DGN-PPO: A multi-agent continuous control framework for mixed autonomy cooperation

Tianyu Shi[*]
Student ID: 260936236
Department of Civil Engineering
McGill University
Montreal, QC H3A 0C3,Canada
`tianyu.shi3@mail.mcgill.ca`

Jiawei Wang[†]
Student ID: 260913656
Department of Civil Engineering
McGill University
Montreal, QC H3A 0C3,Canada
`jiawei.wang4@mail.mcgill.ca`

1 Introduction

In the coming years, connected automated vehicles (CAVs) will become increasingly prevalent, providing new opportunities to improve mobility and transportation systems. In light of the introduction of automated vehicles into transportation systems, there is a new need for understanding the mixed-autonomy traffic 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 different agents' cooperation so as to maximize the total 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 moved up immediately, the following vehicle in the line will also increase its speed, which will end up a shock wave in traffic flow. Instead, if the automated vehicle learns to cooperate with other agents, it will adjust its speed steadily and try to mitigate the negative impact on the whole system and make the system become more stable.

A considerable body of literature have been investigating how to modeling driver behavior in order to improve system efficiency based on rule-based model (e.g. Intelligent Driver Model [1]). For example, some researchers [2] investigate how to model and calibrate adaptive driving behavior by calculating the appropriate acceleration or deceleration as a function of the input quantities and the driver's settings. However, as in a real-world scenario, some irrational and unforeseen behavior (e.g. suddenly overtaking) may render the aforementioned methods inefficient. There are also some researchers proposing the deep reinforcement learning framework to achieve a robust and reliable automated driving policy. Some researchers integrated a traffic simulator with Deep Reinforcement Learning library to develop reliable policy for complex multi-agent problems [3]. However, how to explore traffic domain knowledge to make the training more stable and encourage cooperation are not well explored.

In this project, we mainly focus on how to utilize the shared information for cooperation between each CAV to mitigate the shock-wave and improving the transportation efficiency. Our main contributions are:
(1) Utilize the graph attention networks in the setting of multi-agent reinforcement learning for mixed-autonomy cooperation.
(2) Integrate Proximal Policy Optimization (PPO) for continuous action generation.
(3) Build up adjacency matrix based on both velocity and position information to formalize

[*]Main contributions: Model improvement, Experiment analysis, edit video, report writing
[†]Main contributions: Model design, code analysis, report writing

This is our project report for McGill course COMP 767.
Source code can be found at:
`https://github.com/SHITIANYU-hue/COMP-767-project`
Presentation video can be found at:
`https://youtu.be/rL95dglox2c`

attention features among surrounding agents.

(4) Conduct extensive experiments based on neural network structures, percentage of CAVs, and different difficult levels of transportation networks against different baseline methods to demonstrate the effectiveness of our approach.

2 Methods

In this study, we manage to improve traffic through CAV control in a road network. It is essentially a multi-agent continuous control problem and we model the CAVs in a road network as a graph. In the following we will illustrate the detailed methods.

2.1 Proximal Policy Optimization (PPO)

Proximal Policy Optimization [4] is a model-free RL algorithm in the family of policy gradients. By introducing a new surrogate objective function based on the idea of importance sampling, PPO can solve the problems rooted in off-policy algorithm. Specifically, the PPO objective function is $J\text{ppo} = min(\frac{\pi_\theta(a_t|S_t)}{\pi_{\theta old}(a_t|S_t)}\hat{A}_t, \text{clip}(\frac{\pi_\theta(a_t|S_t)}{\pi_{\theta old}(a_t|S_t)}, 1-\epsilon, 1+\epsilon)\hat{A}_t)$, where $\hat{A}_t = r_t + V(S_{t+1}) - V(S_t)$ is estimator of advantage function and it can be trained in the similar way of DQN.

2.2 Graph Convolutional Reinforcement Learning PPO (DGN-PPO)

Graph Convolutional Reinforcement Learning (DGN) [5] is one of the state-of-the art multi-agent reinforcement learning algorithms. DGN is DQN-based algorithm and the major idea lies in utilizing graph convolutional layer to estimate Q-value for each agents in the system. While the original DGN solves multi-agent reinforcement learning (MARL) problem based on the value function estimation, thus it is only suitable for discrete action space. In this report, we try to modify this algorithm to solve continuous action space using PPO, which is named as DGN-PPO.

In DGN-PPO, the critic is a graph convolutional neural network as original DGN, the output of critic will be used for advantage estimation to train the actor. Specifically, we utilize centralized critic with decentralized actor framework as many state-of-the art MARL methods[6, 7, 8].

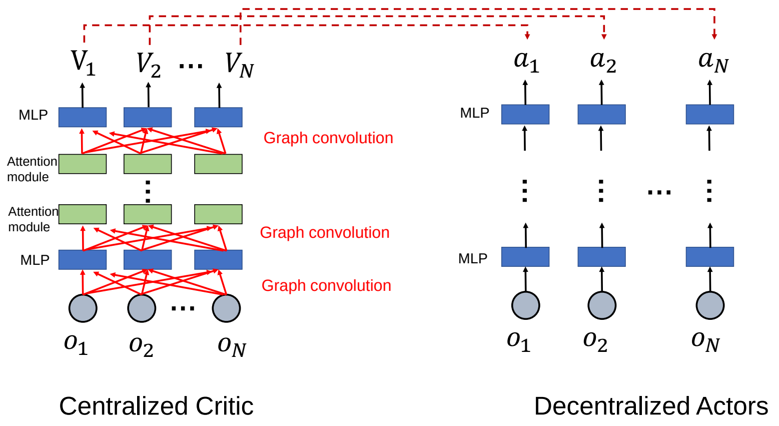

Figure 1: Framework of DGN-PPO.

As shown in Fig. 1, we denote $o_i, i = 1, 2, ..., N$ as the local observation for N agents in the system and S as the global information (i.e., concatenation of local observation). As for the critic, we adopt centralized estimator to estimate the state value $v_i, i = 1, 2, ..., N$ for each agent. Like [5], we use adjacency matrix $M_i \in \mathbb{R}^{Nb \times N}$ to characterize first Nb neighbors of agent i, through graph convolution for each agent, the global information can be extracted in view of different agents based on their adjacency matrices (i.e., $H_i = M_i H^- W$, W is the model parameters to learn and H^- is global information from last layer). In addition, the information after convolution is passed to an attention module to further highlight the local feature (i.e., agent-wise). Notably, instead of only considering spatially near neighbors like previous research, we take both the position and velocity influence into account and build up the corresponding adjacency matrix based on a relation index m_i, if index is smaller, the priority for considering influence will be higher, which is defined as:

$$m_i = (x_i - x_0) + \frac{v_t}{v_0(|v_i - v_0|) + \epsilon} \tag{1}$$

In (1), x_i and v_i are the position and velocity of the surrounding i agent, x_0 and v_0 are the position and velocity of ego agent, ϵ is a small positive number to keep it from dividing by 0. Intuitively, if there is two vehicles running slowly on road far away from each other, their relation should be thin, so the index will be bigger. On the contrary, if a vehicle is running fast along with a slow vehicle and they are very closed, then they are more likely to be affected by each other, either because the safe or efficiency consideration, therefore, the index will be smaller, and priority will be higher.

In this study, the critic network is utilized for state value estimation. And the update gradient for centrealized critic is based on Temporal-Difference learning, which can be formulated as:

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

As the action $a_i, i = 1, 2..., N$ is sampled from independent policy learned by each actor. Their policies $\pi_i(a_i|s_i)$ can be modelled as distribution (i.e., normal distribution for continuous control) and parameterized through deep neural network with parameters θ_i, respectively. Then the police gradient is based on the advantage $\hat{A}_i$ derived from centralized critic:

$$\triangledown_{\theta_i} J(\theta_i) = \triangledown_{\theta_i} \mathbb{E}_{\pi_{\theta_i^{old}}}\left[\min\left(\frac{\pi_{\theta_i}}{\pi_{\theta_i^{old}}}\hat{A}_i, \text{clip}\left(\frac{\pi_\theta}{\pi_{\theta_i^{old}}}, 1-\epsilon, 1+\epsilon\right)\hat{A}_i\right)\right] \tag{3}$$

Specifically, we perform roll-out to collect the experience and estimate the advantage, thus advantage estimation for agent i at step T is calculated as: Where $\hat{A}_i^t = \sum_t^T \gamma^t r_i^t - \hat{V}_i(S_t)$,. The training algorithm is given as follows:

Algorithm 1 Training algorithm for CAVs control based on traffic simulation

```
Set time horizon $T$ steps for each simulation.
Initialize memory buffer $M = \emptyset$.
Initialize parameters $\alpha$, $\theta_i$ for centralized critic network and actor networks.
for each episode do
    for $t = 1$ to $T$ do
        Obtain state observation $o_i, i = 1, ..., N$ and global observation $S = [o_1, .., o_N]$ .
        for actor $i = 1, ..., N$ do
            Sample action $a_i$ from $\pi_\theta^{old}(o_i)$ to control autonomous vehicle.
        end for
        Obtain next state observation $(o_i')_{i=1}^N$ and global observation $S' = [o_1', .., o_N']$ as well as the
        reward signal for each autonomous vehicle $r_i$.
        $M \leftarrow M \cup (a_i, o_i, o_i', r_i)_{i=1}^N$
        if collision happens then
            Break
        end if
    end for
    Fetch experience from $M$ and calculate advantage $A_i^t$ for each agent at each step.
    Update $\alpha$ based on Eq. 2
    for actor i = 1,...,N do
        Update $\theta_i$r based on Eq. 3
    end for
    $M = \emptyset$
end for
```

3 Experiments

3.1 Settings

We conduct experiments on Flow[3], an open-source traffic simulator that supports mixed autonomy control. Three closed loop networks (as shown in Figure 2) are built to evaluate our algorithms. Some important characteristics 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 [1], 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. [4]
- **Action**: Actions are a list of acceleration for each RL vehicles, bounded by the maximum accelerations and decelerations specified in environment parameters.
- **Reward**: The reward function rewards high average speeds from all vehicles in the network, and penalizes accelerations by the RL vehicle.
- **Termination**: An episode is terminated if the time horizon is reached or if two vehicles collide into one another.

As we consider both single agent training and multi-agent training methods, so we also build a multi-agent environment for training and testing based on the aforementioned characteristics. Instead of calculating actions, state, and reward for a single agent, in these functions, the actions, state, and reward will be calculated for all the agents in the system. To do so, we create a dictionary with agent ids as keys and different parameters (actions, state, and reward) as values.

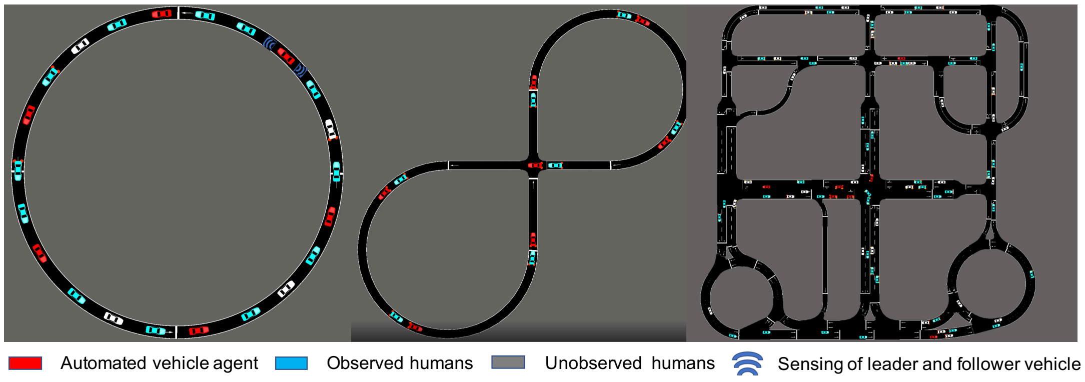

Figure 2: Various difficult levels of networks are built for evaluating our algorithms, **left:** ring network. It consists of only one single lane ring network. **middle:** figure eight network. It consists of two single lane ring network and an intersection. **right:** minicity network. It consists of multiple lane road, many roundabout and intersections.

3.2 Methods Comparison

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

- **Intelligent driver model (IDM)** [1]: 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.

[3]`https://github.com/flow-project`

[4]Note that for some scenario, we found that the information from follower vehicle will make little difference on overall performance, so we didn't include these.

Table 1: Returns within different scenarios

Model.	ring	figure eight	minicity
DGN-PPO	**3040.61**	**388.33**	**153.86**
MAPPO	2975.76	372.9	125.22
DDPG	2660.89	238.74	141.97
IDM	2525.00	52.29	144.59

- **Deep Deterministic Policy Gradient (DDPG)** [9]: DDPG is a 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 [9] to compare with mulit-agent framework.
- **Multi-agent Proximal Policy Optimization (MAPPO)**: This is a multi-agent framework , which allows the policies to use extra information to ease training. The critic is augmented with extra information from the policies of other agents.

From the Table 1, we can find that our model outperforms all baselines in three different networks from easy to hard. In the simulation[5], we found that DGN-PPO will learn to coordinate the velocity with each other, rather than blindly accelerate to achieve high speed in ring network. We also find that the DGN-PPO agent will learn to follow closely with the leader vehicle in figure eight network, so as to mitigate congestion in the intersection. We also find that the DGN-PPO agent will learn to accelerate smoothly in roundabout within minicity network in order to reduce shockwave generation.

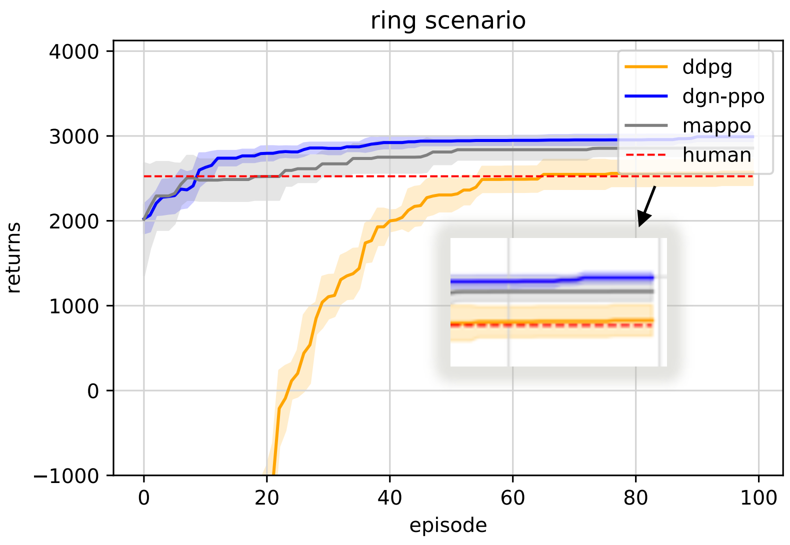

Figure 3: Training performance with 10 random seeds in ring network of 6 RL agents controlled by DGN-PPO, MAPPO, DDPG models against all human drivers controlled by IDM model.

Firstly, we evaluate the training performance in ring network. We found that DGN-PPO agent has overall best training performance against all baseline methods. Furthermore, multi-agent training methods have better initial performance than single agent method. In order to explore different environment parameters' impact, we have conducted experiments on ring network with different percentage of CAVs in the scenario. We find that DGN-PPO agent has overall best performance than all baseline methods. And multi-agent methods are better than single-agent method. We are also surprised to find that for both single and multi agent methods, highest percentage of CAVs will

[5]Simulation link: `https://youtu.be/rL95dglox2c`

Table 2: Hyper-parameter tuning for DGN-PPO model

Model config.	units of encoder layer[a]	activation function	ϵ	learning rate[b]	return
a	(128, 128)	ReLU	0.3	(1e-4,1e-4)	**2982.97**
b	(512, 128)	ReLU	0.3	(1e-4,1e-4)	2956.09
c	(128,64,128)	ReLU	0.3	(1e-4,1e-4)	2900.04
d	(128, 128)	ReLU	0.15	(1e-4,1e-4)	2972.85
e	(128, 128)	eLU	0.3	(1e-4,1e-4)	2900.04
f	(128, 128)	ReLU	0.3	(2.5e-4,1e-3)	2898.77

[a]It represents different units in each layers, e.g. (128,64,128) stands for 128, 64, 128 units in layer 1, 2,3 respectively.

[b]The units represent the learning rates for actor and the critic network respectively.

undermine the overall performance. The best percentage in ring network is 0.27, which means that a suitable proportion of CAVs and human driving vehicle can achieve the overall best performance.

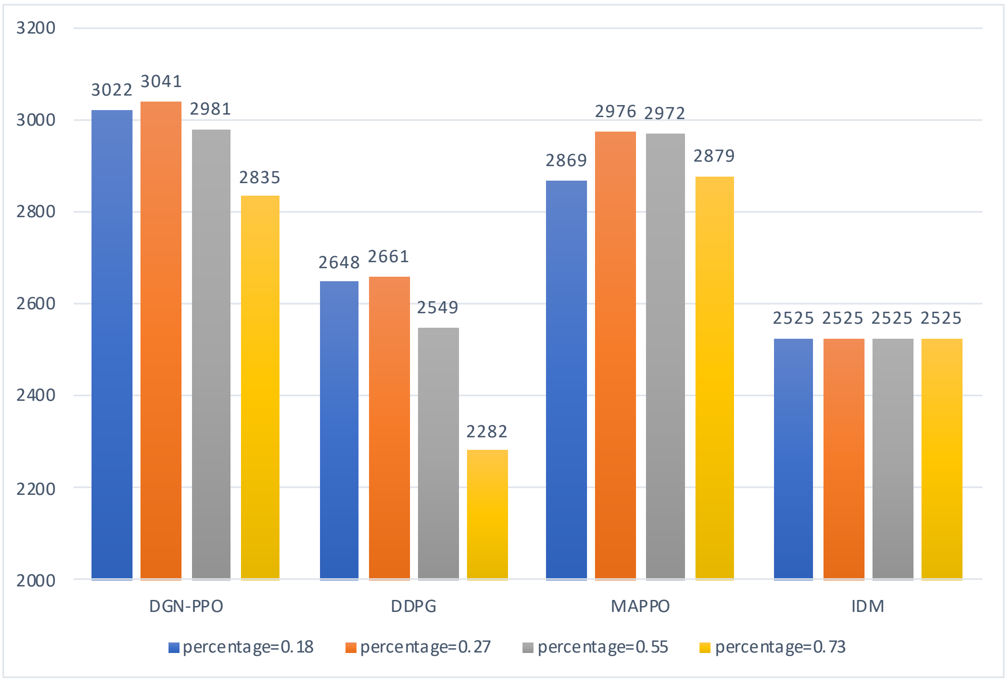

Figure 4: Evaluation of average returns with different percentage of CAVs in ring network.

3.3 Study of DGN-PPO Framework

In this section, we evaluate how different components (e.g., neighborhood definition, number of neighbors, and model structures) affect on DGN-PPO within ring network.

To begin with, we set up different model structures based on DGN-PPO architectures. The average returns are collected and compared in Table 2. We fixed the discount factor as 0.9, attention head number as 8, soft update τ as 0.1, and the optimizer chosen as Adam. We evaluate different model structures in ring network with 12 automated vehicles controlled by RL and 8 human driving vehicles. According to our experiment, the best model structure is config.a with the best return as 2982.97.

Firstly, from config. a,b and a,c we can find that change the encoder layer number or each units in encoder layer will have no improvements on model's performance. Furthermore, if we remove the attention layer in actor network, the performance will drop down, which demonstrate that the graph convolution method do encourage the cooperation between each agent. Secondly, in config.e,f,g, we

Table 3: Returns with different neighbor numbers

Neighbours number.	3	6	8	12
Return	2975.25	2980.60	2974.65	2755.37

Table 4: Returns with different information in adjacency index

adjacency index.	position	velocity	both
Returns	2890.99.	2901.87	2910.32

modified the activation function, ϵ in clipped loss and learning rate for both actor and critic networks, the results indicate that our model suites the configurations in a best.

Afterwards, we also evaluate the effects of numbers of neighbors and the definition of adjacency matrix on model performance. The results are shown in Table 3 and 4. From the results in Table 3, we can find that if we consider the neighbors number as equal as agents' number, the performance is the worst, the reason we suggest is that many redundant information will deprecate the cooperation. From the results in Table 4, we can conclude that information with both position and velocity integrate into the adjacency index formulation will boost the overall performance.

4 Conclusion

In this project, we propose a graph convolutional proximal policy optimization(DGN-PPO) approach to encourage the cooperation within 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 methods. There are also some directions for improvements. Firstly, multi-agent training is quite unstable, a small change of environment setting will result in large return shift. We will explore how to stabilize the training in the future. Moreover, in real-world problems, the number of agents in an environment can be large and diverse. Furthermore, the number of agents might change due to agents leaving or entering the environment. As in this project, we only consider the closed loop road network without inflow and outflow change, we will explore various open loop networks (e.g. bottleneck, highway) in future research.

References

[1] Masako Bando, Katsuya Hasebe, Akihiro Nakayama, Akihiro Shibata, and Yuki Sugiyama. Dynamical model of traffic congestion and numerical simulation. *Physical review E*, 51(2):1035, 1995.

[2] Arne Kesting. Microscopic modeling of human and automated driving: Towards traffic-adaptive cruise control. 2008.

[3] Cathy Wu, Aboudy Kreidieh, Kanaad Parvate, Eugene Vinitsky, and Alexandre M Bayen. Flow: Architecture and benchmarking for reinforcement learning in traffic control. *arXiv preprint arXiv:1710.05465*, 2017.

[4] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[5] Jiechuan Jiang, Chen Dun, and Zongqing Lu. Graph convolutional reinforcement learning for multi-agent cooperation. *arXiv preprint arXiv:1810.09202*, 2(3), 2018.

[6] Gregory Palmer, Karl Tuyls, Daan Bloembergen, and Rahul Savani. Lenient multi-agent deep reinforcement learning. In *Proceedings of the 17th International Conference on Autonomous*

Agents and MultiAgent Systems, pages 443–451. International Foundation for Autonomous Agents and Multiagent Systems, 2018.

[7] Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in neural information processing systems*, pages 6379–6390, 2017.

[8] Shimon Whiteson. Qmix: Monotonic value function factorisation for deep multi- agent reinforcement learning. 2018.

[9] Wenhui Huang, Francesco Braghin, and Stefano Arrigoni. 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.