Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 1859020, 10 pages
https://doi.org/10.1155/2022/1859020

Research Article

Reinforcement Learning-Based Path Planning Algorithm for Mobile Robots

ZiXuan Liu,[1] **Qingchuan Wang**,[2] **and Bingsong Yang**[3]

[1]*School of Electrical and Information Engineering, Tianjin University, Tianjin 300072, China*
[2]*School of Electronic Information and Artificial Intelligence, Leshan Normal University, Leshan, 614000 Sichuan, China*
[3]*Service Robot Development Department, Infore Robotics & Automation Co., Ltd., Shenzhen, 518000 Guangdong, China*

Correspondence should be addressed to ZiXuan Liu; liuzixuan19960215@tju.edu.cn

Received 27 March 2022; Revised 6 April 2022; Accepted 13 April 2022; Published 19 May 2022

Academic Editor: Kalidoss Rajakani

A robot path planning algorithm based on reinforcement learning is proposed. The algorithm discretizes the information of obstacles around the mobile robot and the direction information of target points obtained by LiDAR into finite states, then reasonably designs the number of environment model and state space, and designs a continuous reward function, so that each action of the robot can be rewarded accordingly, which improves the algorithm and improves the training efficiency of the algorithm. Finally, the agent training simulation environment is built on gazebo, and the training results verify the effectiveness of the algorithm. At the same time, the navigation experiment is carried out on the actual robot. The experimental results show that the algorithm can also complete the navigation task in the real environment.

1. Introduction

In autonomous robot navigation, path planning is a very important part. The robot path planning problem can be described as finding an optimal path from the current point to a specified target point in the robot's working environment based on one or more optimization objectives, such as minimizing the work cost, minimizing the trajectory length, and minimizing the movement time, given that the robot's own pose is known [1]. According to the understanding of the working environment, the robot path planning algorithm can be divided into global path planning and local path planning: (a) global path planning: when the global static environment map is known, the robot searches the collision free path under the static environment according to the specific algorithm; (b) local path planning mainly considers the obstacle avoidance ability of the robot in the dynamic environment. The robot only knows part of the environmental information, or if the robot only knows part of the environment or does not know the environmental information at all, it will constantly update the environmental information according to the information obtained by the sensor [2, 3]. The local path planning algorithm considers the motion parameters, path direction, obstacles, and other information of the robot itself. Therefore, the local path planning problem has gradually become a research hotspot in recent years [4].

The common algorithms for local path planning are artificial potential field method, vector histogram method, genetic algorithm, fuzzy logic method, reinforcement learning method, etc. [5]. The reinforcement learning algorithm is a learning method that does not require any a priori knowledge of the environment or the robot itself, in which the robot acts while perceiving the current state of the environment [6]. The robot decides on the next action based on the reward information.

The robot includes a variety of technologies, but path planning for mobile robots is one of the core technologies. The path planning of mobile robots can be broadly classified into two categories: global path planning and local path planning [7]. Global path planning was aimed at enabling mobile robots to understand the environment, which can identify the starting point and end point as well as obstacles in the environment model and plan a collision free path

from the starting point to the end point using a variety of different methods, and the widely used global path planning methods are topology method, visual graph method, free space method, raster method, etc. [8].

Local path planning is aimed at mobile robots that do not understand the environment and do not use environmental modeling technology to plan collision free paths from the starting point to the end point [9]. The mobile robot only needs to use some of its own sensors to obtain the shape, size, and distance between it and the obstacles and then carry out path planning through the algorithm. The mobile robot moves according to the planning results to achieve the effect of avoiding collision from the starting point to the end point. The earliest path planning methods include fuzzy logic algorithm, artificial potential field method, and annealing algorithm [10]. For the local path planning method, the core technology is the algorithm. The development direction of the algorithm is closer and closer to bionics, such as neural network algorithm. In recent years, it has a very wide range of applications and is the representative of bionic algorithm. The more mature bionic path planning algorithm and ant colony algorithm, genetic algorithm and reinforcement learning Q-learning algorithm [11].

Nowadays, reinforcement learning Q-learning algorithm is widely used in the path planning of mobile robots. It does not need any prior empirical knowledge or understanding of the environment. It only needs to adjust the action through trial and error in the unknown environment and learn independently to achieve the goal of path planning [12]. Nowadays, it is gradually becoming the main technology of mobile robot path planning.

Although the advantage of Q-learning is that path planning does not require prior knowledge, it still has the disadvantages of slow convergence and poor generalization ability. Therefore, Q-learning has begun to develop in combination with other methods [13].

In this paper, a path planning algorithm of mobile robot based on reinforcement learning is proposed. The number of environment model and state space is designed reasonably by using discrete LiDAR information. We study the continuous payment function to speed up the convergence of the algorithm. Finally, we conduct simulation learning training in the gazebo environment and conduct navigation experiments on real robots to verify the effectiveness of the algorithm [14].

2. Related Work

Reinforcement learning was first combined with fuzzy logic to achieve path planning for mobile robots, proposed by [15], by establishing fuzzy rules and introducing fuzzy controllers to achieve reinforcement learning; this combination can speed up the convergence of single use Q-learning, but the method of setting the degree of fuzzification in fuzzy logic methods can affect the effectiveness of mobile robots to complete path planning. [16] used the artificial potential field method to start using in combination with reinforcement learning, aiming to set the potential at each point of the robot's movement in the environmental space, where the target point attracts and rewards the mobile robot and the obstacle repels and negatively rewards the mobile robot, through this strategy to make the mobile robot complete the path planning task; the disadvantage of this method is that it requires a large amount of a priori knowledge, but this approach tends to fall into the problem of local optimality. In [17], the environment model is defined as three categories, i.e., the quadrant around the robot where the target point is located, the quadrant around the robot where the nearest obstacle is located, and the magnitude of the angle between the robot and the obstacle line and the robot and the target point line, and the robot is classified into success, failure, safety, and insecurity states according to the distance between the robot and the obstacle target point, and the payoff function is defined by the transfer of states. The disadvantage of this method is that only the nearest obstacle information is considered, and the learning ability of the algorithm for simple scenes is strong, and the adaptation ability of complex scenes is weak. [18] introduced neural network fitting Q_table to improve the convergence of the algorithm, but its distribution of surrounding obstacles is not clearly delineated, and it is easy to fall into the local minimum state. [19] proposed a mobile robot obstacle avoidance algorithm based on the combination of reinforcement learning and neural network; the state is the obstacle information obtained by the sensor in front of the robot, the state and action are input into the network, the output is its corresponding Q value, and the action with the largest Q value is selected for execution; the algorithm can complete the obstacle avoidance task better, but the path planning cannot be completed because of the lack of information about the target point.

The combination of reinforcement learning and neural network algorithms is widely applied to the path planning problem of mobile robots, as proposed by [20], which divides the path planning of mobile robots into two processes: the first process is to collect and train state-action value pairs until the Q -learning algorithm converges, and the second process is to train the neural network using the collected state-action value pairs as samples. In this way, when the training is sufficient, the internal structure of the neural network is determined and the neural network outputs a new action when a new state of the mobile robot appears, but at this time, the data source of the neural network still belongs to manual collection, and its collection process is very time-consuming and labor-intensive. This problem was solved by [11], which used the data generated by the mobile robot during its motion as the samples of the neural network, and the mobile robot realized that training and collecting samples were carried out simultaneously in the self-learning process, which effectively improved the efficiency of completing path planning and also reduced the problem of too few training samples of the neural network; since then, the method of reinforcement learning combined with neural network began to mature and gradually became the main technique for path planning of mobile robots.

3. Local Path Planning Method Design

3.1. Basic Principle of Reinforcement Learning Algorithm. Q -learning is a typical model free reinforcement learning

algorithm, which is characterized by the fact that the Q values of the actions taken in each state can be stored in the form of Q_table, making it easy to obtain the Q values of each state and action. It is an iterative update learning method, which is a model free learning algorithm that approximates the objective function 11 by the value function Q without any prior knowledge of the environment. Q^* The update iteration of Q-learning takes the form

$$Q(s,a) = Q(s,a) + \alpha\left(r(s,a) + \gamma \max_{a' \in A} Q\left(s', a'\right) - Q(s,a)\right), \tag{1}$$

where r is the payoff when taking action a under state s, s' is the next state, a is the learning rate, which represents the degree of learning new knowledge and is taken between 0 and 1, and γ is the discount rate, which represents the degree of considering future payoffs and is taken between 0 and 1.

3.2. Motion State Space Design. The environment model of the robot should take into account both the surrounding obstacle information and the location of the target point to avoid collision. Since the continuous high-dimensional state space makes it difficult for the reinforcement learning algorithm to converge, i.e., the number of states and actions is very large, so it is necessary to discretize the robot environment state space. The environment model consists of the robot, target points, and obstacles, and the states are defined as

$$s = \left(R_g, R_{01}, R_{02}, R_{03}, R_{04}\right). \tag{2}$$

In this equation, R_g describes the orientation of the target point to be reached by the robot, and $R_{01} \sim R_{04}$ describes the distribution of obstacles in front of the robot.

Since the differential robot can only move in the direction of the robot's heading angle, the angle θ between the line of the navigation target point and the robot and the robot's forward direction is important for the robot path planning, which can control the robot to adjust the direction towards the target point continuously, so θ is discrete into seven states according to a certain angle (see Figure 1); i.e., the states of the location of the target point around the robot R_g are defined as follows:

$$R_g = \begin{cases} 1 & (-25^\circ < \theta \le 25^\circ), \\ 2 & (25^\circ < \theta \le 80^\circ), \\ 3 & (80^\circ < \theta \le 130^\circ), \\ 4 & (130^\circ < \theta \le 180^\circ), \\ 5 & (-180^\circ < \theta \le -130^\circ), \\ 6 & (-130^\circ < \theta \le -80^\circ), \\ 7 & (-80^\circ < \theta \le -25^\circ). \end{cases} \tag{3}$$

The distribution of obstacles in the range of 120° in front of the robot is discrete into four states according to the angle

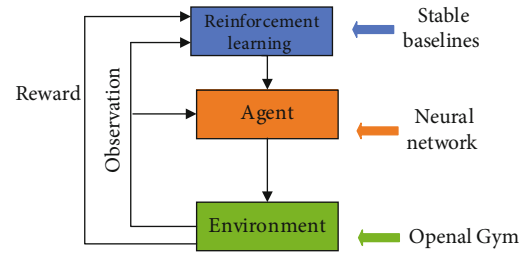


Figure 1: Inspection path planning implementation method.

and distance to characterize the distribution of obstacles, and the direction of the robot is 0°, and the following four states are discrete continuous space with 0.5 m resolution. The robot is in state R_{01} when the angle between the robot and the obstacle line is [20°, 60°], which is defined as follows:

$$R_{01} = \begin{cases} 1 & \left(0.0 < d_{r\_0} \le 0.5\right), \\ 2 & \left(0.5 < d_{r\_0} \le 1.0\right), \\ 3 & \left(1.0 < d_{r\_} \le 1.5\right), \\ 4 & \left(1.5 < d_{r\_0}\right), \end{cases} \tag{4}$$

where d_{r_0} is the distance between the robot and the nearest obstacle in the area of the angle. The states R_{02}, R_{03}, and R_{04} are defined similarly to R_{01} when the robot is at [-20°, 0°), [0°, 20°), and [-60°, -20°) from the obstacle line. The total number of input states is $7 \times 4 \times 4 \times 4 \times 4 = 1792$.

3.3. Motion State Space Design. The velocity sampling points in the vector space are mainly divided into three action sets, forward, left turn, and right turn, so that the designed action is more in line with the robot kinematic characteristics than other algorithms that fix the movement speed and only output the steering angle, making the robot movement more flexible, mainly including three parts: the forward velocity (linear velocity v and angular velocity ω) is (0.3, 0.0), the left turn velocity is (0.1, -0.6), and the speed of the right turn is (0.1, 0.6).

The actual selection of the action process also takes into account the robot motion characteristics, motor load capacity, and other factors and sets a dynamic window in which the action is selected to ensure the executability of the output action, continuity of motion, etc. According to the load capacity of the motor, there is a limit to the maximum acceleration and deceleration of the robot, so the actual speed that the robot can reach in the control cycle is the dynamic window, and the dynamic window V_d is defined as follows:

$$V_d = (V, \omega) | V \in [v_c - \dot{v}_a \Delta t, v_c + \dot{v}_a \Delta t] \omega \in [\omega_c - \dot{\omega}_a \Delta t, \omega_c + \dot{\omega}_a \Delta t]\}, \tag{5}$$

where V is the range of linear velocity in one sampling period of the robot, ω is the range of angular velocity in one sampling period of the robot, v_c is the current linear velocity of the robot, ω_c is the current angular velocity of the robot, $\dot{v}_a$ is the maximum linear acceleration of the

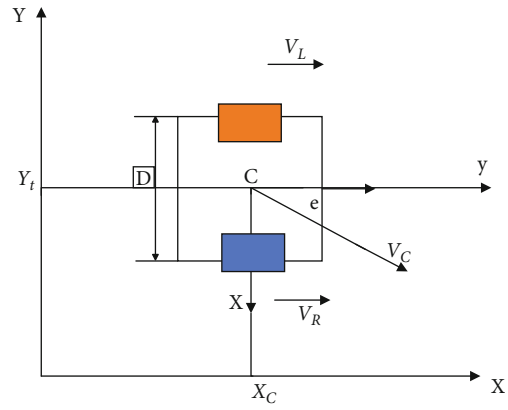


Figure 2: Kinematic model of the inspection robot.

robot, $\dot{\omega}_a$ is the maximum angular acceleration of the robot, and Δt is the sampling period of the robot. Considering the robot motion with dynamic window constraint, the feasible speed can be selected according to the robot's current speed value, with the maximum acceleration and deceleration limit that the motor can withstand.

3.4. Payoff Function Design. The payoff function is an estimate of the action taken by the robot in its state, indicating how good or bad the action is in a given state. If the payoff function is continuous, i.e., the payoff value exists at all times during the training process, the algorithm can effectively use this information for continuous learning. Therefore, in this study, a continuous reward function is designed, i.e.,

$$r(s_t, a_t) = \begin{cases} r_{\text{rea}} & (d_{r-t}(t) \le d_{\text{win}}), \\ r_{\text{col}} & (d_{r-o}(t) \le d_{\text{col}}), \\ c_t(d_{r-t}(t-1) - d_{r-t}(t)) + \eta_0 \dfrac{d_{r-0}(t)}{d_{r-o}(t-1)}, \end{cases} \tag{6}$$

where r_{rea} is the payoff value obtained by the robot to reach the target point, r_{col} is the value of the collision payoff of the robot, η_0 and c_t are coefficients, $d_{r-t}(t-1)$ is the distance between the robot and the target point at t-1, $d_{r-t}(t-1)$ is the distance between the robot and the target point at t, $d_{r-t}(t)$ is the distance between the robot and the nearest obstacle at t, d_{win} is the threshold value for the robot to reach the target point, and d_{col} is the threshold value for the robot to collide with the obstacle.

If the robot is less than a certain threshold d_{win} from the target point, a positive payoff r_{rea} is given; if the robot is less than a certain threshold d_{col} from the obstacle, a negative payoff d_{col} is given; otherwise, the payoff value is the change in the environment caused by the robot taking the previous action and the change in the distance between the target point; if the action reduces the distance to the target point, a positive payoff is given, otherwise a negative payoff. The design is to make the robot keep moving toward the target point, so that the robot can get feedback in time for each action it takes to ensure the continuity of the payoff function and speed up the convergence of the algorithm.

4. Implementation Method of Inspection Path Planning Based on Reinforcement Learning

The method relies only on the distance information acquired by the solid-state LiDAR, the distance between the robot and the detection point and the robot's action history, uses a neural network to make decisions about the angular velocities of the left and right wheels of the inspection robot, constructs an input-output mapping of the neural network, and designs a continuous reward function to evaluate each action of the robot, as shown in Figure 1, and uses the stable baselines. The proximal policy optimization provided by baselines changes the weights inside the neural network according to the high evaluation, inducing the robot to achieve the goals of detection point traversal and obstacle avoidance simultaneously. Finally, a simulation environment is established in OpenAI Gym to verify the effectiveness of the algorithm, which proves that the method in this paper can accomplish the goals of traversing detection points and being collision free throughout the process for the mobile robot.

The inspection robot first explores aimlessly in a finite unknown space, where there are several detection points. The detection points will be designed as signal transmitters in the simulation experiment, which can emit a range signal of a certain radius, and the robot will no longer maintain interest in that point [21, 22].

In order to provide an accurate description of the robot's motion and to achieve motion control of the robot, the inspection robot should first be analyzed from a kinematic point of view. Figure 2 shows the kinematic model of the robot.

The motion control of the inspection robot is mainly achieved by controlling the speed difference between the left and right wheels. The speed difference causes friction, and the frictional side slip caused by the inspection robot in the process of motion is ignored in this paper. In Figure 2, the projection coordinate of the center of mass C of the inspection robot in the XOY plane is (X_C, Y_C). θ is the attitude angle of the inspection robot (indicating the angle between the y-axis of the inspection robot coordinate system and the direction of robot motion, and the direction of inspection robot motion is the same as the direction of the velocity of the center of mass, which is represented by V_C) [23, 24]. Let the width of the body of the inspection robot be$D$, and the angular velocity of the center of mass$\omega_c$is denoted; V_L, R_L is the actual velocity of the left and right wheels of the body, respectively. According to Figure 2, the kinematic equation of the robot can be obtained as

$$\begin{cases} \dot{Xc} = V_c \cos\theta, \\ \dot{Y}_c = V_c \sin\theta, \\ \theta = \omega_c, \end{cases} \tag{7}$$

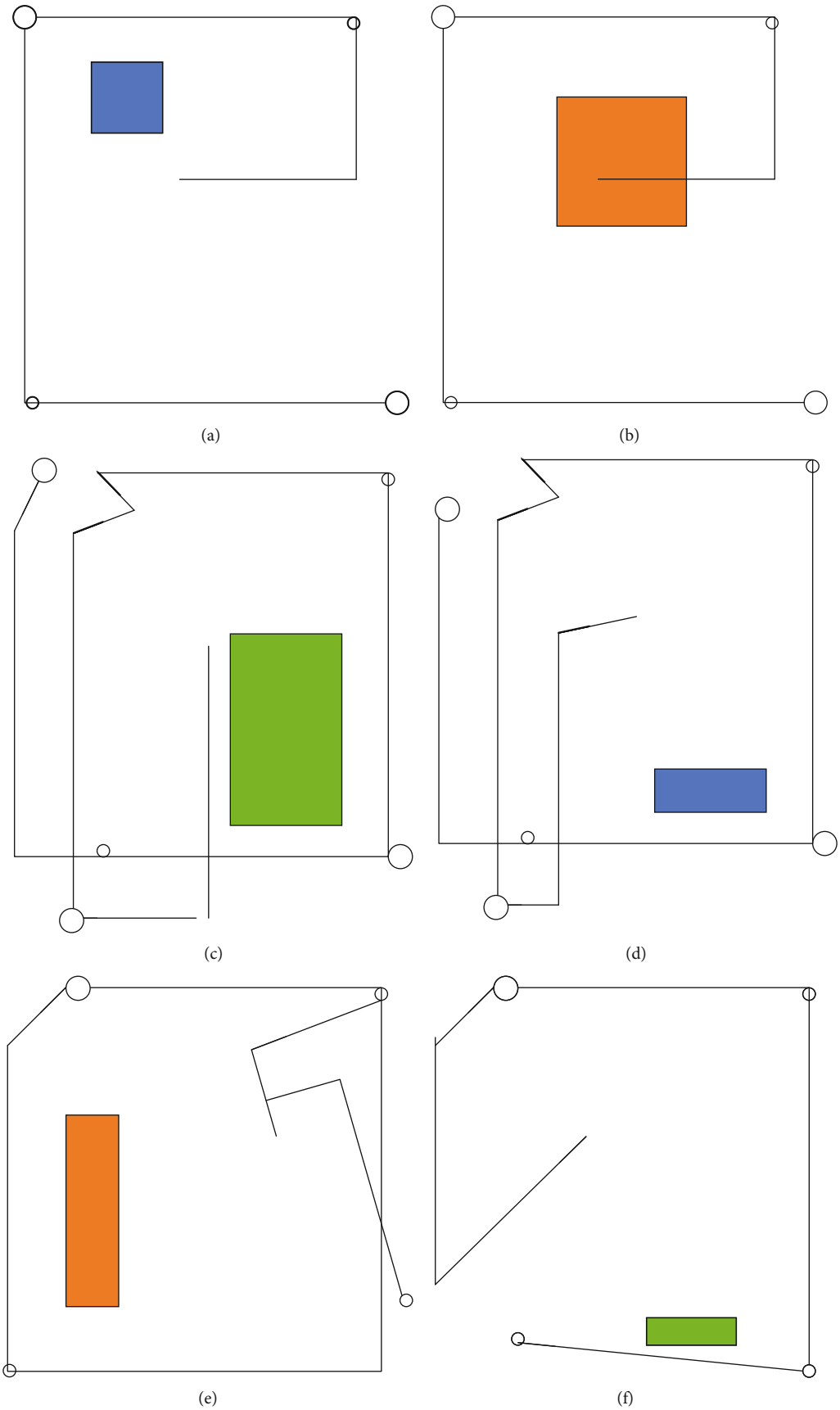
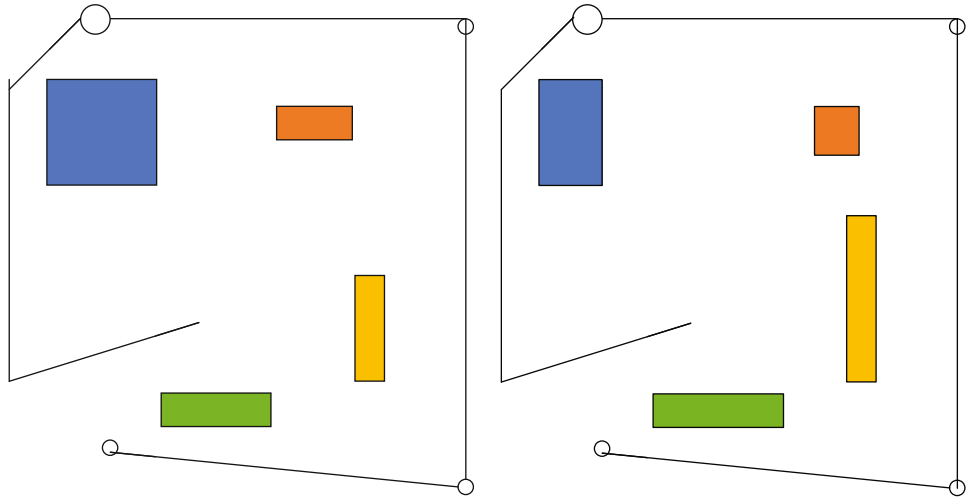


Figure 3: Simulation comparison chart of changing obstacles.



(a) (b)

Figure 4: Simulation comparison of changing multiple obstacles.

$$\begin{cases} V_c = \dfrac{V_L + V_R}{2}, \\ \omega_c = \dfrac{V_R - V_L}{D}. \end{cases} \tag{8}$$

The kinematic equation of the inspection robot is obtained as

$$\begin{bmatrix} X_c \\ \omega_c \\ Y_c \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\cos\theta & \frac{1}{2}\cos\theta \\ \frac{1}{2}\sin\theta & \frac{1}{2}\sin\theta \\ -\frac{1}{D} & \frac{1}{D} \end{bmatrix} \begin{bmatrix} V_L \\ V_R \end{bmatrix}. \tag{9}$$

From the equation, it can be seen that to change the robot's attitude or direction of travel, it is only necessary to change the left and right wheel speeds V_L, R_L of the robot. The velocity vector S of the robot with respect to the linear and angular velocities can be written as

$$S = \begin{bmatrix} V_c \\ \omega_c \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ -\frac{1}{D} & \frac{1}{D} \end{bmatrix} \begin{bmatrix} V_R \\ V_L \end{bmatrix}. \tag{10}$$

When $V_L = V_R$, the angular velocity of the center of mass of the inspection robot is 0, and then, the inspection robot does linear motion with speed V_L or V_R; when $V_L = -V_R$, the linear velocity of the center of mass of the inspection robot is 0, and then, the robot can rotate in situ around C [25, 26]. The rotation speed is determined by the wheel speed and wheel distance of the inspection robot; when $V_L \neq V_R$ and $V_L \neq -V_R$, the robot does circular motion with a certain radius of curvature, and the radius of curvature is calculated as follows.

$$\rho = \frac{V_c}{\omega_c} = \frac{D(V_R + V_L)}{2(V_R - V_L)}. \tag{11}$$

The mobile robot uses an optical encoder to detect the vehicle position and orientation. The motion control system detects the vehicle position and orientation information on a cycle basis, so the obtained robot motion information is the motion information within one sampling cycle.

5. Experimental Results

5.1. Analysis of the Generalizability of the Changing Obstacle Algorithm. In the training 4×10^6 times, the robot achieved the goal of traversing the detection point and obstacle avoidance, for the road blockage and traffic changes caused by the sudden situation in the substation; now, randomly change the number, size, and shape of obstacles in the simulation environment, fix the location of the detection point, fix the starting coordinates of the inspection robot to (5,5) [27], and carry out the adaptive test of whether the inspection robot can complete the task goal of traversing the detection point without touching; six groups of path planning simulation results are shown in Figure 3.

The number of obstacles is increased for adaptive testing, and the results are shown in Figures 4 and 5.

The simulation results show that the robot still has the ability to traverse the detection points when the obstacles change.

It proves that the method proposed in this paper can help the inspection robot cope with the traffic changes caused by the unexpected situation in the substation.

5.2. Analysis of the Generalizability of the Changing Detection Point Algorithm. In order to further explore the robot's adaptive capability, the number and location of detection points of the mobile robot are changed randomly, the size and location of obstacles are fixed [28, 29], and the

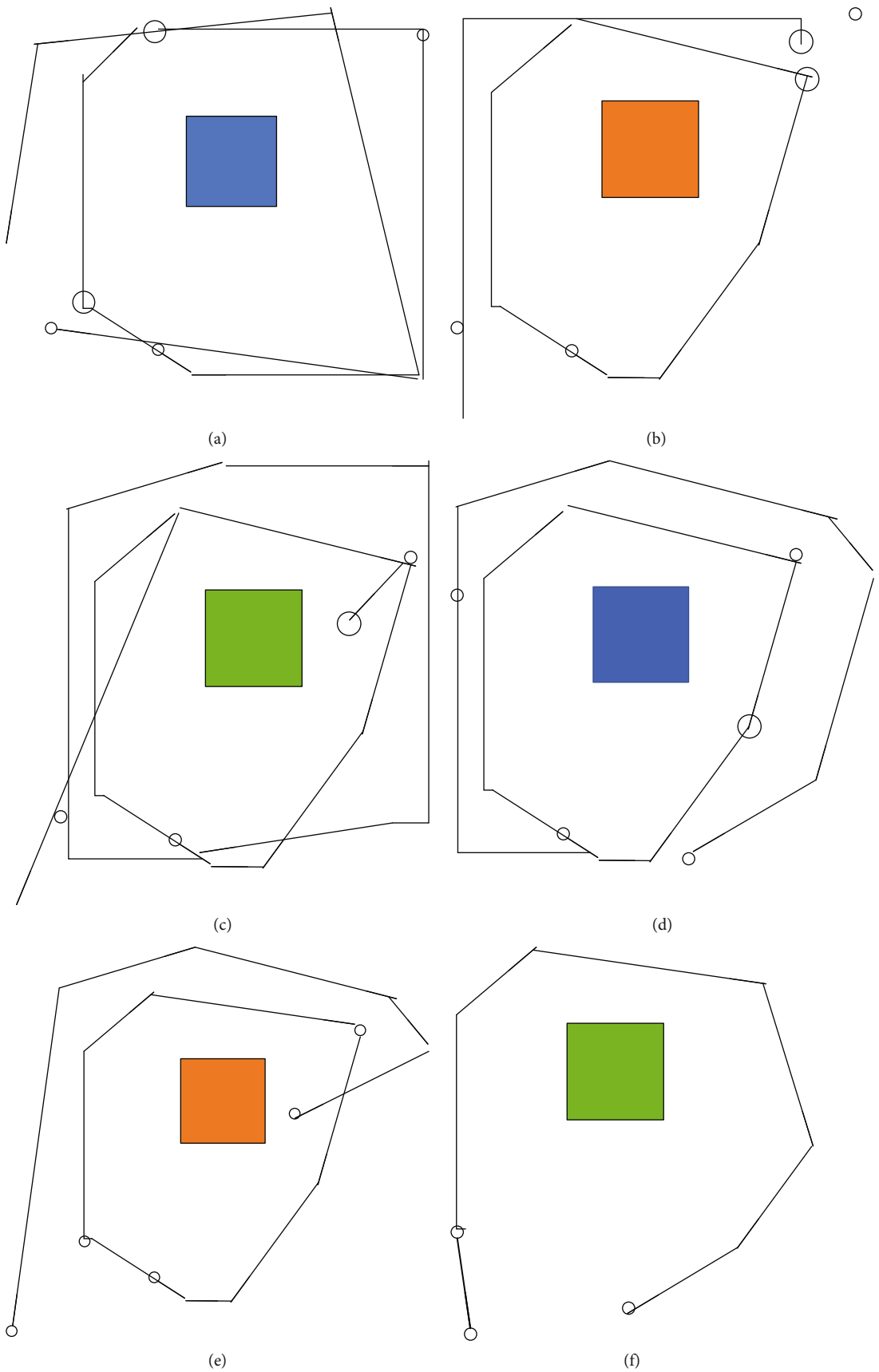


Figure 5: Simulation comparison of changing the number and position of detection points.

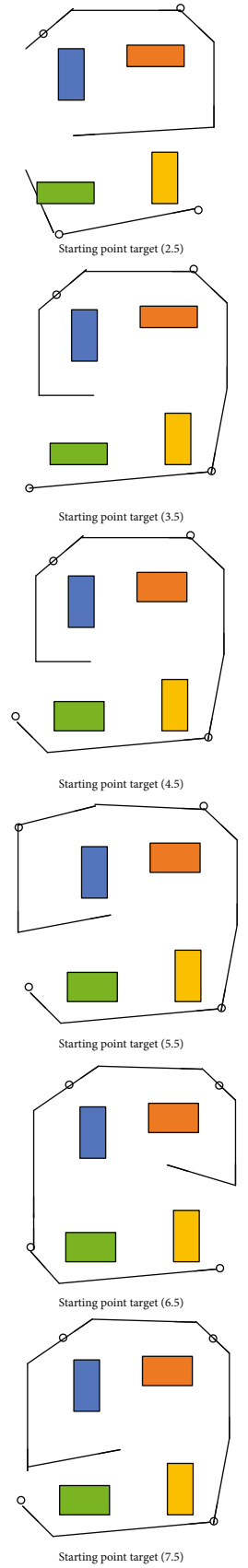


Figure 6: Simulation comparison chart of changing starting point algorithm.

starting coordinates of the inspection robot are fixed to (1,1) to test whether the inspection robot can still accomplish the inspection goal of traversing the detection points without touching.

The simulation results show that the inspection robot starts inspection at the same starting point, and no matter how the detection points change, the mobile robot can still complete the goal of traversing the detection points without understanding the environment and without collision throughout the whole process through self-learning.

It proves that the method of this paper can make the inspection robot cope with the changing situation of the inspection points in the substation, and even if the inspection robot is applied to a new substation, it can achieve the goal of traversing the inspection points without collision through self-learning.

5.3. Generalization Analysis of the Changing Starting Point Algorithm. The above paper explored the effect of the inspection robot traversing the inspection points under the change of obstacles and the location and number of inspection points, and the simulation test results proved that the ability of the inspection robot traversing the inspection points is still unaffected under the change of environment, and now, we continue the simulation experiment of changing the starting point of the inspection robot to explore the generalization of this paper's algorithm, and the simulation test results are as follows.

The simulation results show that the change of the starting point of the inspection robot does not affect the goal of the inspection robot to complete the traversal of the inspection points, and the method of this paper has the self-adaptive ability for the change of the starting point shown in Figure 6.

In summary, the simulation analysis proves that the method of this paper can make the mobile robot face the upgrade change of the substation and still can effectively complete the traversal of the inspection point and the requirement of no collision throughout.

6. Conclusion

The work of the robot needs to cross the detection point and always keep collision free, which depends on effective and reasonable path planning. In this paper, a method without environmental modeling is proposed, which uses reinforcement learning control strategy to make the inspection robot complete the path planning tasks of obstacle avoidance and crossing detection points without collision in the whole process when the substation scene changes, and provides detailed reward and punishment judgment. The convergence and generalization of reinforcement learning are analyzed on the OpenAI Gym platform. The convergence analysis shows that with the increase of reinforcement learning times, the algorithm proposed in this paper can enable the inspection robot to achieve the task goal of collision free passing through the detection point.

Data Availability

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

Conflicts of Interest

The authors declared that they have no conflicts of interest regarding this work.

References

[1] Y. Wang, Y. Fang, P. Lou, J. Yan, and N. Liu, "Deep reinforcement learning based path planning for mobile robot in unknown environment," in *Journal of Physics: Conference Series, Volume 1576, 4th International Conference on Artificial Intelligence, Automation and Control Technologies (AIACT 2020)*, Hangzhou, China, 2020.

[2] L. Kästner, C. Marx, and J. Lambrecht, "Deep-reinforcement-learning-based semantic navigation of mobile robots in dynamic environments," in *2020 IEEE 16th International Conference on Automation Science and Engineering (CASE)*, pp. 1110–1115, Hong Kong, China, 2020.

[3] S. I. A. Meerza, M. Islam, and M. M. Uzzal, "Q-learning based particle swarm optimization algorithm for optimal path planning of swarm of mobile robots," in *2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT)*, pp. 1–5, Dhaka, Bangladesh, 2019.

[4] H. Çetin and A. Durdu, "Path planning of mobile robots with Q-learning," in *2014 22nd Signal Processing and Communications Applications Conference (SIU)*, pp. 2162–2165, Trabzon, Turkey, 2014.

[5] L. Tai, G. Paolo, and M. Liu, "Virtual-to-real deep reinforcement learning: continuous control of mobile robots for mapless navigation," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 31–36, Vancouver, BC, Canada, 2017.

[6] B. Wang, Z. Liu, Q. Li, and A. Prorok, "Mobile robot path planning in dynamic environments through globally guided reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6932–6939, 2020.

[7] H. Xu, N. Wang, H. Zhao, and Z. Zheng, "Deep reinforcement learning-based path planning of underactuated surface vessels," *Cyber-Physical Systems*, vol. 5, no. 1, pp. 1–17, 2019.

[8] S. M. Sombolestan, A. Rasooli, and S. Khodaygan, "Optimal path-planning for mobile robots to find a hidden target in an unknown environment based on machine learning," *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, no. 5, pp. 1841–1850, 2019.

[9] K. Zhu and T. Zhang, "Deep reinforcement learning based mobile robot navigation: a review," *Tsinghua Science and Technology*, vol. 26, no. 5, pp. 674–691, 2021.

[10] X. Ma, Y. Xu, G. Q. Sun, L. X. Deng, and Y. B. Li, "State-chain sequential feedback reinforcement learning for path planning of autonomous mobile robots," *Journal of Zhejiang University Science C*, vol. 14, no. 3, pp. 167–178, 2013.

[11] K. G. S. Apuroop, A. V. Le, M. R. Elara, and B. J. Sheu, "Reinforcement learning-based complete area coverage path planning for a modified hTrihex robot," *Sensors*, vol. 21, no. 4, p. 1067, 2021.

[12] P. Gao, Z. Liu, Z. Wu, and D. Wang, "A global path planning algorithm for robots using reinforcement learning," in *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 1693–1698, Dali, China, 2019.

[13] L. Tai and M. Liu, "Towards cognitive exploration through deep reinforcement learning for mobile robots," 2016, http://arxiv.org/abs/1610.01733.

[14] G. Ryou, Y. Sim, S. H. Yeon, and S. Seok, "Applying asynchronous deep classification networks and gaming reinforcement learning-based motion planners to mobile robots," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6268–6275, Brisbane, QLD, Australia, 2018.

[15] P. T. Kyaw, A. Paing, T. T. Thu, R. E. Mohan, A. V. Le, and P. Veerajagadheswar, "Coverage path planning for decomposition reconfigurable grid-maps using deep reinforcement learning based travelling salesman problem," *IEEE Access*, vol. 8, pp. 225945–225956, 2020.

[16] Z. H. Zhengwan, Z. H. Chunjiong, L. I. Hongbing, and X. I. Tao, "Multipath transmission selection algorithm based on immune connectivity model," *Journal of Computer Applications*, vol. 40, no. 12, p. 3571, 2020.

[17] T. Li, D. Ho, C. Li, D. Zhu, C. Wang, and M. Q. H. Meng, "Houseexpo: a large-scale 2d indoor layout dataset for learning-based algorithms on mobile robots," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5839–5846, Las Vegas, NV, USA, 2020.

[18] W. Duan, J. Gu, M. Wen, G. Zhang, Y. Ji, and S. Mumtaz, "Emerging technologies for 5G-IoV networks: applications, trends and opportunities," *IEEE Network*, vol. 34, no. 5, pp. 283–289, 2020.

[19] A. Radwan, K. M. S. Huq, S. Mumtaz, K. F. Tsang, and J. Rodriguez, "Low-cost on-demand C-RAN based mobile small-cells," *IEEE Access 4*, pp. 2331–2339, 2016.

[20] N. A. Khan, O. I. Khalaf, C. A. T. Romero, M. Sulaiman, and M. A. Bakar, "Application of intelligent paradigm through neural networks for numerical solution of multiorder fractional differential equations," *Computational Intelligence and Neuroscience*, vol. 2022, Article ID 2710576, 16 pages, 2022.

[21] H. S. Gill, O. I. Khalaf, Y. Alotaibi, S. Alghamdi, and F. Alassery, "Fruit image classification using deep learning," *CMC-Computers, Materials & Continua*, vol. 71, no. 3, pp. 5135–5150, 2022.

[22] P. An, Z. Wang, and C. Zhang, "Ensemble unsupervised autoencoders and Gaussian mixture model for cyberattack detection," *Information Processing & Management*, vol. 59, no. 2, article 102844, 2022.

[23] A. Revathy, C. S. Boopathi, O. I. Khalaf, and C. A. T. Romero, "Investigation of AlGaN channel HEMTs on β-Ga2O3 substrate for high-power electronics," *Electronics*, vol. 11, no. 2, p. 225, 2022.

[24] G. Cai, Y. Fang, J. Wen, S. Mumtaz, Y. Song, and V. Frascolla, "Multi-carrier M-ary DCSK system with code index modulation: an efficient solution for chaotic communications," *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 6, pp. 1375–1386, 2019.

[25] A. V. Le, P. Veerajagadheswar, P. Thiha Kyaw, M. R. Elara, and N. H. K. Nhan, "Coverage path planning using reinforcement learning-based TSP for hTetran—a polyabolo-inspired self-reconfigurable tiling robot," *Sensors*, vol. 21, no. 8, p. 2577, 2021.

[26] M. Wei, S. Wang, J. Zheng, and D. Chen, "UGV navigation optimization aided by reinforcement learning-based path tracking," *IEEE Access*, vol. 6, pp. 57814–57825, 2018.

[27] C. He, Y. Wan, Y. Gu, and F. L. Lewis, "Integral reinforcement learning-based multi-robot minimum time-energy path planning subject to collision avoidance and unknown environmental disturbances," *IEEE Control Systems Letters*, vol. 5, no. 3, pp. 983–988, 2021.

[28] Z. W. Zhang, D. Wu, and C. J. Zhang, "Study of cellular traffic prediction based on multi-channel sparse LSTM," *Computer Science*, vol. 48, no. 6, pp. 296–300, 2021.

[29] B. Zuo, J. Chen, L. Wang, and Y. Wang, "A reinforcement learning based robotic navigation system," in *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 3452–3457, San Diego, CA, USA, 2014.