Study on the Path Planning Algorithm Based on Dueling Deep Q Network

Cao Jingwei¹ᵃ, Zhao Shancheng¹ᵇ, Guo Caiqin¹ᶜ, Hu Youtao¹ᵈ, Gao Qianxu¹ᵉ, Shen Fengjun²ᶠ, Ye Yujie²ᵍ, Zhang Yong¹ʰ

¹School of information engineering, Tianjin University of Commerce, Tianjin, Tianjin, 300134, China
²School of science, Tianjin University of Commerce, Tianjin, Tianjin, 300134, China

ᵃemail:jingwei_1219@qq.com,ᵇemail: asd2667910142@163.com,
ᶜemail: 1350350128@163.com,ᵈemail: m16602272367@163.com,
ᶠemail: 2776280197@qq.com,ᵍemail: sfj980113@163.com,
ᵉemail: 705643415@qq.com,ʰemail: zhangyong@tjcu.edu.cn

*Corresponding author’s e-mail: zhangyong@tjcu.edu.cn

Abstract: In traditional path planning methods based on Deep Q-Network (DQN) algorithm for mobile robots, the target Q value was usually obtained by a greedy algorithm, and the estimation of Q value was usually high, which would result in slow training speed of the algorithm. To solve this problem, a Dueling DQN algorithm based on Dueling network was proposed in this paper, which was mainly realized by the structure optimization of neural network. The initial algorithm structure was divided into two states and a higher return is obtained through competitive comparison. Then, the long training time and slow speed convergence problem could be effectively solved than the traditional DQN. The simulation results show that the DQN algorithm based on Dueling DQN has high efficiency in Q value obtaining. Compared with the traditional DQN method, it has the better anti-interference ability and decision-making ability, less training time, and stable convergence.

1. Introduction
Path planning is an inevitable problem for mobile robots. Its essence is to find an optimal path from the starting point to the target state in the workspace according to the criteria of minimum cost, optimal route, or shortest walking time. However, when the task and environment become complex, it becomes extremely heavy to design the basic behavior according to the traditional algorithm. In this case, the robot with self-learning ability has become a new research trend[1]. Among learning methods, reinforcement learning is considered to be a more appropriate method to obtain the control strategy of an autonomous robot in an unknown environment.

Reinforcement Learning (RL) algorithm is an unsupervised learning method, which and has universal applicability [2]. The Agent of reinforcement learning learns through trial and error and seeks the strategy to obtain the maximum cumulative reward in the current environment [3]. At present, reinforcement learning has achieved abundant research results in optimization, control, navigation, and other fields [4-5]. For example, in urban traffic, a reinforcement learning algorithm can be applied to
the path planning algorithm, which can effectively help vehicle path planning and alleviate urban traffic congestion [6]. Although reinforcement learning has many successful applications, it is limited to low-dimensional problems due to sampling and computational complexity.

Recent developments in Deep Learning (DL) have made it possible to extract high-level features from raw sensory input, that is, high-dimensional data can be reliably represented in low dimensions [7]. Deep learning has made great progress in the fields of computer vision, natural language processing, and speech recognition, and has many practical applications [8]. For example, NIC (Neural Image Caption), a representative method for generating captions based on deep learning, can autonomously extract features from images and generate corresponding image captions [9]. If deep learning is applied to reinforcement learning, the computational complexity of reinforcement learning can be solved [10].

Deep Reinforcement Learning (DRL) is an algorithm that can integrate the decision-making ability of reinforcement learning and the perceptual ability of deep learning. For example, the Deep Q-Network (DQN), which combines the Q-Learning method in reinforcement learning with the convolutional neural network (CNN), can be used to solve the problem. DQN is an important method in the field of deep reinforcement learning[11]. At present, DQN has shown its magic power in different fields: defeating top human players in video games[12] and chess and card games[13]; controlling complex machines [14], and allocating network resources[15]. In the aspect of path planning, Tai[16]. applied DQN algorithm to path planning of mobile machine, but this method still has many problems, for example, the target Q value obtained in a complex environment is directly obtained through the greedy algorithm, and the Q value estimation will be higher, resulting in slower algorithm training speed. Therefore, this paper proposes a DQN algorithm based on a dueling network, to obtain better performance.

2. Algorithm Design

2.1 Q-learning algorithm

Q-learning algorithm is a widely used reinforcement learning algorithm. It is an algorithm that can be used to explore the unknown environment model. In the process of exploration, the mobile robot continuously learns to optimize an iteratively computed Q function, and the goal is to find the optimal strategy in each state to maximize the expected return.

The Q value is updated as follows:

\[ Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha (R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)) \]  

(1)

Wherein \( Q(S_t, A_t) \) is the expected return value of the action \( A_t \) selected by the mobile robot in the state \( S_t \); \( R_{t+1} \) is the immediate return value of the action \( A_t \) selected by the mobile robot in the state \( S_t \); \( \gamma \max_a Q(S_{t+1}, a) \) represents the maximum expected return value of various actions selected in a state \( S_{t+1} \); \( \gamma \) is the discount factor, which reflects the impact of future return on immediate return. The lower the value, the smaller the impact; \( \alpha \) is a learning rate.

2.2 Deep Q Network

The Q-learning algorithm uses a Q table to record the Q value of each action in each state and update it repeatedly. However, in practice, there may be too many States to be saved in a table, while deep learning provides the possibility of extracting high-level features from raw sensor data.

DQN takes \( \gamma \max_a Q(S_{t+1}, a) \) as the target Q value and defines a loss function \( L \) based on the deviation between the Q value output by the network and the target Q value:

\[ L(w) = E[(R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t, w))^2] \]  

(2)

Wherein \( (S_{t+1}, a) \) represents the next state and action after the state \( S_t \) takes the action \( A_t; Q(S_t, A_t, w) \) represents the output value of the Q network. In calculation, the weights of DQN can be updated by stochastic gradient descent, and the neural network as a function approximation model can avoid the curse of dimension when the dimension is very large.
2.3 Dueling deep Q network

In the DQN algorithm, the value of each state needs to be estimated, but for many states, it is not necessary to estimate the value of each action. We propose a DQN algorithm based on the Dueling network (Dueling DQN), which optimizes the algorithm by optimizing the structure of the neural network, and divides the initial algorithm structure into two states: the estimated (scalar) state value and the dominant state of each action.

The state-action value function \( Q(s, a) \) represents the expected return value when the action \( a \) is selected by the strategy \( \pi \) in the state \( s \), the state value \( V(s) \) represents the value of the state \( s \) and is the expected value of all action values generated by the strategy \( \pi \) in the state, and the difference between the two values represents the advantage of selecting the action \( a \) in the state \( s \), which is defined as

\[
A(s, a) = Q(s, a) - V(s)
\]

(3)

![Dueling DQN Model Framework.](image)

Figure 1. Dueling DQN Model Framework.

The framework flow of the Dueling DQN model is shown in Figure 1. Firstly, several records are extracted from the memory bank and input into the input layer of the Dueling DQN, and then reach the output layer after passing through several hidden layers. Dueling DQN has two data streams, one stream outputs status value \( V(s; \theta, \beta) \), and the other stream outputs action advantage \( A(s, a; \theta, \alpha) \).

Where in \( \theta \) represents a network neuron parameter for performing feature processing on the input layer; \( \alpha, \beta \) are the parameters of the two flows respectively. The output formula of DQN with dueling network structure is as follow

\[
Q(s, a; \theta, \alpha, \beta) = V(s; \theta, \beta) + A(s, a; \theta, \alpha)
\]

(4)

Since the network outputs the Q value directly, the state value \( V \) and the action advantage \( A \) cannot be known, so the advantage of the forced action advantage estimation under the selected action is 0, and the modified Q value is expressed as

\[
Q(s, a; \theta, \alpha, \beta) = V(s; \theta, \beta) + A(s, a; \theta, \alpha) - \max_{a' \in A} A(s', a'; \theta, \alpha)
\]

(5)

In the practical application of Dueling DQN, the average value of action dominance is usually used to replace the maximum value of action dominance in the calculation of Q value, which ensures the performance and improves the stability of optimization.

3. Simulation Analysis

The experimental key parameter values of all learning processes are shown in Table 1.
Table 1. Parameter Settings.

| Parameter               | Value |
|-------------------------|-------|
| Learning Rate           | 0.01  |
| Reply Memory Size       | 20000 |
| Discount Factor         | 0.9   |
| Gradient Momentum       | 0.9   |
| Batch Size              | 32    |

The environment is shown in Figure 2. The upper left circle represents the mobile robot, and the initial coordinates are (0,0); the lower right circle represents the target point of the mobile robot, and the coordinates are (19,19); the black square represents the obstacle; the white area represents the safety area. The mobile robot starts from the starting point and explores the environment independently. If it encounters obstacles, the reward value is -1, and it starts from the starting point to explore the environment again to find the target point. If the mobile robot does not encounter an obstacle and arrives at the target point, it will receive a reward value of +1.

In the environment shown in Figure 2, the time required for training 200 episodes of Dueling DQN and 2000 episodes of the DQN algorithm is compared, and the experimental results are shown in Table 2.

Table 2. Comparison of Algorithm Training Time.

| Algorithm    | Episode | Time       |
|--------------|---------|------------|
| Dueling DQN  | 200     | 78.163503s |
| DQN          | 200     | 101.873417s|
| Dueling DQN  | 2000    | 491.350165s|
| DQN          | 2000    | 576.421812s|

In the training of 200 and 2000 episodes, the required time of the Dueling DQN algorithm is significantly less than that of the DQN, and the required time is reduced by 23.27% and 14.76% respectively. The Dueling DQN algorithm is better than DQN in training speed.
Figures 3, 4, 5, and 6 describe the comparison of the reward value of each episode and the cumulative reward value of the two algorithms in the simulation environment with the same parameters. The results show that the algorithm based on Dueling DQN can find a path to reach the target point after only one trial and error, and obtain a positive reward, and the cumulative reward increases linearly, that is, it can find the optimal path in the direction of maximizing the reward. However, the path planning algorithm based on DQN has the trend of finding the optimal strategy after 40 episodes of learning, and the cumulative reward is always negative in 100 episodes, and the convergence speed is significantly slower than that of Dueling DQN.

If the algorithm is unstable, the reward will be repeated between +1 and -1, which will cause dark areas in the graph. Comparing Figure 7 and Figure 8, in the 2000 rounds of training, the Dueling DQN algorithm has fewer dark areas, and the algorithm reaches stability after 1750 times of training, the reward obtained each time is +1, and the target point can be reached each time to obtain the reward. However, the DQN algorithm has more dark areas, and after 2000 episodes of training, the algorithm still does not converge steadily. Dueling DQN converges more stably than DQN.
The above experiments show that the Dueling DQN, which is formed by adding a dueling network to DQN, is superior to the DQN algorithm in training time, algorithm convergence speed, and algorithm stability. The algorithm has high efficiency, better anti-interference ability, and decision-making ability than the DQN algorithm, less training time, and stable convergence.

4.Conclusion
In this paper, a Dueling DQN model based on a competitive Deep Q-learning network for mobile robot path planning. Dueling the DQN algorithm, the state output value of the Q network value function was combined with the reward signal obtained by environmental feedback, and adds it to the reinforcement learning training in the form of total reward. The simulation results show that the training time of 200 episodes and 2000 episodes of Dueling DQN is reduced by 23.27% and 14.76% respectively compared with DQN, and the cumulative reward shows a significant linear increase and the convergence speed and stability are significantly better than DQN.

Acknowledgments
The authors wish to thank the financial support of the National College Students Innovation and Entrepreneurship Training Program (No. 202010069003), Tianjin Natural Science Foundation(No. 20JCYBJC00320).

Reference
[1] Lu, J., Xu,L., Zhou,X. (2004) Application of Reinforcement Learning Method in Navigation of Mobile Robot. Journal of Harbin Engineering University, 2004(02):176-179.
[2] Sutton, R. S., Barto, A. G.. (1998) Reinforcement learning: an introduction. IEEE Transactions on Neural Networks, 9(5), 1054-1054.
[3] Huang, D., Jiang, C., Han, K. (2020) A 3D Path Planning Algorithm Based on Deep Reinforcement Learning. Computer Engineering and Applications, 56(15):30-36.
[4] Xu, X., Zuo, L., Huang, Z. (2014) Reinforcement learning algorithms with function approximation: Recent advances and applications. Information Sciences, 261, 1-31.
[5] Silver, D., Sutton, R. S., Müller, M. (2012) Temporal-difference search in computer Go. Machine learning, 87(2), 183-219.
[6] Liu, S, Tong, X. (2021) Urban Traffic Route Planning Based on Reinforcement Learning. Journal of Computer Applications, 41 (01): 185-190.
[7] Liu, Q, Zhai, J, Zhang, Z., et al. (2018) Overview of Deep Reinforcement Learning. Chinese Journal of Computer Science, 41(01):1-27.
[8] Zhang, Q. , Yang, L. T. , Chen, Z. , Li, P. . (2018) A survey on deep learning for big data. Information Fusion, 42, 146-157.
[9] Karpathy, Andrej,Li Fei-Fei.Deep Visual-Semantic Alignments for Generating Image
Descriptions[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017, 39(6):664-676.

[10] Arulkumaran, K., Deisenroth, M. P., Brundage, M., Bharath, A. A. (2017) Deep reinforcement learning: A brief survey. IEEE Signal Processing Magazine, 34(6), 26-38.

[11] Li, H., Wei, T., Ren, A., Zhu, Q., Wang, Y. (2017) Deep reinforcement learning: Framework, applications, and embedded implementations. In 2017 IEEE/ACM International Conference on Computer-Aided Design (ICCAD). Irvine. pp. 847-854.

[12] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... Hassabis, D. (2015) Human-level control through deep reinforcement learning. nature, 518(7540), 529-533.

[13] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., ... Hassabis, D. (2016) Mastering the game of Go with deep neural networks and tree search. nature, 529(7587), 484-489.

[14] Levine, S., Finn, C., Darrell, T., Abbeel, P. (2015) End-to-end training of deep visuomotor policies. Journal of Machine Learning Research, 17(1), 1334-1373.

[15] Mao, H., Alizadeh, M., Menache, I., Kandula, S. (2016) Resource management with deep reinforcement learning. In Proceedings of the 15th ACM workshop on hot topics in networks.50-56.

[16] Tai, L., Liu, M. (2016) Towards cognitive exploration through deep reinforcement learning for mobile robots. arXiv preprint arXiv:1610.01733.