

Comparison of Single- and Multi-Agent Reinforcement Learning for Highway Driving

1st Daniel Tamas Gujiczer

*Department of Control for Transportation
and Vehicle Systems*

Budapest University of Technology and Economics
Budapest, Hungary
gujiczerd@edu.bme.hu

2nd Adam Szabo

*Department of Control for Transportation
and Vehicle Systems*

Budapest University of Technology and Economics
Budapest, Hungary
szabo.adam@kjk.bme.hu

3rd Tamas Becsi

*Department of Control for Transportation
and Vehicle Systems*

Budapest University of Technology and Economics
Budapest, Hungary
becsi.tamas@kjk.bme.hu

Abstract—Single-agent reinforcement learning has been widely used to solve several traffic-related problems. These algorithms show promising results in their training environments, but their performance remains questioned when they meet other agents with the same strategy. This paper aims to show the advantages of multi-agent reinforcement learning in autonomous driving using a simple multilane highway scenario as an example. The performance of single- and multi-agent learning policies have been compared in mixed traffic. Furthermore, the effects of additional information in the observation space have also been analyzed.

Index Terms—reinforcement learning, autonomous vehicles, multi-agent reinforcement learning,

I. INTRODUCTION

Following the enormous success of Google Deepmind's AlphaGo Zero [1], [2], research in deep reinforcement learning has been rising in popularity. Since then, it has been successfully used in several other applications, such as board games [3], video games [4], and robotics [5]. Its capability to solve complex problems involving sequential decision-making tasks and the increasing availability of computational power widened the range of applications even further. As a consequence, more and more researchers utilize the power of Deep Reinforcement Learning (DRL) in the realm of road transportation [6], which includes autonomous driving [7], traffic signal control [8], energy efficient driving [9], and route optimization [10].

Although there are available end-to-end frameworks (see [11]) to overcome the challenge of autonomous driving, most researchers try to solve it by using modular systems and focus on some of its sub-tasks. Such tasks include route planning,

environmental sensing [12], local trajectory planning [13], and local feedback control.

While lower layers of the motion planning architecture are independent of other participants, strategic decisions on higher levels must consider the behavior of the other drivers. Other drivers are often modeled based on the different car following models, such as the Intelligent Driver Model (IDM) [14].

In [15], an intelligent overtaking decision-making method is presented for highway autonomous driving, which uses Q-learning. Based on the presented results, the algorithm outperforms conventional decision-making policies. The research presented in [16] proposes an attention-based architecture to handle various nearby vehicles, which is then compared with a fully-connected network and a convolutional neural network using DQN. As deep learning algorithms become increasingly complex, understanding their decision-making is crucial to deploy them successfully. Hence, in [10], a new framework is presented to analyze their behavior. The proposed method includes an episode timeline, frame-by-frame, and aggregated statistical analysis. It also includes heatmaps to provide better spatial understanding.

Single-agent algorithms learn to maximize their own rewards in these situations, which could lead to either selfish or altruistic behavior, depending on the applied rewarding strategy. While the problem is formulated as a Partially Observable Markov Decision Process, the agent must prepare to handle other self-learning vehicles that share his policy or have a different one. To prepare the agent for a wide variety of driving situations, in [17], a new simulation technique is presented to enrich existing traffic simulators with behavior-rich trajectories corresponding to varying levels of aggressiveness. Other researchers propose Multi-Agent Reinforcement Learning (MARL) to handle other self-learning agents and human drivers simultaneously in mixed traffic scenarios. A scalable MARL framework and an action masking scheme

The research was supported by the European Union within the framework of the National Laboratory for Autonomous Systems. (RRF-2.3.1-21-2022-00002)

This paper was also supported by the János Bolyai Research Scholarship of the Hungarian Academy of Sciences.

are presented in [18]. The algorithm has been validated in a highway onramp merging scenario to overcome the time-varying communication topology. In [19], a novel MARL architecture is proposed to learn multiple driving behaviors simultaneously. The algorithm utilizes the homogeneity of the agents and parameter sharing, and the results indicate faster convergence and a more generic behavior.

In this paper, the results of single- and multi-agent reinforcement learning (RL) are compared in a highway scenario. In multi-agent learning, the training was repeated with a different number of agents, while the total number of vehicles remained the same. As agents share the same goal, parameter sharing has been used to fasten the learning process. The trained agents have been deployed in highway scenarios with different traffic compositions, and their performances have been evaluated. Furthermore, the effects of different observations have been analyzed on the cooperation of the agents without direct communication.

The paper is organized as follows: Section II presents the applied algorithms and the hyperparameters used for training. Section III introduces the used simulation environment. Section V presents the simulation results and the analysis of the algorithm's performance. Section VI shows some concluding remarks.

II. METHODOLOGY

Reinforcement learning is one of the three main machine learning paradigms alongside supervised learning (SL) and unsupervised learning. Unlike SL agents, there is no need for labeled input/output pairs. RL agents learn through a continuous interaction with their environment, which is formulated as a Markov Decision Process (MDP) in the form of a tuple: $\{S, A, T, R\}$, where S contains the actual state of the environment, A is the possible set of actions of the agent, T is the state-transition function, which updates the current state depending on the selected action, and R is the reward function. By developing a policy, the agent aims to maximize its cumulative reward:

$$G = \sum_{t=1}^T \gamma^t r_t \quad (1)$$

where γ is the discount factor, which determines the importance of future rewards, and r_t is the reward at the current t time step.

At each time step, the agent receives the actual state s_i of the environment, then chooses an action a_i based on its policy π_i . The execution of the selected action causes a $P(s_i, a_i | s_{i+1})$ state transition in the environment. The quality of the selected action is encompassed in the reward, which the agent uses to find an optimal policy. The training loop of reinforcement learning is shown in Fig. 1.

A. Double Deep Q Network

Double Deep Q Network (DDQN) is an enhanced variant of the popular Deep Q Network (DQN). Both DQN and DDQN are value-based algorithms where the function approximator is

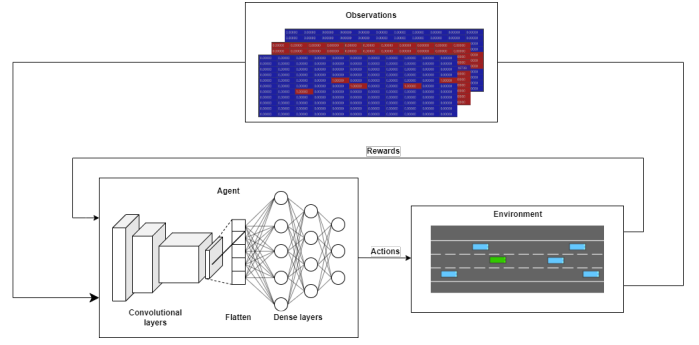


Fig. 1. Interaction between the agent and the environment in reinforcement learning

tuned to predict the action-value function. The DQN algorithm is prone to overestimate the Q -values slightly. In DDQN, the evaluation process is decoupled by using two networks. The greedy policy is evaluated according to the online network, but the target network (a periodic copy of the online network) is used to estimate its value. Thus, the original Bellman equation is slightly modified, and the update rule is written as follows:

$$Q(s_t, a_t; \theta_t) = r_{t+1} + \gamma Q(s_{t+1}, \underset{a}{\operatorname{argmax}} Q(s_{t+1}, a; \theta_t); \theta_t^-). \quad (2)$$

where $Q(s, a)$ is the action-value of the action a in state s , and s' is the new state of the environment, which is reached by performing the chosen action a in state s . θ_t is the weights of the online neural network, and θ_t^- is the weights of the target network in time step t .

B. Multi-Agent Reinforcement Learning

When multiple learning entities exist in the environment, the MDP can be generalized as a Markov Game introduced in [20]. Dealing with multiple agents simultaneously raises problems, such as the non-stationarity, shadowed equilibrium, partial observability, and credit assignment problem. [21]

Depending on the desired behavior of the agent, traffic scenarios can be viewed as competitive, cooperative, or mixed games. While selecting the rewarding strategy seems straightforward, reward shaping and credit assignment can be complicated. For instance, purely cooperative rewarding is expected to result in altruistic behavior, although it could also raise the lazy agent problem. [22]

In the case of MARL, there are three main training schemes. The first one is the centralized approach, when a single neural network handles the observations and actions of all agents. This approach has good results in the case of a small number of agents, but it has scalability issues. In the case of decentralized learning, each agent has its own policy, which is trained in parallel; hence it has an increased learning time as the agents continue to adapt to each other. The third approach is parameter sharing when the agents develop a joint policy. Parameter sharing is the least resource-intensive, and it can be a natural choice if the agents share the same goal. On

the other hand, joint performance can be further enhanced by communication. This way, agents can share their individual observations and their intentions with other agents.

III. ENVIRONMENT DESCRIPTION

A. Highway environment

The environment used for the simulations is Highway-env [23], a part of the Farama Foundation project. It provides a collection of environments (e.g., highway, intersection, roundabout) for decision-making in autonomous driving. Out of those, the highway environment was used throughout the research, which represents a multilane highway traffic situation. Fig. 2 shows the visualization of the environment, with vehicles controlled by an agent marked as green and others as blue.



Fig. 2. Visualization of the environment

The agent's task is to drive at the highest possible speed while avoiding collisions with other vehicles. In addition, changing to or staying in the rightmost lane is also considered. The reward function is composed of these features. The speed-based reward is given at a specific speed range, linearly mapped from zero to the maximum reward. It can be formulated as follows:

$$R_s = \begin{cases} \frac{v-v_{min}}{v_{max}-v_{min}} R_{s_{max}}, & \text{if } v_{min} \leq v \leq v_{max} \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

where R_s is the speed-based reward, v is the current speed of the controlled vehicle, v_{min} and v_{max} are the range of the rewarded speeds, and $R_{s_{max}}$ is the reward given at maximum speed.

The collision reward is a penalty that is only given when the agent crashes with another vehicle:

$$R_c = \begin{cases} R_{crashed} & \text{if crashed} \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

where R_c is the collision reward term and $R_{crashed}$ is the value of the penalty.

The right lane reward is also linearly mapped for the lanes the ego-vehicle drives in.

Those terms are summarized, and we get the final reward function by normalizing it from 0 to 1:

$$r = R_s + R_c + R_l \quad (5)$$

$$R = \frac{r - R_{collision}}{R_{s_{max}} + R_{l_{max}} - R_{collision}} \quad (6)$$

It is important to mention that the agents only make decisions on a higher level, which are then followed by low-level P and PD longitudinal and lateral controllers. The Kinematic Bicycle Model [24] describes the kinematics of each vehicle. Non-agent vehicles behave as follows. The longitudinal

behavior (acceleration) is described by the Intelligent Driver Model (IDM) [14], and the lateral behavior (lane changing) is represented by the Minimizing Overall Braking Induced by Lane change (MOBIL) model [25]. The observation space is one of the built-in observations in the environment, called Occupancy Grid. It is a three-dimensional array, with the first two representing the discretized space around the vehicle discretized in the form of a grid. The ego vehicle is always in the middle, and all metrics are given relative to it. The number of observed features determines the size of the third dimension. Each of the features is stored in separate channels. This observation type was chosen because its configuration does not depend on the number of vehicles in the environment, so this parameter could be randomized, approximating real-life scenarios.

As agents only make decisions about the actions to take, thus the action space consists of discrete actions, which are lane changing to the left or the right, accelerating or decelerating, and idle action, which represents no lane or speed change.

B. Modifications

The environment had to be updated, including modifying the observation and action space and the calculation of the rewards to handle multiple agents simultaneously. After the modifications, they were handled as vectors with the observations and actions of each controlled vehicle. The number of agents determines the size of these vectors, so it can also be used for single-agent training. To speed up the training process, a faster variant of the environment, called highway-fast, was used. It decreases the simulation frequency and the number of vehicles in the environment, thus the duration of the episode. Besides that, the collisions are not checked between non-agent vehicles.

IV. TRAINING

After the configuration of the environment, a DDQN agent was implemented to it. Considering that the observation of the vehicle is represented as a grid, a convolutional neural network was applied. With the observation being the input and the action the output, other hyperparameters of the agent and the network can be found in Table I.

The training was done in an environment with three lanes and twenty vehicles, of which 1 or 4 were agents. The training process, as shown in Fig. 3, lasted 35000 episodes.

During the first iteration, the observed features have been the following:

- An indicator of the presence of a vehicle in a grid cell (*presence*)
- An indicator to separate the road and surroundings (*on_road*)
- Longitudinal velocity of the vehicle in the grid cell (v_x)
- Lateral velocity of the vehicle in the grid cell (v_y)

Then, the following features have been added to the observation space:

- The cosine of the vehicle's reference heading angle ($\cos \gamma_{ref}$)
- The sine of the vehicle's reference heading angle ($\sin \gamma_{ref}$)

This update of the observation space allowed the controlled vehicles to receive information about the other agents' future intentions, not only about the actual state.

TABLE I
HYPERPARAMETERS OF THE AGENT

Parameter	Value
Learning rate	0.0001
Discount factor	0.99
Epsilon decay	0.999995
Epsilon min	0.01
Batch size	128
Burn-in	256
Sync every	20
Num. of hidden conv. layers	3
Num. of filters	32, 128, 2
Num. of hidden dense layers	3
Num. of neurons	4096, 256, 128
Activation function	ReLU
Speed reward range	[20, 30]
High speed reward	0.4
Collision reward	-1
Right lane reward	0.1

During this research, parameter sharing has been used as a training scheme for MARL with fully competitive rewarding.

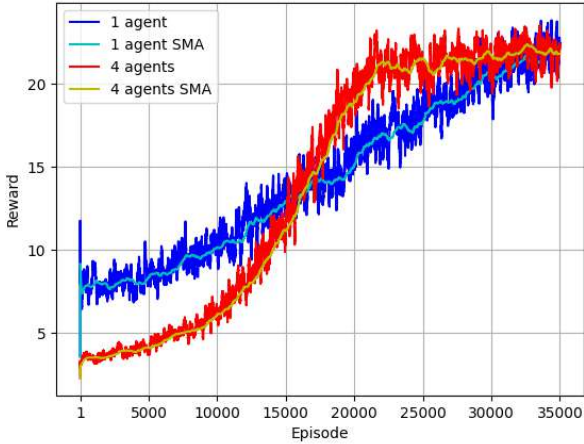


Fig. 3. Average rewards during the training process

V. SIMULATION RESULTS

During the research, a neural network controlling a single agent was trained with both observation spaces. Then, the training was repeated using MARL with four agents controlled simultaneously.

The agents have been evaluated in test batches consisting of a hundred episodes. Each batch contained tests with a different number of controlled vehicles, while the total number

of vehicles remained constant. The examined metrics are the following:

- Number of collisions per 100 episodes
- Average speeds of the agents
- Average cumulative reward

A. Effects of the number of agents

The first batch of tests was done with the original observation, and the results are summarized in Table II and Fig. 4.

TABLE II
TEST RESULTS USING THE ORIGINAL OBSERVATION SPACE

Num. of agents (evaluation)	Num. of agents (training)	Collisions	Avg. reward
1	1	5	24.56
	4	2	23.94
4	1	30	21.19
	4	14	23.39
8	1	55	18.82
	4	32	21.07
12	1	79	14.26
	4	46	19.92

Table II shows that the multi-agent policy performed better even with more agents, as collisions occurred less frequently. The more robust behavior can also be seen in the higher average rewards. The single-agent algorithm has slightly higher average rewards when no other trained agent exists in the environment, but the number of unsuccessful episodes is also higher. In every other case, the multi-agent policy is superior. It can also be noted that while the multi-agent policy has a worsening performance as the number of agents is increased during evaluation, the decay is more significant in the case of the single-agent policy. Hence, the gap between the algorithms' performances also increases. As seen in Fig. 4, the single-agent policy chooses higher speeds, which scores higher immediate rewards, but it often leads to a collision. The multi-agent policy chooses a lower reference speed, resulting in fewer collisions, thus, more consistent rewards during the episodes.

B. Effects of the observed features

Next, the agents trained with the augmented observation were tested. The results are shown in Table III and Fig. 5

TABLE III
RESULTS OF THE SECOND TRAINING BATCH

Num. of agents (evaluation)	Num. of agents (training)	Collisions	Avg. reward
1	1	7	20.91
	4	0	22.33
4	1	15	20.32
	4	2	22.33
8	1	45	17.01
	4	16	21.02
12	1	79	12.42
	4	51	17.49

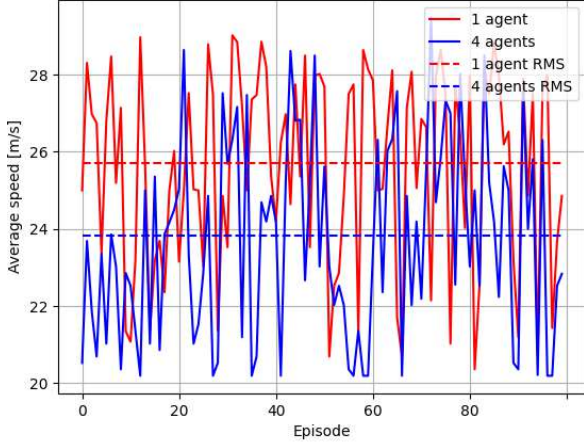


Fig. 4. Average speeds of the first agent

With the modifications of the observation space, both algorithms have better performance in the second and third test cases. They have a slightly lower average reward, but there is also a considerable improvement in the number of collisions. The advantage provided by the additional information ceases if the number of agents is increased above a threshold. On the other hand, in the first (single-agent) test case, it is interesting to see that the number of collisions did not change notably, but the average reward decreased due to the more careful policies. Both algorithms learned to use the additional information, but they need it only to predict and adapt to other agents' behavior. In the last test case, the augmented observation space does not mean a higher reward or better performance.

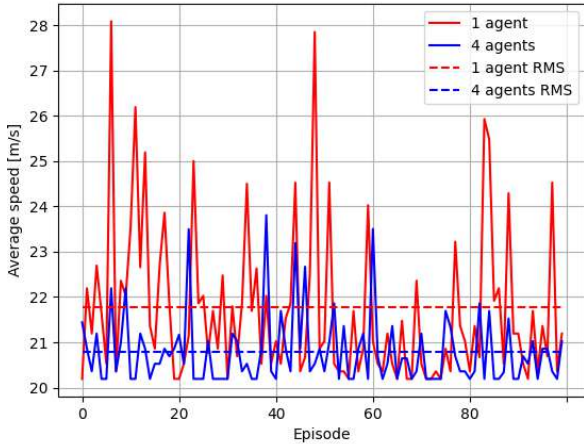


Fig. 5. Average speeds of the first agent

By comparing the number of collisions between the two training batches, it can be seen that adding the desired $\sin(d)$ and $\cos(d)$ to the observation space has a more prominent effect in the case of the multi-agent policy (e.g., in the

second test case the number of collisions decreased by 85.71% for the multi-agent policy and by 50% for the single-agent policy). While this is obviously the result of the multi-agent training setup (the behavior of other agents was observed during training), it also foreshadows the positive effects of communication.

The average speeds presented in Fig. 5 show the same tendency as in the previous batch. The single agent tends to be braver, choosing higher speeds that sometimes end in a collision, causing more fluctuating episode rewards. The previously experienced fluctuation of the rewards has decreased in both cases. However, the results are more prominent in the multi-agent policy, where most rewards are between 22 and 24.

It is important to highlight that the multi-agent policy has been trained using fully competitive rewarding. Hence, one would expect more selfish behavior from the agents. However, due to penalizing collisions and the more erratic behavior observed by the agents during training, the multi-agent algorithm learned a more robust strategy, which led to a better overall performance.

VI. CONCLUSIONS AND FUTURE WORK

This paper presents the difference between single- and multi-agent reinforcement learning in a highway decision-making scenario. The trained agents have been tested in scenarios with different numbers of agents controlled by the learned policy. The multi-agent method resulted in notably fewer collisions and reduced speeds compared to the single-agent method. In addition, the impact of the updated observation space has been analyzed. It is shown that it helped to decrease the number of crashes by stabilizing the speeds around a safe average. It also must be highlighted that the presented algorithms are still not applicable in real-life traffic scenarios, as even two collisions out of a hundred episodes is an unacceptably high value. Still, the conclusions can be drawn based on the trends of the results.

Future work will focus on a more stable initialization of the environment, as in the case of a few episodes, the collision was unavoidable due to the initial states of the vehicles. Furthermore, several other parameters, such as the number of lanes, will be randomized during training. Then, curriculum learning will be introduced to the training process, where the number of agents (and the complexity of the task) will increase as the training continues.

REFERENCES

- [1] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," 2013.
- [2] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.
- [3] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, and D. Hassabis, "Mastering chess and shogi by self-play with a general reinforcement learning algorithm," 2017.

- [4] OpenAI, :, C. Berner, G. Brockman, B. Chan, V. Cheung, P. Debiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, R. Józefowicz, S. Gray, C. Olsson, J. Pachocki, M. Petrov, H. P. d. O. Pinto, J. Raiman, T. Salimans, J. Schlatter, J. Schneider, S. Sidor, I. Sutskever, J. Tang, F. Wolski, and S. Zhang, "Dota 2 with large scale deep reinforcement learning," 2019.
- [5] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," 2019.
- [6] N. P. Farazi, T. Ahamed, L. Barua, and B. Zou, "Deep reinforcement learning and transportation research: A comprehensive review," 2020.
- [7] S. Aradi, "Survey of deep reinforcement learning for motion planning of autonomous vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 2, pp. 740–759, 2020.
- [8] K. Bálint, T. Tamás, and B. Tamás, "Deep reinforcement learning based approach for traffic signal control," *Transportation Research Procedia*, vol. 62, pp. 278–285, 2022, 24th Euro Working Group on Transportation Meeting. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2352146522001624>
- [9] Y. Wu, H. Tan, J. Peng, H. Zhang, and H. He, "Deep reinforcement learning of energy management with continuous control strategy and traffic information for a series-parallel plug-in hybrid electric bus," *Applied Energy*, vol. 247, pp. 454–466, 2019. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S030626191930652X>
- [10] I. Bello, H. Pham, Q. V. Le, M. Norouzi, and S. Bengio, "Neural combinatorial optimization with reinforcement learning," 2017.
- [11] A. O. Ly and M. Akhloufi, "Learning to drive by imitation: An overview of deep behavior cloning methods," *IEEE Transactions on Intelligent Vehicles*, vol. 6, no. 2, pp. 195–209, 2020.
- [12] M. S. Farooq, H. Khalid, A. Arooj, T. Umer, A. B. Asghar, J. Rasheed, R. M. Shubair, and A. Yahyaoui, "A conceptual multi-layer framework for the detection of nighttime pedestrian in autonomous vehicles using deep reinforcement learning," *Entropy*, vol. 25, no. 1, 2023. [Online]. Available: <https://www.mdpi.com/1099-4300/25/1/135>
- [13] M. Moghadam, A. Alizadeh, E. Tekin, and G. H. Elkaim, "A deep reinforcement learning approach for long-term short-term planning on frenet frame," in *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2021, pp. 1751–1756.
- [14] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Phys. Rev. E*, vol. 62, pp. 1805–1824, Aug 2000. [Online]. Available: <https://link.aps.org/doi/10.1103/PhysRevE.62.1805>
- [15] X. Li, X. Xu, and L. Zuo, "Reinforcement learning based overtaking decision-making for highway autonomous driving," in *2015 Sixth International Conference on Intelligent Control and Information Processing (ICICIP)*, 2015, pp. 336–342.
- [16] E. Leurent and J. Mercat, "Social attention for autonomous decision-making in dense traffic," 2019.
- [17] A. Mavrogiannis, R. Chandra, and D. Manocha, "B-gap: Behavior-rich simulation and navigation for autonomous driving," 2022.
- [18] D. Chen, M. Hajidavalloo, Z. Li, K. Chen, Y. Wang, L. Jiang, and Y. Wang, "Deep multi-agent reinforcement learning for highway on-ramp merging in mixed traffic," 2022.
- [19] M. Kaushik, P. S, and K. M. Krishna, "Parameter sharing reinforcement learning architecture for multi agent driving behaviors," 2018.
- [20] M. L. Littman, "Markov games as a framework for multi-agent reinforcement learning," in *Machine learning proceedings 1994*. Elsevier, 1994, pp. 157–163.
- [21] S. Gronauer and K. Diepold, "Multi-agent deep reinforcement learning: a survey," *Artificial Intelligence Review*, pp. 1–49, 2022.
- [22] P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, V. Zambaldi, M. Jaderberg, M. Lanctot, N. Sonnerat, J. Z. Leibo, K. Tuyls, and T. Graepel, "Value-decomposition networks for cooperative multi-agent learning," 2017.
- [23] E. Leurent, "An environment for autonomous driving decision-making," <https://github.com/eleurent/highway-env>, 2018.
- [24] P. Polack, F. Althé, B. d'Andréa Novel, and A. de La Fortelle, "The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles?" in *2017 IEEE intelligent vehicles symposium (IV)*. IEEE, 2017, pp. 812–818.
- [25] A. Kesting, M. Treiber, and D. Helbing, "Mobil: General lane-changing model for car-following models," in *Proceedings of the Transportation Research Board Annual Meeting*, 2006.