Multi-agent Graph Reinforcement Learning for Connected Automated Driving

Jiawei Wang [*1] **Tianyu Shi** [*1] **Yuankai Wu** [1] **Luis Miranda-Moreno** [1] **Lijun Sun** [1]

Abstract

Automated driving in multi-agent setting has attracted more and more attention recently. Cooperative driving between multiple automated vehicles is important to guarantee a safe and efficient transportation road system. In this paper, we demonstrate the effectiveness of integration with graph information sharing between each agent based on multi-agent reinforcement learning. Extensive experiments in various scenarios show that our method can achieve better performance in dynamic mixed-autonomy system against several state of the art baselines. To the best of our knowledge, while previous automated driving studies mainly focus on enhancing individual's driving performance, this work serves as a starting point for research on system-level multi-agent cooperation performance based on graph information sharing.

1. Introduction

Nowadays, the popularization of connected and automated vehicles (CAVs) has become unprecedentedly promising given the achievement in control and information technologies. One of the benefits from the utilization of CAVs is that the randomness in road network can be significantly reduced, thus there should be a new opportunity to improve the traffic efficiency. With the aim to introduce CAVs into transportation systems, there is a new need for understanding the mixed-autonomy traffic system where both automated and human-driven vehicles exist and interact with each other.

The main challenge in such a multi-agent system is how to encourage automated vehicle agents' cooperation so as to maximize the total expected returns of the whole system. For example, when there is a gap in front of the adjacent line of the automated vehicle, if the automated vehicle move up immediately, the following vehicle in the line will also increase its speed, which will end up a shock wave in traffic flow. Furthermore, as automated vehicles have different characteristics compared to human-driven agents (e.g., reacting time and action generation), how to formalize reasonable behavior for automated vehicle agent is also worth to explore. Finally, to guarantee a safe and efficient policy generation in multi-agent automated driving setting, combination of expert knowledge (e.g., three phase theory, catch-effect) in transportation domain is also extremely necessary.

In this research, we mainly focus on how to utilize the shared information for cooperation among CAVs to improve the traffic efficiency. To the best of our knowledge, this is the first approach to utilize graph neural networks in automated driving motion planning domain, which provides a valuable strategy for cooperative multi-agent control in the mixed autonomy systems. Our main contributions are:

(1) **Graph level information sharing:** Utilize the graph attention networks in the navigation setting of multi-agent reinforcement learning for mixed-autonomy cooperation.

(2) **Continuous action generation:** Integrate Proximal Policy Optimization (PPO) for continuous action generation.

(3) **Dynamic adjacency matrix:** Propose dynamic adjacency matrix scheme to exploit the information from neighbors and avoid waste from aimless surrounding consideration.

(4) **Generalization ability analysis:** Conduct extensive experiments in different difficult levels of transportation networks against several state of art baselines to demonstrate the effectiveness of our proposed approach.

2. Related work

According to our knowledge, most existing researches in automated vehicles control and motion planning field consider to maximize the travelling efficiency for individual agent. A large proportion of research formulates it as an optimization problem and solve it with rule based models (Xi et al., 2020; Luo et al., 2019). However, such methods may fail in the real world due to the complex interactions between each agents. The most recent progress for automated driving appears in reinforcement learning methods. Some

[*]Equal contribution [1]Department of Civil Engineering and Applied Mechanics, McGill University,Montreal, Canada. Correspondence to: Lijun Sun <lijun.sun@mcgill.ca>.

Proceedings of the 37^{th} International Conference on Machine Learning, Vienna, Austria, PMLR 108, 2020.

researchers designed a Q-function approximator that has a closed-form greedy policy to generate continuous control policy (Wang et al., 2018). Meanwhile, integration of micro-traffic simulator SUMO with deep reinforcement learning library rllib enables easy implementation of different traffic control tasks, e.g. lane change, ramp merge, etc (Wu et al., 2017). However, how to explore traffic domain knowledge to make the training more efficient and encourage cooperation among agents are still an open problem. There are also research demonstrates the effectiveness of using graph neural networks to model traffic participant interaction, e.g. (Diehl et al., 2019). Unlike their task, our approach focus on decision making for a group of agents. Multi-agent reinforcement learning (MARL) has also become a direction for cooperative automated driving, some researchers introduced a hierarchical temporal abstraction with a gating mechanism that significantly reduces the variance of the gradient estimation (Shalev-Shwartz et al., 2016). However, most MARL methods are still limited to local control rather than explore multi-agent cooperation in a mixed autonomy transportation system.

3. Methods

3.1. CAVs Control Framework

The CAVs control framework is given in Fig. 1. Specifically, N CAVs as N homogeneous agents are modeled in the mixed-autonomy traffic network. Their decision procedure can be divided into three stages: At the beginning of each decision, the agents $c_i, i = 1, ..., N$ will first have a local observation and identify their current state s_t^i; then each of them will manage to locate and communicate with their neighbors; once the agents acquire both information from itself and neighbors, they will accelerate/decelerate speed accordingly.

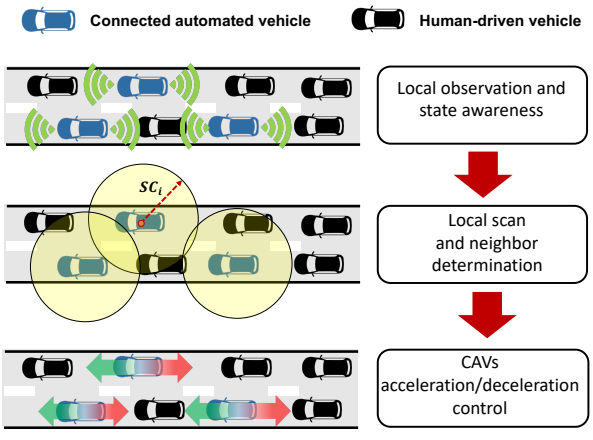

Figure 1. The illustration of CAVs control framework.

3.2. Dynamic Neighborhood Determination

With the aim to improve network traffic by controlling CAVs, cooperation should be exploited. Through cooperation, the scope of each CAV is extended thus they have more access to information from the environment. Moreover, establish control in cooperation manner promotes safety and efficiency in dynamic traffic flow. In this work, we introduce dynamic neighbor determination to exploit the relation among CAVs and facilitate their cooperation. First, for each CAV i, we suggest scan scale SC_i which can be considered as LiDar range in real automated driving settings and a relation vector m_i for neighbors identification. Specifically, j th entry of m_{ij} can be any predefined index (i.e., distance) to evaluate the strength of relation between CAV i and j. This setting makes sense in two-folds: On the one hand, in real-world scenarios the CAVs should not have unlimited scale of communication with other CAVs. On the other hand, the number of each CAVs' neighbor varies in traffic network, and the scan scale makes the framework more flexible without specifying fixed number of neighbors compared to traffic light control settings (Wei et al., 2019). After determining the neighbors, we can further construct weighted selection matrix $M_i, i = 1, ..., N$ for each CAV. Notably, j th row of M_i represents j th neighbors and it is a one-hot like vector, where entry with index of this neighbor can be binary or relation strength, if addition weight is imposed on this neighbor. In this way we can dynamically select feature from neighbors for each CAV.

3.3. MARL-CAVG Model

Graph attention network (GAT) is widely applied in mutli-agent control because of the its effectiveness to incorporate multi-dimension surrounding information for cooperation (Jiang et al., 2018; Wei et al., 2019). Based on the control framework and dynamic relation determination introduced above, in this study we suggest a graph attention based multi-agent reinforcement learning model. As shown in Fig. 2, the model to consider CAVs graph is highlighted. After state observation, the CAVs will integrate information from their neighbors to develop a more comprehensive awareness on current traffic dynamic. Intuitively, The observation and extracted features of each agent are integrated through graph convolution based on the weighted selection matrix $M_i, i = 1, ..., N$:

$$h_i^k = f\left(\text{concat}\left[M_i H^{k-1}, D_i^{-1} M_i H^{k-1}\right] W_i\right) \quad (1)$$

Where f is the activate function and h_i^k denotes extracted feature by agent i at k th layer, which depends on the current adjacency matrix M_i as well as the feature of its neighbors extracted at previous layer $H^{k-1} = \left[h_1^{k-1}, ..., h_N^{k-1}\right]$.

Furthermore, attention module is added to emphasize the impact from the surrounding agents. As mentioned before,

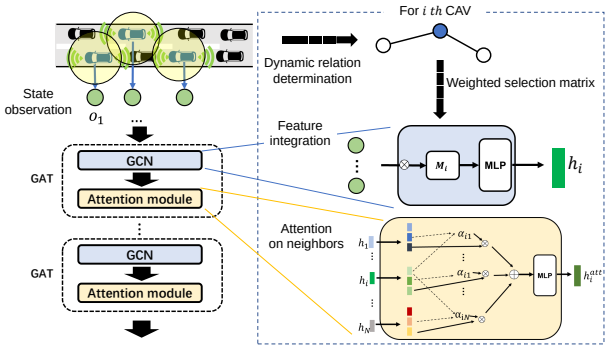

Figure 2. Graph attention on CAVs.

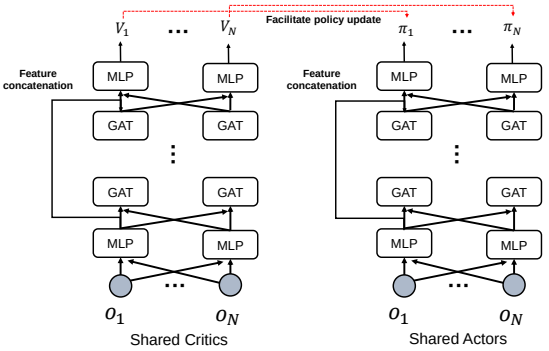

Figure 3. The architecture of MARL-CAVG.

the neighbors are selected within the scan scale SC_i for each CAV respectively. Considering N_i neighbors of ego CAV i, the attention score on neighboring CAV j can be calculated as:

$$q_i = f^{\text{query}}\left(h_i * W^{\text{query}}\right) \tag{2}$$

$$k_j = f^{\text{key}}\left(h_j * W^{\text{key}}\right), j \in \mathcal{N}_i \tag{3}$$

$$\alpha_{ij} = \text{softmax}\left(\frac{q_i * k_j^T}{\sum_{l\in\mathcal{N}_i} q_i * k_l^T}\right) \tag{4}$$

Note that we omit the layer index here for simplicity. f^{query} and f^{key} are used to encode input feature as query-value pairs, W^{query} and W^{key} are learning parameters, then dot-product between query and value vector is conducted, with which we can quantify the strength of relationship between two entities (Vaswani et al., 2017). With attention scheme, ego CAV can further selectively utilize information from neighboring CAVs and thus promote a more effective cooperation.

Based on the graph attention on CAVs, the multi-agent reinforcement learning architecture is established as shown in Fig. 3. Specifically, CAVs learn their policies with PPO as basic optimization scheme to handle continuous action space. The overall architecture is based on the actor-critic algorithm (Sutton & Barto, 2018) as shown in Fig. 3 The critic network is a graph convolutional neural network parametrized with α. Notably, the output of critic network at each time step t is state value estimation V_i (i.e., short for $V(S_t, M_i)$), it will be further used for advantage estimation to train the actor network.

The update gradient for critic is based on Temporal-Difference (TD) learning, which can be formulated as:

$$\bigtriangledown_\alpha L(\alpha) = \bigtriangledown_\alpha E\left[\sum_{n=1}^{N}(r_i^t + \gamma\hat{V}(S_{t+1}, M_i) - \hat{V}(S_t, M_i))^2\right] \tag{5}$$

The policy π_i (i.e., short for $\pi(a_i|S, M_i)$) can be modelled as a distribution (i.e., Gaussian distribution for continuous control) and also parameterized through the graph convolutional network with parameters θ. Then the policy gradient can be derived with the advantage $\hat{A}_i, i = 1, .., N$ from critic:

$$\bigtriangledown_\theta J(\theta) = \bigtriangledown_\theta E_{\pi_{\theta^{old}}}\left[\sum_{n=1}^{N}\min\left(\frac{\pi_\theta(a_i|S_t, M_i)}{\pi_{\theta^{old}}(a_i|S_t, M_i)}\hat{A}_i,\right.\right.$$
$$\left.\left.\text{clip}\left(\frac{\pi_\theta(a_i|S_t, M_i)}{\pi_{\theta^{old}}(a_i|S_t, M_i)}, 1-\epsilon, 1+\epsilon\right)\hat{A}_i\right)\right] \tag{6}$$

It should be pointed out that the selection matrices are kept the same for next state value prediction for model simplicity. Such assumption makes sense since the variation is limited between two consequent state observation, especially when the experiment is studied in fine granularity (i.e., simulation resolution is 1 s). In addition, we perform on-policy roll-out to collect the experience and the advantage estimation for agent i at step T is calculated as: $\hat{A}_i^t = \sum_t^T \gamma^t r_i^t - \hat{V}(S_t, M_i)$.

4. Experiments

We conduct experiments on Flow[1], an open-source traffic simulator that supports mixed autonomy control. Three

[1] https://github.com/flow-project

closed loop networks (as shown in Figure 4) are built to evaluate our algorithms.

In this section, we compare our method with different baselines[2] from both transportation engineering community and reinforcement learning community.

From the Table 1, we can find that our model outperforms all baselines in three different networks. In the simulation, we find that MARL-CAVG can learn to coordinate the velocity with each other, rather than blindly accelerate to achieve high speed in ring network. In addition, the MARL-CAVG agent is capable of learning to follow closely with the leader vehicle in figure eight network, so as to mitigate congestion in the intersection. Positive result also appears in roundabout within mini-city network, the MARL-CAVG agent can to accelerate smoothly in order to reduce shock-wave generation.

Table 1. Returns within different scenarios

Model.	Ring	Figure eight	Mini-city
MARL-CAVG	**2779.51**	**386.46**	**291.91**
MADDPG	2668.84	365.02	287.86
DDPG	1987.69	238.73	252.90
PPO	1999.37	358.28	249.91
IDM	424.12	37.34	12.35

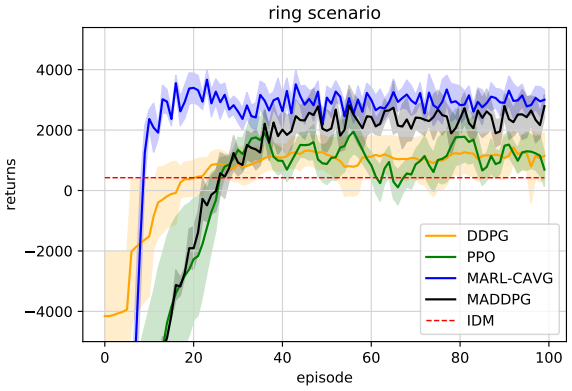

Figure 5. Training performance with 10 random seeds in ring network of 6 RL agents controlled by MARL-CAVG, MAPPO, DDPG models against all human drivers controlled by IDM model.

First, we evaluate the training performance in ring network as shown in Fig. 5. We find that MARL-CAVG agent has overall best training performance against all baseline methods including both traditional transportation method and reinforcement learning methods. Furthermore, multi-agent training methods have better initial performance than single agent training method. We also compare the the velocity performance before and after the automation is turned on. The results are shown in Fig 6. We can see from the picture that after automation turns on, the velocity become stable and MARL-CAVG achieves the best velocity performance with 4.99 m/s.

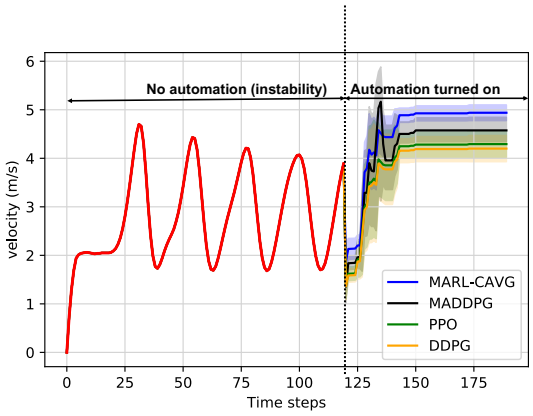

Figure 6. Velocity performance with 10 random seeds in ring network of 6 RL agents controlled by MARL-CAVG, MADDPG, DDPG, and PPO models.

5. Conclusion

In this paper, we propose a graph convolutional reinforcement learning approach for connected automated driving to encourage the cooperation during mixed autonomy traffic control. Specifically, our method constructs the multi-agent reinforcement learning model by leveraging multi-head dot-product attention as the convolutional kernel to compute interactions between agents. We conduct extensive experiments based on different road networks and demonstrate the superior performance of our proposed method over both reinforcement learning and transportation baselines. However, it should be noted that there are also some directions for improvements. First, as multi-agent training is quite unstable, a small change of environment setting will result in large return shift, explore how to better stabilize the training is necessary. In addition, we only consider the closed loop road network without inflow and outflow change. However, for real-world application, the number of agents in an environment can be large and sparse. Besides, the number of agents might change due to agents leaving or entering the environment. As a result, research on various open loop networks (e.g. bottleneck, highway) will be an important direction .

[2]PPO, DDPG, and MAPPDG are based on RLlib implementation: https://docs.ray.io/en/latest/rllib.html

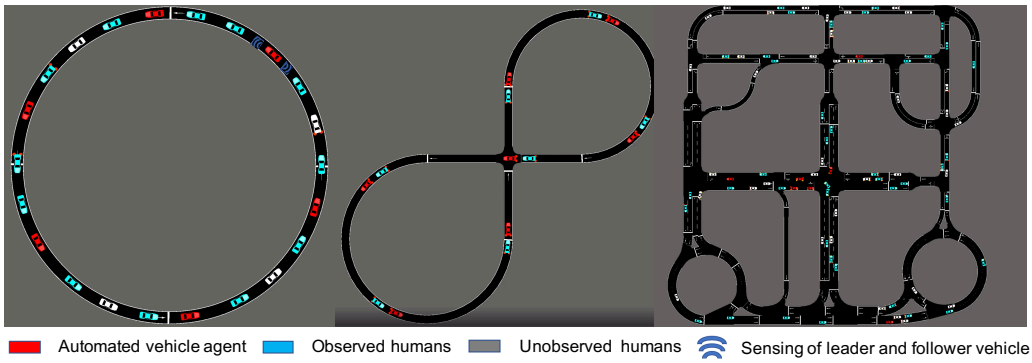

Figure 4. Various difficult levels of networks are built for evaluating our algorithms, **left:** ring network. It consists of only one single lane ring network. **middle:** figure eight network. It consists of two single lane ring network and an intersection. **right:** minicity network. It consists of multiple lane road, many roundabout and intersections.

References

Bando, M., Hasebe, K., Nakayama, A., Shibata, A., and Sugiyama, Y. Dynamical model of traffic congestion and numerical simulation. *Physical review E*, 51(2):1035, 1995.

Diehl, F., Brunner, T., Le, M. T., and Knoll, A. Graph neural networks for modelling traffic participant interaction. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pp. 695–701. IEEE, 2019.

Huang, W., Braghin, F., and Arrigoni, S. Autonomous vehicle driving via deep deterministic policy gradient. In *ASME 2019 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. American Society of Mechanical Engineers Digital Collection, 2019.

Jiang, J., Dun, C., and Lu, Z. Graph convolutional reinforcement learning for multi-agent cooperation. *arXiv preprint arXiv:1810.09202*, 2(3), 2018.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Lowe, R., Wu, Y. I., Tamar, A., Harb, J., Abbeel, O. P., and Mordatch, I. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in neural information processing systems*, pp. 6379–6390, 2017.

Luo, Y., Yang, G., Xu, M., Qin, Z., and Li, K. Cooperative lane-change maneuver for multiple automated vehicles on a highway. *Automotive Innovation*, 2(3):157–168, 2019.

Shalev-Shwartz, S., Shammah, S., and Shashua, A. Safe, multi-agent, reinforcement learning for autonomous driving. *arXiv preprint arXiv:1610.03295*, 2016.

Shi, T., Wang, P., Cheng, X., Chan, C.-Y., and Huang, D. Driving decision and control for autonomous lane change based on deep reinforcement learning. *arXiv preprint arXiv:1904.10171*, 2019.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*. MIT press, 2018.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Wang, P., Chan, C.-Y., and de La Fortelle, A. A reinforcement learning based approach for automated lane change maneuvers. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1379–1384. IEEE, 2018.

Wei, H., Xu, N., Zhang, H., Zheng, G., Zang, X., Chen, C., Zhang, W., Zhu, Y., Xu, K., and Li, Z. Colight: Learning network-level cooperation for traffic signal control. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pp. 1913–1922, 2019.

Wu, C., Kreidieh, A., Parvate, K., Vinitsky, E., and Bayen, A. M. Flow: Architecture and benchmarking for reinforcement learning in traffic control. *arXiv preprint arXiv:1710.05465*, 2017.

Xi, C., Shi, T., Wu, Y., and Sun, L. Efficient motion planning for automated lane change based on imitation learning and mixed-integer optimization. *arXiv preprint arXiv:1904.08784*, 2020.

Appendix

Experiment details

Some important definitions of these scenarios are:

- **Agent**: In the simulation, we mainly consider two types of agents. One is human driving agent whose acceleration or deceleration is calculated based on rule-based intelligent driver model (Bando et al., 1995), the other one is automated vehicle agent which is controlled by deep reinforcement learning framework.
- **State**: Speed and position of the ego vehicle, as well as the difference in speed and position between the ego vehicle, its leader and follower. [3]
- **Action**: Actions are a list of acceleration for each CAVs, bounded by the maximum accelerations and decelerations specified in environment parameters.
- **Reward**: The reward function encourages high average speeds from all vehicles in the network, and penalizes accelerations by the CAVs.
- **Termination**: An episode is terminated if the time horizon is reached or collision happens.

Table 2. Parameters of different scenarios

Scenarios	horizon	roll-out	total vehicles	noise	velocity limit
ring	3000	20	22	0.2	30km/h
figure eight	20	20	14	0.2	30km/h

Some important definitions of the baseline methods are:

- **Intelligent driver model (IDM)** (Bando et al., 1995): A common used adaptive cruise control method for vehicles that automatically adjusts the acceleration based on position and velocity information to maintain a safe distance from vehicles ahead. This is commonly considered as human-driven behavior (Wang et al., 2018; Shi et al., 2019).
- **Deep Deterministic Policy Gradient (DDPG)** (Lillicrap et al., 2015): DDPG is a deterministic version of model-free RL algorithm to deal with continuous action space. Automated vehicle agent can learn the optimal policy with continuous actions reliably. We construct a single agent training framework based on DDPG method, which is similar in (Huang et al., 2019) to compare with mulit-agent framework.
- **Multi-agent Deep Deterministic Policy Gradient(MADDPG)**(Lowe et al., 2017): This is a common used multi-agent framework with centralized critic and decentralized actors. However, it do not introduce graph neural network to specifically consider information from neighbors.

In this research, we run all the experiments in ALIENWARE AURORA R8 GAMING DESKTOP, with 9th Gen Intel® Core™ i7 9700. For all experiments we run 100 episodes and collect the average results of 10 random seeds. The configurations of these scenarios are shown in Table 2.

[3]Note that for all scenarios, we run 100 episodes with 10 random seeds. And for some scenarios, we found that the information from follower vehicle will make little difference on overall performance, so we didn't include these.