Path planning of mobile robot based on improved DDQN

Yang yunxiao\textsuperscript{1,2,3*}, Wangjun\textsuperscript{1}, Zhang Hualiang\textsuperscript{2,3}, and Dai shilong\textsuperscript{2,3}

\textsuperscript{1} School of Computer Science, Shenyang University of Chemical Technology, Shenyang 110142, China.
\textsuperscript{2} Industrial Control Network and System Laboratory, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, China.
\textsuperscript{3} Institutes for Robotics and Intelligent Manufacturing, Chinese Academy of Sciences, Shenyang 110169, China.

*Corresponding author’s e-mail: 1970241@stu.neu.edu.cn

Abstract. Aiming at the problem of overestimation and sparse rewards of deep Q network algorithm in mobile robot path planning in reinforcement learning, an improved algorithm HERDDQN is proposed. Through the deep convolutional neural network model, the original RGB image is used as input, and it is trained through an end-to-end method. The improved deep reinforcement learning algorithm and the deep Q network algorithm are simulated in the same two-dimensional environment. The experimental results show that the HERDDQN algorithm solves the problem of overestimation and sparse reward better than the DQN algorithm in terms of success rate and reward convergence speed, which shows that the improved algorithm finds a better strategy than the DQN algorithm.

1. Introduction

In recent years, the field of mobile robots has achieved rapid development. Path planning is the core issue. Robots need to plan an optimal path in an environment with obstacles. In the face of increasingly complex environments and the development of artificial intelligence, the intelligentization of mobile robot path planning is the main development trend. Traditional path planning algorithms include A*\textsuperscript{[1]} algorithm, genetic algorithm \textsuperscript{[2]}, ant colony algorithm \textsuperscript{[3]} and so on. Deep reinforcement learning has the strong perception ability of deep learning and the intelligent decision-making ability of reinforcement learning. It has outstanding performance in the face of complex environments and tasks. This is helpful for autonomous learning and obstacle avoidance planning of robots, and has gradually become the research of path planning for mobile robots hot spot.

Literature \textsuperscript{[4]} defines the environment model into three categories, namely, the quadrant of the target point around the robot, the quadrant of the nearest obstacle around the robot, the angle between the line between the robot and the obstacle and the line between the robot and the target point.

The distance of the obstacle target point is divided into four states, and the reward function is defined by the transition of the state. The disadvantage of this method is its weak adaptability to complex scenes. Literature \textsuperscript{[5]} The reward setting uses the obstacle expansion layer to better avoid the collision between the robot and the obstacle. It is determined by the robot radius and safety distance and the distance between the current position and the target position. However, it is difficult to obtain positive rewards in the initial training stage and the convergence is slow. Tai et al. \textsuperscript{[6]} proposed using the DQN algorithm
for simulation path planning of mobile robots, but the DQN algorithm has the disadvantage of overestimating the action value.

OpenAI [7] analyzes the negative impact of artificially designed rewards on learning. The sparse reward problem is a thorny problem in deep reinforcement learning to solve the path planning of mobile robots. In reinforcement learning, the agent needs rewards as a guiding signal. In the early stage of training, the robot's exploration adopts a random strategy and requires frequent interaction with the environment. The rewards are difficult to obtain. In the path planning task, the agent can only obtain positive rewards if it successfully avoids obstacles and reaches the end point. Sparse rewards can cause slow iterations of reinforcement learning and even difficult to converge. Therefore, how to solve the problems caused by the sparse reward environment is crucial to whether reinforcement learning can be better applied to the path planning of mobile robots.

Therefore, this paper proposes an improved path planning algorithm based on DDQN to solve the DQN overestimation problem, and through the use of the hindsight playback experience mechanism, solves the problem of the algorithm's difficulty in converging in the sparse reward environment, improving the sample utilization rate and accelerating the convergence speed, to a certain extent Avoid the design of complex reward functions and improve the effectiveness of reinforcement learning in mobile robot path planning applications.

2. Materials and Methods

The basic model of reinforcement learning is shown in Figure 1. The agent takes an action, and the environment is affected by the action. It will get the next environment state information and reward. The agent then chooses the next action based on the feedback and the current environment state information. Always interact with the environment and continuously improve the strategy.

![Figure 1. Reinforcement learning basic model](image)

Hasselt et al. proposed the DDQN algorithm based on the double Q learning algorithm [8]. Although DQN uses two Q networks and uses the target Q network to calculate the Q value, DDQN solves the problem of overestimating the Q value in the DQN algorithm. The DDQN algorithm uses two different parameters $\theta$ and $\theta^-$, among which the parameter $\theta$ is used to select the target Q value of the action, and the parameter $\theta^-$ is used to evaluate the target Q value of the optimal action. DDQN effectively solves the problem of over-estimation of the Q-learning algorithm of DQN in an experimental environment, so that the agent can choose relatively better actions. However, the agent cannot achieve the goal in most rounds of the training process and will stop in different states. The agent will not learn from these failed episodes, which results in low sample utilization efficiency and slow convergence. In a sparse reward environment, effective exploration is more important, so the reward function must be well designed. In many complex situations, rewards are only given when certain conditions are met, and the negative effects of sparse rewards are difficult to solve. In 2017, Andrychowicz et al. proposed Hindsight Experience Replay [9] (HER). By taking the reached state as the target, mobile robots can make full use of samples. This method is suitable for reinforcement learning algorithms for offline strategies.

Therefore, this paper proposes the HERDDQN algorithm combined with the DDQN algorithm. At this time, the $q$ function not only needs state and action, but also a goal. Therefore the $q$ function is
defined as: \( Q(s,a,g) \). The reward function relies on \( g \in G, r_s : S \times A \times G \to R \) to sample a goal in each round and keep it fixed throughout the round. In a sparse reward environment, a relatively simple dichotomous reward function is used. If the goal is not achieved, the reward is -1 to achieve the goal reward. Is 0. The reward function is shown in formula 1:

\[
r_i = r_g(s,a) = \begin{cases} 
0, & \text{if } s_i = g \\
-1, & \text{otherwise}
\end{cases}
\]

The HERDDQN target value function is shown in formula 2:

\[
y^\text{HERDDQN}_t = r(s,a,g,s') + \gamma Q(s',\arg \max_{a' \in A} Q(s',a',g';\theta^-) ; \theta^-) \]

Using deeply convolutional neural networks to approximate value functions, \( Q(s,a,g;\theta) \approx Q(s,a,g) \).

First, the \( \varepsilon \)-greedy strategy is adopted to randomly select actions to explore the environment. In order to improve the update efficiency of the neural network and the convergence effect of the algorithm, the state of the corresponding trajectory at the last moment is taken as the new target, namely \( s(S,s_i,\ldots,s_r) = m(s_r) \), and the target corresponding to the additional storage of samples becomes \( (s || g', a, r', s_i, || g') \), plus the original The trajectory generated by the target, that is, the experience pool D stores 2 times the actual sampled samples. Perform gradient descent (SGD) on the loss function, update the network parameters \( \theta^- \), and update the TD target network parameters \( \theta^- \) every C steps, that is, let \( \theta^- = \theta \).

The loss function is defined as shown in formula 3:

\[
L(\theta) = E \left[ r(s,a,g,s') + \gamma Q(s',\arg \max_{a' \in A} Q(s',a',g';\theta^-) ; \theta^-) - Q(s,a,g;\theta) \right]^2
\]

The training flowchart is shown in Figure 2:
3. Results & Discussion
The experimental environment of this article is GPU RTX 2080ti, Python3.7, Torch 1.8.1, Cuda11.1. The simulation environment is a two-dimensional grid map with a size of 20×20 pixels. As shown in Figure 3.

![Simulation Environment](image)

**Figure3.** Simulation Environment

When the mobile robot is training, the method is used to explore the strategy, and then the transferred samples with additional targets are collected and stored in the playback experience pool. The samples in the playback experience pool are re-attached to store the new targets for the corresponding round, and then the samples in the playback experience pool are used Train the network. In the training process of each round, the parameters of the target network need to be updated every 3000 steps. The mobile robot has motions in four directions: up, down, left, and right. The parameter settings are shown in Table 1.

| Parameter                  | Numerical Value |
|----------------------------|-----------------|
| Parameter $\epsilon$       | 0.1             |
| Discount Rate $\gamma$     | 0.99            |
| Learning rate $\alpha$     | 0.0001          |
| Number of iterations       | 15000           |
| Playback experience pool size | 500000         |
| Target network update steps | 3000            |

In the same two-dimensional grid simulation environment, and using the same binary reward function to train 15,000 times, the DQN algorithm and the HERDDQN algorithm are compared respectively. The experimental results are as follows.

![Graph](image)

**Figure4.** DQN success rate

![Graph](image)

**Figure5.** HERDDQN Success rate
From Figure 4 and Figure 5, the abscissa represents the number of training rounds, and the ordinate represents the success rate of the mobile robot reaching the end point. In the same environment, after about 15000 times of training, the success rate of DQN was the highest and failed to reach 60%, and there was no convergence and large fluctuations. At the beginning of training, the success rate of HERDDQN rises faster. When training about 3000 times, the HERDDQN algorithm can reach a success rate of about 60%, gradually tends to converge, and finally can reach a success rate of more than 70%.

**Figure 6. DQN Return**

**Figure 7. HERDDQN Return**

From Figure 6 and Figure 7, the abscissa represents the number of training rounds, and the ordinate represents the reward value that the mobile robot can get. During training, the higher the reward value obtained, the closer the path of the mobile robot when avoiding obstacles to the target point is closer to the optimal point. In the environment of sparse rewards, the DQN algorithm has too many samples of negative rewards in the early stage of training, and the samples cannot be used efficiently. The rewards gradually converge after about 7000 trainings. The HERDDQN algorithm maps the reached state to a new goal. And replace the original goal, the goal can be achieved in most rounds, so the reward becomes dense, greatly improving the efficiency of sample use, so the reward value gradually converges after about 2000 trainings.

4. Conclusion
Sparse reward is a tricky problem in the application of reinforcement learning in the path planning of mobile robots. In response to this problem, this paper proposes an improved HERDDQN algorithm based on DQN. Through the combination with the HER algorithm, samples can be more fully utilized and the algorithm iterations faster. It is easier to converge, which solves the problem of sparse rewards to a certain extent. The simulation experiment in the same environment shows that the algorithm in this paper compares the DQN algorithm with a certain degree of improvement and optimization in the training process reward value convergence speed and the success rate of the planning path. In short, the HERDDQN method can achieve greater gains in fewer iterations. The reward value of, so that the mobile robot can obtain the optimal path in the shortest time, so that the deep reinforcement learning algorithm can be more widely used in the path planning problem of the mobile robot.

Acknowledgement
This work was partially supported by National Key Research and Development Program of China under Grant(NO. 2018YFE0205801)

References
[1] Hart P E, Nilsson N J, Raphael B. A formal basis for the heuristic determination of minimum cost paths[J]. IEEE transactions on Systems Science and Cybernetics, 1968, 4(2): 100-107.
[2] Dorigo M, Birattari M, Stutzle T. Ant colony optimization[J]. IEEE computational intelligence magazine, 2006, 1(4): 28-39.
[3] Ismail A T, Sheta A, Al-Weshah M. A mobile robot path planning using genetic algorithm in static environment[J]. Journal of Computer Science, 2008, 4(4): 341-344.

[4] Jaradat M A K, Al-Rousan M, Quadan L. Reinforcement based mobile robot navigation in dynamic environment [J]. Robotics and Computer-Integrated Manufacturing, 2011, 27(1): 135-149.

[5] Gao J, Ye W, Guo J, et al. Deep reinforcement learning for indoor mobile robot path planning[J]. Sensors, 2020, 20(19): 5493.

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

[7] OPENAI. Faulty Reward Functions in the Wild[EB/OL]. https://blog.openai.com/faulty-reward-functions.2017.

[8] Van Hasselt H, Guez A, Silver D. Deep reinforcement learning with double q-learning[C]//Proceedings of the AAAI conference on artificial intelligence. 2016, 30(1).

[9] Andrychowicz M, Wolski F, Ray A, et al. Hindsight experience replay[J]. arXiv preprint arXiv:1707.01495, 2017.