Multi-agent Decision-making at Unsignalized Intersections with Reinforcement Learning from Demonstrations

Chang Huang, Junqiao Zhao*, *Member, IEEE,* Hongtu Zhou, Hai Zhang, Xiao Zhang, Chen Ye

***Abstract*— Intersections are key nodes and also bottlenecks of urban road networks, so improving the traffic efficiency at intersections is beneficial to improving overall traffic throughput and mitigating traffic congestion. Previous methods such as rule-based, planning-based, and single-agent reinforcement learning usually oversimplify the policies of the surrounding vehicles and thus have difficulty modeling the complex interaction behaviors between vehicles, which limits the performance of these methods to some extent. Instead, we adopt a multi-agent reinforcement learning (MARL) approach to train and coordinate the policies of all vehicles to handle unsignalized intersection scenarios. Nevertheless, due to complex interactions between multiple agents, it is challenging to efficiently explore the environment and obtain high-reward samples. We therefore propose to pre-train the policy using demonstration data consisting of expert data and interaction data to improve the initial performance of agents and improve exploration, as well as to reduce the distributional shift between the demonstration data and the environmental interaction data. We experimentally prove that using interaction data generated by the algorithm in the demonstration data improves training stability. The proposed method enables effective exploration and greatly speeds up the training process.**

I. INTRODUCTION

The policy learning at intersections is one of the most challenging problems and has been an important topic of research. Many previous methods such as rule-based [1], planning-based [2] and reinforcement learning (RL) [3] methods usually treat it as a single-agent problem and focus on the policy learning of ego-vehicle. These methods consider other vehicles as part of the environment and assume that they act according to fixed strategies. Therefore, the complex interactions between vehicles can not be modeled.

Recent studies are starting to apply multi-agent reinforcement learning (MARL) to policy learning at intersections [4] [5] [6], as it can be expected that in the future vehicles can be connected and a unified strategy can be adopted to coordinate and control all vehicles. We therefore consider policy learning at unsignalized intersections as a multi-agent problem and employ MARL method to efficiently learn cooperative policies.

However, interactions between vehicles at unsignalized intersections are very complex. Each vehicle must decide when to cross the intersection or slow down to give way, which is a complex problem even for humans. If collisions occur frequently, vehicles may become too cautious to avoid interactions. Furthermore, the global state-action space dimension of MARL grows exponentially with the number of agents, which requires a local-global exploration trade-off [7]. Exploring only from a global perspective is inefficient while exploring only from a local perspective lacks clear guidance for more valuable spaces. Therefore, the complex interactions and large global state-action space make it extremely difficult to successfully pass through the intersection, and the algorithm may converge to a suboptimal policy early in training due to insufficient exploration.

So it is of great importance to use an effective exploration method that can make full use of the interaction information. Many such methods model the interaction process by computing mutual information [8], causal influence [9], and social influence [10] among agents, and use this interaction information to encourage agents to explore those spaces where interactions between agents occur. These methods have achieved remarkable progress in the field of games. However, the MARL methods applied in the field of autonomous driving [4] [5] [6] do not pay much attention to the exploration problem. This motivates us to investigate better exploration methods to deal with complex interaction scenarios like unsignalized intersections more effectively.

In this paper, we propose to add a pre-training stage to the original training process of the value decomposition MARL method QMIX [11] to improve initial performance and exploration ability, which helps to obtain high-reward samples in the early stage of subsequent online training. The demonstration data used for pre-training consists of expert data and interaction data, where the interaction data is generated by the algorithm itself through interacting with the environment, rather than a fixed dataset like other similar methods [12] [13] [14]. This is important because using self-generated interaction data helps to reduce the mismatch between the data distribution induced by the algorithm and the data distribution contained in the demonstration data [15], which makes the transition between pre-training and online training smoother and more stable.

In pre-training, supervised margin loss, $TD(\lambda)$ [16], and regularization term are used to imitate the demonstrator while avoiding overfitting. The experimental results show that the number of interaction samples required by our proposed method QMIXwD is much smaller than that of QMIX, which confirms the effectiveness of our method in improving exploration and sample efficiency.

One of the contributions of this paper is that we use

*This work is supported by the National Key Research and Development Program of China (No. 2021YFB2501104, No. 2020YFA0711402)

All the authors are with the Department of Computer Science and Technology, Tongji University, China and the MOE Key Lab of Embedded System and Service Computing, Tongji University, China, e-mail: (zhaojunqiao@tongji.edu.cn)

2023 IEEE Intelligent Vehicles Symposium (IV) | 979-8-3503-4691-6/23/$31.00 ©2023 IEEE | DOI: 10.1109/IV55152.2023.10186792

a MARL method based on value decomposition to learn control policies for multiple vehicles at intersections and combine it with learning from demonstrations to efficiently improve exploration. Another contribution is that we experimentally prove that using self-generated interaction data in the demonstration data improves training stability.

II. Related Works

A. Policy learning at intersections

A lot of work has focused on policy learning of one single vehicle at intersections. [1] calculates time-to-collision (TTC [17]) with the assumption that surrounding vehicles move at a constant speed to judge whether it is safe to cross the intersection. In [2], a partially observable markov decision process (POMDP) model is constructed to estimate the intention of other vehicles, and the model is combined with a planning method to generate an optimal trajectory. [3] proposed a driving policy based on spatial and temporal attention to handle unprotected intersections using RL.

In order to achieve more complex interactive behaviors and further improve traffic efficiency at intersections, other researchers adopt MARL methods. [4] developed Coordinated Policy Optimization (CoPO) to coordinate agents' behaviors while still maximizing individual objectives. Each agent receives its own reward and can become competitive with others, which is not adaptable to our fully-cooperative task. [5] incorporates a vehicle selection method into MADDPG [18] to cooperate between different numbers of vehicles. But they conducted experiments on a simplified intersection scenario where vehicles only go straight. [6] applies QMIX to control various autonomous vehicles in mixed-autonomy traffic of different densities. The method in [6] is most relevant to ours but it only applies some implementation techniques including $Q(\lambda)$, reward clipping, and network-level improvements on QMIX. In contrast, our method effectively encourages exploration and greatly improves sample efficiency.

B. Multi-agent exploration

Many methods considering interaction information have been proposed to encourage exploration. EMC [19] encourages collaborative exploration with intrinsic reward calculated by the prediction error of individual action values and introduces episodic memory to efficiently utilize explored informative experience to speed up policy training. MAVEN [8] uses mutual information between the trajectories of agents to capture the interaction relationship to achieve coordinated exploration. Other methods such as causal influence [9] and social influence [10] are proposed to model the interaction process. Rather than modeling the interactions explicitly or implicitly, our approach directly imitates the demonstrator to perform high-quality interactive behaviors. Exploration is encouraged by improving the initial performance of the algorithm to avoid premature convergence due to insufficient exploration in the early stage of training.

C. RL from demonstrations

Reinforcement learning from demonstrations (RLfD) is a combination of RL and imitation learning and was first introduced in [20], which shows how learning from demonstrations is beneficial for the RL process. Deep Q-learning from demonstrations (DQfD) [12] and DDPG from demonstrations (DDPGfD) [13] respectively combine DQN and DDPG with learning from demonstrations and significantly accelerate the training process. [21] applies a similar method in soft actor-critic structure and has validated its effectiveness in a simulated urban roundabout scenario under a single-agent setting. [14] extends RLfD in a multi-agent actor-critic method for flocking control. Instead, we adopt RLfD on a value decomposition MARL method and propose to use self-generated interaction data in pre-training.

III. Methodology

A. Problem definition and revisiting value decomposition

1) Dec-POMDP: The multi-agent decision process at an intersection can be modeled as a decentralized partially observable markov decision process (Dec-POMDP [22]) consisting of a tuple $G =< S, A, O, \Omega, P, r, n, \gamma >$. $s \in S$ is the global state. After performing the joint action of n agents $\boldsymbol{a} \in \boldsymbol{A}, \boldsymbol{a} = \{a_1, ..., a_n\}$, a transition from s to the state at the next moment s' occurs according to the state transition function $P(s'|s, \boldsymbol{a})$ and all agents get a shared reward $r(s, \boldsymbol{a}) \in \mathbb{R}$. In partially observable scenarios, each agent can only obtain the observation $o_i \in \Omega$ of part of the environment according to the observation function $O(o_i|s, \boldsymbol{a})$ and has an individual policy $\pi_i(a_i|\tau_i)$ where τ_i is the action-observation history. The objective is to find an optimal joint policy $\boldsymbol{\pi} = \{\pi_1, ..., \pi_n\}$ to maximize the joint value function $V^{\pi}(s) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_t]$.

2) Value decomposition: Value decomposition based on centralized training and decentralized execution (CTDE) framework is one of the most commonly used methods in MARL. Global state information is provided during training to get a more accurate joint action value function $Q_{jt}(s, \boldsymbol{a})$ while individual action value function $Q_i(\tau_i, a_i)$ only receives its own observation to achieve full decentralization. Q_{jt} can be decomposed into Q_i and is trained by the TD error.

$$L_{TD} = \sum_{i=1}^{b} [(y - Q_{jt}(s, \boldsymbol{a}))^2] \tag{1}$$

where $y = r + \gamma max_{\boldsymbol{a}'} Q_{jt}(s', \boldsymbol{a}')$ and b is the mini-batch size. However, the complexity of calculating target y is too high for practical application since the joint action space grows exponentially with the number of agents. In order to achieve decentralized execution while reducing the computational complexity, individual-global-max (IGM) condition is proposed, where a global $argmax$ performed on Q_{jt} yields the same result as a set of individual $argmax$ operations performed on each Q_i:

$$\arg\max_{\boldsymbol{a}\in\boldsymbol{A}} Q_{jt}(s,\boldsymbol{a}) = \\ (\arg\max_{a_1\in A} Q_1(\tau_1,a_1),...,\arg\max_{a_n\in A} Q_n(\tau_n,a_n)) \tag{2}$$

3) QMIX: QMIX is a state-of-the-art algorithm in value decomposition MARL. It adds monotonicity constraints through a mixing network generated based on global state through a hypernet to meet the IGM condition.

$$Q_{jt}^{QMIX} = f_{mixing}(Q_1,...,Q_n;s) \tag{3}$$

$$\frac{\partial Q_{jt}^{QMIX}(s,\boldsymbol{a})}{\partial Q_i(\tau_i,a_i)} \geq 0, i=1,...,n \tag{4}$$

B. MARL from demonstrations

At present, QMIX and other value decomposition MARL methods most commonly use individual ϵ-greedy exploration techniques, which cannot effectively deal with complex interactive scenarios [23]. We therefore propose to add a pre-training stage to learn from demonstrations to improve exploration and name our method QMIXwD.

Unlike previous methods where the demonstration data used for pre-training was a fixed dataset completely generated by experts, we propose to use both expert data and self-generated interaction data to improve the stability of the algorithm. This will, on the one hand, avoid the algorithm overfitting to the expert policy and thus save time in the subsequent online training stage for correcting the Q-values. On the other hand, the interaction data generated by the currently trained algorithm can reduce the distributional shift between the current policy and the policy in demonstration data, which further stabilizes the transition from pre-training to subsequent online training.

The pre-training loss function consists of three parts, supervised margin loss, TD error and regularization term. Supervised margin loss makes the action of policy execution as consistent as possible with the demonstration data and is a key component to improve the initial performance of the algorithm. It will only be applied to the individual action value function. Keeping $\max_{a\in A} Q_i(\tau_i,a)$ constant after gradient update by supervised margin loss helps improve training stability as $\max_{a\in A} Q_i(\tau_i,a)$ is used when calculating TD target y. Therefore, in order to imitate the demonstrator while trying to stabilize training, we design the supervised margin loss to ensure that the maximum individual action value remains as constant as possible:

$$L_{demo} = [Q_i(\tau_i,a_i) - (\max_{a\in A} Q_i(\tau_i,a) - l(a_{demo},a_i)]^2 \tag{5}$$

where a_{demo} is the actual action in demonstration data and $l(a_{demo},a_i)$ is a margin function that is 0 when $a_i = a_{demo}$ and a positive constant otherwise.

TD error is designed to satisfy the Bellman equation to achieve a better transition between the pre-training stage and the online training stage and improve the stability of the algorithm. We use $TD(\lambda)$ to balance short-term and long-term rewards:

$$L_{TD(\lambda)} = \sum_{i=1}^{b}[(G_t^\lambda - Q_{jt}(s,\boldsymbol{a}))^2] \tag{6}$$

where G_t^λ is $TD(\lambda)$ at time t.

Finally, a regularization item is added to prevent the algorithm from overfitting in the pre-training stage, making it easier to adapt to subsequent online training stage. This regularization term is applied to both the individual action value network and the mixing network:

$$L_{L2} = \sum_{i=1}^{b}(||\theta||_2^2) \tag{7}$$

where θ is the network parameters of QMIX.

The overall loss for updating the network parameters is:

$$L_{tot} = \lambda_1 L_{demo} + \lambda_2 L_{TD(\lambda)} + \lambda_3 L_{L2} \tag{8}$$

Only $L_{TD(\lambda)}$ is considered during online training and the demonstration data is no longer used. The online training process is exactly the same as the original QMIX. Individual action value function shares parameters across all agents.

The whole algorithm is presented in Algorithm 1.

Algorithm 1 QMIXwD

```
1: Inputs: $\theta$: weights for individual action value network, $\phi$: weights for mixing network, $\eta$: expert data ratio in demonstration data, $E$: demonstration data buffer size, $k$: number of pre-training gradient updates, $m$: number of online-training gradient updates
2: collect $E$ episodes using pre-trained expert and initialized $\theta$ according to $\eta$ and store them in buffer $D_{demo}$.
3: for $steps \leftarrow 1, k$ do
4:     Sample a mini-batch from $D_{demo}$
5:     Update $\theta$ and $\phi$ using loss $L_{tot}$
6: end for
7: for $steps \leftarrow 1, m$ do
8:     episode_data ← []
9:     reset environment status
10:    while episode not terminated do
11:        each agent $i$ selects action $a_i$ based on $\epsilon$-greedy
12:        play actions $\boldsymbol{a}$ and observe $(s', r)$
13:        append $(s, \boldsymbol{o}, \boldsymbol{a}, s', \boldsymbol{o}', r)$ into episode_data
14:    end while
15:    store episode_data in buffer $D$
16:    Sample a mini-batch from $D$
17:    Update $\theta$ and $\phi$ using loss $L_{TD(\lambda)}$
18:    Update target networks
19: end for
```

IV. EXPERIMENTS AND ANALYSIS

In this section, we test the effectiveness of the proposed method. We designed an intersection scenario where careful interactions are required to successfully cross the intersection.

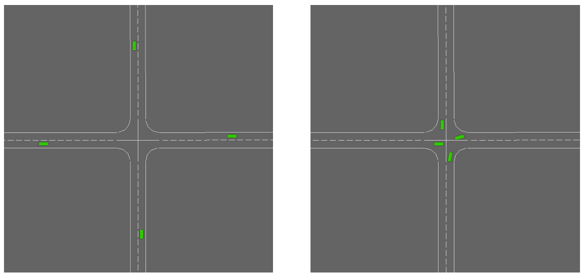

Fig. 1. An intersection scenario in highway-env.

A. Scenario setting

1) Task: We conduct our experiments in the highway-env environment [24]. The scenario is a crossroad with one vehicle coming from each direction as shown in Figure 1.

To meet the condition of decentralized execution, each vehicle is controlled individually by a policy. Each vehicle is initialized with a predetermined route with a random choice of turning left, going straight, or turning right. Vehicles will travel along the predetermined routes, and the policies need to control the speed. The goal of this task is that all vehicles will cross the intersection and reach their respective destinations as quickly and safely as possible.

2) State: The global state consists of the position (x, y), velocity (v_x, v_y), heading angle (ψ), and the future trajectory $\{x_0, y_0, ..., x_T, y_T\}$ within a period of time calculated with a constant speed of every vehicle following the predetermined routes.

3) Observation: The observation contains the information of ego-vehicle including position, velocity, heading angle, and the future trajectory which are identical to their counterparts in the global state. At the same time, the current relative position, relative velocity, and relative heading angle of surrounding vehicles can be observed. Under the partially observable setting, we assume that the intention and future trajectories of surrounding vehicles are not available.

4) Action: The steering wheel angle of the vehicle is calculated by the built-in control algorithm following the predetermined route, so we set the discrete action space as $\{decelerate, constant\ speed, accelerate\}$.

5) Reward: The reward consists of three parts including a success reward, a collision penalty, and a time penalty to encourage all vehicles to cross the intersection safely and quickly:

$$r = \begin{cases} 100, \text{if all vehicles cross the intersection safely} \\ -100, \text{if there is a collision} \\ 0, \text{otherwise} \end{cases} \tag{9}$$

Positive rewards are only earned when the task is successfully completed, so it is a sparse reward problem that requires a greater exploration ability. Adopting dense rewards at each step can guide the agent to learn basic skills faster, but engineering such a dense reward is not straightforward. Meanwhile, a sparse reward setting makes it more likely that the algorithm will learn a variety of high-level skills [25]. Therefore, we sought to enhance the exploration ability of the algorithm, with the intention of learning complex interaction skills.

B. Baseline

1) TTC: We design a rule-based strategy based on TTC for multiple agents. For agent i, we calculate the predicted trajectory for a period of time along the predetermined route, while for other agents, we still assume that the intention is unknowable and predict the future trajectory with constant speed and heading angle. If it can be inferred that agent i will collide with other agents following the predicted trajectories and TTC is less than a certain threshold, agent i will perform $deceleratee$ action, otherwise, it will perform $accelerate$ action.

2) PPO: PPO is a state-of-the-art reinforcement learning method and it has excellent adaptability in a multi-agent environment [26]. Each vehicle is controlled by a PPO agent and receives the same global reward.

3) QMIX: All agents share the parameters of individual action value network but receive their own partial observation to make decisions.

C. Experiment and algorithm setup

At the beginning of each episode, a vehicle is generated at a distance of $(60 + 5 \times \mathcal{N}(0, 1))m$ from the intersection on each entry road and the travel direction is randomly set. An episode has a maximum of 100 time steps. When all vehicles leave the intersection $25m$ away without collision, the episode is marked as $done$ and is successfully completed. If there is a collision, the episode terminates immediately. The policy execution frequency is 5Hz. For each run of a method, we test for 20 episodes with each agent performing greedy decentralized action selection every 20000 steps. Each result is averaged under 3 different seeds with 95% confidence intervals.

We design our algorithm based on the pymarl2 framework [27]. QMIX and QMIXwD both use the basic ϵ-greedy exploration technique. We tried to set a high initial ϵ which decays over time but we found that the performance of this method is very sensitive to the decay time parameter and the final performance varies greatly among different experiment trials. Thus we set ϵ to 0.1 in the whole training process to reduce the interference of this factor on our proposed method. The expert data in demonstration data is generated by a trained QMIX model. For the sake of fair comparison, the number of steps contained in the demonstration data is also counted into the final result curve. The demonstration data contains 100 episodes of expert data and 900 episodes of interaction data. The size of replay buffer for online training is 5000 episodes.

D. Main experiment

We tested our proposed method and other baselines at unsignalized intersections, as shown in Figure 1. The success rate, episodic return, and the average travel time needed in successful episodes are used as performance indicators. The results are shown in Figure 2.

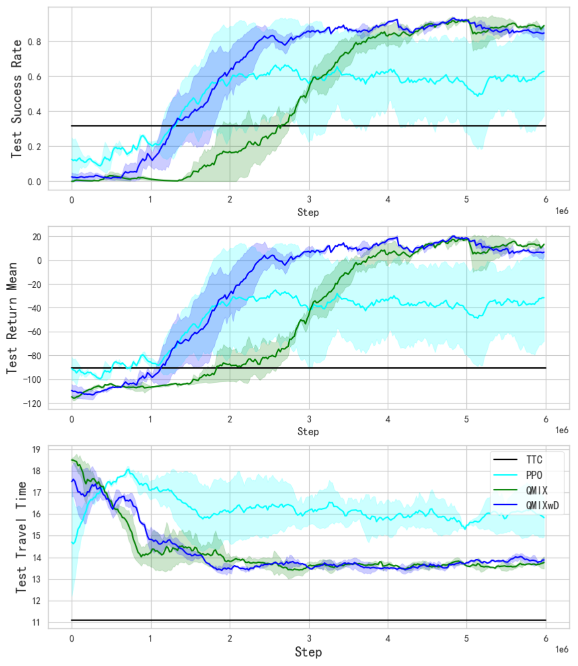

Fig. 2. Test success rate, test episodic return, and test travel time for each algorithm. The performance of TTC is averaged over 5M steps and shown as a straight line.

The travel time for TTC to cross the intersection is short but the success rate is very low. The reason is that when only a small number of vehicles' trajectories intersect, the TTC algorithm needs to slow down appropriately to avoid collision. But when it is necessary to deal with the interactions between multiple vehicles or all vehicles enter the intersection at the same time, TTC cannot determine which vehicle has the priority to cross the intersection first and is unable to handle this situation.

PPO fails to learn an effective policy in this complex situation. Because from the perspective of a single agent, other agents are part of the environment and such an environment is non-stationary [18], which will seriously hinder the training of the algorithm. Especially in this kind of intersection scenario with complex interactions, the actions of ego-vehicle are often punished due to collisions between other vehicles, which gives a wrong signal to its own policy learning and greatly increases the variance and instability in the training process.

The success rate of QMIX once dropped to around 0 at about 1M steps and the learned policy avoids interacting with other vehicles to reduce collisions at that time. After a period of exploration by ϵ-greedy it jumps out of the local optimal solution.

QMIXwD can obtain high-reward samples more easily in the early stage of training process, which reduces the risk

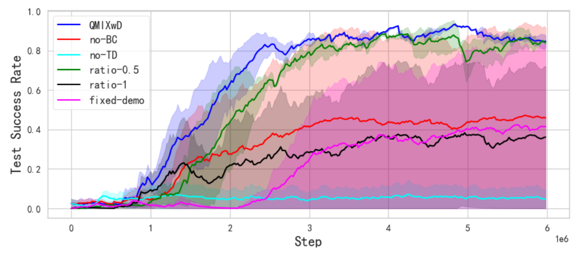

Fig. 3. Test success rate for ablation methods

of the algorithm converging to a suboptimal policy and also saves time for subsequent corrections. The training curves of QMIX and QMIXwD show that QMIXwD requires nearly 30% fewer interaction samples than QMIX, which means the pre-training process with demonstration data can help the agent learn the appropriate policy faster and significantly improve sample efficiency. However, QMIXwD requires more travel time than TTC because QMIXwD tends to adopt a more cautious strategy overall to deal with complex interactions. The question of how to learn more efficient policies while ensuring safety remains to be addressed.

E. Ablation study

We also designed ablation experiments to test the effectiveness of each part of our proposed method.

1) no-BC: The supervised margin loss term is canceled.

2) no-TD: The $TD(\lambda)$ loss term for pre-training is canceled.

3) ratio-0.5, ratio-1: The proportions of expert data in the demonstration data are 0.5 and 1 respectively.

4) fixed-demo: The demonstration data is fixed and the interaction data is generated by a randomly initialized model.

The test success rate results are shown in Figure 3.

Experimental results show that each part of our proposed method is meaningful. Canceling the supervised margin loss item or $TD(\lambda)$ hurts the performance of the algorithm and $TD(\lambda)$ is particularly important because after pre-training the individual action value functions by supervised margin loss, there is a severe mismatch between current joint action value estimates and the return in the episodes collected by individual agents, which seriously hinders training. From the training curves of experiments with different proportions of expert data or fixed demonstration data, it can be seen that both ratio-1 and fixed-demo suffer from huge variance during training and the final performance is far worse than QMIXwD, indicating that the smooth transition achieved by self-generated interaction data between pre-training and online training is of great benefit to improving the stability of the algorithm.

V. CONCLUSIONS

In this paper, we propose QMIXwD, combining the value decomposition method QMIX and learning from demonstration, to improve exploration and sample efficiency at complex unsignalized intersections. Dynamic demonstration data and self-generated interaction data are used to improve the stability of the algorithm. Supervised margin loss, $TD(\lambda)$, and regularization loss also help to achieve a smooth transition between pre-training and online training while imitating the demonstrator. Experimental results demonstrate that our proposed method can significantly speed up training.

In the future, we will research on improving the scalability of the algorithm to handle scenarios with more vehicles and try to learn a policy that can adaptively pursue high efficiency in simple scenarios while ensuring safety in complex scenarios.

REFERENCES

[1] A. Cosgun, L. Ma, J. Chiu, J. Huang, M. Demir, A. M. Añon, T. Lian, H. Tafish, and S. Al-Stouhi, "Towards full automated drive in urban environments: A demonstration in GoMentum Station, California," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, June 2017, pp. 1811–1818.

[2] C. Xia, M. Xing, and S. He, "Interactive Planning for Autonomous Driving in Intersection Scenarios Without Traffic Signs," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–11, 2022.

[3] H. Seong, C. Jung, S. Lee, and D. H. Shim, "Learning to Drive at Unsignalized Intersections using Attention-based Deep Reinforcement Learning," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. Indianapolis, IN, USA: IEEE, Sept. 2021, pp. 559–566.

[4] Z. Peng, Q. Li, K. M. Hui, C. Liu, and B. Zhou, "Learning to Simulate Self-driven Particles System with Coordinated Policy Optimization," in *Advances in Neural Information Processing Systems*, vol. 34. Curran Associates, Inc., 2021, pp. 10 784–10 797.

[5] T. Wu, M. Jiang, and L. Zhang, "Cooperative Multiagent Deep Deterministic Policy Gradient (CoMADDPG) for Intelligent Connected Transportation with Unsignalized Intersection," *Mathematical Problems in Engineering*, vol. 2020, pp. 1–12, July 2020.

[6] Z. Guo, Y. Wu, L. Wang, and J. Zhang, "Coordination for Connected and Automated Vehicles At Non-Signalized Intersections: A Value Decomposition-Based Multiagent Deep Reinforcement Learning Approach," *IEEE Transactions on Vehicular Technology*, pp. 1–11, 2022.

[7] T. Yang, H. Tang, C. Bai, J. Liu, J. Hao, Z. Meng, P. Liu, and Z. Wang, "Exploration in deep reinforcement learning: a comprehensive survey," *arXiv preprint arXiv:2109.06668*, 2021.

[8] A. Mahajan, T. Rashid, M. Samvelyan, and S. Whiteson, "MAVEN: Multi-Agent Variational Exploration," in *Advances in Neural Information Processing Systems*, vol. 32. Curran Associates, Inc., 2019.

[9] T. Zhang, Z. Liu, Z. Pu, and J. Yi, "Peer Incentive Reinforcement Learning for Cooperative Multi-Agent Games," *IEEE Transactions on Games*, pp. 1–14, 2022.

[10] N. Jaques, A. Lazaridou, E. Hughes, C. Gulcehre, P. Ortega, D. Strouse, J. Z. Leibo, and N. D. Freitas, "Social Influence as Intrinsic Motivation for Multi-Agent Deep Reinforcement Learning," in *Proceedings of the 36th International Conference on Machine Learning*. PMLR, May 2019, pp. 3040–3049.

[11] T. Rashid, M. Samvelyan, C. S. De Witt, G. Farquhar, J. Foerster, and S. Whiteson, "Monotonic value function factorisation for deep multi-agent reinforcement learning," *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 178:7234–178:7284, June 2022.

[12] T. Hester, M. Vecerik, O. Pietquin, M. Lanctot, T. Schaul, B. Piot, D. Horgan, J. Quan, A. Sendonaris, I. Osband, G. Dulac-Arnold, J. Agapiou, J. Leibo, and A. Gruslys, "Deep Q-learning From Demonstrations," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, Apr. 2018.

[13] M. Vecerik, T. Hester, J. Scholz, F. Wang, O. Pietquin, B. Piot, N. Heess, T. Rothörl, T. Lampe, and M. Riedmiller, "Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards," *arXiv preprint arXiv:1707.08817*, 2017.

[14] Y. Qiu, Y. Zhan, Y. Jin, J. Wang, and X. Zhang, "Sample-Efficient Multi-Agent Reinforcement Learning with Demonstrations for Flocking Control," in *2022 IEEE 96th Vehicular Technology Conference (VTC2022-Fall)*, Sept. 2022, pp. 1–7.

[15] S. Fujimoto, D. Meger, and D. Precup, "Off-Policy Deep Reinforcement Learning without Exploration," in *Proceedings of the 36th International Conference on Machine Learning*. PMLR, May 2019, pp. 2052–2062.

[16] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, ser. Adaptive Computation and Machine Learning Series. Cambridge, Massachusetts: The MIT Press, 2018.

[17] A. Van der Horst and J. Hogema, "Time-to-collision and collision avoidance systems," *Verkeersgedrag in Onderzoek*, 1994.

[18] R. Lowe, Y. WU, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch, "Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments," in *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017.

[19] L. Zheng, J. Chen, J. Wang, J. He, Y. Hu, Y. Chen, C. Fan, Y. Gao, and C. Zhang, "Episodic Multi-agent Reinforcement Learning with Curiosity-driven Exploration," in *Advances in Neural Information Processing Systems*, vol. 34. Curran Associates, Inc., 2021, pp. 3757–3769.

[20] S. Schaal, "Learning from Demonstration," in *Advances in Neural Information Processing Systems*, vol. 9. MIT Press, 1996.

[21] H. Liu, Z. Huang, J. Wu, and C. Lv, "Improved Deep Reinforcement Learning with Expert Demonstrations for Urban Autonomous Driving," in *2022 IEEE Intelligent Vehicles Symposium (IV)*, June 2022, pp. 921–928.

[22] F. A. Oliehoek and C. Amato, *A Concise Introduction to Decentralized POMDPs*. Springer, June 2016.

[23] H. T. Tse and H.-f. Leung, "Exploiting semantic epsilon greedy exploration strategy in multi-agent reinforcement learning," *arXiv preprint arXiv:2201.10803*, 2022.

[24] E. Leurent, "An environment for autonomous driving decision-making," *GitHub*, 2018.

[25] T. Bansal, J. Pachocki, S. Sidor, I. Sutskever, and I. Mordatch, "Emergent complexity via multi-agent competition," *arXiv preprint arXiv:1710.03748*, 2017.

[26] C. Yu, A. Velu, E. Vinitsky, Y. Wang, A. Bayen, and Y. Wu, "The surprising effectiveness of ppo in cooperative, multi-agent games," *arXiv preprint arXiv:2103.01955*, 2021.

[27] J. Hu, S. Jiang, S. A. Harding, H. Wu, and S.-w. Liao, "Rethinking the implementation tricks and monotonicity constraint in cooperative multi-agent reinforcement learning," *arXiv preprint arXiv:2102.03479*, 2021.