Generalized Single-Vehicle-Based Graph Reinforcement Learning for Decision-Making in Autonomous Driving

Yang, Fan; Li, Xueyuan; Liu, Qi; Li, Zirui; Gao, Xin

DOI
10.3390/s22134935

Publication date
2022

Document Version
Final published version

Published in
Sensors

Citation (APA)
Yang, F., Li, X., Liu, Q., Li, Z., & Gao, X. (2022). Generalized Single-Vehicle-Based Graph Reinforcement Learning for Decision-Making in Autonomous Driving. *Sensors*, *22*(13), Article 4935. https://doi.org/10.3390/s22134935

sensors

MDPI

Article

Generalized Single-Vehicle-Based Graph Reinforcement Learning for Decision-Making in Autonomous Driving

Fan Yang [1], Xueyuan Li [1,*], Qi Liu [1], Zirui Li [1,2] and Xin Gao [1]

[1] School of Mechanical Engineering, Beijing Institute of Technology, Beijing 100081, China; yangfanbitdb@163.com (F.Y.); 3120195257@bit.edu.cn (Q.L.); 3120195255@bit.edu.cn (Z.L.); 13403627345@163.com (X.G.)
[2] Department of Transport and Planning, Faculty of Civil Engineering and Geosciences, Delft University of Technology, Stevinweg 1, 2628 CN Delft, The Netherlands
* Correspondence: lixueyuan@bit.edu.cn

Abstract: In the autonomous driving process, the decision-making system is mainly used to provide macro-control instructions based on the information captured by the sensing system. Learning-based algorithms have apparent advantages in information processing and understanding for an increasingly complex driving environment. To incorporate the interactive information between agents in the environment into the decision-making process, this paper proposes a generalized single-vehicle-based graph neural network reinforcement learning algorithm (SGRL algorithm). The SGRL algorithm introduces graph convolution into the traditional deep neural network (DQN) algorithm, adopts the training method for a single agent, designs a more explicit incentive reward function, and significantly improves the dimension of the action space. The SGRL algorithm is compared with the traditional DQN algorithm (NGRL) and the multi-agent training algorithm (MGRL) in the highway ramp scenario. Results show that the SGRL algorithm has outstanding advantages in network convergence, decision-making effect, and training efficiency.

Keywords: autonomous driving; decision-making; graph convolution; deep reinforcement learning

Citation: Yang, F.; Li, X.; Liu, Q.; Li, Z.; Gao, X. Generalized Single-Vehicle-Based Graph Reinforcement Learning for Decision-Making in Autonomous Driving. *Sensors* **2022**, *22*, 4935. https://doi.org/10.3390/s22134935

Academic Editors: Yafei Wang, Chao Huang, Peng Hang, Zhiqiang Zuo and Bo Leng

Received: 24 May 2022
Accepted: 28 June 2022
Published: 29 June 2022

1. Introduction

In autonomous driving, the decision-making system is mainly used to produce advanced actions of vehicles, such as lane changing, acceleration, braking, and so on. Tactical decision-making for autonomous driving is challenging due to the diversity of environments, the uncertainty in the sensor information, and the complex interaction with other road users [1,2]. Traditional vehicle trajectory modeling and tracking control methods, such as genetic algorithms, neural networks, and their optimizations, have played a positive role in the research of decision-making [3].

The operational space of an autonomous vehicle (AV) can be diverse and vary significantly. Due to this, formulating a rule-based decision-maker for selecting driving maneuvers may not be ideal [4]. With the development of deep learning, the domain of reinforcement learning (RL) has become a robust learning framework now capable of learning complex policies in high-dimensional environments [5]. Therefore, using reinforcement learning to solve decision-making problems has gradually become the mainstream of research. Carl-Johan Hoel et al. introduce a method based on deep reinforcement learning for automatically generating a general-purpose decision-making function [6]. They trained an RL agent to handle a truck–trailer combination's speed and lane change decisions in a simulated environment. Hongbo Gao et al. solved sequential decision optimization problems based on the inverse reinforcement learning algorithm, and the proposed method was verified in terms of efficiency [7]. Some algorithms for planning and decision-making are based on analyzing driver behavior [8,9].

The realization of the decision-making is based on the understanding and analysis of high-dimensional environmental information. However, the traditional reinforcement learning algorithm only has a good capacity for decision-making based on low-dimension features [10]. It will have the problem of insufficient understanding in the face of more complex scenario information. The deep neural network (DNN) has a strong ability to learn representations and for the generalization of matching patterns from high-dimensional data. Therefore, deep reinforcement learning (DRL) algorithms are effective in tasks requiring feature representation and policy learning, e.g., autonomous driving decision-making [11]. Using the functional approximation ability of a deep neural network (DNN), an intelligent controller integrating artificial intelligence technologies such as deep learning (DL) and reinforcement learning (RL) is designed to maintain and avoid obstacles in lanes [12–14], decision-making [15–21], longitudinal control [22], merger maneuvers [23], human-like driving strategies [24–26], and other large-scale autonomous driving control tasks. Yingjun Ye et al. put forward a framework for decision-making training and learning. It consists of a deep reinforcement learning (DRL) training program and a high-fidelity virtual simulation environment [27]. Compared with the DQN algorithm based on a value function, the deep deterministic policy gradient (DDPG) algorithm based on an action policy can solve the continuity problem of the action space. Haifei Zhang et al. used the DDPG algorithm to solve the control problem of automatic driving based on a reasonable reward function, deep convolution network, and exploration policy [28].

Interaction between vehicles in common public transport scenarios is necessary and pervasive. However, there are relatively few studies on how autonomous vehicles interact in public environments with reinforcement learning. Realizing coordination between vehicles in a shared environment is challenging due to the unique feature of vehicular mobility, which makes it infeasible to apply the existing reinforcement learning methods directly. Chao Yu et al. proposed using a dynamic coordination graph to model the continuously changing topology during vehicles' interactions and developed two basic learning approaches to coordinate the driving maneuvers for a group of vehicles [29].

Cooperative vehicle–infrastructure systems and automatic driving technology are developing rapidly [30]. The graph neural network (GNN) has gained increasing popularity in various domains, including social network analysis [31,32]. GNN can extract the relational data representations and generate useful node embeddings on the node features and the features from neighboring nodes. The interactions between the ego vehicle and other surrounding vehicles can also be represented by the dynamic potential field (DPF) and embedded in the gap acceptance model to ensure safety and personalization during driving [33]. Jiqian Dong et al. proposed a novel deep reinforcement learning (DRL)-based approach combining the graphic convolution neural network (GNN) and deep Q network (DQN), namely the graphic convolution Q network, as the information fusion module and decision processor [34]. The proposed model can aggregate the information obtained by collaborative perception and output collaborative lane change decisions for multiple vehicles. Even in the case of highly dynamic and partially observed mixed traffic, the intention can be satisfied.

However, the above multi-agent training-based GNN reinforcement learning (MGRL) has the following problems in the actual verification process (the scenario of highway ramp exiting):

1. Multi-agent training simultaneously increases the computational network complexity, resulting in a higher overall training time cost. Therefore, each parameter modification requires a more prolonged verification time, which is not conducive to the development and adjustment of the algorithm.
2. The reward and punishment offset each other in multi-agent overall training, resulting in poor network convergence in the training process. Due to the mutual influence of multiple agents' reward values, it cannot accurately evaluate the current state, resulting in the very unstable fluctuation of the loss curve.

3. Through the test of the final training model, the final task success rate of the GCQ algorithm is maintained at around 50%, which cannot meet the basic driving needs of vehicles.
4. The GCQ decision-making model stays in the lateral lane change and cannot control the longitudinal behavior of the vehicle at the same time. This is an incomplete driving control model, which leads to a low success rate, serious collisions, and low traffic efficiency.

Given the above problems, this paper proposes an improved single-agent GNN-based RL algorithm (SGRL algorithm). This paper has the following contributions:

1. Based on retaining interactive feature extraction (GNN), the trained object is transferred from multi-agent to single-agent. Through the internal processing of the network model, the training results of single-agent can be applied to the application scenarios of multi-agent. This training method can significantly reduce the time cost of model training and eliminate the interaction of rewards and punishments between agents to improve the training effect.
2. An inductive reward function is designed to improve the convergence speed of the model. The reward function incorporates driving intention, collision, lane change frequency, and vehicle speed into the calculation. The trained model simultaneously performs well in terms of task success rate, safety, driving stability, and traffic efficiency.
3. The dimension enhancement of the action space is realized by changing the network structure. Adding vehicle longitudinal velocity control gives the model a more robust control ability for task success rate, safety, and traffic efficiency.
4. In the training process, the real-time screening of stored data can improve the training speed. The random generation mechanism of vehicles' number, type, and position in the training scenario can improve the model's generalization ability and avoid overfitting.

The paper is organized as follows. Section 2 introduces the proposed SGRL algorithm in detail. Section 3 shows the model training and testing of the SGRL algorithm and comparison algorithms (MGRL and NGRL). Section 4 shows the results of training and testing and analyzes the comparison. Finally, Section 5 derives the conclusions and proposes future improvement directions.

2. Method

The proposed SGRL methods are used to model the Markov Decision Process (MDP). The agents can explore the environment by observing states, taking actions, and receiving rewards, as shown in Figure 1.

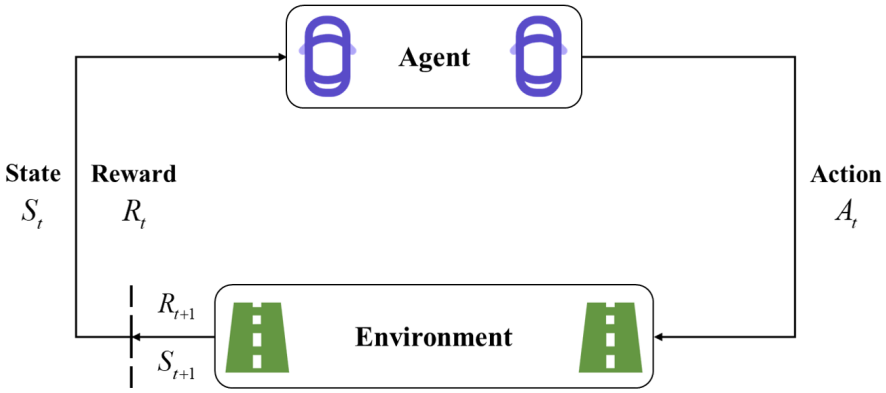

Figure 1. Markov Decision Process.

The implementation core of the SGRL method is based on the preprocessing of input data, the structure of the deep neural network, and the setting of the output end.

2.1. Network Input

In each time step in the training process, the vehicles in the scenario can be divided into human-driven vehicles (HVs) and autonomous vehicles (AVs). At each step t, the input of the training model can be divided into two parts: feature matrix X_t and correlation matrix C_t. The state $s_t = (X_t, C_t)$. Concerning the feature matrix X_t, the necessary basic information based on highway scenarios is included. The total number of human-driven vehicles (HVs) and autonomous vehicles (AVs) is set to $N = N_{HV} + N_{AV}$. Then, the specification of the matrix is $N \times 8$. The eight characteristic parameters of each vehicle describe the speed, lateral position, longitudinal position, and destination information, denoted as $x_i = (v_i, p_i, l_i, I_i)$, To serialize multiple data at the same scale, the parameters are set as follows:

- $v_i = \frac{v_{i-current}}{v_{\max}}$ is the relative speed, where $v_{\max} = \max(v_{\max-vehicle}, v_{\max-highway})$ is the maximum speed for the current vehicle.
- $p_i = \frac{x_i}{l_{highway}}$ is the relative longitudinal position normalized by the entire length of the highway. (The algorithm belongs to local planning. Its application scenarios are specific ramps and expressway sections within short distances, so the normalization of location information is more favorable for network computing.)
- l_i is the lateral lane position for the vehicle i. The position of the vehicle is represented by three-bit binary coding. For example, vehicles in the rightmost lane are denoted as $[1, 0, 0]$, the middle lane $[0, 1, 0]$ and the leftmost lane $[0, 0, 1]$.
- I_i is the destination intention feature for the vehicle i. The data type is similar to l_i. The three destinations are expressed as $[1, 0, 0]$ (merge out from the first ramp), $[0, 1, 0]$ (merge out from the second ramp) and $[0, 0, 1]$ (go straight along the highway).

Based on GNN, we can introduce a correlation matrix C_t to calculate the relationship between vehicle nodes. Considering that the sensors installed on autonomous vehicles have fixed sensing ranges, only vehicles within the sensor sensing range are recorded in the correlation matrix. C_t is a square matrix with the specification $N \times N$. The row data i of the matrix represent the relationship between vehicle i and other vehicles through binary data. The value C_{ij} of the vehicle j within the set distance of the target vehicle i will be set to 1, as shown in Figure 2.

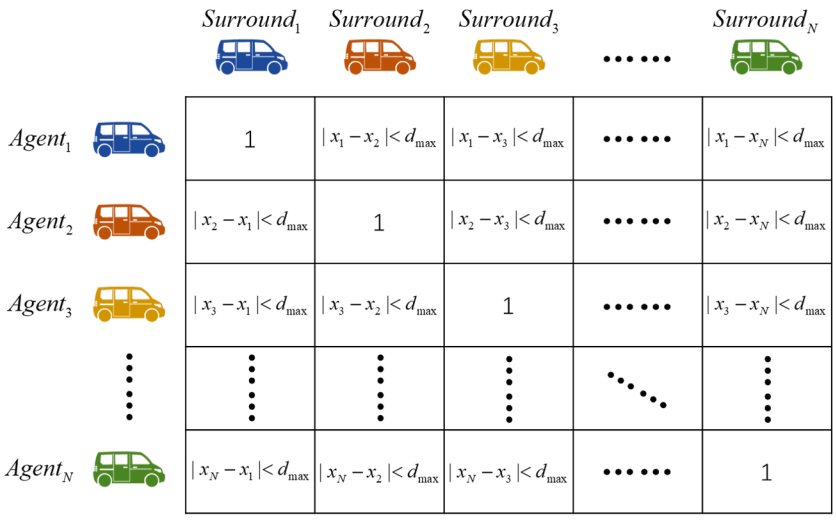

Figure 2. The schematic diagram of the correlation matrix setting. Each row in the matrix represents the correlation between a vehicle and all other vehicles, specifically the logical values of 0 and 1, where 0 represents the actual distance greater than the set distance, and 1 represents the actual distance less than the set distance.

Considering that the number of vehicles in the scene is dynamic, this situation has been considered when setting the feature matrix and the correlation matrix. The vehicles in the environment are divided into HVs and AVs, located in the feature matrix's upper and lower parts. When the number of vehicles is less than N, the positions in the matrix are occupied by 0.

To ensure the authenticity of the simulation process, all vehicles appear and disappear successively in the scenario, so the feature information and correlation information obtained at the step t cannot reach the predetermined matrix size. To ensure the standard calculation of GNN, matrix data filling is needed. Matrix segmentation prevents data confusion and error correspondence caused by random packing, as shown in Figure 3.

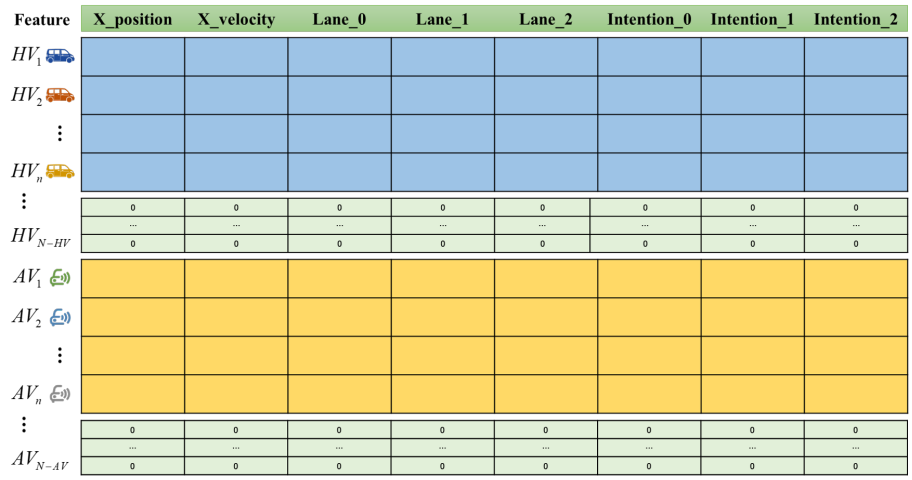

Figure 3. Matrix segmentation diagram. Each row in the matrix represents the characteristic information of a vehicle, and the matrix is divided into the upper and lower parts to separate AV and HV. The green part indicates that the vehicle is not currently in the scenario.

2.2. Network Structure

In the whole network structure, the function of graph convolution is to obtain the interaction information between vehicles. The function of the full connection layer is to parse the matrix information. Each newly obtained matrix in the network needs to be input into the full connection layer for recording and information analysis. In addition, the number of nodes in each layer also plays a decisive role in the model's performance. The optimal number of nodes is selected through comparative experiments.

Data records for comparative tests are shown in Table 1.

Table 1. Training effect for different numbers of nodes.

N of Nodes	Training Time for 1000 Episodes	Convergence Effect
32	1.5179 h	Poor (large fluctuation)
64	2.6438 h	Acceptable (occasional large fluctuations)
128	**3.4756 h**	**Good (small fluctuation)**
256	6.0987 h	Good (small fluctuation)
512	10.9542 h	Good (small fluctuation)

At each step t, the feature matrix X_t is first fed to a Fully Connected Network (FCN) encoder ψ and then the matrix $H_t \in {}^{N\times 128}$ is obtained (Equation (1)). The original eight characteristic values will be recoded to 128 values. FCN will be executed twice consecutively.

$$H_t = \psi(X_t) \in {}^{N\times 128} \quad (1)$$

Next is the calculation of graph convolution. The input of the convolution function is the newly obtained feature matrix H_t and correlation matrix C_t. The function output matrix K_t has the same specification as the original matrix H_t (Equation (2)).

$$K_t = \chi(H_t, C_t) \in \mathbb{R}^{N \times 128} \tag{2}$$

The original feature information and the new information obtained by graph convolution are fused by matrix stitching online as the total input of the subsequent neural network η.

The feature data density changes at each stage are as follows:

- Fully Connected Network (FCN): $Dense(8) \rightarrow Dense(128) \rightarrow Dense(128)$;
- Graph Neural Network (GNN): $Dense(128) \rightarrow Dense(128) \rightarrow Dense(128)$;
- Q Network: $Dense(256) \rightarrow Dense(128) \rightarrow Dense(128)$;
- Output: $Dense(128) \rightarrow Dense(33)$.

The overall structure of the SGRL algorithm is shown in Figure 4.

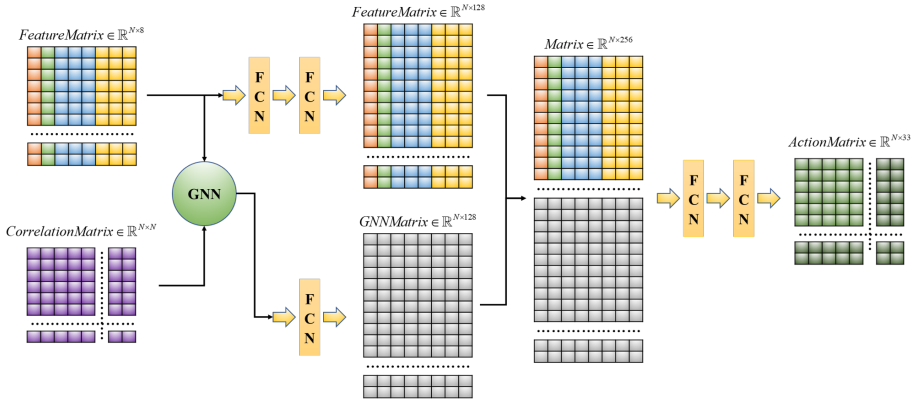

Figure 4. The overall structure diagram of the model. FCN represents the full connection layer, and GNN represents the graph neural network.

2.3. Network Output

Since the vehicle longitudinal control model built into the simulation environment is still rule-based, it cannot incorporate the interaction between vehicles into the control model. The SGRL algorithm fuses longitudinal and lateral control into the same model by increasing the output dimension, which significantly simplifies the complexity of the control process and solves the coupling problem of two directions.

The SGRL algorithm is trained based on DQN, so the output action space can only be discrete. For AVs, the longitudinal control is mainly reflected in the acceleration, and the lateral control is primarily reflected in the lane change direction. The improvement of the output action resolution is conducive to improving the control sensitivity, but it also increases the computational complexity and reduces the control frequency. In the algorithm of this paper, a compromise solution is selected. The longitudinal control is set to 11 discrete values in the interval $[-5, 5]$, and the lateral control is set to three actions: keeping, turning left and turning right. Thus, the output matrix of the model is set as $A_t \in \mathbb{R}^{N \times 33}$, as shown in Figure 5.

Action space correspondence relationship		Longitudinal Acceleration										
		-5	-4	-3	-2	-1	0	1	2	3	4	5
Lane Change Action	left	0	3	6	9	12	15	18	21	24	27	30
	keep	1	4	7	10	13	16	19	22	25	28	31
	right	2	5	8	11	14	17	20	23	26	29	32

Figure 5. Model action space diagram. The matrix row direction divides the longitudinal acceleration into discrete values, and the matrix column direction represents the lateral lane change of the vehicle.

2.4. Reward Function

The setting of the reward function needs clear guidance and a strong correlation with training objectives. In the SGRL algorithm, the task success rate, security, and traffic efficiency should be considered simultaneously. The corresponding reward values are set as intention reward, crash reward, and speed reward.

2.4.1. Intention Reward

To guide autonomous vehicles to complete driving tasks from the corresponding ramps out of the highway, the SGRL algorithm constructs reward gradients for different lanes, as shown in Figure 6. Merge_0 and merge_1, respectively, refer to the autonomous vehicles that need to drive away from ramp_0 and ramp_1 according to the task requirements. In the simulation experiment, the driving route of all autonomous vehicles is entirely determined by the reinforcement learning controller. Therefore, vehicles belonging to the merge_0 category will not necessarily exit from ramp_0; that is, they cannot complete the driving task.

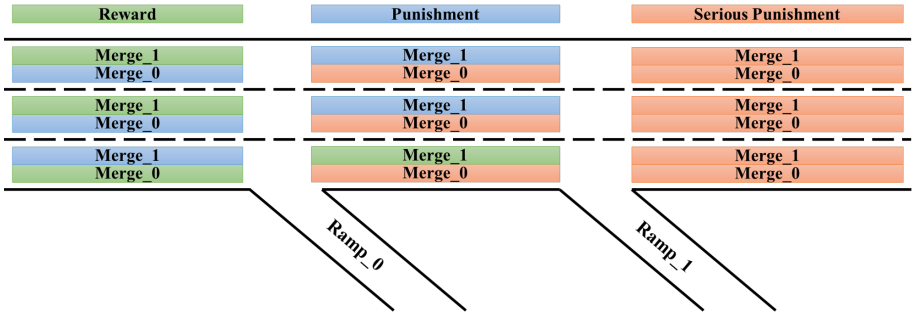

Figure 6. Intention reward gradient diagram. The task completion of autonomous vehicles is seen as a factor in judging the quality of the current reinforcement learning model, which is manifested in the reward values that can be obtained when each vehicle is in different driving sections and lanes. The strip area represents the reward types that the corresponding vehicles can obtain from the area. Green represents reward ($1 \times R_{I-Base}$), blue represents punishment ($-1 \times R_{I-Base}$), and orange represents serious punishment ($-2 \times R_{I-Base}$). R_{I-Base} is set to 1 for normalization.

To ensure practical guidance for all locations, the reward value R_{I-t} for the fixed area is set to a constant value. To ensure the interaction between various rewards and to pass it to the training process, it is necessary to pay attention to the consistency of the numerical scale when setting R_{I-t}. Moreover, the R_{I-t} needs to distinguish between positive and negative values, which is more conducive to the training. In addition, if the autonomous vehicle completes the task, it can obtain greater R_{I-t}, and if the task fails, it will also obtain a greater negative R_{I-t}.

2.4.2. Crash Reward

The safety of autonomous driving is the basis of all other characteristics. The simulation platform SUMO can detect collisions at step t and output the number $N_{collision-t}$ of vehicles involved in collisions. The collision reward is calculated according to $N_{collision-t}$ (Equation (3)).

$$R_{C-t} = -N_{collision-t}/2 \tag{3}$$

2.4.3. Speed Reward

To control the consistency of the reward scale, the speed reward value R_{S-t} needs to be normalized. $v_{\max} = \max(v_{\max-vehicle}, v_{\max-highway})$ is the max speed for the AV. To ensure the positive and negative values of R_{S-t}, the calculation is shown as Equation (4).

$$R_{S-t} = \frac{v_{i-t}}{v_{\max}} - 0.3 \tag{4}$$

2.4.4. Total Reward

The general method to calculate the total reward is a weighted summation of each component. Direct addition will lead to mutual coverage of rewards, resulting in poor training effects. The SGRL algorithm uses a new aggregation method, as shown in Equation (5).

$$R_t = \begin{cases} \omega_I \times R_{I-t} \times R_{S-t} + \omega_C \times R_{C-t} & (R_{I-t} > 0) \\ \omega_I \times R_{I-t} + \omega_C \times R_{C-t} & (R_{I-t} \leq 0) \end{cases} \tag{5}$$

This calculation method takes the task success rate and safety as the primary considerations and considers traffic efficiency simultaneously. The problem of vehicle parking for intention rewards can be completely avoided. Moreover, the weight relationship between rewards can be adjusted by parameters ω_I and ω_C. The parameter settings of this paper are $\omega_I = 1, \omega_C = 2$.

2.5. Model Training and Testing

The most prominent feature of the SGRL algorithm is training for a single agent, but the obtained model can be applied to a multi-agent environment. For trained agents, the vehicles around them can be considered the same category—surrounding vehicles. Therefore, the work done by the GNN-based reinforcement learning model can be interpreted as planning and decision-making based on the characteristics and relationships of the target agent and its surrounding vehicles. The model obtained by single-agent training can be directly transplanted to other agents in the same scenario, as shown in Figure 7.

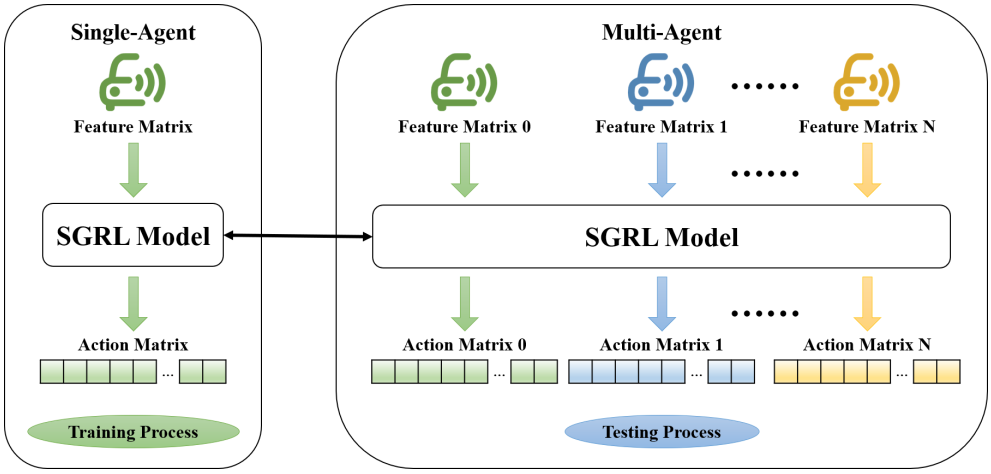

Figure 7. Reinforcement learning model transplant diagram.

To prevent the overfitting of the reinforcement learning process, we randomize the distribution of training vehicles and the task of autonomous vehicles in each episode based on fixed rules, as shown in Figure 8.

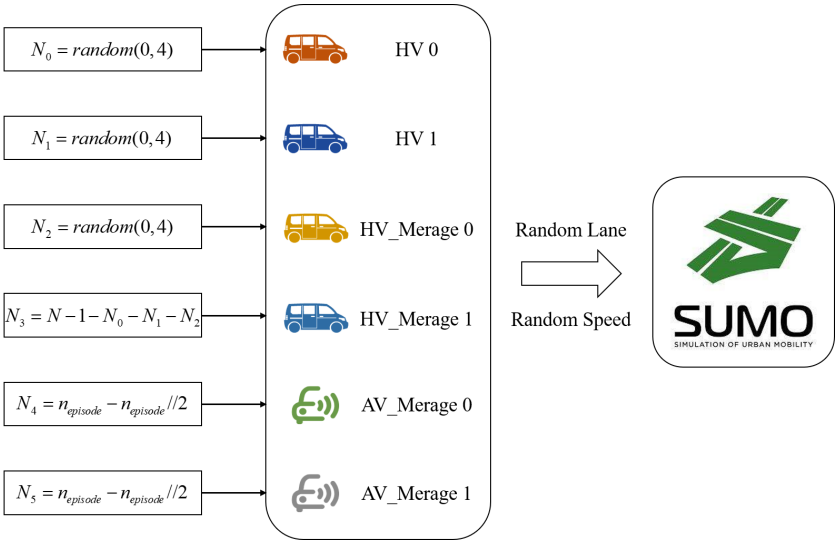

Figure 8. Scenario random setting diagram.

Algorithm 1 shows the detailed steps of training.

Algorithm 1 SGRL Q Learning Steps

Initialize the reply memory R to capacity N
Initialize the weights for the SGRL Net $(\psi, \chi, \eta \ldots\ldots)$
Jointly Current Network $\hat{Q}_\theta$ and Target Network $\hat{Q}_t = \hat{Q}_\theta$
Warming up
For step $t = 1$ to T_0(warming up steps) **do**
  Choose random action for agent i: $a_{t-r} = np.random.choice(np.arrange(3), N)$
  Get the transition (s_t, a_t, r_t, s_{t+1})
  Store in the buffer
Training step
For step $t = T_0 + 1$ to T(total steps) **do**
  With the probability e choose random action for agent i: a_{t-r}
  With the probability $1 - e$ do:
    State decoding: $(X_t, C_t) = s_t$
    Double FCN: $H_{t-1} = \psi_0(X_t) \in \mathbb{R}^{N\times 128}$; $H_{t-2} = \psi_1(H_{t-1}) \in \mathbb{R}^{N\times 128}$
    GNN + FCN: $K_t = \chi(H_{t-2}, C_t) \in \mathbb{R}^{N\times 128}$; $K_{t-1} = \psi_2(K_t) \in \mathbb{R}^{N\times 128}$
    Feature Stitching: $F_t = (H_{t-2}, K_{t-1}) \in \mathbb{R}^{N\times 256}$
    Double FCN: $F_{t-1} = \psi_3(F_t) \in \mathbb{R}^{N\times 128}$; $F_{t-2} = \psi_4(F_{t-1}) \in \mathbb{R}^{N\times 128}$
    Compute Q values: $\hat{Q}_\theta(s_t) = \psi_5(F_{t-3}) \in \mathbb{R}^{N\times 33}$
    Select $a_t^* = \arg\max \hat{Q}_\theta(s_t)$
  Execute a_t^* and get a new state s_{t+1}
  Store in the buffer
  Set $s_t = s_{t+1}$
Training model at training step
Sample a batch from the buffer and calculate the Q target:

$$Q_{target} = \begin{cases} r_t + \gamma \max_a \hat{Q}_\theta(s_t) & done = 0 \\ r_t & done = 1 \end{cases}$$

Get average *Loss*
Update parameters θ of the model $\hat{Q}_\theta$
Updating Target Net every n steps
$\hat{Q}_{target} = \hat{Q}_\theta$

Algorithm 2 shows the detailed steps of testing.

Algorithm 2 SGRL Testing Steps

Initialize the simulation environment
Testing step
For step $t = 1$ to T(total test steps) **do**
  State decoding: $(X_t, C_t) = s_t$
  For AV $i = 1$ to N_{AV} **do**
    Move other AV's features to the back of HV: $X_{t0} = \lambda(X_t) \in \mathbb{R}^{N\times 8}$
    Calculate Q based on trained model: $\hat{Q}_\theta(s_t) = \pi_{SGRL}(X_{t0}, C_t) \in \mathbb{R}^{N\times 33}$
    Select $a^*_{t-i} = \arg\max \hat{Q}_\theta(s_t)$
  Get the action matrix $a^*_t = (a^*_1, a^*_2, \ldots\ldots a^*_{N_{AV}})$
  Execute a^*_t and get a new state s_{t+1}
  Set $s_t = s_{t+1}$

3. Simulation

3.1. Baseline Models

Two baseline models are introduced for comparative analysis in the simulation: the traditional DQN algorithm (NGRL) and the multi-agent training algorithm (MGRL). In the NGRL algorithm, the GNN part is removed, and the correlation matrix is used as input by splicing with the feature matrix.

The model of the MGRL algorithm is consistent with the SGRL proposed in this paper in terms of network structure, but its training process is based on multi-agent environment interaction.

By comparing the three models, the specific effects of the GNN structure and single-agent training of SGRL can be effectively analyzed. The simulation scenario is shown in Figure 9.

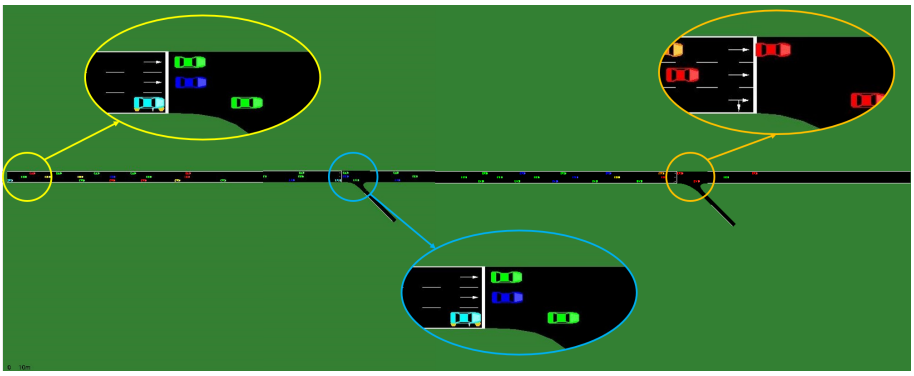

Figure 9. Simulation scenario diagram.

3.2. Simulator Parameters

The simulation scenario is a long highway with three lanes. There are exit ramps at one-third and two-thirds of the total length respectively. The speed limit for the whole road is set as 20 m/s (76 km/h) for all the vehicles. The AVs and HVs are put into the scenario at the probability of 0.1 and 0.4 from the left side of the road. And the initial speed and lane position are random.

There are six types of vehicles in the scenario, including HVs that travel straight through the highway, vehicles (HVs and AVs) that want to exit from ramp_0 and vehicles (HVs and AVs) that want to exit from ramp_1. The simulation environment controls the driving of HVs, and the AVs are completely controlled by the reinforcement learning model SGRL in real time.

The number of vehicles in the experiment is set as shown in Table 2.

Table 2. Number setting of experimental vehicles.

Algorithm Type		N of Vehicles		
		AVs		HVs
		Merge_0	Merge_1	
Training Process	SGRL	1		19
	MGRL	5	5	10
	NGRL	5	5	10
Testing Process	SGRL	5	5	10
	MGRL	5	5	10
	NGRL	5	5	10

4. Results and Discussion

4.1. Training Results

All three models were trained for 1000 episodes. The symbolic data of the training process are the reward, average Q value and loss value. Average rewards are obtained by averaging rewards against steps. The specific changes are shown in Figures 10–13.

For the comparison of reward values in the training process, under the same reward value calculation, the reward values and average reward values of the SGRL algorithm can converge faster and have better final convergence.

The changing trend of the Q value and loss value of the SGRL algorithm in the training process is consistent with the learning process. According to the loss results, SGRL has a faster convergence speed.

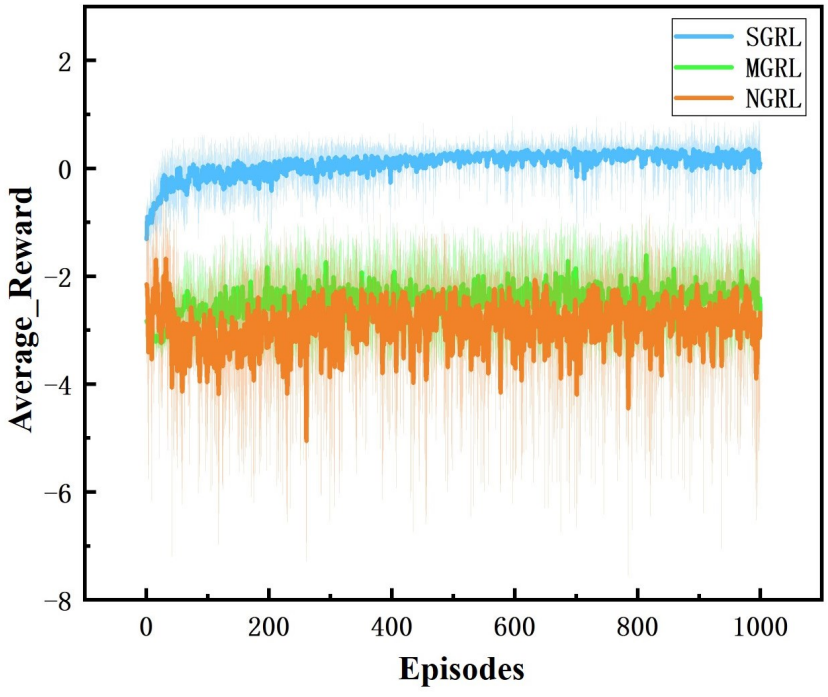

Figure 10. Diagram of average reward. This value is the average of each step reward value.

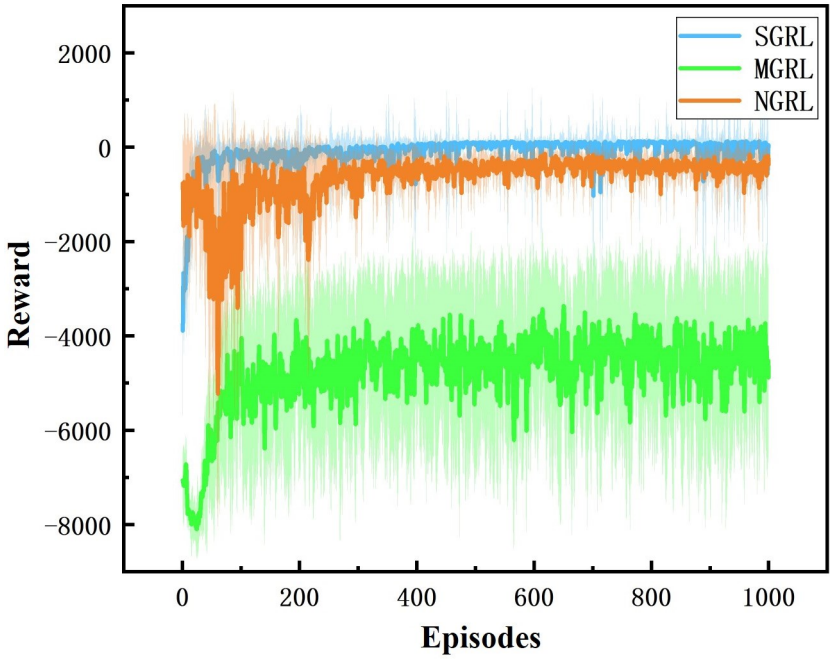

Figure 11. Diagram of reward. This value is the accumulation of each single step reward value.

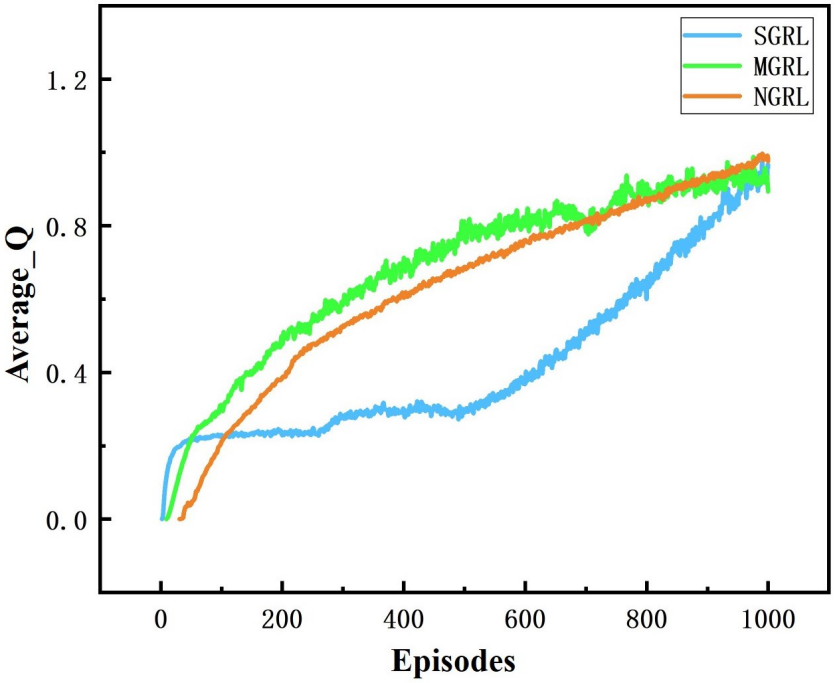

Figure 12. Diagram of average Q. This value is a training mark value in reinforcement learning.

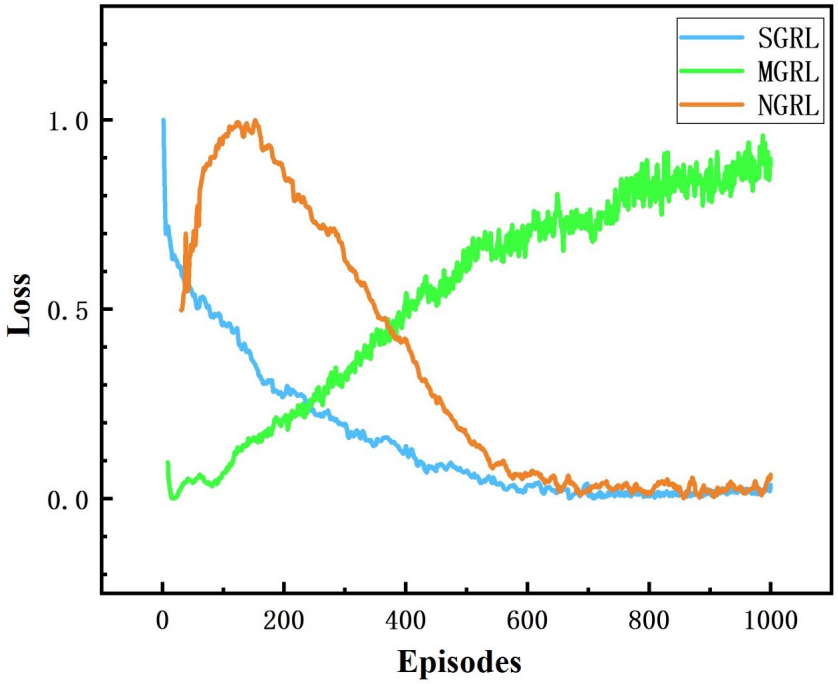

Figure 13. Diagram of loss. This value represents the difference between the real network and the ideal network.

To compare the task success rate, security, and traffic efficiency, the average velocity, number of collisions, success rate, and average steps per episode must be collected. The results of the above training data are shown in Figures 14–17.

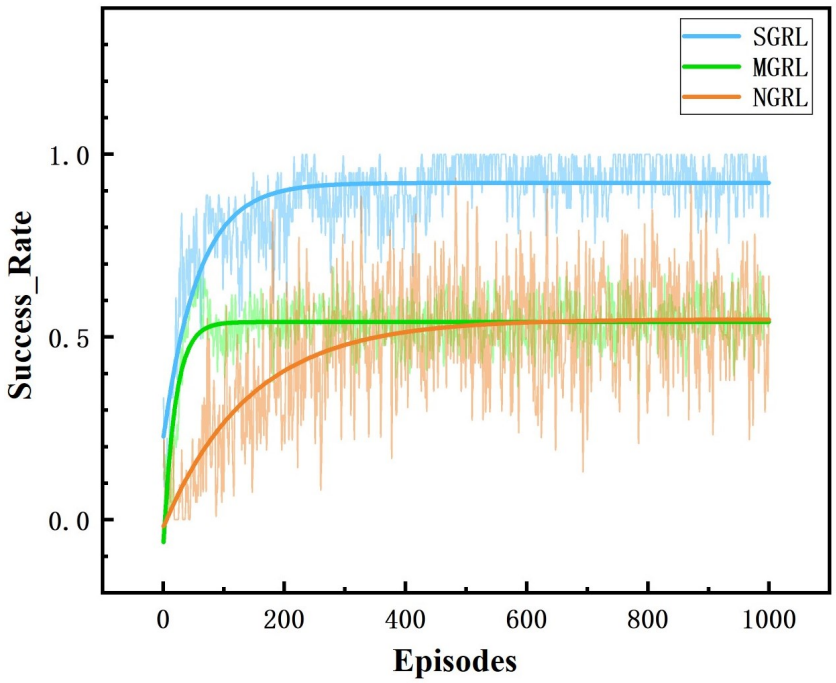

Figure 14. Diagram of success rate. This value is obtained by the ratio of the number of vehicles completing the task (entering the corresponding ramp) to the total number.

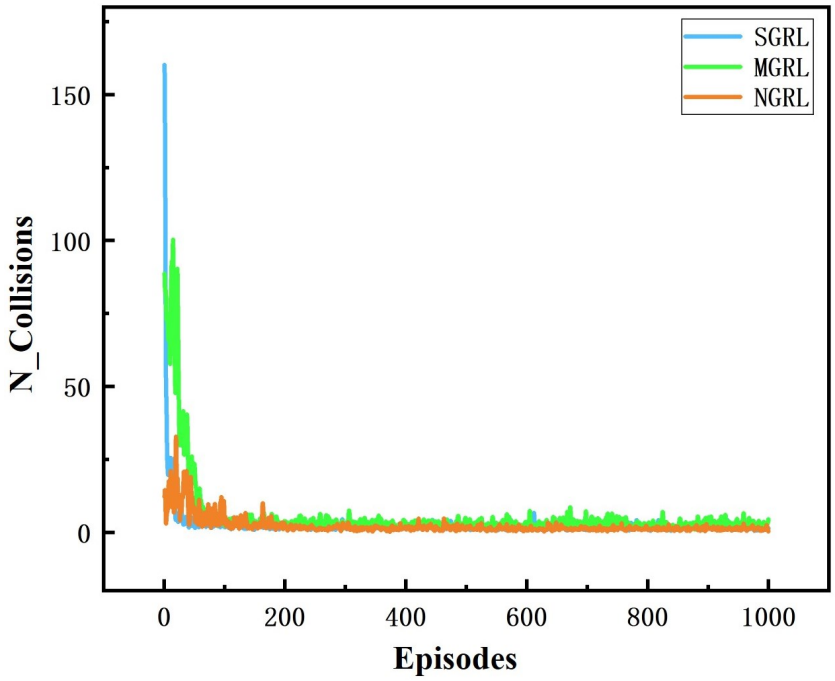

Figure 15. Diagram of collisions. This value is the number of collisions between vehicles obtained by real-time detection in the simulation scenario.

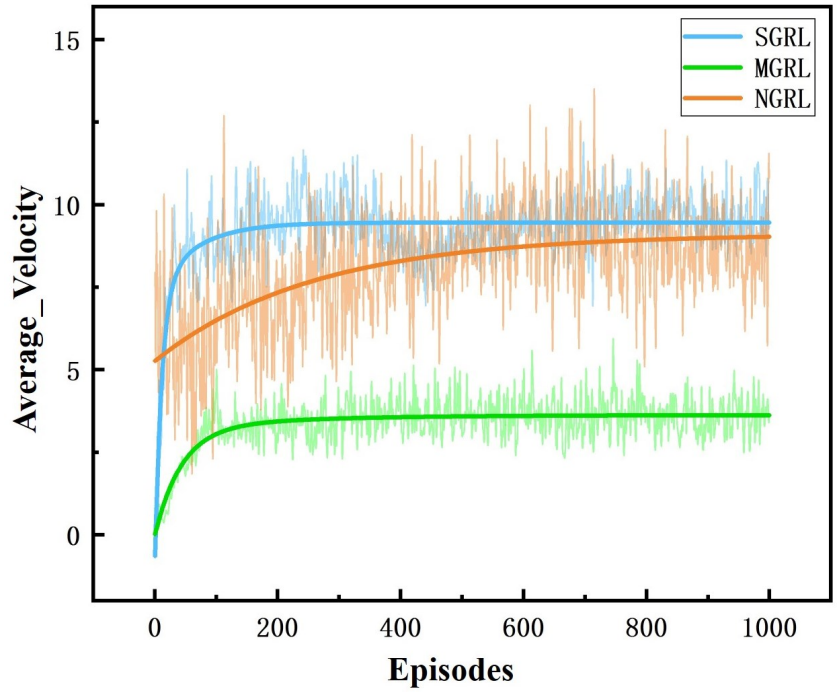

Figure 16. Diagram of average velocity. This value is the average velocity of all AVs in the scenario.

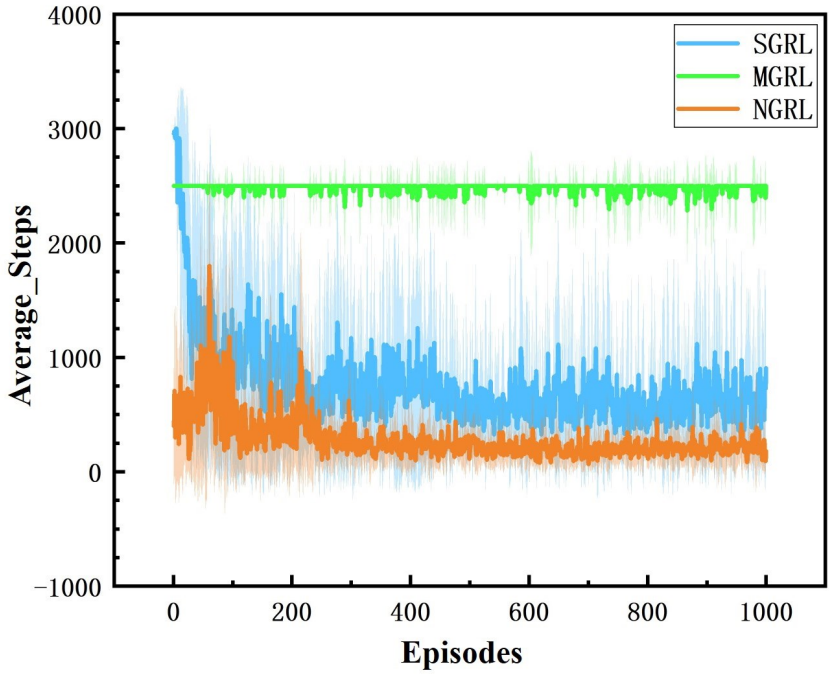

Figure 17. Diagram of average steps. This value is the number of steps experienced at the end of each episode.

The SGRL algorithm has obvious advantages regarding task success rate and average vehicle speed. In terms of collision times and average training step length, SGRL also meets driving safety requirements.

In addition, SGRL has apparent advantages in training efficiency under the premise of the same training episodes. The specific hardware parameters and training time are shown in Table 3. The comparison of the training time is shown in Table 4.

Table 3. Computer hardware information.

Item	Type
CPU	Intel I9 10980XE
GPU	NVIDIA RTX3090(24G)
RAM	Crucial DDR4 3200MHz 32G × 4
SSD	SAMSUNG 970 EVO Plus 1T × 2
OS	Ubuntu 20.04

Table 4. Training time statistics (ten experiments for each algorithm; time unit is hours).

	SGRL	MGRL	NGRL
1	3.335665	66.40787	3.251174
2	3.687935	78.94237	4.812508
3	3.782653	71.37449	3.737191
4	3.763133	82.16632	4.360045
5	3.38374	66.463	3.259155
6	3.286891	72.25948	3.890604
7	3.874574	67.38896	3.840993
8	3.043649	67.68016	3.668797
9	3.368947	65.51202	3.017189
10	3.832845	69.72951	4.072374
Mean	3.536003	70.79242	3.791003

4.2. Testing Results

The trained model needs to be verified by the test process. To fully verify the algorithm's effectiveness, we adjust the total length of the road under the premise of the same traffic flow (20 vehicles per episode). The three algorithms are tested on 1000 m, 750 m and 500 m roads to simulate different traffic flow and congestion levels.

Each test process includes 1000 episodes. In the test process, the reward value can still be used as an essential evaluation of model performance. The simulation results of reward value and average reward value are shown in Figures 18 and 19.

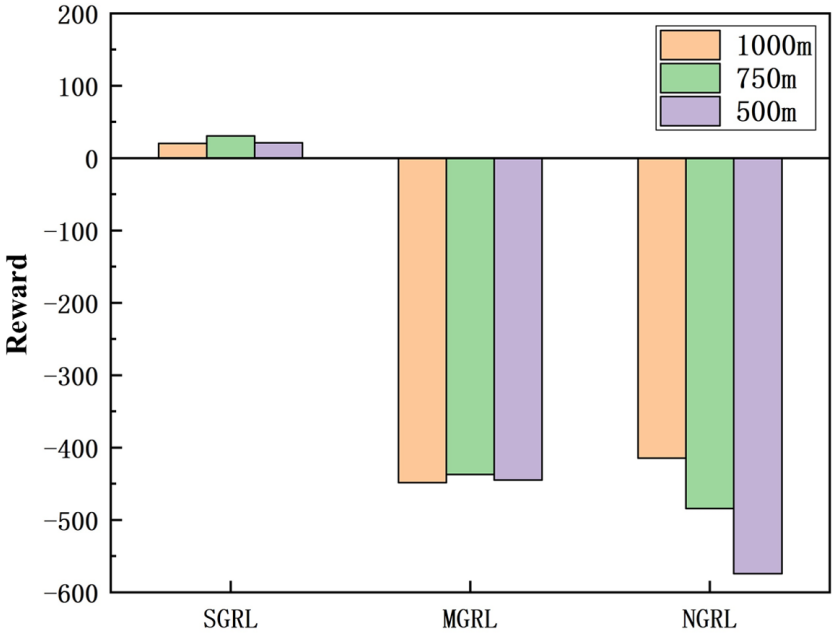

Figure 18. Diagram of testing reward. This value is the average of the total reward value for each episode in the test.

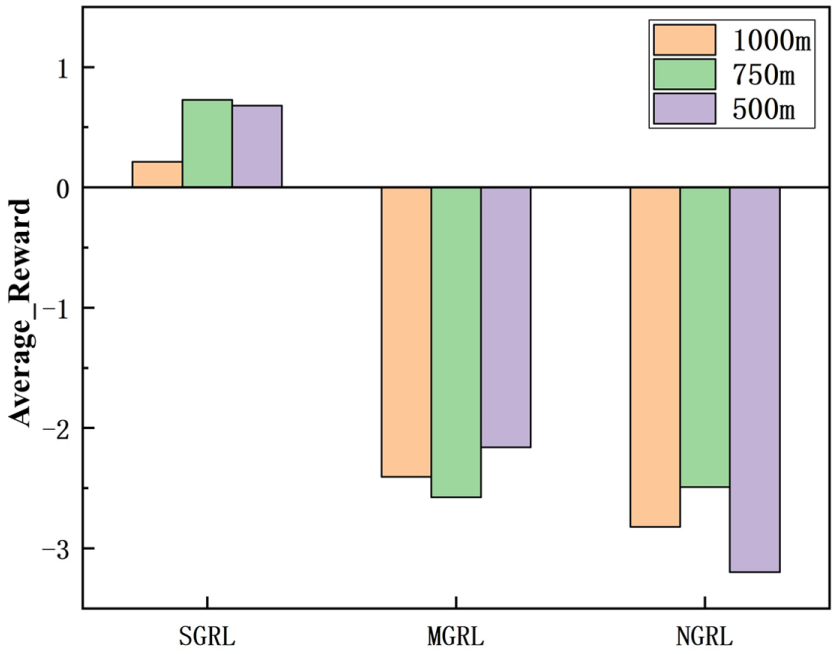

Figure 19. Diagram of testing average reward. This value is the average of the average reward value of each episode in the test.

For the most complex and congested 500 m highway scenario, the longitudinal motion spatial distribution of the three algorithms throughout the test cycle is as shown in Figure 20.

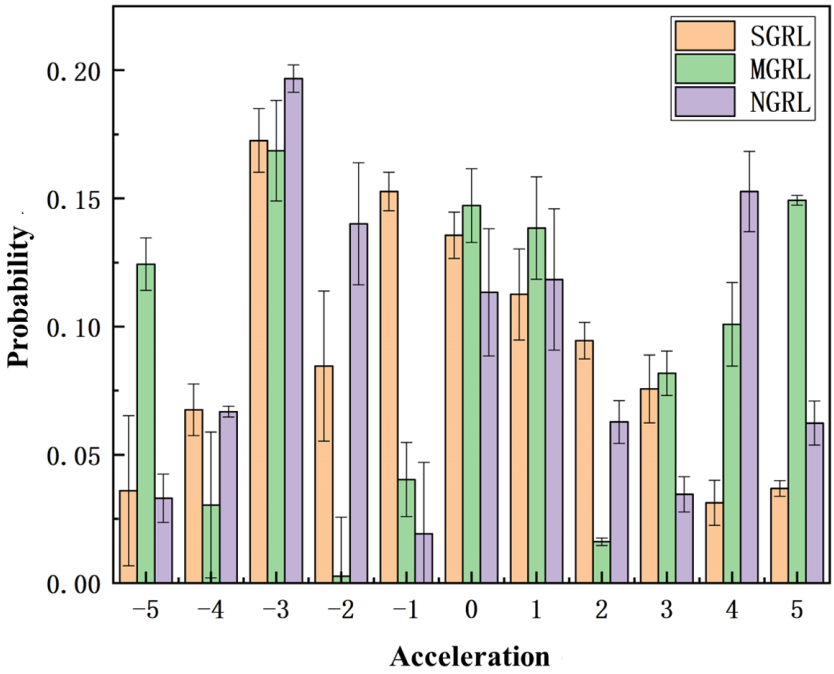

Figure 20. Spatial distribution diagram of longitudinal movement. Based on the frequency statistics of each action output in each testing process, the probability distribution of the longitudinal action can be obtained through probability calculation. The data in the figure are obtained from the average of ten repeated experiments.

The longitudinal control of autonomous vehicles will become more and more complex with the increase in road congestion, so the test results of the 500 m scene are the optimal reference. The test results show that the action output of the SGRL algorithm is mainly concentrated near 0, and the probability of large acceleration is acceptable.

To compare the task success rate, security, and traffic efficiency, the average velocity, number of collisions, success rate, and average step per episode must be collected.

The data of different methods are listed in Table 5, and the mean of the above data is shown in Figures 21–24.

Table 5. Performance comparison for different models.

Model	Road Length	Average_V	N_Collisions	Success_Rate	Average_Steps
SGRL	1000 m	**9.55588**	1.66888	**0.94068**	601.1014
	750 m	**8.45047**	**1.73714**	**0.92385**	494.9505
	500 m	**7.50548**	**2.06911**	**0.91493**	307.8586
MGRL	1000 m	3.60231	2.90205	0.54129	2478.409
	750 m	3.33545	2.96804	0.53864	1836.932
	500 m	3.20753	3.02719	0.47095	1334.704
NGRL	1000 m	8.92665	**1.23373**	0.53529	**204.3344**
	750 m	7.16045	1.83906	0.52708	**166.5145**
	500 m	6.70472	2.31418	0.4839	**111.9219**

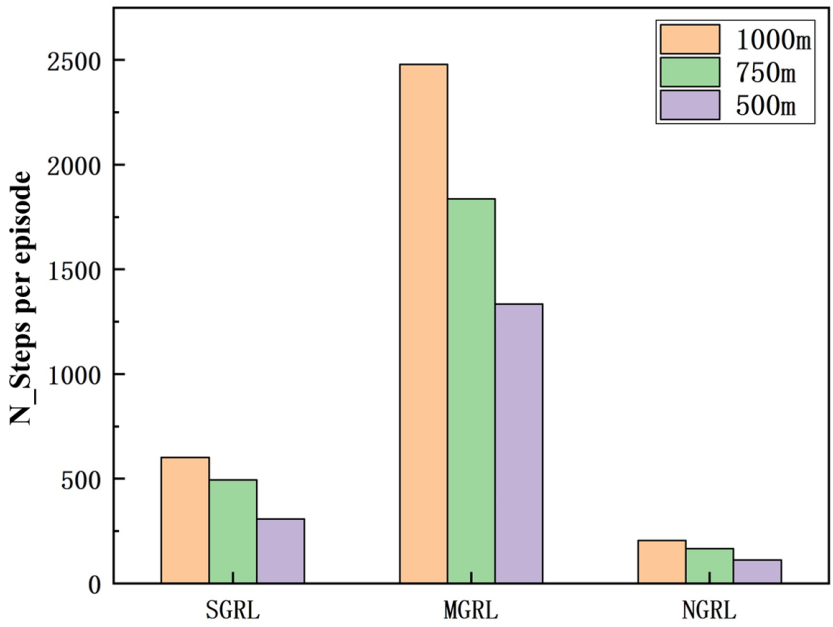

Figure 21. Diagram of testing average steps. This value is the number of steps experienced at the end of each episode.

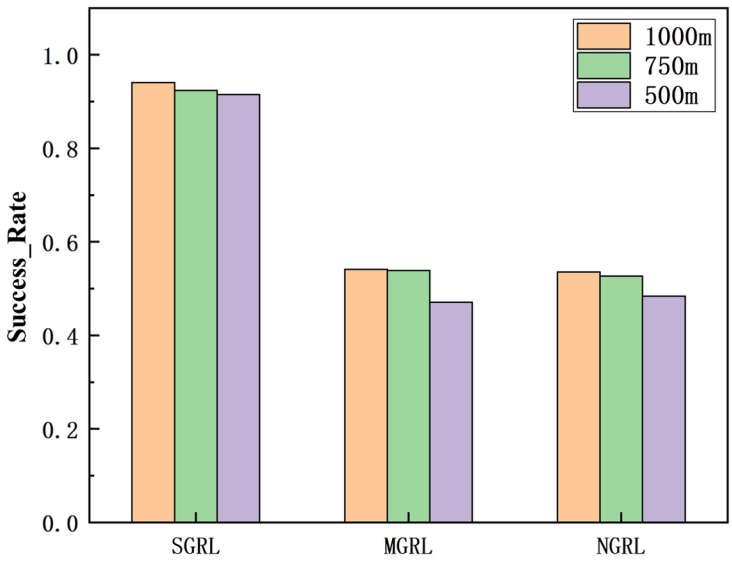

Figure 22. Diagram of testing success rate. This value is obtained by the ratio of the number of vehicles completing the task (entering the corresponding ramp) to the total number.

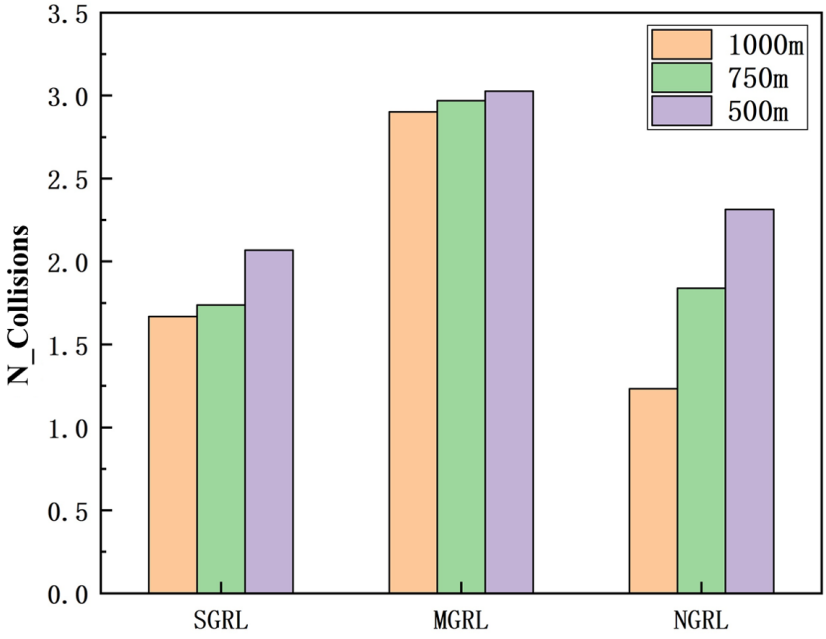

Figure 23. Diagram of testing collisions. This value is the number of collisions between vehicles obtained by real-time detection in the simulation scenario.

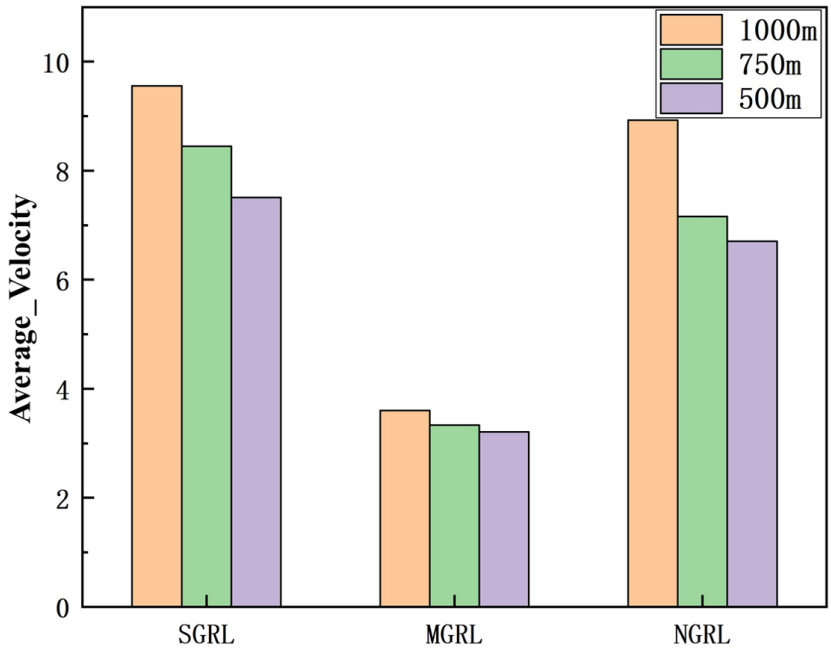

Figure 24. Diagram of testing average velocity. This value is the average velocity of all AVs in the scenario.

It can be seen from the figure and table that SGRL has apparent advantages in task success rate and average velocity. In terms of the number of collisions and the test steps, although the SGRL value is not the best, it is consistent with the best value.

4.3. Results Discussion

It can be seen that the SGRL algorithm has outstanding advantages over the MGRL algorithm and the NGRL algorithm. The SGRL algorithm can converge faster during the training process and achieve better data performance. In the testing process, the SGRL algorithm has outstanding data performance in terms of task success rate, average vehicle speed, and security.

The MGRL algorithm directly sums the reward value of all autonomous vehicles in the process of reward value calculation, so there is a phenomenon in which the excellent performance of vehicle behavior and the poor performance of vehicle behavior offset each other, which is not conducive to the adequate updating of parameters, and also causes the final convergence speed to be slow, and thus the convergence effect is not good.

The NGRL algorithm lacks the calculation consideration of the interaction process between vehicle individuals. Therefore, in the decision-making process, due to the relatively simple understanding of the environment, it cannot obtain sufficient data support, so the data performance is poor.

The SGRL algorithm considers and solves the above problems and optimizes the network structure. According to the comparison of data, it can be verified that the improvement of SGRL is obviously effective.

5. Conclusions

This paper proposes a generalized single-vehicle-based graph neural network reinforcement learning algorithm (SGRL algorithm). This algorithm combines GNN with deep reinforcement learning to solve the vehicle planning problem in the scenario of highway driving out of the ramp. The SGRL model is trained for a single agent and can be tested in multi-agent scenarios. At the same time, the algorithm sets up an improved reward function to provide a clear direction for training.

Comparing the three algorithms, the conclusions are as follows:

- Firstly, a training mode for single-agent training extended to multi-agent scenarios is proposed and verified in terms of training effectiveness and performance. The algorithm improves the analytical ability of the DRL by increasing the number of network nodes, thereby increasing the control dimension of vehicle decision-making to longitudinal and lateral dimensions.
- Secondly, the proposed SGRL algorithm simplifies the training mode and improves the training efficiency without affecting the training effect. This helps to adjust complex parameters and reduce time costs.
- Thirdly, SGRL is more sufficient in the training process to achieve a better convergence effect. The fluctuation is the smallest after the training data are stable. This shows that the proposed SGRL algorithm has outstanding training ability and is more suitable for decision-making based on reinforcement learning in multi-agent scenarios.
- Finally, the newly designed reward function effectively solves the problem of mutual influence between longitudinal and lateral control. SGRL can achieve higher task success rates and average velocity in the training and testing process. This shows that the new reward function, the training method for a single agent, and the incorporation of GNN effectively improve the decision performance of the model.

In future research, the continuity of model action space can be added to this algorithm, which will effectively improve the driving fluency of vehicles. In addition, the relationship between multiple agents in the scenario should not be limited to physical characteristics: decision-making, driving intention, and task priority can also be incorporated into the calculation process.

Author Contributions: Conceptualization, F.Y. and Q.L.; Data curation, F.Y.; Methodology, F.Y. and Z.L.; Project administration, X.L.; Resources, X.L.; Software, F.Y., Q.L. and X.G.; Supervision, X.L.; Validation, F.Y.; Visualization, F.Y.; Writing—original draft, F.Y.; Writing—review & editing, X.L. and Q.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Hoel, C.J.; Driggs-Campbell, K.; Wolff, K.; Laine, L.; Kochenderfer, M.J. Combining Planning and Deep Reinforcement Learning in Tactical Decision Making for Autonomous Driving. *IEEE Trans. Intell. Veh.* **2020**, *5*, 294–305. [CrossRef]
2. Liu, Q.; Li, Z.; Yuan, S.; Zhu, Y.; Li, X. Review on Vehicle Detection Technology for Unmanned Ground Vehicles. *Sensors* **2021**, *21*, 1354. [CrossRef] [PubMed]
3. Peng, T.; Su, L.; Zhang, R.; Guan, Z.; Zhao, H.; Qiu, Z.; Zong, C.; Xu, H. A new safe lane-change trajectory model and collision avoidance control method for automatic driving vehicles. *Expert Syst. Appl.* **2020**, *141*, 112953. [CrossRef]
4. Nageshrao, S.; Tseng, H.E.; Filev, D. Autonomous highway driving using deep reinforcement learning. In Proceedings of the 2019 IEEE International Conference on Systems, Man and Cybernetics (SMC), Bari, Italy, 6–9 October 2019; pp. 2326–2331.
5. Kiran, B.R.; Sobh, I.; Talpaert, V.; Mannion, P.; Sallab, A.A.A.; Yogamani, S.; Perez, P. Deep Reinforcement Learning for Autonomous Driving: A Survey. *IEEE Trans. Intell. Transp. Syst.* **2021**, *23*, 4909–4926. [CrossRef]
6. Hoel, C.J.; Wolff, K.; Laine, L. Automated speed and lane change decision making using deep reinforcement learning. In Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018; pp. 2148–2155.
7. Gao, H.; Shi, G.; Xie, G.; Cheng, B. Car-following method based on inverse reinforcement learning for autonomous vehicle decision-making. *Int. J. Adv. Robot. Syst.* **2018**, *15*. [CrossRef]
8. Li, Z.; Gong, J.; Lu, C.; Li, J. Personalized Driver Braking Behavior Modeling in the Car-Following Scenario: An Importance-Weight-Based Transfer Learning Approach. *IEEE Trans. Ind. Electron.* **2022**, *69*, 10704–10714. [CrossRef]
9. Lu, C.; Hu, F.; Cao, D.; Gong, J.; Xing, Y.; Li, Z. Transfer learning for driver model adaptation in lane-changing scenarios using manifold alignment. *IEEE Trans. Intell. Transp. Syst.* **2019**, *21*, 3281–3293. [CrossRef]
10. Zhao, D.B.; Shao, K.; Zhu, Y.H.; Li, D.; Wang, C.H.J.C.T. Review of deep reinforcement learning and discussions on the development of computer Go. *Control Theory Appl.* **2016**, *33*, 701–717.
11. Wang, J.; Zhang, Q.; Zhao, D.; Chen, Y. Lane Change Decision-making through Deep Reinforcement Learning with Rule-based Constraints. In Proceedings of the 2019 International Joint Conference on Neural Networks (IJCNN), Budapest, Hungary, 14–19 July 2019; pp. 1–6. [CrossRef]
12. Li, Y.; Chen, S.; Ha, P.; Dong, J.; Steinfeld, A.; Labi, S. Leveraging Vehicle Connectivity and Autonomy to Stabilize Flow in Mixed Traffic Conditions: Accounting for Human-driven Vehicle Driver Behavioral Heterogeneity and Perception-reaction Time Delay. *arXiv* **2020**, arXiv:2008.04351.
13. Gong, C.; Li, Z.; Lu, C.; Gong, J.; Hu, F. A comparative study on transferable driver behavior learning methods in the lane-changing scenario. In Proceedings of the 2019 IEEE Intelligent Transportation Systems Conference (ITSC), Auckland, New Zealand, 27–30 October 2019; pp. 3999–4005.
14. Sallab, A.; Abdou, M.; Perot, E.; Yogamani, S.J.E.I. Deep Reinforcement Learning framework for Autonomous Driving. *Electron. Imaging* **2017**, *2017*, 70–76. [CrossRef]
15. Noh, S. Decision-Making Framework for Autonomous Driving at Road Intersections: Safeguarding Against Collision, Overly Conservative Behavior, and Violation Vehicles. *IEEE Trans. Ind. Electron.* **2019**, *66*, 3275–3286. [CrossRef]
16. Liu, Q.; Li, X.; Yuan, S.; Li, Z. Decision-Making Technology for Autonomous Vehicles: Learning-Based Methods, Applications and Future Outlook. In Proceedings of the 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), Indianapolis, IN, USA, 19–22 September 2021; pp. 30–37. [CrossRef]
17. Schwarting, W.; Alonso-Mora, J.; Rus, D. Planning and Decision-Making for Autonomous Vehicles. *Annu. Rev. Control. Robot. Auton. Syst.* **2018**, *1*, 187–210. [CrossRef]
18. Li, L.; Ota, K.; Dong, M. Humanlike Driving: Empirical Decision-Making System for Autonomous Vehicles. *IEEE Trans. Veh. Technol.* **2018**, *67*, 6814–6823. [CrossRef]
19. Xu, X.; Zuo, L.; Li, X.; Qian, L.; Ren, J.; Sun, Z. A Reinforcement Learning Approach to Autonomous Decision Making of Intelligent Vehicles on Highways. *IEEE Trans. Syst. Man Cybern. Syst.* **2019**, *50*, 3884–3897. [CrossRef]
20. Zhang, Z.; Jiang, Q.; Wang, R.; Song, L.; Zhang, Z.; Wei, Y.; Mei, T.; Yu, B. Research on Management System of Automatic Driver Decision-Making Knowledge Base for Unmanned Vehicle. *Int. J. Pattern Recognit. Artif. Intell.* **2019**, *33*, 1959013. [CrossRef]
21. Duan, J.; Li, S.E.; Guan, Y.; Sun, Q.; Cheng, B.J.I.I.T.S. Hierarchical reinforcement learning for self-driving decision-making without reliance on labelled driving data. *IET Intell. Transp. Syst.* **2020**, *14*, 297–305. [CrossRef]
22. Cheng, X.; Jiang, R.; Chen, R. Simulation of decision-making method for vehicle longitudinal automatic driving based on deep Q neural network. In Proceedings of the 2020 the 7th International Conference on Automation and Logistics (ICAL), Beijing, China, 22–24 July 2020; pp. 12–17.
23. Wang, P.; Chan, C.; Fortelle, A.d.L. A Reinforcement Learning Based Approach for Automated Lane Change Maneuvers. In Proceedings of the 2018 IEEE Intelligent Vehicles Symposium (IV), Changshu, China, 26–30 June 2018; pp. 1379–1384. [CrossRef]
24. Forster, Y.; Hergeth, S.; Naujoks, F.; Beggiato, M.; Krems, J.F.; Keinath, A. Learning to use automation: Behavioral changes in interaction with automated driving systems. *Transp. Res. Part F Traffic Psychol. Behav.* **2019**, *62*, 599–614. [CrossRef]
25. Biondi, F.; Alvarez, I.; Jeong, K.A. Human–Vehicle Cooperation in Automated Driving: A Multidisciplinary Review and Appraisal. *Int. J. Hum. Comput. Interact.* **2019**, *35*, 932–946. [CrossRef]
26. Li, Z.; Gong, C.; Lu, C.; Gong, J.; Lu, J.; Xu, Y.; Hu, F. Transferable driver behavior learning via distribution adaption in the lane change scenario. In Proceedings of the 2019 IEEE Intelligent Vehicles Symposium (IV), Paris, France, 9–12 June 2019; pp. 193–200.

27. Ye, Y.; Zhang, X.; Sun, J.J.T.R.P.C.E.T. Automated vehicle's behavior decision making using deep reinforcement learning and high-fidelity simulation environment. *Transp. Res. Part C Emerg. Technol.* **2019**, *107*, 155–170. [CrossRef]
28. Zhang, H.; Xu, J.; Qiu, J.; Bashir, A.K. An Automatic Driving Control Method Based on Deep Deterministic Policy Gradient. *Wireless Commun. Mob. Comput.* **2022**, *2022*, 7739440. [CrossRef]
29. Yu, C.; Wang, X.; Xu, X.; Zhang, M.; Ge, H.; Ren, J.; Sun, L.; Chen, B.; Tan, G. Distributed Multiagent Coordinated Learning for Autonomous Driving in Highways Based on Dynamic Coordination Graphs. *IEEE Trans. Intell. Transp. Syst.* **2020**, *21*, 735–748. [CrossRef]
30. Yuan, S.; Zhao II, P.; Zhang III, Q. Research on automatic driving technology architecture based on cooperative vehicle-infrastructure system. In Proceedings of the International Conference on Artificial Intelligence, Virtual Reality, and Visualization (AIVRV 2021), Sanya, China, 19–21 November 2021; Volume 12153, pp. 111–117.
31. Li, Z.; Gong, J.; Lu, C.; Yi, Y. Interactive Behavior Prediction for Heterogeneous Traffic Participants in the Urban Road: A Graph-Neural-Network-Based Multitask Learning Framework. *IEEE/ASME Trans. Mechatron.* **2021**, *26*, 1339–1349. [CrossRef]
32. Li, Z.; Lu, C.; Yi, Y.; Gong, J. A hierarchical framework for interactive behaviour prediction of heterogeneous traffic participants based on graph neural network. *IEEE Trans. Intell. Transp. Syst.* **2021**, 1–13. [CrossRef]
33. Huang, C.; Lv, C.; Hang, P.; Xing, Y. Toward Safe and Personalized Autonomous Driving: Decision-Making and Motion Control With DPF and CDT Techniques. *IEEE/ASME Trans. Mechatron.* **2021**, *26*, 611–620. [CrossRef]
34. Dong, J.; Chen, S.; Ha, P.; Li, Y.; Labi, S. A DRL-based Multiagent Cooperative Control Framework for CAV Networks: A Graphic Convolution Q Network. *arXiv* **2020**, arXiv:2010.05437.