotes an experience replay buffer that collects previously encountered experiences and  $\phi'$  denotes the parameters obtained from earlier iterations used in a target network [31].

### B. Multi-Agent Reinforcement Learning (MARL)

MARL has found great success across a wide range of multi-agent systems, including traffic light control [32], games [33], resource management in wireless networks [34], and powergrid control [35], only to name a few. MARL algorithms can be categorized into two main classes: *cooperative* and *non-cooperative*. In this paper, we will focus on the *cooperative* setting where all agents are encouraged to cooperate to achieve a common goal, i.e., safely maneuver with maximum traffic efficiency. We next introduce a few state-of-the-art cooperative MARL algorithms that we will use as benchmarks for comparison in Section V.

An independent MARL framework, called IQN, is proposed in [36], allowing each agent to learn independently and simultaneously while viewing other agents as part of the environment. While fully scalable, it suffers from non-stationarity and partial observability. An off-policy MARL algorithm is proposed in [37] where collaboration is achieved by estimating the state-action value function using a centralized critic network based on global observations and actions. In [32] a learnable communication protocol and a spatial distance factor are proposed to scale down the reward signals of neighboring agents during training. Experimental results show good scalability and improved cooperation among agents. However, these MARL approaches only consider a stationary environment with fixed communication topology, and thus the algorithms need to be re-designed and/or re-trained whenever the communication typology changes.

Recently, parameter sharing is widely applied in MARL settings with homogeneous agents [25], [38], [39], which bootstraps single-agent RL methods and learns an identical policy for each agent, and thus enables the handling of changes in the number of participating agents. In [39], several state-of-the-art single RL algorithms (i.e., PPO [40] and ACKTR [41]) are extended to the MARL with parameter sharing denoted as MAPPO and MAACKTR. A parameter-sharing A2C (MA2C) algorithm is proposed in [38] to solve the fleet management problem and experimental results are given to confirm the performance. These methods will be used as benchmarks for performance comparison in Section V.

Several recent works also address safety issues in MARL problems. For example, a centralized shielding approach is

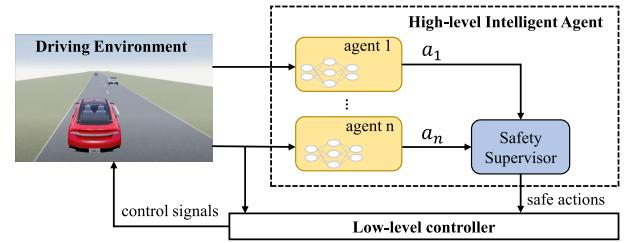


Fig. 2. Schematics of system and simulation setup.

introduced in [42], where a centralized model is used to monitor the joint actions of all agents and restrict unsafe actions. To address the scalability problem of centralized supervision, a local shielding approach is developed for only a subset of agents. Experiments in two-player navigation games in the grid world show good performance on collision avoidance. In addition, [43] proposes a decentralized control barrier function that shields unsafe actions based on available local information. They demonstrate the performance of the proposed approach using patrol tasks where two agents navigate in an environment with obstacles and walls. However, these methods consider an all-autonomous, cooperative agent environment, without considering moving objects like HDVs.

To fill the aforementioned gaps, in this paper, we develop a novel on-policy MARL algorithm for the considered on-ramp merging problem with great efficiency and safety, which features action masking, priority-based safety supervisor, parameter sharing, and local reward shaping. Performance comparisons between the proposed algorithm and the above benchmarks are presented in Section V.

## III. RAMP MERGING AS MARL

In this section, we first formulate the considered on-ramp merging problem as a partially observable Markov decision process (POMDP) [44]. Then we present our actor-critic-based MARL algorithm, featuring a parameter-sharing mechanism, effective reward function design, and action masking, to solve the formulated POMDP, which is denoted as the baseline method in Section V.

### A. Problem Formulation

In this paper, we model the on-ramp merging environment in mixed traffic as a model-free multi-agent network [32], [35],  $\mathcal{G} = (\nu, \varepsilon)$ , where each agent  $i \in \nu$  communicates with its neighbors  $\mathcal{N}_i := \{j | \varepsilon_{ij} \in \varepsilon\}$  through the edge connections  $\varepsilon_{ij}, i \neq j$ . Let  $S := \times_{i \in \nu} S_i$  and  $\mathcal{A} := \times_{i \in \nu} \mathcal{A}_i$  denote the global state space and action space, respectively. The underlying dynamics of the system can be characterized by the state transition distribution  $\mathcal{P}: S \times \mathcal{A} \times S \rightarrow [0, 1]$ . We consider a decentralized MARL framework where each agent  $i$  (AV  $i$ ) only observes a part of the environment (i.e., surrounding vehicles). This is consistent with the reality that AVs can only sense or communicate with vehicles in close vicinity, making the overall dynamical system a POMDP  $\mathcal{M}_{\mathcal{G}}$ , which can be described by the following tuple  $(\{\mathcal{A}_i, \mathcal{S}_i, \mathcal{R}_i\}_{i \in \nu}, \mathcal{T})$ :

- **Action Space:** The action space  $\mathcal{A}_i$  of agent  $i$  is defined as the set of high-level control decisions, including *turn left*, *turn right*, *cruising*, *speed up*, and *slow down* following the designs in [45] and [46]. With a selected high-level decision, lower-level controllers will then produce the corresponding steering and throttle control signals to maneuver the AVs. The system and simulation setup is illustrated in Fig. 2. The overall action space of the system is the joint actions from all AVs, i.e.,  $\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2 \times \dots \times \mathcal{A}_N$ .

- **State Space:** The state of agent  $i$ ,  $\mathcal{S}_i$ , is defined as a matrix of dimension  $N_{\mathcal{N}_i} \times W$ , where  $N_{\mathcal{N}_i}$  is the number of observed vehicles and  $W$  is the number of features used to represent the state of a vehicle, including:

- *ispresent*: a binary variable denoting whether a vehicle is observable in the vicinity of the ego vehicle.
- $x_l$ : the longitudinal position of the observed vehicle relative to the ego vehicle.
- $y$ : the lateral position of the observed vehicle relative to the ego vehicle.
- $v_x$ : the longitudinal speed of the observed vehicle relative to the ego vehicle.
- $v_y$ : the lateral speed of the observed vehicle relative to the ego vehicle.

We assume that only the “neighboring vehicles” can be observed by the ego vehicle. The “neighboring vehicles” are defined as the nearest  $N_{\mathcal{N}_i}$  vehicles that are within a 150 m longitudinal distance from the ego vehicle due to the local observability [29]. In the considered on-ramp merging case as shown in Fig. 1, we found  $N_{\mathcal{N}_i} = 5$  achieves the best performance. The entire state of the system is then the Cartesian product of the individual states, i.e.,  $\mathcal{S} = \mathcal{S}_1 \times \mathcal{S}_2 \times \dots \times \mathcal{S}_N$ .

- **Reward Function:** The reward function  $\mathcal{R}_i$  is crucial to train the RL agents so that it follows desired behaviors. As the objective is to train our agents to safely and efficiently pass the merging area, the reward for the  $i$ th agent at time step  $t$  is defined as follows:

$$r_{i,t} = w_c r_c + w_s r_s + w_h r_h + w_m r_m, \quad (3)$$

where  $w_c$ ,  $w_s$ ,  $w_h$ , and  $w_m$  are positive weighting scalars corresponding to collision evaluation  $r_c$ , stable-speed evaluation  $r_s$ , headway time evaluation  $r_h$ , and merging cost evaluation  $r_m$ , respectively. As safety is the most important criterion, we make  $w_c$  much bigger than other weights to prioritize safety. The performance evaluation on different  $w_c$  values is discussed in Section V-A and Table I. The four performance metrics are defined as follows:

- the collision evaluation  $r_c$  is set to -1 if collision happens, otherwise  $r_c = 0$ .
- the speed evaluation  $r_s$  is defined as

$$r_s = \min \left\{ \frac{v_t - v_{min}}{v_{max} - v_{min}}, 1 \right\}, \quad (4)$$

where  $v_t$  is the current speed of the ego vehicle. Combining the speed recommendation from the US Department of Transportation (20-30 m/s [47]) and

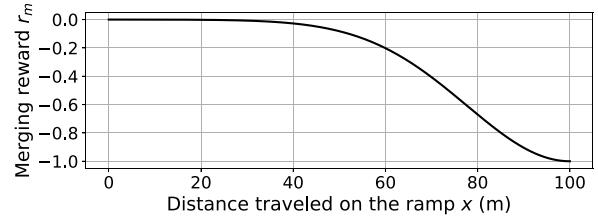


Fig. 3. Illustration of the designed merging reward/penalty.

the speed range observed in the Next Generation Simulation (NGSIM) dataset<sup>2</sup> (the minimum speed at 6-8 m/s [48]), we set the minimum and maximum speeds of the ego vehicle as  $v_{min} = 10$  m/s, and  $v_{max} = 30$  m/s, respectively.

- the time headway evaluation is defined as:

$$r_h = \log \frac{d_{\text{headway}}}{t_h v_t}, \quad (5)$$

where  $d_{\text{headway}}$  is the distance headway and  $t_h$  is a predefined time headway threshold. As such, the ego vehicle will get penalized when the time headway is less than  $t_h$  and rewarded only when the time headway is greater than  $t_h$ . In this paper, we choose  $t_h$  as 1.2 s as suggested in [49].

- The merging cost  $r_m$  is designed to penalize the waiting time on the merge lane to avoid deadlocks [50]. Here we adopt  $r_m = -\exp(-(x - L)^2/10L)$ , where  $x$  is the distance the ego vehicle has navigated on the ramp and  $L$  is the length of the ramp (see Fig. 1). The merging cost function is plotted in Fig. 3, which shows that the penalty increases as the ego vehicle moves closer to the merging end.

- **Transition Probabilities:** the transition probability  $\mathcal{T}(s'|s, a)$  characterizes the dynamics of the system. In the developed simulator, we exploit the intelligent driver model (IDM) [51] and MOBIL model [52] for longitudinal acceleration and lateral lane change decisions of HDVs, respectively. The high-level decisions of AVs are made by the MARL algorithm and will be tracked by the lower-level controller (PID controller) (see Fig. 2). A kinematic bicycle model [53] is used to propagate vehicle trajectories. We do not assume any prior knowledge of the transition probability in the development of our MARL algorithm.

## B. MA2C for CAVs

In the cooperative MARL setting, the objective is to maximize the global reward  $R_{g,t} = \sum_{i=1}^N r_{i,t}$ . Ideally, each agent will be assigned with the same average global reward  $R_t = \frac{1}{N} R_{g,t}$  during training, i.e.,  $r_{1,t} = r_{2,t} = \dots = r_{N,t}$ . However, this shared reward approach does not accurately indicate the contributions of each vehicle and can lead to several issues [54], [55]. First, aggregating the global reward can cause large latency and increase communication

<sup>2</sup><https://ops.fhwa.dot.gov/trafficanalysistools/ngsim.htm>

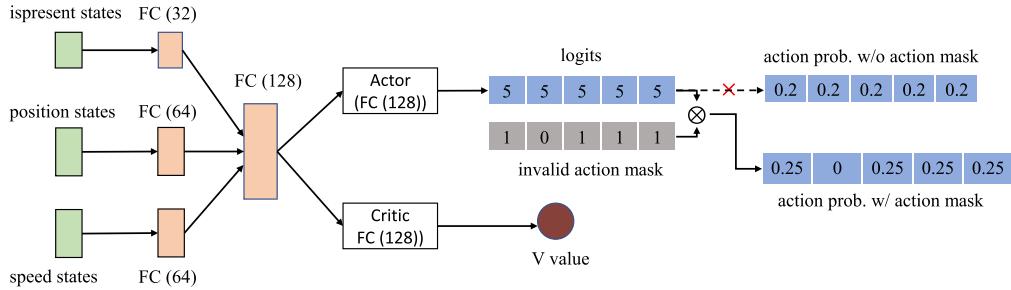


Fig. 4. Architecture of the proposed network structure. The numbers in the parentheses denote the size of the layers. “w/o” and “w/” represent “without” and “with”, respectively.

overheads, which is problematic for systems with real-time constraints such as AVs. Second, a single global reward leads to the credit assignment problem [56], which can significantly impede learning efficiency and limit the number of agents to a small size. Therefore, in this paper, we adopt a local reward assignment strategy, where each ego vehicle is only affected by its neighboring vehicles. Specifically, the reward for the  $i$ th agent at time  $t$  is defined as:

$$r_{i,t} = \frac{1}{|\nu_i|} \sum_{j \in \nu_i} r_{j,t}, \quad (6)$$

where  $\nu_i = i \cup \mathcal{N}_i$  is a set containing the ego vehicle and its neighbors, and  $|\cdot|$  denotes the cardinality of a set. This local reward design only includes rewards from agents that are most related to the success or failure of a task [38], [57]. This is appropriate for on-road vehicles as a vehicle only interacts with its surrounding vehicles and distant vehicles have limited impact on the ego vehicle.

The used network backbone is shown in Fig. 4, where the actor network and the critic network share the same low-level representations, and the policy loss and the value function error loss are thus combined into a single loss function [40]. With the shared network parameters, the overall loss function takes the following form:

$$J(\theta_i) = J^{\pi_{\theta_i}} - \beta_1 J^{V_{\phi_i}} + \beta_2 H(\pi_{\theta_i}(s_t)), \quad (7)$$

where  $\beta_1$  and  $\beta_2$  are the weighting coefficients for the value function loss and the entropy regularization term,  $H(\pi_{\theta_i}(s_t)) = E_{\pi_{\theta_i}}[-\log(\pi_{\theta_i}(s_t))]$ , used to encourage the agents to explore new states [40], [58], respectively. From Eq. 1, it follows that the policy loss can be written as:

$$J^{\pi_{\theta_i}} = E_{\pi_{\theta_i}} \left[ \log \pi_{\theta_i}(a_{i,t}|s_{i,t}) A_{i,t}^{\pi_{\theta_i}} \right], \quad (8)$$

where  $A_{i,t}^{\pi_{\theta_i}} = r_{i,t} + \gamma V^{\pi_{\phi_i}}(s_{i,t+1}) - V^{\pi_{\phi_i}}(s_{i,t})$  is the advantage function and  $V^{\pi_{\phi_i}}(s_{i,t})$  is the state value function. The loss for updating the state value  $V_{\phi_i}$  is in the following form:

$$J^{V_{\phi_i}} = \min_{\phi_i} E_{\mathcal{D}_i} \left[ r_{i,t} + \gamma V_{\phi_i}(s_{i,t+1}) - V_{\phi_i}(s_{i,t}) \right]^2. \quad (9)$$

We use separate experience reply buffers for each agent but the same policy network is updated with the same network parameters among agents. This is suitable as we train a general policy for both on-ramp and through AVs [25], [38]. Minibatches of sampled trajectories are exploited to update the network parameters using Eq. 7 to reduce the variance.

### C. DNN Settings

The deep neural network design is illustrated in Fig. 4. Specifically, to improve scalability and robustness, we regroup the observation  $s_{i,t}$  according to their physical units. For instance, the observation  $s_{i,t}$  is divided into three groups:  $s_{i,t}^1 \cup s_{i,t}^2 \cup s_{i,t}^3$  according to their units, representing *ispresent states*, *position states* and *speed states*, respectively. Each of the three sub-state vectors is encoded by one fully connected (FC) layer and the three encoded states are then concatenated into a single vector. The concatenated vector is fed into a 128-neuron FC layer, the result of which is consumed by both the actor network and the critic network. In a standard setting, the logits  $l_i$  from the actor network will go to a Softmax layer, producing the probability by  $\pi_{\theta_i}(s_i) = \text{softmax}([l_1, l_2, l_3, l_4, l_5])$ , which is used to sample the actions, i.e.,  $a_i \sim \pi_{\theta_i}(s_i)$ .

However, this sampling procedure has several issues. First, invalid/unsafe actions are also assigned with non-zero probabilities; as a stochastic policy is used, these unsafe actions may be sampled during training, which can lead to undesirable system behaviors and even system breakdown. Second, sampling invalid/unsafe actions also impedes policy training as invalid policy updates [59] are executed for invalid actions, since the collected experiences associated with the invalid actions are not meaningful and misleading. To address these issues, we adopt the invalid action masking approach [38] which “masks out” invalid actions and only samples from valid actions. As shown in Fig. 4, with an invalid action mask obtained from the environment (e.g., based on the traffic scenario) where “0” represents an invalid action and “1” denotes a valid action, the corresponding logits of invalid actions are replaced with large negative values, e.g.,  $-1e8$ . As a result, the probability of the invalid actions after the Softmax layer is very close to 0, and sampling from invalid actions can thus be avoided, equivalently “renormalizing the probability distribution” [39]. In this work, we consider the following invalid actions:

- the ego vehicle attempts to make lane changes to a non-existing lane. For example, the ego vehicle tries to make a left turn when it is already on the leftmost lane.
- the ego vehicle attempts to speed up or slow down when its speed has already reached the predefined maximum or minimum speed.

Note that we only include the two most basic invalid actions here and other unsafe actions will be further checked and

regulated by the proposed priority-based safety supervisor in Section IV.

#### IV. PRIORITY-BASED SAFETY ENHANCEMENT

While obvious invalid actions can be avoided using the rule-based action masking scheme described above, it cannot prevent inter-vehicle or vehicle-obstacle collisions. Therefore, a more comprehensive safety supervisor is needed to deal with collisions in complex, dynamic, and cluttered mix-traffic environments. Towards that end, we propose a new safety-enhancement scheme by exploiting vehicle dynamics and multi-step predictions. The goal is to predict any potential collisions over a prediction horizon  $T_n$  and correct the unsafe (exploratory) actions accordingly. As we consider mixed traffic with HDVs, a proper model is needed to predict the high-level decisions of human drivers. In this paper, we use IDM [51] to predict the longitudinal acceleration of the HDVs, based on the current speed and distance headway. In addition, we exploit the MOBIL lane change model [52] to predict the lane-changing behavior of HDVs, which makes a lane-changing decision when it is safe and there is an extra acceleration gain. The high-level decisions of AVs are generated by the MARL agent with the actions defined in Section III-A. These high-level acceleration and lane-change decisions will be realized through low-level PID controllers. The vehicle trajectories are then propagated based on the kinematic bicycle model [53]. We call the high-level decision-induced trajectories motion primitives and show the described framework and simulation setup in Fig. 2.

##### A. Priority Assignment

With the HDV motion models, one can predict whether a collision can happen in the next  $T_n$  steps based on the joint motion primitives from all AVs. Therefore, it is attempting to use joint action from all AVs to design the safety-enhancement scheme. However, while it is relatively straightforward to determine whether collisions can happen given a joint action, it is very computationally costly to determine a joint safe action, if a collision is detected, as the action space is  $|\mathcal{A}_i|^N$  with  $N$  being the number of AVs. It quickly becomes computationally intractable as  $N$  grows, especially considering that the considered application has stringent real-time constraints. As such, we propose a sequential, priority-based safety enhancement scheme that has great computation efficiency and is thus suitable for real-time implementations. The principle is that we coordinate the AVs in a sequential order, prioritizing AVs with smaller safety margins. For example, AVs near the end of the merge lane or near the defined safety boundary (e.g. distance headway very close to the defined threshold) should have higher priorities.

More specifically, the following rationales are considered for priority assignments:

- 1) Vehicles on the merge lane should have higher priorities compared to vehicles on the through lane as vehicles on the merge lane face a time-critical merging task (due to the merging lane end).

- 2) Merging vehicles closer to the merge lane end should have higher priorities as they are more probable to cause collisions and deadlocks [50].
- 3) Vehicles with smaller time headway should have a higher priority as they are more likely to collide with the preceding vehicles.

Based on the above rationales, we construct the priority index  $p_i$  of the ego vehicle  $i$  as follows:

$$p_i = \alpha_1 p_m + \alpha_2 p_d + \alpha_3 p_h + \sigma_i, \quad (10)$$

where  $\alpha_1$ ,  $\alpha_2$ , and  $\alpha_3$  are positive weighting factors for the merging priority metric  $p_m$ , distance-to-end metric  $p_d$ , and time headway metric  $p_h$ , respectively; and  $\sigma_i \sim \mathcal{N}(0, 0.001)$  is a small random variable introduced to avoid two vehicles having the same priority indices. Specifically,  $p_m$  is defined as:

$$p_m = \begin{cases} 0.5, & \text{if on merge lane;} \\ 0, & \text{otherwise,} \end{cases} \quad (11)$$

which assigns priority scores to vehicles on the merge lane. The distance-to-end priority score  $p_d$  is defined as:

$$p_d = \begin{cases} \frac{x}{L}, & \text{if on merge lane;} \\ 0, & \text{otherwise,} \end{cases} \quad (12)$$

where  $x$  and  $L$  are the distance the ego vehicle has traveled on the connecting ramp and the length of the ramp (see Fig. 1), respectively. Finally, we define the time headway priority score to measure the headway priority as  $p_h = -\log \frac{d_{\text{headway}}}{t_h v_t}$ , where we use the time headway definition in Eq. 5.

##### B. Priority-Based Safety Supervisor

In this subsection, we present the proposed priority-based safety supervisor. Specifically, at each time step  $t$ , with the predicted HDV motions and assigned priority scores for all AVs as discussed above, the safety supervisor first generates a priority list for the AVs,  $\mathcal{P}_t$ , with their priority scores in a descendant order, i.e., a vehicle with the highest priority is on top of the list. Then the AV on the top of the obtained list, indexed by  $\mathcal{P}_t[0]$ , is selected for a safety check. More specifically, based on the (exploratory) action generated from the action network of vehicle  $\mathcal{P}_t[0]$ , the safety supervisor will examine whether the motion primitive induced by the exploratory action will conflict with its neighboring vehicles  $\mathcal{N}_{\mathcal{P}_t[0]}$  (both AVs and HDVs) in a considered time horizon  $T_n$ , where  $T_n$  is a hyper-parameter that can be tuned. The motions of HDVs are predicted using the human-driver decision models and vehicle kinematic model discussed above, whereas the motions of all other (lower-priority) AVs are predicted assuming the same actions from the last step. As the predicted trajectories of the considered vehicles (i.e.,  $\mathcal{P}_t[0] \cup \mathcal{N}_{\mathcal{P}_t[0]}$ ) are all  $T_n$ -step sequences, a collision can be detected if any two sequences have a distance below a prescribed threshold at any step  $k$ ,  $k = 1, \dots, T_n$ . If no collision is predicted, the exploratory action will be chosen as the actual action for vehicle  $\mathcal{P}_t[0]$ .

On the other hand, if the predicted trajectory of vehicle  $\mathcal{P}_t[0]$  is conflicting with other considered vehicles, the exploratory action is determined as unsafe and a new “safe”

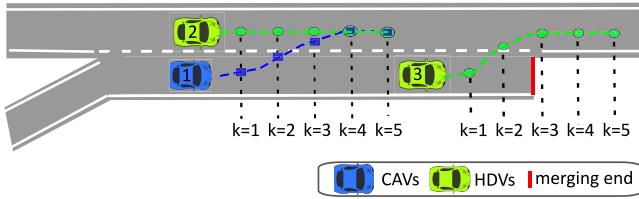
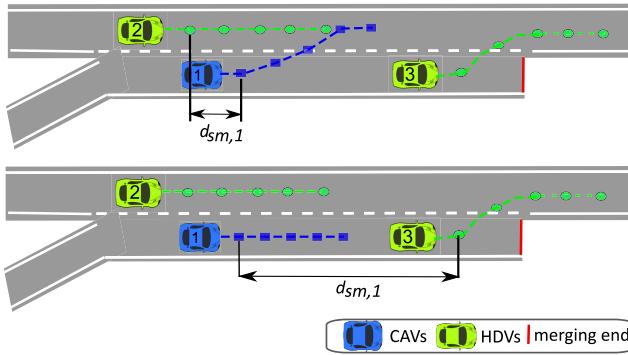
Fig. 5. Illustration of trajectory conflict for  $T_n = 5$  steps.

Fig. 6. Illustration of safety margin definitions. Top: safety margin if vehicle 1 turns left; and Bottom: safety margin when vehicle 1 keeps straight.

action will be used to replace the original action. A trajectory conflict is illustrated in Fig. 5, where the motion primitive of vehicle 1 conflicts with the predicted trajectory of vehicle 2 (HDV) at time steps 4 and 5. The exploratory action from vehicle 1 is deemed unsafe. Here we assume no collisions among HDVs (i.e., rational drivers), which can (almost) be guaranteed as the IDM and MOBIL models are extremely safety-focused. Then the safety supervisor enumerates other (valid) candidate actions and picks the best action based on the safety margin as follows:

$$a'_t = \arg \max_{a_t \in \mathcal{A}_{\text{valid}}} \left( \min_{k \in T_n} d_{\text{sm},k} \right). \quad (13)$$

where  $\mathcal{A}_{\text{valid}}$  is the set of valid actions at time step  $t$ . The safety margin  $d_{\text{sm},k}$  at the prediction time step  $k$  can be obtained as follows:

- if the action is changing lanes, i.e., *turn left* or *turn right*, the safety margin is defined as the minimum distance to the preceding and the following vehicles on the current and target lanes. An example is shown in the top subfigure of Fig. 6.
- if the action is *speed up*, *idle*, or *slow down*, the safety margin is set as the minimum distance headway. An example is shown in the bottom subfigure of Fig. 6.

After the action of the vehicle  $\mathcal{P}_t[0]$  is decided, its trajectory can be (re-)generated. Then vehicle  $\mathcal{P}_t[0]$  is removed from the list and the second highest becomes the first, i.e.,  $\mathcal{P}_t[i] \leftarrow \mathcal{P}_t[i+1], i = 1, 2, \dots$ . Then the same safety-check procedure discussed above is applied to the vehicle corresponding to new  $\mathcal{P}_t[0]$ , except that when determining collisions, instead of using actions from the last step to generate the trajectories, the motion primitives of the higher-priority vehicles (those who have been processed through the safety check procedure)

with the safety-proved actions are used. The procedure will continue until the priority list  $\mathcal{P}_t$  is emptied. The details of the proposed priority-based safety supervisor are given in Algorithm 1.

---

**Algorithm 1** Priority-based Safety Supervisor

---

**Parameter:**  $L, \alpha_1, \alpha_2, \alpha_3, t_h, w, T_n$ .

**Output :**  $a_i, i \in \mathcal{V}$ .

```

1 for  $i = 0$  to  $N$  do
2   | compute the priority scores according to Eq. 10;
3   | rearrange ego vehicles to list  $\mathcal{P}_t$  according to their
4   | priority scores.
5 end
6 for  $j = 0$  to  $|\mathcal{P}_t|$  do
7   | obtain the highest-priority vehicle  $\mathcal{P}_t[0]$ ;
8   | find its neighboring vehicles  $\mathcal{N}_{\mathcal{P}_t[0]}$ ;
9   | predict trajectories  $\zeta_v, v \in \mathcal{P}_t[0] \cup \mathcal{N}_{\mathcal{P}_t[0]}$  for  $T_n$ 
10  | time steps.
11  | if trajectories are overlapped then
12    | | replace the risky action as  $a_t \leftarrow a'_t$  according
13    | | to Eq. 13;
14    | | replace the trajectory  $\zeta_{\mathcal{P}_t[0]}$  with  $\zeta'_{\mathcal{P}_t[0]}$ 
15 end
16 remove  $\mathcal{P}_t[0]$  from  $\mathcal{P}_t$ ;
17 update  $\mathcal{P}_t[i] \leftarrow \mathcal{P}_t[i+1], i = 1, 2, \dots$ 
18 end

```

---

*Remark 1:* The priority-based safety supervising scheme can be realized through vehicle-to-infrastructure [60] (V2I) communication where a central communication station near the ramp can observe the HDVs and communicate with the AVs. At each discrete time  $t$ , the infrastructure agent will determine the priority scores for the AVs based on the observed traffic. AVs send their exploratory actions to the infrastructure and Algorithm 1 is then performed to generate the updated, safe actions for the AVs. As Algorithm 1 is sequential and thus computationally efficient (approximately 28.13 ms for the safety supervisor with  $T_n = 8$  to make a decision in the Hard traffic model, see Table II in Section V-C), with reasonable computation power at the infrastructure, it is expected that the algorithm can be implemented in real-time, i.e., providing updated controls within one sampling time step. For future real deployment, some further computation optimization can be performed to improve the computational efficiency, which is left for our future work.

*Remark 2:* The prediction horizon  $T_n$  is an important hyper-parameter in the safety-enhancement scheme. If  $T_n$  is too small, the safety supervisor is “short-sighted” and can lead to no feasible solutions after a few steps. On the other hand, if  $T_n$  is too large, the uncertainty of HDVs (the actual vehicle motion in the simulation has noisy perturbations from the human driver models used to predict the trajectories) are propagated and the results tend to be conservative in order to guarantee the safety in a large horizon. In our work, we use cross-validations and find that  $T_n = 8$  or 9 is the best choice (see e.g., Fig. 11 and Table II).

**Algorithm 2** MARL for AVs with Safety Supervisor

---

**Parameters:**  $\gamma, \eta, T, M, \beta_1, \beta_2$ .  
**Outputs :**  $\theta$ .

```

1 initialize  $s_0, t \leftarrow 0, \mathcal{D} \leftarrow \emptyset$ ;
2 for  $j = 0$  to  $M - 1$  do
3   for  $t = 0$  to  $T - 1$  do
4     for  $i \in \mathcal{V}$  do
5       observe  $s_i$ ;
6       update  $a_{i,t} \sim \pi_{\theta_i}(\cdot|s_i)$  with action masking.
7     end
8     for  $i \in \mathcal{V}$  do
9       check the actions by Algorithm 1;
10      if safe then
11        execute  $a_{i,t}$ ;
12        update  $\mathcal{D}_i \leftarrow (s_{i,t}, a_{i,t}, r_{i,t}, v_{i,t})$ 
13      end
14      else
15        update  $a_{i,t} \leftarrow a'_{i,t}$  and execute  $a'_{i,t}$  ;
16        update  $\mathcal{D}_i \leftarrow (s_{i,t}, a'_{i,t}, r_{i,t}, v_{i,t})$ .
17      end
18    end
19    update  $t \leftarrow t + 1$ 
20    if DONE then
21      for  $i \in \mathcal{V}$  do
22        update  $\theta_i \leftarrow \theta_i + \eta \nabla_{\theta_i} J(\theta_i)$ 
23      end
24    end
25    initialize  $\mathcal{D}_i \leftarrow \emptyset, i \in \mathcal{V}$ ;
26    update  $j \leftarrow j + 1$ 
27  end
28  update  $s_0, t \leftarrow 0$ 
29 end

```

---

Pseudo-code of the proposed MARL with the priority-based safety supervisor is shown in Algorithm 2. The hyperparameters include the (time)-discount factor  $\gamma$ , the learning rate  $\eta$ , the total number of training epochs  $M$ , the epoch length  $T$ , and the coefficients for the loss function  $\beta_1$  and  $\beta_2$ . In each epoch, each agent collects the state information and samples actions by applying the action masking strategy to avoid invalid actions (Lines 4–7). Then the exploratory actions from the MARL will be checked by the priority-based safety supervisor detailed in Algorithm 1 (Line 9). If the action is unsafe, then the safety supervisor will replace the risky action with a safe action according to Eq. 13. The safe action will be taken by the agent and the corresponding experience will be collected and saved to the replay buffer (Lines 10–17). The parameters of the policy network are updated using the collected experience sampled from the on-policy experience buffer after the completion of each episode (Lines 20–26). The DONE signal is flagged if either the episode is completed or a collision occurs. After receiving the DONE flag, all agents are reset to their initial states to start a new epoch (Lines 28).

## V. NUMERICAL EXPERIMENTS

In this section, we evaluate the performance of the proposed MARL algorithm in terms of training efficiency and collision rate in the on-ramp merging scenario illustrated in Fig. 1. The length of the road is 520 m, where the entrance of the merge lane (road segment AB) is 320 m and the length of the merge lane is  $L = 100$  m. There are 12 spawn points (the numbers under the vehicles) evenly distributed on the through lane and the ramp lane from 0 m to 220 m, as shown in Fig. 7. The vehicles exceeding the road will be removed from displaying while the kinematics are still updated. Specifically, we consider three levels of traffic densities with different numbers of initial vehicles defined as:

- *Easy mode*: 1-3 AVs and 1-3 HDVs.
- *Medium mode*: 2-4 AVs + 2-4 HDVs.
- *Hard mode*: 4-6 AVs + 3-5 HDVs.

In each training episode, a different number of HDVs and AVs will randomly appear at the spawn points with a random position noise (uniformly distributed in [-1.5m, 1.5m]) added to each initial spawn position. The initial speed is randomly chosen between 25 to 27 m/s. The vehicle control sampling frequency is 5 Hz, i.e., AVs take an action every 0.2 seconds. A 5% random noise is added to the predicted acceleration and steering angle for HDVs. We train all MARL algorithms over 2 million steps with 3 different random seeds while the same random seeds are shared among the agents, which is around 20,000 episodes with episode horizon  $T = 100$  steps. We evaluate the algorithm over 3 episodes every 200 training episodes. We set  $\gamma = 0.99$  and the learning rate  $\eta = 5e^{-4}$ ; The coefficients  $w_c, w_s, w_h$ , and  $w_m$  for the reward function are set as 200, 1, 4, and 4, respectively. The priority coefficients  $\alpha_1, \alpha_2$  and  $\alpha_3$  are equally set as 1. The weighting coefficients  $\beta_1$  and  $\beta_2$  for the loss function are chosen as 1 and 0.01, respectively. Here we call the MARL algorithm, without the safety supervisor, proposed in Section III as the baseline method.

The simulation environment is modified from the gym-based highway-env simulator [61] and is open-sourced.<sup>3</sup> We use the default parameters of the IDM and MOBIL models which can be found as in the highway-env simulator [61]. The experiments have been performed in a Ubuntu 18.04 server with AMD 9820X processor and 64 GB memory. The video demo of the training process can be found at the site.<sup>4</sup>

## A. Reward Function Designs

In this subsection, we will first evaluate the performance of the proposed MARL framework under different reward function designs, local (baseline) v.s. global rewards (baseline with global reward). Then the impact of the safety penalty weight  $w_c$  in the reward function (Eq. 3) will be evaluated.

We investigate the proposed local reward function by comparing it with the global reward design used in [25] and [28], where the reward of the  $i$ th agent at time step  $t$  is the

<sup>3</sup>See [https://github.com/DongChen06/MARL\\_CAVs](https://github.com/DongChen06/MARL_CAVs)

<sup>4</sup>See <https://drive.google.com/drive/folders/1437My4sDoyPFsUjrThmlu1oJjTkTkvJ7?usp=sharing>

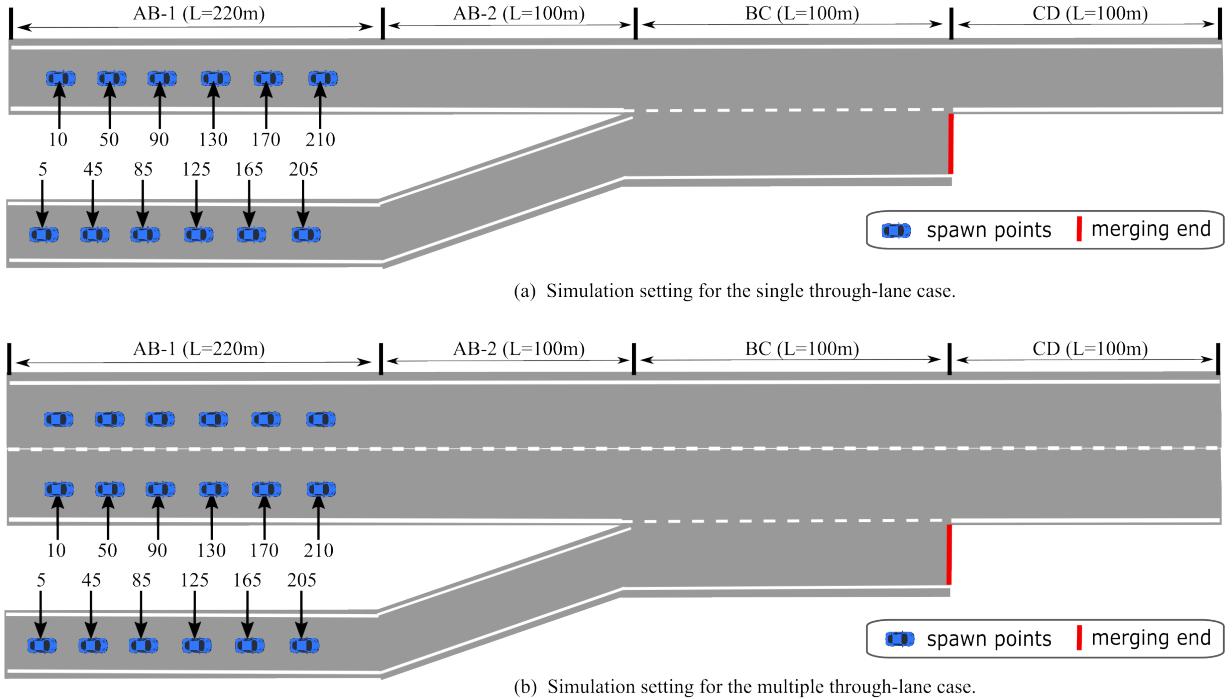


Fig. 7. Simulation settings for the single through lane case (upper) and multiple through lane case (lower). “L” represents the length of the road segments. The numbers under the vehicles are the vehicle spawn points.

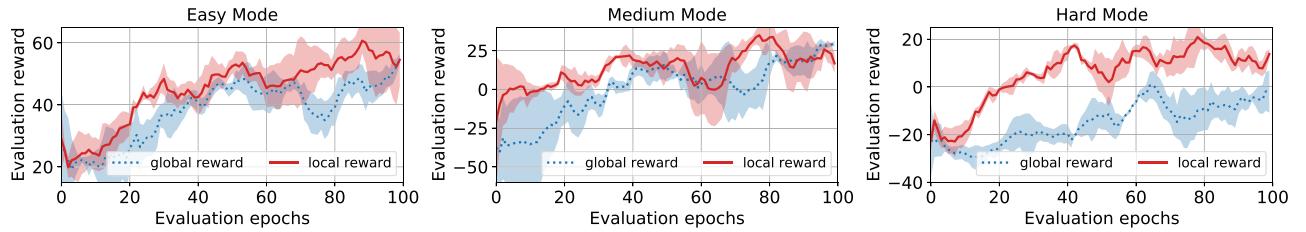


Fig. 8. Evaluation curves during training with different reward functions for different traffic levels. The shaded region denotes the standard deviation over 3 random seeds. The curves are smoothed over the last 9 evaluation epochs.

TABLE I  
PERFORMANCE WITH DIFFERENT  $w_c$ ’S IN TERMS OF COLLISION RATE AND AVERAGE SPEED IN THE *Medium* TRAFFIC MODE. NOTE THAT OTHER WEIGHTING FACTORS ARE KEPT UNCHANGED

|                | $w_c = 10$ | $w_c = 100$ | $w_c = 200$ (we chose) | $w_c = 1000$ | $w_c = 10000$ |
|----------------|------------|-------------|------------------------|--------------|---------------|
| Collision rate | 0.1        | 0           | 0                      | 0            | 0             |
| avg. speed     | 24.77      | 24.09       | 24.08                  | 23.69        | 23.62         |

averaged global reward  $r_{i,t} = \frac{1}{N} \sum_{j=1}^N r_{j,t}$ . Fig. 8 shows the evaluation performance comparison between the proposed local reward design and the global reward design. As expected, the proposed local reward outperforms the global reward design in terms of higher evaluation rewards as well as faster convergence speed across all three traffic scenarios. In the *Easy* and *Medium* modes, the global reward design performs well and achieves a reasonable reward due to the small number of AVs, while it fails the control tasks in the *Hard* mode, with an evaluation reward less than 0, as it suffers from the credit assignment issues [56] and the fact that the assigned average global rewards have less correlation with individual agent’s actions as the number of agents increases.

Table I shows the testing performance with different  $w_c$  values in the *Medium* traffic mode, while other weighting coefficients are kept unchanged in Eq. 3. It can be seen that there are no collisions when a large enough  $w_c$  is selected (e.g.,  $w_c \geq 100$ ), and the average traffic speed decreases as  $w_c$  further increases. This is because the CAVs will behave conservatively if we emphasize too much on the safety. In the following experiments, we set  $w_c$  to 200 since it represents a reasonable trade-off between safety margin and traffic efficiency.

### B. Curriculum Learning

In this subsection, we adopt curriculum learning [62] to speed up the learning and improve the performance for the

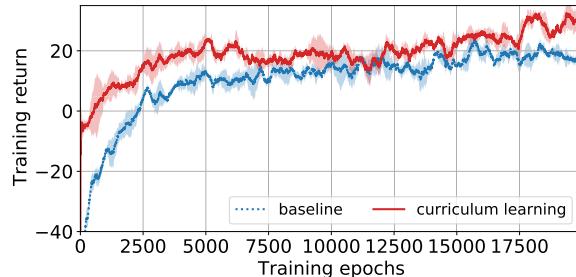


Fig. 9. Training curves with and without curriculum learning for *Hard* traffic mode.

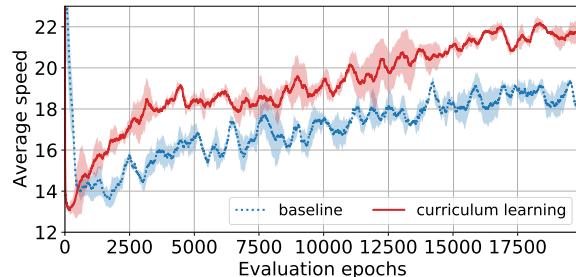


Fig. 10. Average speed during training with and without curriculum learning for *Hard* traffic mode.

*Hard* mode. Specifically, instead of learning the *Hard* mode directly, we build upon the trained model from the easier modes (i.e., *easy* and *medium*) and train the models to achieve higher efficiency. Curriculum learning is especially preferable to safety-critical tasks (e.g., autonomous driving) as starting from a decent model can greatly reduce the number of “blind” explorations that can be risky.

Fig. 9 shows the training performance comparison between the baseline method (i.e., starting from scratch) and curriculum learning (baseline + curriculum learning) for the *Hard* traffic mode. It is obvious that learning based on the trained model from easier tasks greatly expedites the speed of convergence and improves the final model performance. The average speed during the training, as shown in Fig. 10, indicates that the curriculum learning strategy also improves the average vehicle speed up to 22 m/s compared to baseline method at 18 m/s, thus achieving high traffic efficiency. Therefore, we apply the curriculum learning in the following experiments for *Hard* traffic modes.

### C. Performance of the Priority-Based Safety Supervisor

In this subsection, we evaluate the effectiveness of the proposed priority-based safety supervisor. As shown in Fig. 11, the proposed priority-based safety supervisor method has a better sample efficiency, evidenced by faster converge speed in all three traffic densities. In addition, the proposed priority-based safety supervisor method achieves higher evaluation reward even in the *Hard* traffic mode. This is because most unsafe actions are replaced with safe ones by the safety supervisor, especially in the earlier exploration phase, which avoids early terminations and thus improves learning efficiency.

Fig. 12 shows the average vehicle speed during the training, which is an indication of traffic throughput. It is clear that

the algorithms with the safety supervisor maintain higher training speed than the baseline method (i.e., without safety supervision). This shows that the proposed safety supervisor is not only beneficial for training but also leads to better traffic efficiency. It can also be seen that vehicle speeds are slower as the traffic density increases (26 m/s, 24 m/s and 22 m/s for *Easy*, *Medium* and *Hard* traffic densities, respectively), which is reasonable as the interactions are more frequent in dense traffic and a lower speed is safer to avoid collisions.

After training, MARL algorithms for each traffic density are tested over 3 random seeds for 30 epochs and the average collision rates, vehicle speeds, and inference time of the safety supervisor are shown in Table II. We can see that with the safety supervisor ( $T_n \geq 7$ ), the MARL can run without collisions in all traffic modes, while the baseline method has a collision rate of 0.07 and 0.16 for *Medium* and *Hard* traffic densities, respectively. It is clear that with only a short prediction horizon, e.g.,  $T_n = 3$  or 6, the MARL is still failed under challenging cases. For example, the agents still have 0.03 collision rate in the *Hard* traffic mode when choosing  $T_n = 6$ . The reason is that if  $T_n$  is too small, the safety supervisor is “short-sighted” and can lead to no feasible solutions after only a few steps. However, if we further increase  $T_n$  (e.g.,  $T_n = 10, 12, 14$ ), the collision rate may increase, as discussed in Remark 2 in Section IV-B, due to the uncertainty propagation in a longer time window. Also, the average speed will decrease since the CAVs need to behave carefully to guarantee safety in a large time horizon. With a reasonable  $T_n$  (e.g., 7, 8), the average speed indicates that the safety supervisor leads to higher traffic efficiency. In all traffic modes, the safety supervisor always leads to higher average speed while lower collision rate. For example, the best average speed for the *Easy* traffic mode is achieved by baseline +  $T_n = 8$  (27.72 m/s) compared to the baseline method (23.52 m/s). It is noted that the baseline method has a lower average speed in the *Medium* traffic mode than the *Hard* traffic mode, which is not consistent with the methods that employ the safety supervisor. Our observation is that the CAVs (in the baseline) often behave conservatively and hesitate to speed up or slow down to let the merging vehicles merge to avoid traffic congestion, which is the main reason causing traffic inefficiencies. In contrast, with the designed safety supervisor, traffic inefficiencies are largely improved and this situation is minimized. As expected, the inference time increases as the prediction horizon increases. With reasonable computation power in the infrastructure, it is expected that the algorithm can be implemented in real-time.

### D. Comparison With State-of-the-Art Benchmarks

In this subsection, we compare the proposed method with several state-of-the-art MARL benchmarks, MAA2C, MAPPO, and MAACKTR as mentioned in Section II, as well as an improved model predictive control (MPC) method [13], [63]. The dynamics, cost function, and constraint ingredients of the improved MPC approach are illustrated in the Appendix. All the MARL benchmarks are implemented by

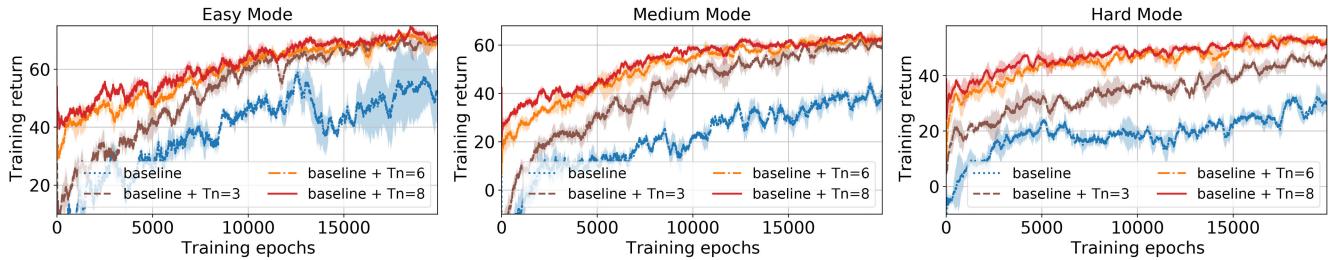


Fig. 11. Training curves for the n-step priority-based safety supervisor.

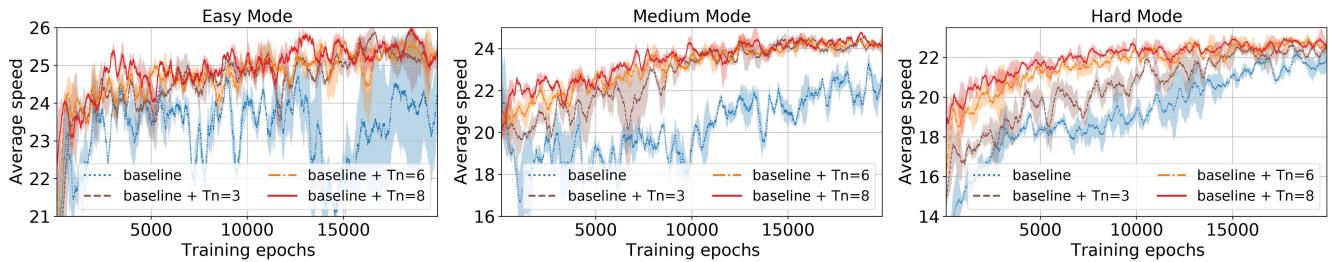
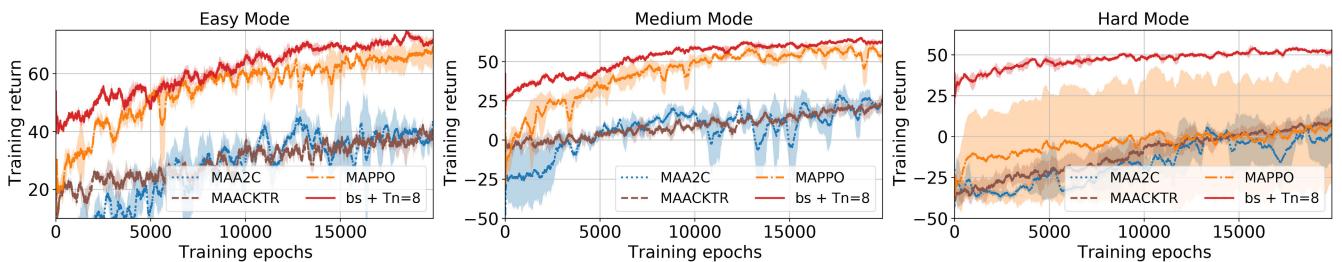


Fig. 12. Average speed during training for the n-step priority-based safety supervisor.

TABLE II  
TESTING PERFORMANCE COMPARISON OF COLLISION RATE, AVERAGE SPEED (m/s), AND INFERENCE TIME (ms)  
BETWEEN N-STEP SAFETY SUPERVISOR BASED ON THE BASELINE (bs) METHOD

| Scenarios   | Metrics        | bs    | bs + $T_n = 3$ | bs + $T_n = 6$ | bs + $T_n = 7$ | bs + $T_n = 8$ | bs + $T_n = 9$ | bs + $T_n = 10$ | bs + $T_n = 12$ | bs + $T_n = 14$ |
|-------------|----------------|-------|----------------|----------------|----------------|----------------|----------------|-----------------|-----------------|-----------------|
| Easy Mode   | collision rate | 0     | 0              | 0              | 0              | 0              | 0              | 0               | 0               | 0               |
|             | avg. speed     | 23.53 | 25.12          | 25.38          | 25.27          | 27.72          | 27.50          | 25.89           | 25.82           | 25.74           |
|             | infrn time     | -     | 4.90           | 7.62           | 8.30           | 8.93           | 10.07          | 11.08           | 12.62           | 14.71           |
| Medium Mode | collision rate | 0.07  | 0.03           | 0.01           | 0              | 0              | 0              | 0               | 0               | 0.01            |
|             | avg. speed     | 20.30 | 24.22          | 24.61          | 24.13          | 24.08          | 24.19          | 23.74           | 24.35           | 24.13           |
|             | infrn time     | -     | 14.75          | 14.64          | 16.78          | 17.55          | 19.40          | 21.29           | 23.22           | 26.89           |
| Hard Mode   | collision rate | 0.16  | 0.14           | 0.03           | 0              | 0              | 0              | 0.03            | 0.05            | 0.05            |
|             | avg. speed     | 21.71 | 22.52          | 22.56          | 22.58          | 22.73          | 23.01          | 22.52           | 21.31           | 21.83           |
|             | infrn time     | -     | 14.75          | 23.13          | 25.69          | 28.13          | 31.45          | 35.93           | 39.43           | 50.13           |

Fig. 13. Training curves comparison between the proposed MARL policy (baseline (bs) +  $T_n = 8$ ) and 3 state-of-the-art MARL benchmarks.

sharing parameters among agents to deal with dynamic numbers of agents and using the global reward and the discrete action space. Specifically, MAA2C [38] implements a context-aware multi-agent actor-critic algorithm with a centralized critic network using an expected update. MAACKTR [41] optimizes the trust region with the Kronecker-factored approximate curvature [64] (K-FAC). MAPPO [65] improves the MARL framework by adopting common practices such as Generalized Advantage Estimation (GAE) [66], advantage normalization, and value clipping.

Fig. 13 shows the evaluation results during training for all the MARL algorithms. The proposed method (baseline +  $T_n = 8$ ) consistently outperforms the benchmarks in all traffic levels. The proposed method shows an even greater advantage in terms of sample efficiency and training performance in the Hard mode over the benchmarks. Fig. 14 shows the proposed method has relatively higher average training speeds which lead to high training efficiency.

After training, all algorithms for each traffic density are tested over three random seeds for 30 epochs, and the resulting

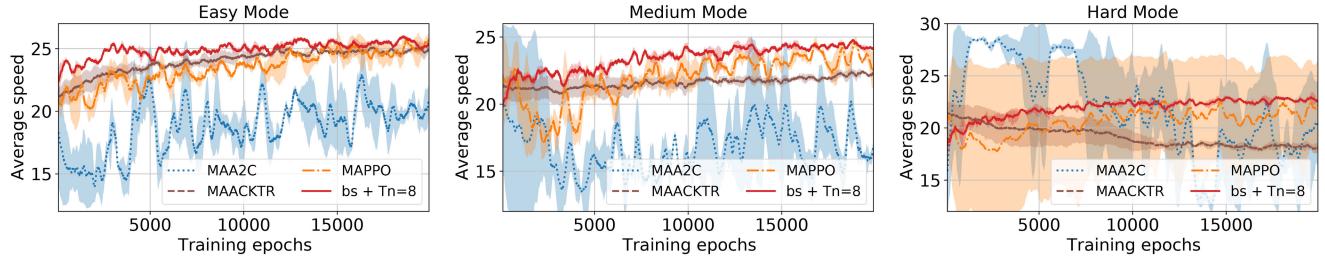
Fig. 14. Average speed comparison between the proposed MARL policy (baseline (bs) +  $T_n = 8$ ) and 3 state-of-the-art MARL benchmarks.

TABLE III  
TESTING PERFORMANCE COMPARISON OF COLLISION RATE AND AVERAGE SPEED BETWEEN THE PROPOSED METHOD AND 4 STATE-OF-THE-ART BENCHMARKS

| Scenarios   | Metrics          | MPC   | MAA2C | MAACKTR | MAPPO | baseline + $T_n = 8$ |
|-------------|------------------|-------|-------|---------|-------|----------------------|
| Easy Mode   | collision rate   | 0.03  | 0.02  | 0.08    | 0     | 0                    |
|             | avg. speed [m/s] | 22.05 | 21.00 | 24.71   | 25.70 | 25.72                |
| Medium Mode | collision rate   | 0.03  | 0.08  | 0.12    | 0.02  | 0                    |
|             | avg. speed [m/s] | 19.67 | 19.33 | 21.94   | 24.00 | 24.08                |
| Hard Mode   | collision rate   | 0.40  | 0.52  | 0.18    | 0.34  | 0                    |
|             | avg. speed [m/s] | 21.02 | 19.68 | 18.19   | 22.41 | 22.73                |

average collision rates and vehicle speeds are presented in Table III. The test results demonstrate that the proposed method achieves no collision and higher efficiency compared to other benchmark algorithms. Specifically, MAPPO exhibits excellent performance in the *Easy* traffic mode, with a zero collision rate, and a relatively low collision rate (0.02) in the *Medium* traffic mode. However, it demonstrates a high collision rate (0.34) in the *Hard* traffic mode as it exhibits frequent sudden speed changes as illustrated in Figs. 13 and 14, causing HDVs little room to respond. On the other hand, MAA2C and MAACKTR, lacking safety checks, fail in the on-ramp merging tasks across all traffic scenarios, representing high collision rates. It is noteworthy that, due to the discrepancy between the exact dynamics used in the highway simulator environment and our model used in MPC, along with the uncertainties injected in the simulator, the MPC can still lead to collisions (0.03, 0.03, and 0.40 collision rate in the *Easy*, *Medium*, and *Hard* traffic modes, respectively). Another observation is on the impact of system complexity on the reliability/performance of model-based methods: when the number of vehicles grows, the merging problem becomes more difficult due to increased model mismatch (system state dimension is higher with more vehicles), and MPC finds it harder to obtain a collision-free policy for on-ramp merging of CAVs. This shows an advantage of model-free approaches that do not rely on explicit models. Another consideration for the model-based MPC implementation is the requirement for powerful computational resources to support the extensive online computations. This is especially significant for the on-ramp merging problem that involves nonlinear dynamics, and a nonlinear program must be solved at each time step in order to calculate the control input, requiring significant onboard computation power. In contrast, our RL-based approach requires much less computational capability [67].

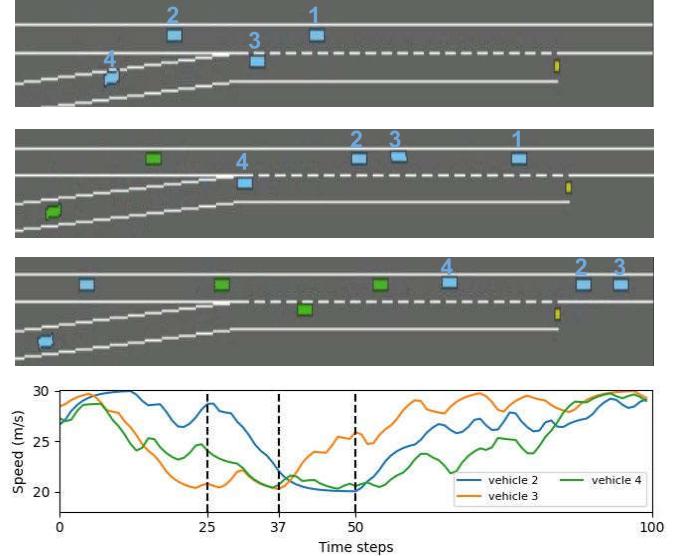


Fig. 15. Frames show the learned policy. The below figure shows the corresponding speed of the AVs.

### E. Policy Interpretation

In this subsection, we attempt to interpret the learned AV behaviors. As an example, Fig. 15 shows the snapshots at time steps 25, 37, and 50, as well as the speeds of agents 2-4. It can be observed that at the time step 25, vehicle 2 starts to slow down and makes space for vehicle 3 to merge. Vehicle 3 accelerates to merge while keeping an adequate distance headway with vehicle 1. Then vehicle 3 successfully merges into the through lane and starts to speed up at time step 37. At the same time, vehicle 2 still keeps a low speed to keep a safe headway distance to vehicle 3. At the time 50, vehicle 2 speeds up while keeping a lower speed than vehicle 3

TABLE IV  
TESTING PERFORMANCE COMPARISON OF COLLISION RATE AND AVERAGE SPEED BETWEEN THE PROPOSED METHOD AND 3 STATE-OF-THE-ART BENCHMARKS ON THE MULTIPLE THROUGH-LANE CASES

| Scenarios   | Metrics          | MAA2C | MAACKTR | MAPPO | baseline + $T_n = 8$ |
|-------------|------------------|-------|---------|-------|----------------------|
| Easy Mode   | collision rate   | 0     | 0.03    | 0     | 0                    |
|             | avg. speed [m/s] | 19.78 | 22.71   | 23.07 | 23.53                |
| Medium Mode | collision rate   | 0.03  | 0.07    | 0.03  | 0                    |
|             | avg. speed [m/s] | 15.16 | 20.07   | 19.59 | 21.05                |
| Hard Mode   | collision rate   | 0.40  | 0.27    | 0.10  | 0                    |
|             | avg. speed [m/s] | 22.04 | 21.60   | 19.65 | 20.95                |

to maintain a safe distance headway. Similar patterns are also observed in vehicle 4.

#### F. Multiple Through-Lane Case

In this subsection, we demonstrate the proposed approach in more challenging multiple through-lane cases illustrated in Fig. 7(b) where vehicles are allowed to change lanes in the through lanes. As shown in Section III-A, we formulated the on-ramp merging as a partially observable Markov decision process (POMDP)  $\mathcal{M}_G$ , which can be described by the following tuple  $(\{\mathcal{A}_i, \mathcal{S}_i, \mathcal{R}_i\}_{i \leq v}, \mathcal{T})$ . In this setting, the action state  $\mathcal{A}$  is extended to the multiple through-lane cases without changes, while the state space is slightly modified to accommodate more surrounding vehicles. Specifically, the observation space (the number of observable neighboring vehicles) is determined by the parameter  $N_{\mathcal{N}_i}$ . For the multiple through-lane case, we choose a larger  $N_{\mathcal{N}_i} = 8$  ( $N_{\mathcal{N}_i} = 5$  in the single through-lane case). For the priority-based safety supervisor, we also extend it to the multi-lane case without any changes.

For the reward function, we first tried to use the original reward design but we then found that the ego vehicles often conducting unnecessary and frequent lane changes which lead to unsafe driving (a demo on frequent lane changes at [https://drive.google.com/file/d/1dO8xPCwLXVRgQFM\\_xwqscRazoId5ksf4/view?usp=sharing](https://drive.google.com/file/d/1dO8xPCwLXVRgQFM_xwqscRazoId5ksf4/view?usp=sharing)). Therefore, we added one more metric  $r_l$  and use the following revised reward function:

$$r_{i,t} = w_c r_c + w_s r_s + w_h r_h + w_m r_m + w_l r_l, \quad (14)$$

where  $w_c$ ,  $w_s$ ,  $w_h$ ,  $w_m$  and  $w_l$  are positive weighting scalars corresponding to collision evaluation  $r_c$ , stable-speed evaluation  $r_s$ , headway time evaluation  $r_h$ , merging cost evaluation  $r_m$ , and lane-changing evaluation  $r_l$ , respectively. The goal of the added lane-changing evaluation  $r_l$  is to penalize unnecessary and frequent lane changes to avoid oscillatory driving following the designs in [68]. Here,  $r_l$  is defined as:

$$r_l = \begin{cases} -1, & \text{if change lanes;} \\ 0, & \text{otherwise.} \end{cases} \quad (15)$$

To further demonstrate the flexibility and effectiveness of the proposed MARL framework, we implemented the aforementioned multiple through-lane cases in the highway environment. Fig. 16 shows that our approach can be easily extended to the multiple through-lane cases and achieves good

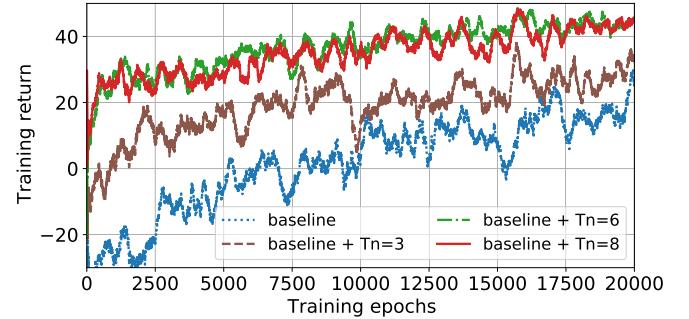


Fig. 16. Training curves for the  $n$ -step priority-based safety supervisor for the multiple through-lane cases.

performance. Table IV shows the evaluation performance on the multi-lane scenarios. As expected, the proposed approach achieves the best performance among the MARL benchmarks in terms of the lowest collision rate. Overall, MAPPO, MAA2C, and MAACKTR achieve better performance in terms of lower collision rate than the single through-lane case since there is an extra through lane providing more operating space for the CAVs. However, nearly all MARL algorithms achieve relatively lower average speeds in the multi-lane case than in the single-lane case due to more complicated traffic scenarios with more CAVs and HDVs. It is noted that MAA2C learns a suboptimal policy in the *Medium* traffic mode, representing the lowest average speed due to conservative operations. Since it is a very involved task to apply MPC approaches to the multi-through lane cases considering the complicated system modeling, we will leave it for future work. The demo video and code for the multiple through-lane scenarios can be found at [https://github.com/DongChen06/MARL\\_CAVs/tree/multi-lane](https://github.com/DongChen06/MARL_CAVs/tree/multi-lane).

## VI. CONCLUSION AND DISCUSSIONS

In this paper, we formulated the problem of on-ramp merging in a mixed-traffic as an on-policy MARL, and we developed an efficient MARL algorithm featuring action masking, local reward design, curriculum learning, and parameter sharing. A novel priority-based safety supervisor was also developed to enhance safety, improve learning efficiency, and increase traffic throughput. Comprehensive experiments were conducted to compare with several state-of-the-art algorithms, which showed that the proposed approach consistently outperformed the benchmark approaches in terms of training efficiency and collision rate.

In future work, we plan to investigate how to fill the gap between simulations and real-world implementations. The initial exploration in general RL training may lead to undesired system behavior and sometimes can even cause crashes in a real-world deployment. For such systems, safety during the exploration can be enhanced by exploiting the dynamic information of the system to limit the exploration actions within an admissible range, see e.g., our previous work [46] as well as others [69], [70] for safe RL algorithms. Before real-world implementations, the policy network is necessary to be trained in high-fidelity simulations until sufficiently good performance is achieved, which can also significantly reduce the number of risky explorations. Also, it needs to pass various tests before field deployment for safety and robustness. Once the policy is deployed on the ego vehicles, periodical updates and maintenance should be conducted to improve the model performance in unseen scenarios. Interested readers are referred to an extensive survey on sim-to-real paradigms for reinforcement learning [71]. Therefore, we will develop a more realistic simulation environment by incorporating data from real-world traffic systems to better fill the sim2real gap.

On the other hand, in this paper, the built-in lane-changing behavior of HDVs in the Highway simulator [61] is governed by the Minimizing Overall Braking Induced by Lane change (MOBIL) model [52], which is not suitable for highway on-ramp scenarios since the HDVs are not aware of the merging end in the MOBIL model. As a result, sometimes, the HDVs may incur unexpected delays on the ramp, see the video demo (blue: CAVs, green: HDVs) at [https://drive.google.com/file/d/1mWFgZrsX4bb0M-D-fmLQ\\_oK5V7OwjWU/view?usp=sharing](https://drive.google.com/file/d/1mWFgZrsX4bb0M-D-fmLQ_oK5V7OwjWU/view?usp=sharing). Therefore, in future work, we will also incorporate more comprehensive human driver models so that HDV motions can be more accurately predicted and the overall traffic statistics (e.g., throughput, queue, and delay on the ramp) can be studied.

## APPENDIX

In this paper, we compared our proposed MARL approach with a model-based control strategy, i.e. Model Predictive Control (MPC). In this regard, we mainly adopted the MPC problem formulation proposed in [13] and [63] for the highway on-ramp merging problem. To capture the vehicle interactions and dynamics in the highway simulator environment, we improved the MPC implementation in [13] and [63] following the specifications:

- 1) The following elements are employed in our problem formulation to simulate vehicle dynamics.
  - Kinematic Bicycle Model [53] is used for vehicle kinematics instead of the point mass model used in [13] and [63].
  - For longitudinal behavior of HDVs, the acceleration of the HDVs is given by the Intelligent Driver Model (IDM) from [51]. For lateral behavior, the discrete lane change decisions of HDVs are modeled by the Minimizing Overall Braking Induced by Lane change (MOBIL) model [52].
  - The CAVs control inputs are the steering angle and the acceleration. This could be translated to

a high-level decisions (i.e. faster, slower, idle, left-lane) similar to the action space used in the MARL framework.

- 2) The cost function is made up of the following components:
  - Collision avoidance term (i.e., Eq. 8 in the reference [13]) in the cost function is adopted.
  - To bound the available moving area for vehicles, a cost term similar to Eq. 13 and Eq. 14 in [13] is incorporated in the cost function.
  - The nominal path for the merging CAVs is defined based on the geometry of the on-ramp lane and the main lane, similar to Eq. 3 in [13].
  - To make vehicles run as closely as possible at their desired speed, a term similar to Eq. 15 in [13] is included in the cost function.
- 3) Hard constraints on the acceleration and steering angle are considered.

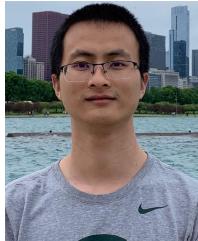
A Nonlinear MPC problem is formulated and implemented following the above descriptions, with the results shown in Table III. The detailed implementation of the MPC approach is open-source and can be accessed at: [https://github.com/DongChen06/MARL\\_CAVs/tree/MPC\\_Merging](https://github.com/DongChen06/MARL_CAVs/tree/MPC_Merging).

## REFERENCES

- [1] *Future of Driving*. Accessed: Mar. 31, 2021. [Online]. Available: <https://www.tesla.com/autopilot>
- [2] *Apollo Open Platform*. Accessed: Mar. 31, 2021. [Online]. Available: <https://apollo.auto/developer.html>
- [3] V. V. Dixit, S. Chand, and D. J. Nair, "Autonomous vehicles: Disengagements, accidents and reaction times," *PLoS ONE*, vol. 11, no. 12, Dec. 2016, Art. no. e0168054.
- [4] F. M. Favaro, N. Nader, S. O. Eurich, M. Tripp, and N. Varadaraju, "Examining accident reports involving autonomous vehicles in California," *PLoS ONE*, vol. 12, no. 9, Sep. 2017, Art. no. e0184952.
- [5] D. Ni and J. D. Leonard, "A simplified kinematic wave model at a merge bottleneck," *Appl. Math. Model.*, vol. 29, no. 11, pp. 1054–1072, Nov. 2005.
- [6] L. Leclercq, J. A. Laval, and N. Chiabaut, "Capacity drops at merges: An endogenous model," *Proc.-Social Behav. Sci.*, vol. 17, pp. 12–26, Jan. 2011.
- [7] Z. H. Khattak, B. L. Smith, H. Park, and M. D. Fontaine, "Cooperative lane control application for fully connected and automated vehicles at multilane freeways," *Transp. Res. C, Emerg. Technol.*, vol. 111, pp. 294–317, Feb. 2020.
- [8] D. Bevly et al., "Lane change and merge maneuvers for connected and automated vehicles: A survey," *IEEE Trans. Intell. Vehicles*, vol. 1, no. 1, pp. 105–120, Mar. 2016.
- [9] J. Rios-Torres and A. A. Malikopoulos, "A survey on the coordination of connected and automated vehicles at intersections and merging at highway on-ramps," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 5, pp. 1066–1077, May 2017.
- [10] L. N. Jacobson, K. C. Henry, and O. Mehyar, *Real-Time Metering Algorithm for Centralized Control*, no. 1232. Transportation Research Board, 1989, pp. 17–26. [Online]. Available: <https://trid.trb.org/view/308666>
- [11] J. Houdakis and P. G. Michalopoulos, "Evaluation of ramp control effectiveness in two twin cities freeways," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1811, no. 1, pp. 21–29, Jan. 2002.
- [12] Y. Lin, J. McPhee, and N. L. Azad, "Anti-jerk on-ramp merging using deep reinforcement learning," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Oct. 2020, pp. 7–14.
- [13] W. Cao, M. Mukai, T. Kawabe, H. Nishira, and N. Fujiki, "Cooperative vehicle path generation during merging using model predictive control with real-time optimization," *Control Eng. Pract.*, vol. 34, pp. 98–105, Jan. 2015.
- [14] J. B. Rawlings, D. Q. Mayne, and M. M. Diehl, *Model Predictive Control: Theory, Computation, and Design*, 2nd ed. Portland, OR, USA: Nob Hill Publishing, 2017.

- [15] M. Papageorgiou, C. Kiakaki, V. Dinopoulou, A. Kotsialos, and Y. Wang, "Review of road traffic control strategies," *Proc. IEEE*, vol. 91, no. 12, pp. 2043–2067, Dec. 2003.
- [16] M. Papageorgiou and A. Kotsialos, "Freeway ramp metering: An overview," *IEEE Trans. Intell. Transp. Syst.*, vol. 3, no. 4, pp. 271–281, Dec. 2002.
- [17] I. Papamichail and M. Papageorgiou, "Traffic-responsive linked ramp-metering control," *IEEE Trans. Intell. Transp. Syst.*, vol. 9, no. 1, pp. 111–121, Mar. 2008.
- [18] Y. Hou, P. Edara, and C. Sun, "Modeling mandatory lane changing using Bayes classifier and decision trees," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 2, pp. 647–655, Apr. 2014.
- [19] F. Marczak, W. Daamen, and C. Buisson, "Key variables of merging behaviour: Empirical comparison between two sites and assessment of gap acceptance theory," *Proc.-Social Behav. Sci.*, vol. 80, pp. 678–697, Jun. 2013.
- [20] Z. E. A. Kherroubi, S. Aknine, and R. Bacha, "Novel decision-making strategy for connected and autonomous vehicles in highway on-ramp merging," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 8, pp. 12490–12502, Aug. 2022.
- [21] J. Lubars et al., "Combining reinforcement learning with model predictive control for on-ramp merging," 2020, *arXiv:2011.08484*.
- [22] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.
- [23] J. Wang, T. Shi, Y. Wu, L. Miranda-Moreno, and L. Sun, "Multi-agent graph reinforcement learning for connected automated driving," in *Proc. 37th Int. Conf. Mach. Learn.*, 2020, pp. 1–6.
- [24] P. Palanisamy, "Multi-agent connected autonomous driving using deep reinforcement learning," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2020, pp. 1–7.
- [25] M. Kaushik and K. M. Krishna, "Parameter sharing reinforcement learning architecture for multi agent driving behaviors," 2018, *arXiv:1811.07214*.
- [26] P. Young Joun Ha, S. Chen, J. Dong, R. Du, Y. Li, and S. Labi, "Leveraging the capabilities of connected and autonomous vehicles and multi-agent reinforcement learning to mitigate highway bottleneck congestion," 2020, *arXiv:2010.05436*.
- [27] S. Bhalla, S. G. Subramanian, and M. Crowley, "Deep multi agent reinforcement learning for autonomous driving," in *Proc. Can. Conf. Artif. Intell.* Cham, Switzerland: Springer, 2020, pp. 67–78.
- [28] J. Dong, S. Chen, P. Y. J. Ha, Y. Li, and S. Labi, "A DRL-based multiagent cooperative control framework for CAV networks: A graphic convolution Q network," 2020, *arXiv:2010.05437*.
- [29] C. Yu et al., "Distributed multiagent coordinated learning for autonomous driving in highways based on dynamic coordination graphs," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 2, pp. 735–748, Feb. 2020.
- [30] V. Mnih et al., "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1928–1937.
- [31] V. Mnih et al., "Playing Atari with deep reinforcement learning," 2013, *arXiv:1312.5602*.
- [32] T. Chu, J. Wang, L. Codeca, and Z. Li, "Multi-agent deep reinforcement learning for large-scale traffic signal control," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 3, pp. 1086–1095, Mar. 2020.
- [33] C. Berner et al., "Dota 2 with large scale deep reinforcement learning," 2019, *arXiv:1912.06680*.
- [34] N. Naderizadeh, J. J. Sydir, M. Simsek, and H. Nikopour, "Resource management in wireless networks via multi-agent deep reinforcement learning," *IEEE Trans. Wireless Commun.*, vol. 20, no. 6, pp. 3507–3523, Jun. 2021.
- [35] D. Chen, K. Chen, T. Chu, R. Yao, F. Qiu, and K. Lin, "PowerNet: Multi-agent deep reinforcement learning for scalable powergrid control," 2020, *arXiv:2011.12354*.
- [36] M. Tan, "Multi-agent reinforcement learning: Independent vs. cooperative agents," in *Proc. 10th Int. Conf. Mach. Learn.*, 1993, pp. 330–337.
- [37] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6379–6390.
- [38] K. Lin, R. Zhao, Z. Xu, and J. Zhou, "Efficient large-scale fleet management via multi-agent deep reinforcement learning," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 1774–1783.
- [39] J. K. Terry, N. Grammel, A. Hari, L. Santos, and B. Black, "Revisiting parameter sharing in multi-agent deep reinforcement learning," 2021, *arXiv:2005.13625*.
- [40] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.
- [41] Y. Wu, E. Mansimov, S. Liao, R. Grosse, and J. Ba, "Scalable trust-region method for deep reinforcement learning using Kronecker-factored approximation," 2017, *arXiv:1708.05144*.
- [42] I. Elsayed-Aly, S. Bharadwaj, C. Amato, R. Ehlers, U. Topcu, and L. Feng, "Safe multi-agent reinforcement learning via shielding," 2021, *arXiv:2101.11196*.
- [43] Z. Cai, H. Cao, W. Lu, L. Zhang, and H. Xiong, "Safe multi-agent reinforcement learning through decentralized multiple control barrier functions," 2021, *arXiv:2103.12553*.
- [44] M. Hausknecht and P. Stone, "Deep recurrent Q-learning for partially observable MDPs," 2015, *arXiv:1507.06527*.
- [45] N. Li, H. Chen, I. Kolmanovsky, and A. Girard, "An explicit decision tree approach for automated driving," in *Proc. Dyn. Syst. Control Conf.*, vol. 58271, 2017, Art. no. V001T45A003.
- [46] D. Chen, L. Jiang, Y. Wang, and Z. Li, "Autonomous driving using safe reinforcement learning by incorporating a regret-based human lane-changing decision model," in *Proc. Amer. Control Conf. (ACC)*, Jul. 2020, pp. 4355–4361.
- [47] *A Policy on Geometric Design of Highways and Streets*, Amer. Assoc. State Highway Transp. Officials, Washington, DC, USA, 2001.
- [48] C. Thiemann, M. Treiber, and A. Kesting, "Estimating acceleration and lane-changing dynamics from next generation simulation trajectory data," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2088, no. 1, pp. 90–101, Jan. 2008.
- [49] T. J. Ayres, L. Li, D. Schleuning, and D. Young, "Preferred time-headway of highway drivers," in *Proc. IEEE Intell. Transp. Syst.*, Aug. 2001, pp. 826–829.
- [50] M. Bouton, A. Nakhaei, K. Fujimura, and M. J. Kochenderfer, "Cooperation-aware reinforcement learning for merging in dense traffic," in *Proc. IEEE Intell. Transp. Syst. Conf. (ITSC)*, Oct. 2019, pp. 3441–3447.
- [51] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 62, no. 2, pp. 1805–1824, Aug. 2000.
- [52] A. Kesting, M. Treiber, and D. Helbing, "General lane-changing model MOBIL for car-following models," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1999, no. 1, pp. 86–94, Jan. 2007.
- [53] P. Polack, F. Alché, B. d'Andréa-Nové, and A. De La Fortelle, "The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles?" in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2017, pp. 812–818.
- [54] D. H. Wolpert and K. Tumer, "Optimal payoff functions for members of collectives," in *Modeling Complexity in Economic and Social Systems*. Singapore: World Scientific, 2002, pp. 355–369.
- [55] J. Wang, Y. Zhang, T.-K. Kim, and Y. Gu, "Shapley Q-value: A local reward approach to solve global reward games," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 5, pp. 7285–7292.
- [56] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.
- [57] D. Bagnell and A. Ng, "On local rewards and scaling distributed reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 18, 2005, pp. 91–98.
- [58] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, nos. 3–4, pp. 229–256, May 1992.
- [59] S. Huang and S. Ontañón, "A closer look at invalid action masking in policy gradient algorithms," 2020, *arXiv:2006.14171*.
- [60] C.-M. Chou, C.-Y. Li, W.-M. Chien, and K.-C. Lan, "A feasibility study on vehicle-to-infrastructure communication: WiFi vs. WiMAX," in *Proc. 10th Int. Conf. Mobile Data Manag., Syst., Services Middleware*, 2009, pp. 397–398.
- [61] Edouard Leurent. (2018). *An Environment for Autonomous Driving Decision-Making*. [Online]. Available: <https://github.com/eleurent/highway-env>
- [62] M. Kaushik, V. Prasad, K. M. Krishna, and B. Ravindran, "Overtaking maneuvers in simulated highway driving using deep reinforcement learning," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2018, pp. 1885–1890.
- [63] W. Cao, M. Mukai, and T. Kawabe, "Two-dimensional merging path generation using model predictive control," *Artif. Life Robot.*, vol. 17, nos. 3–4, pp. 350–356, Feb. 2013.
- [64] Z. Tang et al., "SKFAC: Training neural networks with faster Kronecker-factored approximate curvature," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 2408–2417.

- [65] C. Yu et al., "The surprising effectiveness of PPO in cooperative, multi-agent games," 2021, *arXiv:2103.01955*.
- [66] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," 2015, *arXiv:1506.02438*.
- [67] Y. Lin, J. McPhee, and N. L. Azad, "Comparison of deep reinforcement learning and model predictive control for adaptive cruise control," *IEEE Trans. Intell. Vehicles*, vol. 6, no. 2, pp. 221–231, Jun. 2021.
- [68] D. M. Saxena, S. Bae, A. Nakhaei, K. Fujimura, and M. Likhachev, "Driving in dense traffic with model-free reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 5385–5392.
- [69] M. Alshiekh, R. Bloem, R. Ehlers, B. Konighofer, S. Niekum, and U. Topcu, "Safe reinforcement learning via shielding," in *Proc. AAAI Conf. Artif. Intell.*, 2018, vol. 32, no. 1, pp. 1–10.
- [70] J. Garcia and F. Fernandez, "A comprehensive survey on safe reinforcement learning," *J. Mach. Learn. Res.*, vol. 16, no. 42, pp. 1437–1480, Aug. 2015.
- [71] W. Zhao, J. P. Queraltà, and T. Westerlund, "Sim-to-real transfer in deep reinforcement learning for robotics: A survey," in *Proc. IEEE Symp. Ser. Comput. Intell. (SSCI)*, Dec. 2020, pp. 737–744.



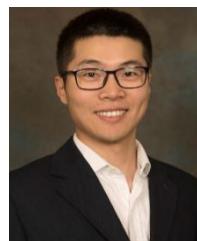
**Dong Chen** (Graduate Student Member, IEEE) received the B.E. degree from the University of Electronic Science and Technology of China, Sichuan, China, in 2017. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, Michigan State University, East Lansing, MI, USA. His research interests include reinforcement learning, multi-agent systems, and autonomous vehicles.



**Kaian Chen** received the M.S. degree in electrical and computation engineering from The Ohio State University, Columbus, OH, USA, in 2017. He is currently pursuing the Ph.D. degree in mechanical engineering with Michigan State University. His research interests include nonlinear system identification, model predictive control, and autonomous vehicles.



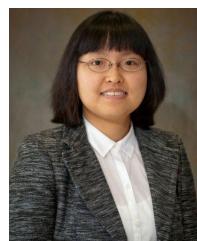
**Yongqiang Wang** (Senior Member, IEEE) was born in Shandong, China. He received the B.S. degree in electrical engineering and automation, the B.S. degree in computer science and technology from Xi'an Jiaotong University, Xi'an, Shaanxi, China, in 2004, and the M.Sc. and Ph.D. degrees in control science and engineering from Tsinghua University, Beijing, China, in 2009. From 2007 to 2008, he was a Visiting Student with the University of Duisburg-Essen, Germany. He was a Project Scientist with the University of California at Santa Barbara, Santa Barbara, CA, USA, before joining Clemson University, Clemson, SC, USA, where he is currently an Associate Professor. His current research interests include decentralized control, optimization, and learning, with an emphasis on privacy and security. He also serves as an Associate Editor for *IEEE TRANSACTIONS ON AUTOMATIC CONTROL* and *IEEE TRANSACTIONS ON CONTROL OF NETWORK SYSTEMS*.



**Longsheng Jiang** received the B.S. degree from the Taiyuan University of Technology, Taiyuan, China, in 2011, and the M.S. degree from the University of Florida, Gainesville, FL, USA, in 2014. He is currently pursuing the Ph.D. degree with the Department of Mechanical Engineering, Clemson University, Clemson, SC, USA. His research interests include human-like decision-making, human behavior modeling, human-robot collaboration, and control systems.



**Mohammad R. Hajidavalloo** received the B.Sc. and M.Sc. degrees in mechanical engineering from the University of Tehran in 2016 and 2018, respectively. He is currently pursuing the Ph.D. degree with the Department of Mechanical Engineering, Michigan State University. His research interests include learning-based control, optimal control, and reinforcement learning.



**Yue Wang** (Senior Member, IEEE) received the B.S. degree in mechanical engineering from Shanghai University, Shanghai, China, in 2005, and the M.S. and Ph.D. degrees in mechanical engineering from the Worcester Polytechnic Institute, Worcester, MA, USA, in 2008 and 2011, respectively. She was a Post-Doctoral Research Associate with the Electrical Engineering Department, University of Notre Dame, Notre Dame, IN, USA. She is currently the Warren H. Owen-Duke Energy Associate Professor of engineering and the Director of the Interdisciplinary and Intelligent Research (I2R) Laboratory, Department of Mechanical Engineering, Clemson University. Her research interests include human-robot interaction systems, multi-robot systems, and cyber-physical systems. She is an active ASME Member. She was a recipient of the Air Force Office of Scientific Research Young Investigator Award in 2016 and the NSF CAREER Award in 2015. She serves as an Associate Editor for the *IEEE Robotics and Automation Magazine* and the *ASME Journal of Autonomous Vehicles and Systems*. She is also a Technical Editor of the IEEE/ASME TRANSACTIONS ON MECHATRONICS.



**Zhaojian Li** (Senior Member, IEEE) received the B.Eng. degree from the Nanjing University of Aeronautics and Astronautics in 2010 and the M.S. and Ph.D. degrees in aerospace engineering (flight dynamics and control) from the University of Michigan, Ann Arbor, MI, USA, in 2013 and 2015, respectively. He is currently an Assistant Professor with the Department of Mechanical Engineering, Michigan State University. His research interests include learning-based control, nonlinear and complex systems, and robotics and automated vehicles. He was a recipient of the NSF CAREER Award.