Robot Path Planning Algorithm for Global Optimization Based on DQN Algorithm

Binghui Ji, Shichao Li(✉), Zou Zhou, and Fei Zheng

School of Information and Communication, Guilin University of Electronic Technology, Guilin 541004, China
shichaoli@guet.edu.cn

Abstract. With the widespread adoption of mobile robots, path planning and intelligent navigation have become hot research topics. In order to enhance the globality of traditional robot path planning algorithms, a deep Q-learning network (DQN) algorithm is proposed. By utilizing the reinforcement learning method, mobile robots can effectively navigate to the target location while avoiding the problems in conventional path planning algorithms, such as redundant pathways, decreased continuity and reliance on local information. The proposed method results in a significant reduction in inflection points of the inspection path, mitigating the occurrence of local optima and providing a highly optimized global solution for path planning under known map conditions. In conclusion, the proposed algorithm has been simulated and compared with conventional path planning techniques utilizing a two-dimensional raster map. Empirical findings attest to the dependability of the proposed global optimization technique, which has culminated in the generation of an optimized planned path.

Keywords: Path Planning · Reinforcement Learning · Global Optimization · Deep Q-learning Network algorithm

1 Introduction

In the past few years, there has been a remarkable advancement in the development and utilization of Autonomous Mobile Robots (AMR) due to the high intelligence. AMR has been widely applied in various fields, including social production and practice. They can perform line inspections in urban areas and forest inspections, reducing the occurrence of personnel accidents. Additionally, AMR plays a crucial role in electric power inspection, mountain fire prevention, and control. It plays a significant role in various fields, such as electric power inspection, mountain fire prevention and control. The utilization of AMR automatic inspection technology not only results in cost savings on labor, but also presents a considerable enhancement to production efficiency [1, 2]. However, battery-powered robots often face energy constraints, hindering their widespread implementation. By improving path planning algorithms, it can reduce the paths traveled by AMR, optimize energy consumption, and enhance energy utilization

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2025
Published by Springer Nature Switzerland AG 2025. All Rights Reserved
J. Wang (Ed.): MobiMedia 2023, LNICST 576, pp. 348–356, 2025.
https://doi.org/10.1007/978-3-031-60347-1_29

efficiency. Therefore, the path planning algorithm has gained significant attention as a promising approach to reduce robot energy consumption and enhance energy utilization efficiency.

Path planning is a challenging issue, and numerous scholars have conducted extensive research on the subject. In the current industry, the most commonly utilized algorithms for the path planning problem of known maps are primarily derived from genetic algorithms [3], particle swarm optimization algorithms [4], ant colony algorithms [5], artificial potential field methods [6], and neural networks [7]. The traditional path planning method can solve the path planning problems in known environments, but the exploration ability in complex environments is limited, often leading to local optima. In addition, when the environment changes, relevant parameters in each algorithm should be adjusted. Most traditional methods are low efficiency and complex operations. Conversely, machine learning and reinforcement learning methods function as heuristic strategies based on data training, making them suitable for various path planning tasks in modern application scenarios.

AMR path planning can be established as standard Markov decision problems using methods, such as value iteration and reinforcement learning to map states [8]. Therefore, we propose a path planning algorithm based on DQN reinforcement learning method. The algorithm selects the optimized moving direction for AMR by using a neural network, and then selects the optimized selection set from the object's initial position to the target, ultimately optimizing the AMR's movement and improving the global search ability of the algorithm.

2 Environment Model and Problem Description

In this paper, effective path planning is proposed to reduce the energy consumption of robots with limited areas obstructed by obstacles. The environmental models used for path planning include the grid method, viewable method, topology method and others. Therefore, we utilize the grid method to construct the model of the inspection robot's mobile range.

This article uses forest environment as the model benchmark, and establishes a simulation environment by simulating obstacles such as trees and rocks in the forest. Figure 1 illustrates a 20×20 overall area grid map, with each grid assumed to measure 1 unit of side length. Black squares on the map denote the positions of obstacles, while white squares represent accessible areas. It is also assumed that the robot moves through the center of each grid, i.e. between the centers of two adjacent grids during its operation.

AMR is equipped with eight possible motion directions in each grid, specifically comprising upward, downward, leftward, rightward, upper-leftward, lower-leftward, upper-rightward, and lower-rightward. AMR requires computation to generate planning paths that move from the starting point (0, 0) to the endpoint (20, 20). Taking into consideration whether the algorithms have reduced the required time for path planning, shortened the length of the planned path, and enhanced the smoothness of path planning simultaneously.

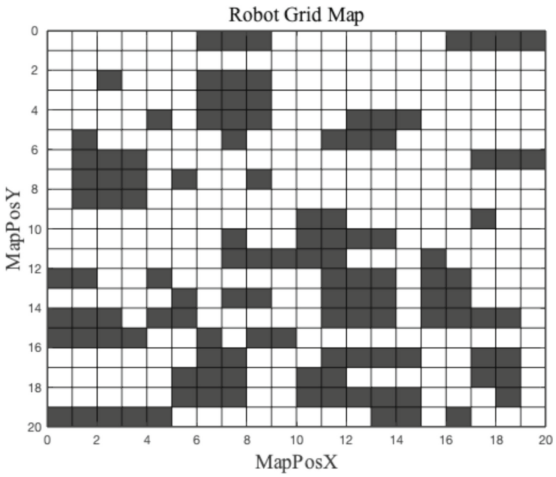


Fig. 1. Modeling the perimeter motion range of the inspection robot.

3 Robot Path Planning Based on DQN Algorithm

3.1 Basic Principle of Q-learning Algorithm

Q-learning algorithm is considered as one offline reinforcement learning algorithm, which was proposed by Watkins in 1989 and is particularly suited for research in route planning strategy selection [9]. It utilizes table-based storage of Q values and can be combined with the sequential difference method and Monte Carlo algorithms. In the application of robot path planning, an initial Q value table is first created to initialize the Q value, after which the robot interacts with the static external environment to earn rewards.

The ε-greedy strategy can be used to modify the action strategy set, and the robot can iteratively modify the Q value table to reach an optimal action set. The objective of the Q-learning algorithm is to determine the optimal policy by iterating through state-action pairs and updating the Q-value table, ultimately resulting in obtaining the optimal policy. In each iteration, the algorithm is used to analyzing the value obtained through each state-action combination that a robot executes, as shown in Eq. (1).

$$Q(s_t, a_t) = \alpha\left[r_t + \lambda \cdot \max_a Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right] + Q(s_t, a_t). \tag{1}$$

where s_t is the robot status at time t, a_t is the activity executed by the robot in status s_t. When the robot changes from the current status s_t to the next status s_{t+1} through action a_t, the reward value r_t can be obtained. α is the learning rate. The larger α is, the faster Q value converges and is prone to oscillation, $\max_a Q(s_{t+1}, a_{t+1})$ means that action selected by algorithm maximizes the value of Q. In this design, the Q-value table established is initialized with no values.

Q-learning algorithm updates Q value with the help of Eq. (2):

$$Q(s_t, a_t) = (1-\alpha)^n \cdot Q(s_t, a_t) + (1-\alpha)^n Q(s_t, a_t)$$

$$+\left[1-(1-\alpha)^n\right]\cdot\left[\lambda Q(s_{t+1},a_{t+1})+r_t\right] \tag{2}$$

At last, the whole Q table will reach a limit.

$$\Delta Q=\lambda\cdot\max_a Q(s_{t+1},a_{t+1})-Q(s_t,a_t)+r_t. \tag{3}$$

when Q value is updated iteratively until $n\rightarrow\infty$, $Q(s_t,a_t)$ converges towards the optimal solution:

$$Q(s_t,a_t)\leftarrow\lambda\cdot\max_a Q(s_{t+1},a_{t+1})+r_t. \tag{4}$$

Equation (4) indicates that the Q-learning algorithm achieves state convergence based on the current state without relying on any initial value, making it an effective reinforcement learning algorithm for modern robot path planning even on unknown paths. However, in complex path environments, the Q-value table can become extremely large, resulting in time-consuming updates. Consequently, Q-learning is not optimally suited for robot path planning in extensive state spaces.

3.2 DQN Algorithm

The DQN algorithm is a variant of Q-learning algorithm, which utilizes deep convolutional neural networks to approximate the value function and an experience replay mechanism. The neural network estimates the value of by Q using parameters θ, as shown in Eq. (5).

$$F(s,a;\theta)\approx Q(s,a). \tag{5}$$

$Q(s,a;\theta)$ is used instead of $F(s,a;\theta)$ in this subsection. The basic idea of using neural network to approximate Q value is shown in Fig. 2.

Neural networks have the ability to map nonlinear functions of any form. In the DQN algorithm, the state space is represented as input and passed through the neural network to generate Q values associated with the action space. The process is shown in Eq. (6):

$$Q_{t,k}=\sum_{\mathrm{k}=1}^{M} W2_{tk}\varphi(W1_{ti}s_{ti}+b1_i)+b2_k. \tag{6}$$

In the equation, k and i respectively represent the dimensions of action space and status space, $W1_{ti}$ and $W2_{tk}$ refers to the connection weights between the hidden layer and output layer. $b1_i$ and $b2_k$ represent the connection thresholds for the hidden layer and output layer.

In Fig. 2, s_t is the input state. For example, s_t represents the features of a small grid, which can be represented by the grid vertex coordinates. Then $s_t=[(x_{t1},y_{t1})|(x_{t2},y_{t2})|(x_{t3},y_{t3})|(x_{t4},y_{t4})]$ represents the current state of robots. The a_t is the sequence of eight potential maneuvers that can be executed by the robot, (up, $down$, $left$, $right$, t - left, b - left, t - $right$, b - $right$). The output $Q(s_t,a_t;\theta)$ of the current neural network represents the value in Q-table of the robot action a_t in the current

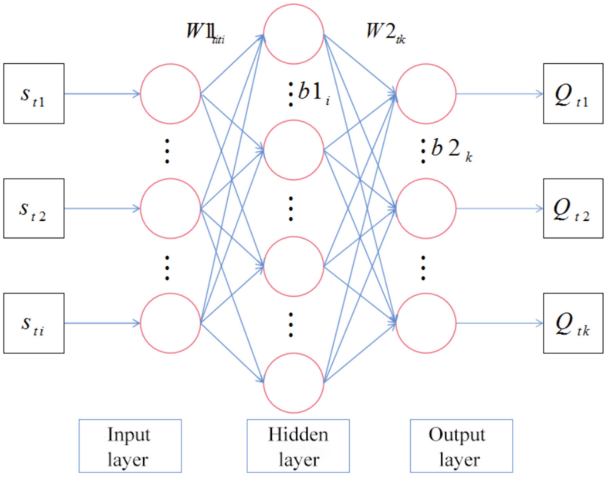


Fig. 2. DQN approximation model.

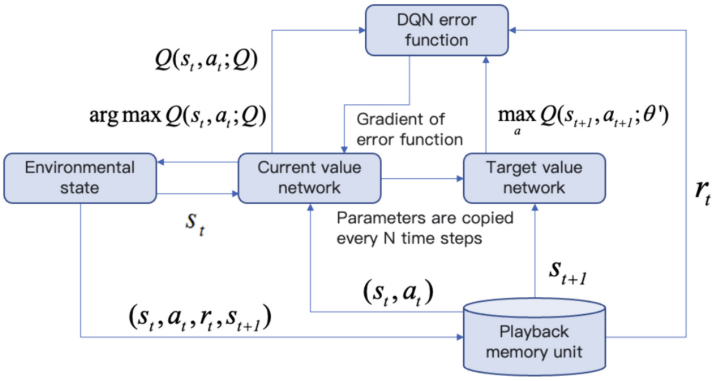


Fig. 3. Flow of DQN algorithm

status s_t. Simply enter the current state s_t to get Q values for all actions in the current state.

Figure 3 shows the DQN algorithm initializes a Q-target neural network with an identical architecture but different parameters from the current Q network. That is a specialized Q-target network which is used to calculate the Q-value of the target in computing, instead of directly using the pre-updated current Q network, in order to reduce the correlation between the target value and the current value.

DQN algorithm uses the same method for data updates as Q-learning algorithm:

$$\left\{Q(s_t, a_t) + \alpha\left[r_t + \lambda \max_a Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right]\right\} \rightarrow Q(s_t, a_t). \tag{7}$$

Select the maximum $Q(s_{t+1}, a_{t+1})$ value multiplied by λ plus return value r according to the next state of the robot as the real Q value, and the previous data $Q(s_t, a_t)$ as estimate Q value.

The loss function of the DQN algorithm can be expressed as:

$$L(\theta) = E\Big[(TragetQ - Q(s_t, a_t; \theta))^2\Big]. \tag{8}$$

$$TragetQ = r_t + \lambda \max_a Q(s_{t+1}, a_{t+1}; \theta). \tag{9}$$

where, $Q(s_t, a_t; \theta)$ denotes the output generated by the present network, which is deployed to appraise the value function that corresponds to the action performed in the current state. The output of the Q-target network denoted as $Q(s_{t+1}, a_{t+1}; \theta)$, which is utilized for the purpose of computing the target value function $TragetQ$.

Q-target network parameter θ is obtained by updating parameterθ. After n iterations, synchronize the current Q-network parameters with the network of Q-Target. By differentiating the parameterθ, the update value function can be obtained, as shown in Eq. (10).

$$\theta_{t+1} = \{\alpha\big[TragetQ - Q(s_t, a_t; \theta)\big]\nabla Q(s_t, a_t; \theta)\} + \theta_t. \tag{10}$$

The reward function can be defined as

$$r_t = \begin{cases} 1 \; if \; s_{t+1} = new\_state \\ 0 \; otherwith \\ -1 \; if \; s_{t+1} = obstacle \end{cases}. \tag{11}$$

In the process of reinforcement learning, the robot selects actions not only based on a certain prior probability but also moves according to the Q-value table constructed by learning. Therefore, the robot can improve the globality of path planning through the DQN algorithm.

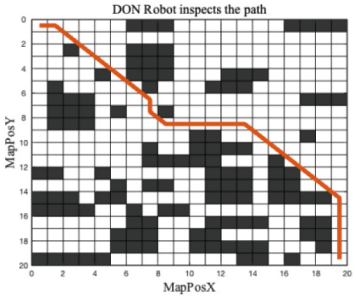
3.3 DQN Algorithm Steps

Algorithm DQN algorithm for AMR path planning

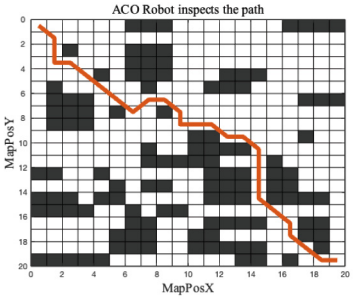
1: Initialize the space size of playback memory unit D as N, and initialize Q-target network parameter θ' and current network parameter θ.
2: The robot interacts with the environment and saves the interactive information (s_t, a_t, r_t, s_{t+1}) in the space of reply memory unit D.
3: Retrieve a batch of interactive information samples from the space of playback memory unit D.
4: Calculate $TragetQ$ using Q-target network.
5: Use Eq. (9) to update the parameters.
6: Synchronizing the parameters θ of current network to the Q target network parameter θ' every N steps.

4 Simulation and Result Analysis

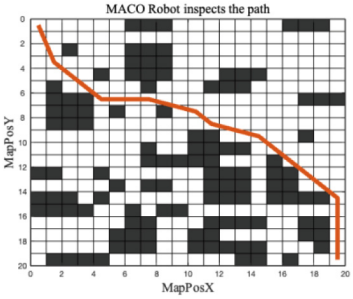
The performance of the algorithm is simulated by using the map in Fig. 1. In order to validate the effectiveness and generalization of the proposed DQN algorithm, we compare it with the Multi-step Ant Colony Optimization (MACO) algorithm designed in [10] and the traditional Ant Colony Optimization (ACO) algorithm. The MACO algorithm aims to accelerate the convergence speed by changing the path of a single ant from moving a fixed distance each time to moving multiple distances. In the simulation, ants are only allowed to move freely between white squares when encountering obstacles that cannot pass.



(a) Path planning of DQN algorithm



(b) Path planning of ACO algorithm



(c) Path planning of MACO algorithm

Fig. 4. Path planning for three algorithms

Figure 4 compares the paths generated by three algorithms. Compared with the traditional ACO algorithm, the MACO algorithm has greatly improved convergence performance and the effectiveness and rationality of path planning. However, there are still too many redundant turning points, which greatly affect the smoothness of the path. The proposed DQN algorithm further reduces the number of turning points in robot traveling pathsand shortens the path distance, making robot traveling paths smoother. This leads to a decrease in energy consumption rate and hardware damage level in practical applications, increasing the running time and service life of AMR.

The results of the path distance, number of turning points, convergence speed, and smoothness of the three algorithms are shown as follows.

Table 1. Comparison of the performance of the three algorithms

Algorithm	Minimum path distance	Maximum path distance	Number of path inflection points	Path smoothness	computational complexity
DQN	29.79	-	6	Good	Moderate
ACO	33.21	41.53	16	Poor	Low
MACO	31.16	41.27	7	Good	High

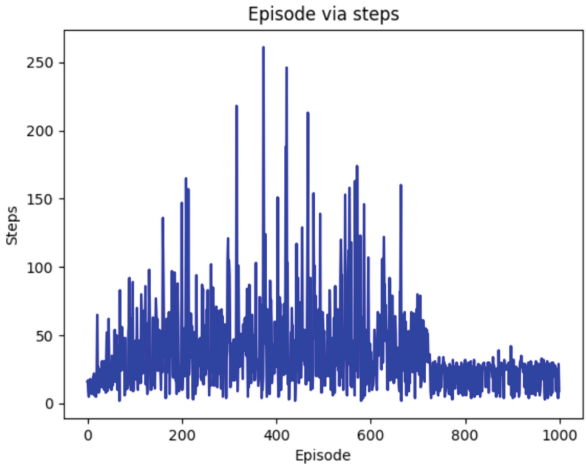


Fig. 5. The proposed DQN algorithm convergence process

Based on the analysis of Table 1, it can be observed that the planned path of the proposed DQN algorithm shows a significant improvement in terms of path distance, number of turning points, and smoothness compared to the traditional path planning algorithm. Moreover, the analysis conducted through post hoc statistics reveals that the computational complexity of the DQN algorithm is lower than that of the MACO algorithm, but higher than that of the ACO algorithm, making it relatively moderate in terms of computational complexity.

Figure 5 is the proposed DQN algorithm convergence process, it can be concluded that AMR path planning based on DQN algorithm achieves a high level of convergence of reinforcement learning.

5 Conclusion

In order to enhance the globality of traditional robot path planning, this paper proposes a deep Q network reinforcement learning algorithm. By using reinforcement learning, environmental information is utilized to move an AMR to a target point, overcoming problems, such as redundant paths, low smoothness, and excessive influence of local paths in traditional path planning algorithms. The proposed DQN algorithm is compared with MACO and ACO algorithms through simulation experiments. The results show that the proposed path planning algorithm can reduce the number of turning points in inspection routes and avoid local optimal solutions, providing better globality for solving path planning problems in known environments.

References

1. Zhang, B., Tang, L., DeCastro, J., et al.: A recursive receding horizon planning for unmanned vehicles. IEEE Trans. Industr. Electron. **62**(5), 2912–2920 (2014)
2. Deng, J., Guo J., Xue, N., et al.: Arcface: Additive angular margin loss for deep face recognition. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 4690–4699 (2019)
3. Zhai, L., Feng, S.: A novel evacuation path planning method based on improved genetic algorithm. J. Intell. Fuzzy Syst. **42**(3), 1813–1823 (2022)
4. Jiang, W., Pan, S., Lu, C., et al.: Label entropy-based cooperative particle swarm optimization algorithm for dynamic overlapping community detection in complex networks. Int. J. Intell. Syst. **37**(2), 1371–1407 (2022)
5. Song, B., Miao, H., Xu, L.: Path planning for coal mine robot via improved ant colony optimization algorithm. Syst. Sci. Control. Eng. **9**(1), 283–289 (2021)
6. Hwang, J., Lee, J., Park, C.: Collision avoidance control for formation flying of multiple spacecraft using artificial potential field. Adv. Space Res. **69**(5), 2197–2209 (2022)
7. Ji, X., Wang, J., Zhao, Y., et al.: Path planning and tracking for vehicle parallel parking based on preview BP neural network PID controller. Trans. Tianjin. Univ. **21**(3), 199–208 (2015)
8. Chen, D., Trivedi, K.S.: Optimization for condition-based maintenance with semi-Markov decision process. Reliab. Eng. Syst. Saf. **90**(1), 25–29 (2005)
9. Jameel, F., Javed, M.A., Zeadally, S., et al.: Secure transmission in cellular V2X communications using deep Q-learning. IEEE Trans. Intell. Transp. Syst. **23**(10), 17167–17176 (2022)
10. Zeng, M.R., Xu, X., Luo, H., et al.: Research of robot path planning of multi-step ant colony algorithm. J. Chin. Comput. Syst. **37**(02), 366–369 (2016)