Hindawi
Journal of Electrical and Computer Engineering
Volume 2022, Article ID 5433988, 7 pages
https://doi.org/10.1155/2022/5433988

Research Article

Path Planning Method of Mobile Robot Using Improved Deep Reinforcement Learning

Wei Wang,[1] **Zhenkui Wu**,[2] **Huafu Luo**,[3] **and Bin Zhang**[4]

[1]*Department of Electrical Engineering, BaoTou Iron & Steel Vocational Technical College, Baotou, Inner Mongolia 014010, China*
[2]*School of Information Engineering, Inner Mongolia University of Science & Technology, Baotou, Inner Mongolia 014010, China*
[3]*Department of Electrical Information Engineering, Sichuan Engineering Technical College, Deyang, Sichuan 618000, China*
[4]*Inner Mongolia Kingdomway Pharmaceutical Limited, Inner Mongolia, Huhehaote 010200, Tuoketuo, China*

Correspondence should be addressed to Wei Wang; emmawangwei@163.com

Received 4 March 2022; Revised 25 March 2022; Accepted 6 April 2022; Published 23 April 2022

Academic Editor: Wei Liu

A mobile robot path planning method based on improved deep reinforcement learning is proposed. First, in order to conform to the actual kinematics model of the robot, the continuous environmental state space and discrete action state space are designed. In addition, an improved deep Q-network (DQN) method is proposed, which takes the directly collected information as the training samples and combines the environmental state characteristics of the robot and the target point to be reached as the input of the network. DQN method takes the Q value at the current position as the output of the network model and uses ε-greedy strategy for action selection. Finally, the reward function combined with the artificial potential field method is designed to optimize the state-action space. The reward function solves the problem of sparse reward in the environmental state space and makes the action selection of the robot more accurate. Experiments show that compared with the classical DQN method, the average loss function value is reduced by 36.87% and the average reward value is increased by 12.96%, which can effectively improve the working efficiency of mobile robot.

1. Introduction

With the continuous integration of informatization and industrialization, the intelligent industry represented by robot technology is booming, which is also the key development field of scientific and technological innovation in various countries [1]. In many application scenarios of robots, the working environment of robots is complex, diverse, and unpredictable. Completing the task of path planning in a complex and unknown environment requires the robot to have a certain degree of intelligence, that is, the ability of autonomous learning and the ability to explore the environment [2–4]. On the other hand, because the robot does not have enough environmental information in the unknown environment, in order for the robot to successfully and efficiently realize path planning in an unknown environment, the robot needs to have a certain degree of adaptability to its working scene and the ability to deal with emergencies [5–7].

Artificial intelligence is one of the important methods of robot navigation [8–11]. In recent years, with the continuous efforts of experts and scholars at home and abroad in the field of artificial intelligence and the continuous progress of artificial intelligence technology, some technologies have been successfully applied to human daily life, including speech recognition and machine translation. As one of the research hotspots in artificial intelligence technology, deep reinforcement learning (DRL) has made a breakthrough [12]. AlphaGo defeated the world's top Go experts. Through deep reinforcement learning algorithm, the DeepMind team can well control YALi arcade games after training, even surpassing the level of ordinary game players, showing the powerful advantages of this method and greatly increasing the confidence of researchers and experts in the research of artificial intelligence technology [13]. Deep reinforcement learning is an autonomous learning method, which does not need to mark the training data, learns appropriate behavior

from the environmental state, and allows the robot to modify its strategy according to the received reward or punishment [14]. By introducing the deep reinforcement learning method into the robot path planning, the robot is equivalent to having a "brain" for autonomous learning and walking. In the unknown and complex information scenes, the robot can adjust the walking path and plan the path independently. In the actual robot path planning task, the deep reinforcement learning method still has some problems to be solved and optimized, such as algorithm training efficiency, reward function design, and convergence stability [15, 16]. Taking the deep reinforcement learning method as the path planning algorithm in the robot path planning task has strong scientific research significance and practical application value for improving the intelligent degree of the robot.

In order to solve the problems of poor exploration ability and sparse reward of environmental state space in mobile robot path planning in unknown environment, a mobile robot path planning method based on improved deep reinforcement learning is proposed. The innovations of the proposed method lie as follows:

(1) The DQN method is improved. The sensor senses the surrounding environmental information, combines its own location information and the target point to form a state space as the input of the network, takes the Q value at the current location as the output of the network model, and uses -greedy strategy for action selection to improve the operation efficiency.

(2) Combining the artificial potential field method with the reward function, the state-action space is optimized, the reward value of the method is improved, and the reward sparsity of the environmental state space is improved.

2. Related Works

Traditional robot path planning methods cannot meet the requirements of modern robot path planning tasks, such as genetic algorithm, artificial potential field method, and neural network [17–19]. These traditional methods require the robot to fully understand the surrounding environment information. When the working environment of the robot is complex and the state is changeable, these traditional methods cannot effectively complete the task of path planning. At the same time, these path planning methods cannot improve the self-study ability, exploration ability, and environmental adaptability of the robot [15]. In order to overcome the shortcomings of these methods, researchers at home and abroad have explored various solutions. Reference [20] used the depth Q-network algorithm to conduct the robot navigation simulation experiment in the maze scene. The experiment directly took color image with a size of 160×120 as input. The output was in three states: straight ahead, right turn, and left turn. The experimental results showed that the robot not only successfully realized obstacle avoidance but also had the ability of self-learning. Reference [21] combined double DQN with competitive architecture DQN, adopted multiview strategy, and took the data collected by four cameras from four different directions as input. The experimental results verified the effectiveness of this method. In order to obtain more training data, reference [22] proposed a deep reinforcement learning method of nonstrategy training based on deep Q function by using the multithreading method. This method adopted the concurrent method, which could quickly collect the samples required for training to a certain extent without too much manual intervention and effectively ensure the training time. Reference [23] proposed a distributed asynchronous strategy method for multirobot cooperative learning because multiple robots could share their experience with each other and learn a strategy together. Each robot used local neural network strategy to optimize the behavior of each robot in different scenes and then saved the training samples of all robots to the same server, so as to supervise the training of global neural network strategy. This method effectively reduced the learning time of robot. Reference [24] applied DQN to model-free obstacle avoidance path planning, but there was a problem of overestimation of state-action values, resulting in sparse rewards for mobile robots, and the planned path was not optimal. Reference [25] proposed an end-to-end path planning method of safety constrained robot based on deep reinforcement learning. Reference [26] proposed a path planning method of mobile robot based on TPR-DDPG. This method preprocessed the state through various normalization methods and designed a complete reward function to make the mobile robot quickly reach the target point through the optimal path in the complex environment. Reference [27] proposed a deep reinforcement learning method based on double deep Q-network (DDQN), which made the robot have the ability of autonomous navigation. However, the above methods often have problems such as poor exploration ability and sparse reward in environmental state space, so it is difficult to obtain the optimal planning path.

To overcome the above problems, a mobile robot path planning method based on improved deep reinforcement learning is proposed.

3. Path Planning Method Based on Deep Reinforcement Learning

3.1. Robot Kinematics Model. Pioneer3-AT is a four-wheel differential driving robot composed of front steering wheel and rear driving wheel. The driving wheel moves forward at the speed v and uses the speed difference between the left and right wheels for steering operation. The model diagram of Pioneer3-AT in plane coordinate system is shown in Figure 1. If the midpoint of the robot's rear axle is selected as the reference point, the robot's attitude at time t can be expressed by a three-dimensional vector $P(x_t, y_t, \theta)$. (x_t, y_t) is the coordinate of the robot in the system coordinate. δ is the steering angle, which is used to describe the angle of the robot's steering wheel. θ is the direction angle, which is used to describe the included angle between the spatial fixed coordinate system and the robot's fixed coordinate system, and d is the distance between the front wheel and the rear wheel.

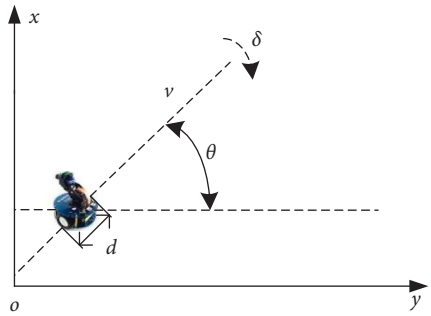

Figure 1: Robot motion coordinates.

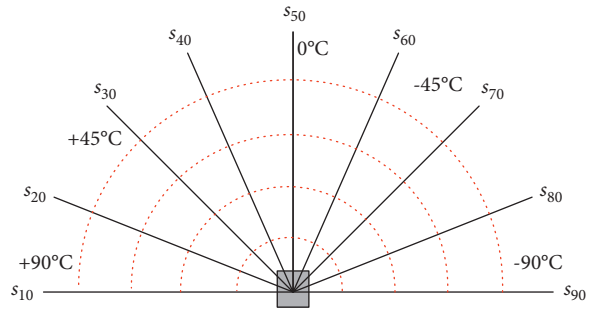

Figure 2: Environmental state space diagram.

Under the spatial fixed coordinate system, the kinematic model of pioneer3-AT is

$$\begin{bmatrix} x_t \\ y_t \\ \theta \end{bmatrix} = \begin{bmatrix} v\cos\theta \\ v\sin\theta \\ v\tan\dfrac{\theta}{d} \end{bmatrix}. \tag{1}$$

where $[x_t\, y_t \theta]^T$ represents state vector at time t.

The discretization equation for obtaining the position and attitude of the robot is

$$\begin{cases} x_{t+1} = x_t + Tv\cos\theta_t \\ y_{t+1} = y_t + Tv\sin\theta_t \\ \theta_{t+1} = \theta_t + \dfrac{Tv\tan\delta_t}{d} \end{cases}, \tag{2}$$

where T is sampling time.

3.2. Environmental State Space Design. State space is the feedback of the whole environment and the basis for agents to choose action space. The robot is equipped with Lidar with a detection range of 10 meters and a scanning range of 360 degrees. Considering the accuracy and calculation, only 180 degrees in front of the robot is considered, and the Lidar data in 9 directions are taken, as shown in Figure 2.

We use s_{io} to indicate the distance of obstacles detected by the Lidar in all directions and use the Lidar's measurement span to normalize s_{io}. The final azimuth state information is $[S_{1o}, S_{2o}, \ldots, S_{9o}]$.

$$S_{io} = \frac{s_{io}}{S}, \tag{3}$$

where S is the maximum range of Lidar.

In order to enable the robot to move towards the target point, the angle β between the current orientation of the robot and the target point is taken as an input state, as shown in Figure 3. When the target point is on the left side of the robot, the value range of β is $[0, 180^\circ]$. When the target point is on the right side of the robot, the value range of β is $(-180^\circ, 0)$.

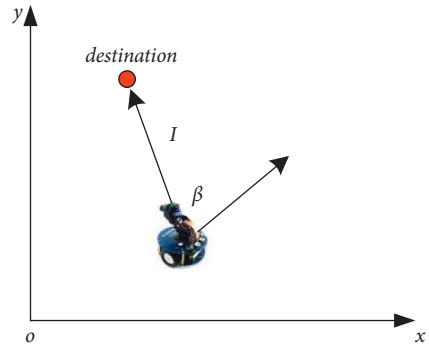

Figure 3: Target angle.

3.3. Action State Space Design. The motions of Pioneer3-AT robot include rotational motion and translational motion. The robot controls the moving direction according to the rotational motion. The action space of the robot is continuous and will produce a huge action space. Because the huge action space will make the deep reinforcement learning method difficult to converge, the translation motion of the robot is set as 0.8 m/s, and the rotation motion is divided into five discrete actions according to

$$A = \{A_i, \quad i = 1, 2, 3, 4, 5\} = \{0^\circ, \pm 30^\circ, \pm 45^\circ\}. \tag{4}$$

When the robot turns right according to its own motion direction, the action A_i is a negative angle; When it turns left, A_i is a positive angle. Each rotation angle corresponds to a Q value. The Q value of each rotation angle is obtained through neural network, and then the corresponding rotation angle is selected according to the Q value.

3.4. Improved Action Selection Strategy. The model of the proposed method is shown in Figure 4. The robot takes the directly collected depth information as the training sample, combines its own environmental state characteristics and the target point to be reached as the input of the network, takes the Q value at the current position as the output of the network model, and uses the ε-greedy strategy for action selection to reach the next state. When the next step is reached, the corresponding reward value r is calculated to obtain a complete data tuple $(s, a, r, s\prime)$, so the series of data is stored in the experience playback pool P, and then a small batch of samples is extracted from the experience playback

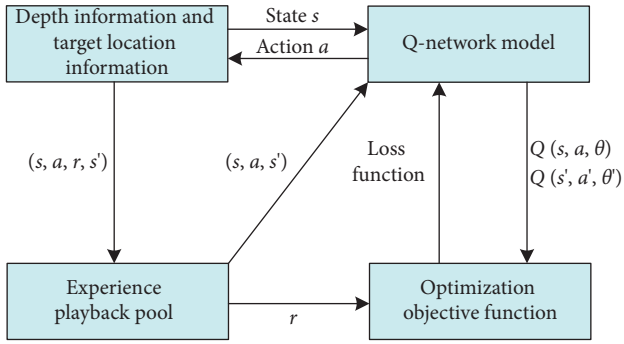

Figure 4: Improved DQN method model.

pool P and put them into the neural network for training. In the process of exploring the optimal path, it is very important for the robot to select the reward value r from the experience playback pool P. The reward value r determines the quality of the robot path planning. The robot sends the reward value r to the optimization objective function to update the network parameters and iterates until the training is completed.

In the process of network training, if the robot uses sensors to identify obstacles, the robot can effectively avoid obstacles by improving the deep reinforcement learning method. The design of the improved deep reinforcement learning algorithm is shown in Algorithm 1.

3.5. Reward Function Design. The reward function is used to evaluate the quality of the decision-making of the deep learning framework and reward each step of the decision-making. It plays a guiding role in the whole learning process. The neural network makes specific decisions according to the state, the environment is updated according to the decision, and the reward value is calculated. The neural network updates the network parameters according to the feedback reward value, so that the network can make better decisions in the next calculation. The quality of reward function directly affects the effectiveness and convergence of the whole reinforcement learning framework. The reward function is designed as follows:

$$R = \begin{cases} -100 \ d(t) < d_o, \\ -2 \text{ others}, \\ -1 \ d(t) = d(t-1), \\ 1 \ d(t) < d(t-1), \\ 2 \ d(t) < d_g, \\ 20 \ d(t) < d_g, \end{cases} \tag{5}$$

$$R = r_{\text{ate}} * n_{\text{orm}}(P_0 - P_1) + (1 - r_{\text{ate}}) * R, \tag{6}$$

where $d(t)$ represents the distance between the robot and the target point at the current time; $d(t-1)$ represents the distance between the robot and the target point at the last time; d_o represents the safe distance of the obstacle. If distance is less than d_o, it indicates the obstacle is encountered; d_g represents the threshold value from the target point. If distance is less than d_g, it is considered to have reached the target point; d_n represents the distance close to the target point. A positive reward will be given when reaching this range, which can promote the robot to reach the target point faster. When the position of the robot is an obstacle, it will get a negative reward; When the robot reaches the target point, it will get a larger positive reward. When the robot approaches the target point, it will get 1 as a reward; When the robot stays in place, it will get −1 as a reward. In other cases, it will get −2 as a reward. This reward is smaller than 1 because it can prevent the robot from moving back and forth to obtain a positive reward and promote the robot to find the shortest path.

In order to solve the problem of blind selection of action by traditional methods, the artificial potential field method is introduced as an auxiliary of early training, so that the system can train the model faster. After selecting the action space, first calculate the next position P_0 of the robot according to the artificial potential field method and then update the environment to get the actual position P_1 of the robot. On the basis of the original reward function, add the distance between P_0 and P_1. In equation (5), $n_{\text{orm}}(P_0 - P_1)$ is calculation of the distance between 2 positions. r_{ate} is a weight value, which represents the weight of the artificial potential field method. In the early stage, r_{ate} has a large proportion and guides the robot to the end faster. With the increase of training times, the weight gradually decreases.

4. Experiment and Analysis

4.1. Simulation Environment Construction and Parameter Setting. Considering that the environment of the actual robot is diverse, the shapes of obstacles in the experimental environment are diverse, and the placement positions are as random as possible. The configuration environment of the computer used in the experiment is NVIDIA GTX 2080Ti GPU server, and the operating system is Ubuntu 16 04. Pioneer3-AT robot and its working environment are simulated with ROS and Gazebo. Python programming language and TensorFlow framework are used. Some relevant parameter values in the experiment are shown in Table 1.

4.2. Comparison between Loss Functions and between Average Value of Rewards. The parameters are updated iteratively through the loss function $L(\theta)$ and Adam optimizer. Under the framework of Keras based on TensorFlow, using GPU acceleration, all training samples pass through 1000 epochs to complete the training of the proposed improved deep reinforcement learning model. Figure 5 shows the comparison of the loss function values in the training process of the network model. It can be seen that the loss values of the two methods tend to be stable with the continuous increase of the number of training iterations. The value of loss function tends to be stable when the training times of DQN algorithm reach about 270. When the training times of the proposed method reach about 200, the value of the loss function can quickly become stable. The average loss function value of DQN algorithm is 0.178, and the average loss function value of the proposed method is 0.112, which is 36.87% lower than

Inputs: information $\varepsilon = [s_t, a_{t-1}]$, target point G, the parameters of Q network is θ, and the parameters of target Q network is θ'.
Output: action direction of robot.
Initialization: initialize s_t, experience playback pool P with capacity of N, action value function Q, the weights of target Q network is θ, $\theta\prime = \theta$.
for episode = 1, M do
  for $t = 1$, T do
  Get from experience pool P, input $s_t = [s_p, s_{p-1}, \ldots, s_0]$
  The robot chooses a random action a_t with a certain probability ε; otherwise, it chooses the optimal action $a_t = \max_a Q\prime((s_t), a; \theta)$
  Perform action a_t in the environment, get reward $r_t = 2v^2 \cos(2vw) - 0.1$ and s_{t+1}, put (s_t, a_t, r_t, s_{t+1}) into experience pool P.
  Randomly take a small batch of samples (s_j, a_j, r_j, s_{j+1}) from the experience pool P
  Loss function is $L(\theta) = E[(y\prime - Q(s, a, \theta)^2)]$
  Perform policy gradient update
  End for
End for

Algorithm 1: Improved deep reinforcement learning algorithm for mobile robot.

Table 1: Experimental parameters of robot path planning.

Parameter	Value
Learning rate	0.005
Reward decline	0.8
Network update frequency	400
Experience pool size	10000
Number of batches	16
Number of samples in each layer	15
β	0.3
Starting value, ε	0.1
Termination value, ε	0.0001

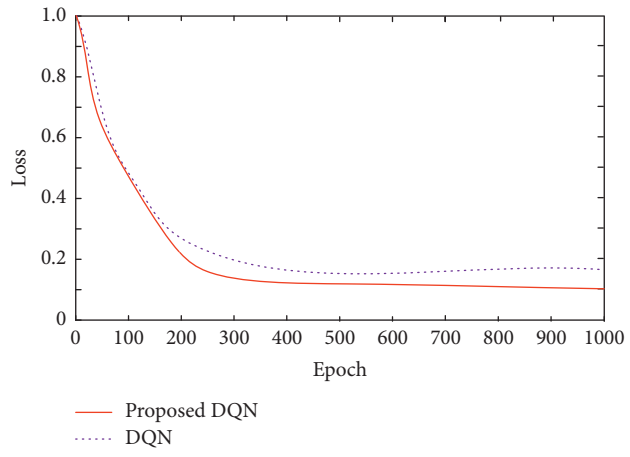

Figure 5: Comparison diagram of loss function values.

that of DQN algorithm. Therefore, the improved method has certain advantages and speeds up the convergence speed of the network.

Figure 6 shows the comparison of the average reward values between DQN algorithm and the proposed method. At the beginning of training, the reward value is between −9 and 0, which is the process that the robot just begins to explore, learn, and avoid obstacles, and it fails to make correct judgment on obstacles, so getting a negative reward value. When the reward value is (0~80) stage, the training

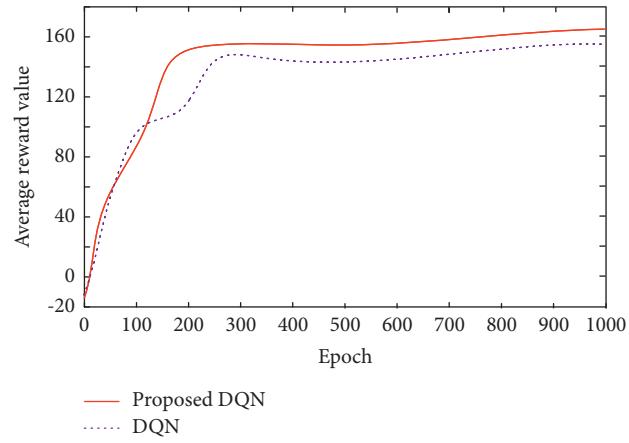

Figure 6: Comparison of average reward values.

times reach 100, and the robot is in the exploratory learning stage, which means that the robot begins to recognize and can avoid some obstacles, but it is still learning interactively with the environment and further adjust the action selection strategy to obtain the corresponding positive reward value. When the reward value is (80~160) stage and the training times are 100~200, the reward value obtained by the robot in DQN algorithm and improved DQN algorithm is unstable. When the training times reach about 300, the reward value obtained by DQN algorithm tends to be balanced. When the training times of the improved method reach about 200, the average reward value tends to be stable. The average reward value of the improved DQN is 12.96% higher than that of DQN. Therefore, the improved method can shorten the network training time, improve the average reward value, and improve the reward sparsity.

4.3. Comparison of Several Methods of Path Planning. In order to prove the advantages of the proposed improved DQN mobile robot path planning method, path planning methods in reference [25, 26] are compared with the proposed method under the same experimental conditions. The comparison indicators include the following two: the

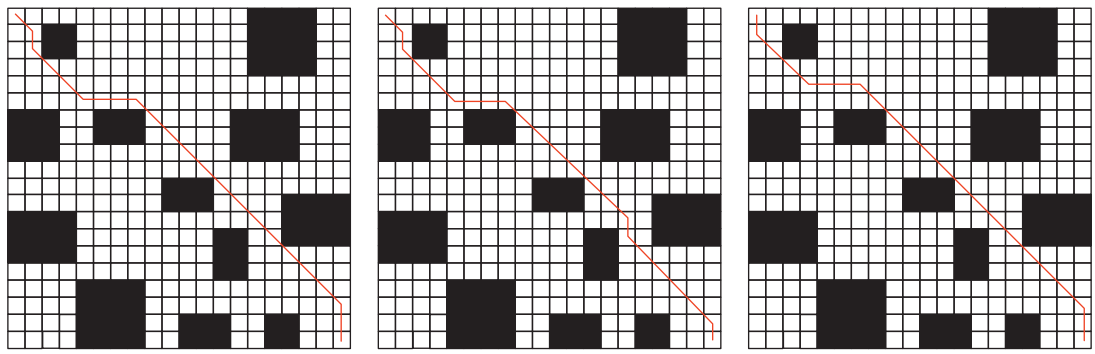

Figure 7: Comparison of path planning of different methods. (a) Ref [26]. (b) Ref [27]. (c) Proposed methods.

Table 2: Comparison of path length and number of turning points of different methods.

Method	Ref. [25]	Ref. [26]	Proposed method
Path length	28.627	28.627	28.627
Number of turning points	5	7	4

planned path length and the number of turning points in planned path. The obtained path planning diagram is shown in Figure 7, and the comparison data is shown in Table 2.

Figures 7(a)–7(c) are the paths planned by reference [25, 26] and the proposed improved DQN method according to the action corresponding to the maximum Q value output by the network after 500 iterations. From the path planning map and Table 2, each method has planned a collision-free path. In terms of the advantages and disadvantages of the path, the proposed DQN method plans an optimal path with a path length of 28.627 and a number of turning points of 4, while the methods in reference [25, 26] do not plan an optimal path. Although the path length is the same as the proposed method, the number of turning points of the two methods are 5 and 7 respectively, which increases the time-consumption of robots. The proposed method aims at the global path planning and combines the artificial potential field method with the reward function to optimize the state-action space, improve the reward value of the method, and guide the robot to the target faster. The comparison method has the problem of overestimation of state-action value, resulting in sparse rewards for mobile robots, and the planned path is not optimal. In the actual environment, the reduction of turning points in path planning will also reduce the time spent by the robot from the starting point to the target point.

5. Conclusion

In the face of mobile robot path planning in unknown environment, the current research often has the problems of poor exploration ability and sparse reward in environmental state space, so it is difficult to obtain the optimal planning path. In order to overcome the above problems, a mobile robot path planning method based on improved deep reinforcement learning is proposed. The continuous environmental state space and discrete action state space are designed, an improved DQN method is proposed to speed up the path search speed, and an improved reward and punishment function is designed to improve the reward value of the method, which alleviates the problem of sparse reward to a certain extent. Experiments show that compared with the comparative references, the proposed method can effectively improve the working efficiency of mobile robot. Using images as data input will get richer environmental information than Lidar. In the future, robot autonomous path planning through camera will be studied. In addition, multirobot cooperative path planning task is also the focus of the next research of this topic.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the Major Science and Technology Projects of Inner Mongolia Autonomous Region (no. 2020ZD0017).

References

[1] K. Karur, N. Sharma, C. Dharmatti, and J. E. Siegel, "A survey of path planning algorithms for mobile robots," *Vehicles*, vol. 3, no. 3, pp. 448–468, 2021.

[2] M. N. Zafar and J. C. Mohanta, "Methodology for path planning and optimization of mobile robots: a review," *Procedia Computer Science*, vol. 133, no. 1, pp. 141–152, 2018.

[3] A. Khan, I. Noreen, and Z. Habib, "On complete coverage path planning algorithms for non-holonomic mobile robots: survey and challenges," *Journal of Information Science and Engineering*, vol. 33, no. 1, pp. 12–21, 2017.

[4] H. Choset, "Coverage for robotics–a survey of recent results," *Annals of Mathematics and Artificial Intelligence*, vol. 31, no. 1, pp. 113–126, 2001.

[5] S. Aggarwal and N. Kumar, "Path planning techniques for unmanned aerial vehicles: a review, solutions, and challenges," *Computer Communications*, vol. 149, no. 5, pp. 270–299, 2020.

[6] M. M. Costa and M. F. Silva, "A survey on path planning algorithms for mobile robots," in *Proceedings of the 2019 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pp. 1–7, IEEE, Porto, Portugal, April 2019.

[7] H. Jahanshahi, M. Jafarzadeh, N. N. Sari, V.-T. Pham, V. V. Huynh, and X. Q. Nguyen, "Robot motion planning in an unknown environment with danger space," *Electronics*, vol. 8, no. 2, pp. 201–212, 2019.

[8] P. Victerpaul, D. Saravanan, S. Janakiraman, and J. Pradeep, "Path planning of autonomous mobile robots: a survey and comparison," *Journal of Advanced Research in Dynamical and Control Systems*, vol. 9, no. 12, pp. 1535–1565, 2017.

[9] B. Sahu, P. K. Das, M. R. Kabat, and R. Kumar, "Multi-robot cooperation and performance analysis with particle swarm optimization variants," *Multimedia Tools and Applications*, vol. 5, no. 9, pp. 1–24, 2021.

[10] R. Liu and X. Zhang, "A review of methodologies for natural-language-facilitated human–robot cooperation," *International Journal of Advanced Robotic Systems*, vol. 16, no. 3, pp. 172–179, 2019.

[11] R. Bormann, F. Jordan, J. Hampp, and M. Hagele, "Indoor coverage path planning: survey, implementation, analysis," in *Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1718–1725, IEEE, Brisbane, Australia, May 2018.

[12] G. S. Chirikjian, "Design and analysis of some non-anthropomorphic, biologically inspired robots: an overview," *Journal of Robotic Systems*, vol. 18, no. 12, pp. 701–713, 2001.

[13] B. Chandra Mohan and R. Baskaran, "A survey: ant Colony Optimization based recent research and implementation on several engineering domain," *Expert Systems with Applications*, vol. 39, no. 4, pp. 4618–4627, 2012.

[14] A. Injarapu and S. K. Gawre, "A survey of autonomous mobile robot path planning approaches," in *Proceedings of the 2017 International conference on recent innovations in signal processing and embedded systems (RISE)*, pp. 624–628, IEEE, Bhopal, India, October 2017.

[15] B. K. Patle, G. Babu L, A. Pandey, D. R. K. Parhi, and A. Jagadeesh, "A review: on path planning strategies for navigation of mobile robot," *Defence Technology*, vol. 15, no. 4, pp. 582–606, 2019.

[16] B. Velusamy and S. C. Pushpan, "A review on swarm intelligence based routing approaches," *Int. J. Eng. Technol. Innov*, vol. 9, no. 3, pp. 182–195, 2019.

[17] Z. Masoumi, J. Van Genderen, and A. Sadeghi Niaraki, "An improved ant colony optimization-based algorithm for user-centric multi-objective path planning for ubiquitous environments," *Geocarto International*, vol. 36, no. 2, pp. 137–154, 2021.

[18] R. S. Pol and M. Murugan, "A review on indoor human aware autonomous mobile robot navigation through a dynamic environment survey of different path planning algorithm and methods," in *Proceedings of the 2015 International Conference on Industrial Instrumentation and Control (ICIC)*, pp. 1339–1344, IEEE, Pune, India, May 2015.

[19] I. Noreen, A. Khan, and Z. Habib, "Optimal path planning using RRT based approaches: a survey and future directions," *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 11, pp. 97–107, 2016.

[20] L. Tai and M. Liu, "A robot exploration strategy based on q-learning network," in *Proceedings of the 2016 IEEE international conference on real-time computing and robotics (RCAR)*, pp. 57–62, IEEE, Angkor Wat, Cambodia, June 2016.

[21] J. Chen, T. Bai, X. Huang, J. Yang, X. Guo, and Y. Yao, "Double-task deep q-learning with multiple views," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 1050–1058, Venice, Italy, October 2017.

[22] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates," in *Proceedings of the 2017 IEEE international conference on robotics and automation (ICRA)*, pp. 3389–3396, IEEE, Marina sands bay, Singapore, May 2017.

[23] A. Yahya, A. Li, M. Kalakrishnan, Y. Chebotar, and S. Levine, "Collective robot reinforcement learning with distributed asynchronous guided policy search," in *Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 79–86, IEEE, Vancouver, BC, Canada, September 2017.

[24] T. Tai, S. Li, and M. Liu, "A deep-network solution towards model less obstacle avoidance," in *Proceedings of the International Conference on Intelligent Robots and Systems(IROS)*, pp. 2759–2764, Daejeon, Korea (South), October 2016.

[25] X. Yu, P. Wang, and Z. Zhang, "Learning-based end-to-end path planning for Lunar Rovers with safety Constraints," *Sensors*, vol. 21, no. 3, pp. 796–804, 2021.

[26] Y. Zhao, X. Wang, R. Wang, Y. Yang, and F. Lv, "Path planning for mobile robots based on TPR-DDPG," in *Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, IEEE, Shenzhen, China, July 2021.

[27] X. Xue, Z. Li, D. Zhang, and Y. Yan, "A deep reinforcement learning method for mobile robot collision avoidance based on double DQN," in *Proceedings of the 2019 IEEE 28th International Symposium on Industrial Electronics (ISIE)*, pp. 2131–2136, IEEE, Vancouver, Canada, June 2019.