Hindawi
Mobile Information Systems
Volume 2022, Article ID 3386116, 11 pages
https://doi.org/10.1155/2022/3386116

Research Article

Path Planning of Industrial Wheeled Robots Based on Wireless Communication and Machine Learning Algorithms

Jingmin Li

Experimental Training Management Center, School of Jilin Business and Technology College, Changchun 130507, China

Correspondence should be addressed to Jingmin Li; 120212202034@ncepu.edu.cn

Received 28 December 2021; Accepted 25 January 2022; Published 10 March 2022

Academic Editor: Hasan Ali Khattak

With the advancement of science and technology, robotics has made considerable progress. Robots can free humans from heavy repetitive labor. From the industrial field to the lives of the general public, robots are playing an increasingly important role. Path planning is one of the core contents of industrial wheeled robotics and has very important significance. Based on the reinforcement Q learning and BP network, this work studies the path planning of industrial wheeled robots. According to task requirements of path planning, design learning strategies, and control rules, use wireless communication to transmit environmental perception information and propose corresponding control strategies. The main researches are as follows: (1) Based on grid map environment, a path planning algorithm with Q-CM learning is designed. The algorithm first designs robot states and actions based on reinforcement Q learning and grid map and establishes Q matrix. Secondly, a coordinate matching (CM) obstacle avoidance control rule is designed to improve the efficiency of robot avoidance. Then, a reward function is designed for the evaluation problem of action execution. (2) Based on the map environment of free space and the generalization ability of BP neural network, a robot path planning with Q-BP learning is designed. The algorithm first designs the sensor detection mechanism and action selection strategy according to the state of the robot in the map environment. Secondly, a dynamic reward function is designed. Then, according to the obstacle avoidance requirements of special obstacles, the obstacle avoidance rules after three shocks were designed. The experimental results show robots can perform better path planning in a discrete and continuous free space map, and the obstacle avoidance effect is good.

1. Introduction

Industrial wheeled robots gather cutting-edge knowledge in many fields and are currently one of the most active areas of scientific research. With the improvement of robot performance and continuous reduction of cost, the application range for wheeled robots has gradually expanded, and they have been well applied in industries and fields such as industry, agriculture, medical treatment, service, national defense, and space exploration. The key technologies of industrial wheeled robots include navigation technology, path planning, and multisensor fusion. Navigation technology means that when there are obstacles in the environment, the robot can move to the target autonomously based on the information collected from the environment and its own situation. Path planning means that under certain evaluation criteria, the robot can search for an optimal path from starting point to target point, which can be mainly divided into global path planning and local path planning. Multisensor fusion means that the robot comprehensively processes the information collected from multiple sensors to obtain a unified description of the environment information, which can accurately represent the environment. The path planning of the industrial wheeled robot is to plan the optimal path according to a certain algorithm according to the information collected by the sensor when the robot has obstacles in the environment [1–5].

If a robot has access to all of the information in its environment, it can plan its course using the global path planning algorithm. The effect of path planning is greatly influenced by the robot's ability to gather accurate

information about its surroundings. If you are trying to identify the best way for a large group of people, you can use global path planning, but it has certain drawbacks. The configuration space approach, the free space method, and the grid method are all options for global path planning. The configuration space technique is to regard the robot as a point with no shape, expand the obstacle according to the robot's length or width, and then plan the path, using the view method and the Dijkstra method. The free space method entails creating a connected graph representation of the free space and then using a time-consuming graph search method to find a path through it. The grid approach refers to the practice of creating equal-sized grids throughout the environment. As long as the grid does not have any barriers, the robot can easily move through [6–10].

Robot path planning when the environment is unknown or partially unknown is referred to as the local route planning algorithm. In terms of real-time performance and practicality, the local path planning algorithm is superior. Because the robot lacks global information, it is easy to slip into the local extreme point and sometimes miss the target position. Artificial potential field methods, particle swarm algorithms, fuzzy logic algorithms, neural network algorithms, and hybrid methods are among the most popular local path planning techniques. Repulsion and gravitational forces are generated by barriers and target points in the artificial potential field approach. The greater the distance between the robot and the obstacle, the greater the force exerted by the robot. The combined force of the repulsive force propels the robot to its destination. The neural network algorithm uses a specific neural network structure to represent the environment, and on this basis, the energy function is defined or the network output information is used for robot path planning [11–15].

This work mainly studies the local path planning of industrial wheeled robots and uses wireless communication technology to collect and transmit real-time information about the location environment of wheeled robots. Then, the information is processed through the machine learning algorithm, and finally, the optimal path planning is obtained. The contributions of this work are as follows: (1) from the perspective of improving the efficiency of robot avoidance, a Q-CM obstacle avoidance algorithm is proposed. It ensures the robot can quickly avoid surrounding obstacles while reaching the target point smoothly. (2) Combining with the designed Q-BP path planning algorithm, in the operating environment containing special obstacles, a three-oscillation back-off obstacle avoidance algorithm is proposed. This keeps the robot as far away as possible from the deadlock trap and improves the stability of path planning.

2. Related Work

The early research phase for robot path planning technology was mainly based on traditional methods. Literature [16] proposed a new obstacle potential field function model and could adaptively change the weight of this function so that the robot could avoid the local minimum during the optimization process. Literature [17] used the potential gradient descent algorithm to identify a path through the potential field and added repulsion potential in the case of a blocking configuration. At this point, new potential fields would be developed and a final value will be discovered. In literature [18], in the Dijkstra algorithm, it was considered that certain nodes in a given environment would reduce the speed of the robot. Therefore, only the nearest node was considered and the most effective path was found, thereby reducing time consumption and increasing speed. Literature [19] combined tabu search and firefly algorithm, TS algorithm was applied to each small segment, the best segment was recombined to form the best path, and FA optimized this path so that the algorithm was not easy to fall into local optimum and search time was reduced. Literature [20] introduced a regression mechanism in the RRT method and also adopted an adaptive extension mechanism. By refining the boundary nodes, the information of the reachable space was continuously improved, and repeated searches for expansion nodes were avoided.

With the development of intelligent algorithms, it was gradually used in path planning. In literature [21], in the improved ant colony algorithm, the number of pheromones was increased on some short paths in each cycle, and the evaporation rate was dynamically adjusted. Thereby increasing the transition probability, the search speed was obviously improved. Literature [22] proposed a new multiobjective gray wolf optimization to solve path planning. Literature [23] applied the Firefly algorithm to path planning in dynamic space. Tests had proved that this algorithm could successfully plan the required path and was relatively low in terms of length and computational cost. Literature [24] proposed to use cuckoo and bat algorithms together and first used cuckoo to find the optimal solution in a small range. Then, the obtained optimal solution was used as the input of the bat algorithm to obtain the global optimal solution. Then, the hybrid algorithm was used, which could quickly and effectively find a path that meets the requirements. Literature [25] combined a directed acyclic graph with an initial population, generated an initial path based on this graph, and then optimized it. The algorithm could output high-quality paths in a short execution time. Literature [26] proposed an improved genetic algorithm combined with directional guidance factors to bypass obstacles through the direction of motion. Therefore, a shorter path was found, infeasible paths were avoided as much as possible, the search in unnecessary areas was reduced, and the convergence speed was accelerated. Literature [27] redistributed genes according to the rotation angle of the path, thereby optimizing the path, making the path smoother, and turning flexible. In order to solve path planning in dynamic environment, literature [28] combined genetic algorithm and Bezier curve to get a new algorithm, which could react to changes in the environment in real time and dynamically searched for the best path to the target location. Literature [29] used the three-time commutative crossover operator, which could generate more optimal offspring than the traditional two-time commutative crossover operator. At the same time, the dual-path constraint was applied to obtain optimal shortest total path. Literature [30] proposed a new algorithm, which used particle swarm optimization for global planning. Then use the improved

probabilistic roadmap for local planning. The algorithm considered both the shortest and smoothest paths at the same time. The improved algorithm was significantly more efficient than the basic algorithm. Literature [31] provided an optimization algorithm based on particle swarm algorithm and designed an evaluation function with the position of obstacles, and the robot could avoid all possible obstacles and move. Literature [32] designed the introduction of a disturbance global update mechanism at the global optimal position to avoid the stagnation of the algorithm. Literature [33] used the distance of the robot to the obstacle and the target point to construct a new fitness evaluation function, which could successfully avoid the obstacle and move towards the target position.

3. Path Planning Based on Q-CM Learning

This section proposes an industrial wheeled robot path planning algorithm with Q-CM on basis of grid map. The algorithm utilizes a reinforcement Q learning algorithm to learn the best state-action pair, allowing the industrial wheeled robot to autonomously obtain the optimal or suboptimal path. The coordinate variables of obstacles in the grid map are introduced, and the CM algorithm is used for coordinate matching to solve the obstacle avoidance problem during the operation for robot and improve operation efficiency for mobile robot. Adjust the learning rate and discount factor of Q learning to solve the path redundancy problem, and plan the optimal or suboptimal path to improve convergence speed.

3.1. CM Algorithm. The coordinate matching (CM) algorithm is based on a given grid map, defines the coordinates of the robot and each obstacle in the map, and implements the algorithm for robot avoidance applications. First, the coordinates of the obstacles are calibrated according to the position of a single square obstacle in the grid map, and the coordinates of all obstacles form a coordinate set. When the robot performs an action and its sensor detects an obstacle, it transmits position information through wireless communication and compares the current robot's coordinates with the coordinates in the preset coordinate set. If a coordinate pair equal to the robot coordinate can be found in the coordinate set, then the robot is considered to have collided. The execution flow of the CM algorithm is shown in Figure 1.

3.2. Grid Map Construction. Establish a grid map as an environment for path planning of industrial wheeled robots. The actual operating environment of the wheeled robot is quantified as a grid. Each grid represents a coordinate point and is the same size as the wheeled robot. The starting point and the ending point occupy a grid position, respectively. The step size of each step of wheeled robot is a grid. The task is to find a path avoiding obstacles safely through the detection and learning of the environment by using the given starting point and ending point in the established grid map. At the same time, it is necessary to meet the shortest or shorter path of the plan.

Mark the grid map as G_{EN}, it is an unknown bounded static environment, and the size is $G_{EN} = X_{\lim} \times Y_{\lim}$. Remember that the start position and target position of the robot in G_{EN} are G_{start} and G_{end}. There are a finite number of unknown static obstacles $o_1, o_2, \ldots, o_n$ distributed in G_{EN}. Establish a G_{table} table with the same size of G_{EN} as a two-dimensional grid environment table to record the position of obstacles in the grid environment detected. The rows and columns in the G_{table} table, respectively, represent the horizontal and vertical coordinates of the o_i. It is stipulated that each minimum unit obstacle o_i in the grid environment two-dimensional table G_{table} consists of 9 coordinates, and each coordinate is represented by the horizontal and vertical coordinates of the diagonal intersection of a single obstacle o_i and its corresponding form, which is recorded as b. The seat marks that make up obi are s, which is expressed as

$$\begin{aligned} s &= \{b_i(i, j)\}, \\ i &= [x - 1, x, x + 1], \\ j &= [y - 1, y, y + 1]. \end{aligned} \tag{1}$$

All single obstacle coordinate sets s compose all obstacle coordinate sets, represented by op. Mark the coordinates of the obstacle location in G_{EN} according to the following formula:

$$G_{\text{table}} = \begin{cases} 1, & G_{\text{table}}(i, j) = b_n(i, j), \\ 0, & \text{other}, \end{cases} \tag{2}$$

where G_{table} and b_n are mentioned parameters.

3.3. State and Action Design. The wheeled robot starts from the starting point and continuously explores the unknown environment. The path it takes each time it successfully reaches the end point is represented by the trajectory connected by the corresponding coordinate points in G_{EN}. Among them, each coordinate point of the robot in its operating environment represents a state, denoted as s_t, and the coordinate is G_s. According to the dimension of the grid map model, there are a total of $X_{\lim} \times Y_{\lim}$ states, and the state set S composed of all states is

$$S = \{s_t | s_t = G_s(i, j)\}. \tag{3}$$

In each state, the actions that the robot can perform are denoted as a_k. The action can be randomly selected according to the current state. It is stipulated here that the robot can only perform four actions up, down, left, and right, and each action moves 1 coordinate (or 1 grid) position at a time. The action set A is expressed as $A = \{a_1, a_2, a_3, a_4\}$. The selection of the current action affects the representation of the next state, where it is stipulated that the coordinates are converted between the state and the action. Table 1 shows the corresponding state changes when the robot performs 4 actions.

3.4. Q Matrix Establishment. The Q matrix is a finite two-dimensional table utilized to store Q value of the best state-action pair. Its size is determined by the state and the action.

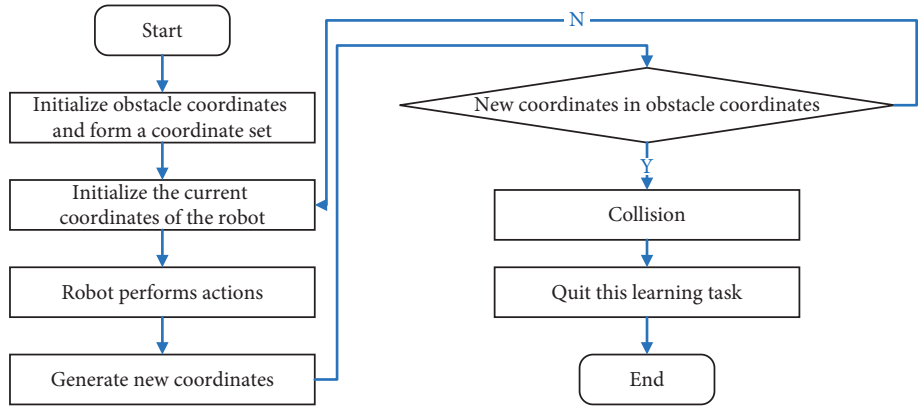


Figure 1: The flowchart of the CM algorithm.

Table 1: The relationship between state and actions.

Set	Relationship			
State set	$s_t + [0, 1]$	$s_t + [-1, 0]$	$s_t + [1, 0]$	$s_t + [0, -1]$
Action set	a_1 (up)	a_2 (left)	a_3 (right)	a_4 (down)

The size of the state is $X_{\lim} \times Y_{\lim}$, and the number of actions is 4. The size of the Q matrix is $X_{\lim} \times Y_{\lim} \times 4$. The Q matrix is expressed as

$$Q = \begin{bmatrix} Q_{s_0 a_1} & \cdots & Q_{s_0 a_4} \\ \cdots & \cdots & \cdots \\ Q_{X_{\lim} \times Y_{\lim} \times a_1} & \cdots & Q_{X_{\lim} \times Y_{\lim} \times a_4} \end{bmatrix}. \quad (4)$$

Each Q value in the Q matrix is the maximum expected reward value obtained when the action a_k is taken to reach the next state in the s_t state during the robot's learning process. When the rate of change of each Q value in the Q matrix is less than the given accuracy, the Q matrix converges and each Q value tends to be stable. Since there may be more than one optimal or suboptimal path, there may be multiple sets of best state-action pairs obtained every time the learning ends.

3.5. Reward Function. The reward function is a judgment criterion used to evaluate the behavior of the wheeled robot to avoid obstacles and tend to target and finally make path that the robot learns and plans safe and shortest. The designed reward function R is represented by the sum of obstacle avoidance reward and punishment function R_1 and goal trend reward and punishment function R_1. Let r be the type of rewards and punishments, used to judge the current state of the robot, and give rewards and punishments based on the state. The return functions are

$$R_1 = \begin{cases} 0, & r = r_1, \\ -100, & r = r_2, \end{cases}$$
$$R_2 = \begin{cases} 10, & r = r_3, \\ -10, & r = r_4, \\ 100, & r = r_5, \end{cases} \quad (5)$$

where r_1 means that no obstacle is encountered, and the reward is 0. r_2 represents collision with obstacles, and the penalty is 100. r_3 means being close to the target and rewarding 10. r_4 means stay away from the target and penalize 10. r_5 means reaching the goal and rewarding 100.

3.6. CM Obstacle Avoidance Design. The wheeled robot needs to avoid obstacles while moving, and adopts the CM obstacle avoidance design method. That is, the coordinates of the robot after each action are matched with the coordinates of the obstacles in the grid map to quickly determine whether there are obstacles in current running direction, and improve its path search efficiency and obstacle avoidance ability. The CM obstacle avoidance method is divided into two parts: obstacle avoidance and boundary obstacle avoidance.

3.6.1. Obstacle Avoidance. Suppose the coordinates of the current robot are $R(x_r, y_r)$, the coordinates of the center point of obstacle o_i are $g(x_o, y_o)$, and the coordinates of the upper right corner are $(x_o + 1, y_o + 1)$. During the exploration and learning process of the robot in the grid environment, after performing action a_4, the robot will collide with the upper right corner of the obstacle o_i. At this time, the coordinates of the robot are $R'(x_r, y_r - 1)$ and match R' with the coordinates of the obstacle o1 to find the coordinates of the collision point. If $x_r = x_o + 1$ and $y_r - 1 = y_o + 1$, it means that the robot has collided with an obstacle after performing action a_4 during the learning process. According to the reward and punishment rules of the reward function, the representation relationship corresponding to the collision is r_2, then $R_1(r_2) = -100$, and the robot is punished during the learning process. When encountering a similar situation next time, the robot will not perform action a_4 but perform other actions.

3.6.2. Border Obstacle Avoidance. The behavior of wheeled robots at the edge of the grid environment is divided into 2 categories and 8 categories, that is, 4 side boundary motion scenarios, left, up, down, and right, and 4 corner boundary motion scenarios, upper left, lower left, lower right, and

upper right. The coordinates of the bottom boundary, left boundary, upper boundary, and right boundary are expressed as $(x, 0)$, $(0, y)$, (x, Y_{lim}), and (X_{lim}, y). When the wheeled robot avoids obstacles according to the CM algorithm, if its coordinate is equal to any boundary coordinate, it will collide with the boundary. According to the reward and punishment rule, $R_1(r_2) = -100$, the wheeled robot will be punished. Through continuous learning, eventually the robot no longer collides with the boundary.

3.7. Q-CM Algorithm Based on Grid Map. Based on the grid map established, the steps of using Q-CM to learn obstacle avoidance and path planning are as follows.

Step 1. Clear the two-dimensional environment table G_{EN} and obstacle coordinate set O; given the robot start point G_{start} and end point G_{end}, the sensor detects and collects obstacle information and utilizes wireless communication to transport information. Establish a short_best linear table to store the historical best state-action pairs. Short_list linear list stores the linear list of the best state-action pairs currently learned from start point to end point. shortbest_size records length of shortest path in history. Initialize the robot to immediately report $R = 0$, the target point return value $R(\text{end}) = 100$, the initial value of the number of learning $L = 0$, and the maximum number of learning N.

Step 2. Initialize the historical shortest path length shortbest_size $= N$, the initial value of the iteration counter *count* $= 0$, and clear the current shortest path record table short_list.

Step 3. The robot judges its state, weakly collides, and uses the Q-CM algorithm to avoid obstacles. Let $s_{t+1} = s_t$ and move to Step 4. Action a_k with the largest Q value in current state s_t is selected for execution. After performing action a_k, the robot turns to state s_{t+1}, count = count + 1, if $G_{\text{start}} = G_{\text{end}}$, go Step 5, otherwise go Step 7.

Step 4. Calculate return value of current environment; give rewards and punishments to the current state of robot, the execution of action; and find value of the return function. If the robot is rewarded, go to step 6. If the robot is punished, it will return to the previous state and go to Step 3.

Step 5. The current state is the target state. Write down the value of the current iteration counter count, the historical shortest path short_best and its length shortbest_size, and the current shortest path short_list, and go to Step 2.

Step 6. Update the enhanced Q value in the current state.

Step 7. If $L < N$, go to Step 2. If $L = N$, the iteration ends, and an optimal or suboptimal collision-free path from starting position to target position is obtained according to short_best.

4. Path Planning with Q-BP Learning

When the design of this section is with Q-BP learning local path planning of a wheeled robot, firstly describe the operating environment of the robot and determine the type of environment. Secondly, given the state space description and action space description of the robot, determine the input and output parameters of the BP neural network. Then establish rules for the obstacle avoidance of the robot so that it can quickly jump out of the shock trap. Finally, the reward function and convergence conditions of Q learning are given to improve the entire algorithm process.

4.1. Environment and State Space Description. The path planning algorithm proposed in this section uses the local path planning method while applying the Q learning algorithm to endow the robot with learning capabilities. First of all, the starting and ending position information required by the industrial wheeled robot is randomly given. Secondly, wheeled robots rely on ultrasonic ranging sensors for environmental monitoring, collect obstacle distance and angle information, and then use wireless communication technology to transmit information and perform calculations, and perform corresponding actions based on the calculation results. Finally, the robot obtains the optimal or suboptimal strategy and plans a shorter path without collision.

The map environment with Q-BP learning path planning algorithm is established using a two-dimensional coordinate system, and the locations of the starting point and the target point in the map environment are represented by two-dimensional coordinates. The number of obstacles and the position of each obstacle are randomly set, and the shape of a single obstacle is defined as a square. The robot can walk at will with the set step length and direction in the map environment, and its relative position in the map determines the behavior strategy to be adopted after obtaining the corresponding environment state information. Based on the given path planning type and the adopted map form, set the relative position information of the robot in the map environment with the end point and obstacles, and describe the state space.

The learning path planning based on Q-BP is different from the way the robot uses CM coordinates to express environmental state information in Section 3. This method focuses on the general orientation and relative distance within the effective detection range of the robot sensor and uses ultrasonic sensors to sense environmental status information. This method can avoid the problem of excessive state space storage caused by coordinate representation. First, the sensor model of the robot is given according to the actual robot sensor situation, including the three sonar sensors on left, center, and right. Angle between sensors is 22.5 degrees. Take center position as coordinate origin, front direction is y-axis, and direction perpendicular to y-axis is x-axis to create a two-dimensional rectangular coordinate system.

The area range robot can perceive and detect is $(30^{\circ}, 150^{\circ})$, and rectangular plane area is divided into three

parts: PA, PB, and PC. The three sensors on the left, center, and right are responsible for detecting obstacles in PC, PB, and PA, respectively. Set maximum distance robot can perceive as $D_{\max}$ and perceptual distance is greater than or equal to $D_{\max}$ to be far away from the obstacle, and less than $D_{\max}$ to be close to the obstacle. Assuming that the minimum sensing distance is $D_{\min}$ and less than $D_{\min}$, it is regarded as a collision. Here, the distance is represented by a parameter, and its value can be set reasonably according to the needs of the experiment. Secondly, the robot needs to use the sensor model to perceive the surrounding environment information, including the distance and direction information between the robot and obstacles and targets. After recognizing the environmental information, the robot accurately performs actions according to its own posture and avoids obstacles, and finally reaches the end to complete the task of learning and path planning.

Suppose the world environment where the robot is located is a two-dimensional coordinate system, and the environment includes an industrial wheeled robot, a target, and several obstacles. The robot determines the state set S based on the above environmental information:

$$S = \left\{ s_t | s_t = \left(d_{ora}, d_{orb}, d_{orc}, d_{rg}, \theta_{rg} \right) \right\}, \tag{6}$$

where the distance that the left sensor detects the obstacle of the robot is set as d_{ora}, the distance that the middle sensor detects the obstacle is defined as d_{orb}, and the distance that the right sensor detects the obstacle is defined as d_{orc}. The distance and direction between robot and target are set as d_{rg} and θ_{rg}, respectively.

4.2. Action Space Description and Action Selection. In order to give the robot the ability to walk freely in map environment, four actions that can be performed are designed, including forward a_1, turn left a_2, turn right a_3, and back a_4. The four actions form the action set $A = \{a_1, a_2, a_3, a_4\}$. The left and right angles are set as $(\pi/6)$ and can be adjusted. The backward behavior a_4 specifies that the robot rotates in situ by an angle of π and then performs forward actions, so no sensor is added at the back.

In order to speed up the robot's approach to the target and avoid obstacles, a state-behavior selection mechanism has been developed. The human experience is added to the action selection mechanism to assist the robot in action selection. The robot will perform actions with reference to its own coordinate system to obtain the state-behavior relationship between obstacles, target points, and actions. The priority of obstacle avoidance is higher than that of approaching targets. The state-behavior selection relationship is illustrated in Table 2.

P indicates that there are no obstacles in the three areas, P_A_O indicates that there are obstacles in the PA area, P_B_O indicates that there are obstacles in the PB area, and P_C_O indicates that there are obstacles in the PC area. P_A_G, P_B_G, and P_C_G, respectively, indicate that the target is in the PA, PB, and PC areas. Refer to the state-behavior selection method in the table, the robot shields a certain action according to some states, and the rest of the actions still implement a random action selection mechanism, which will enable the robot to avoid obstacles accurately and accelerate the speed of reaching the end point. When obstacles appear in PA, PB or PA, PC or PB, and PC at the same time, the robot can still perform actions in combination to avoid obstacles. When an obstacle is detected on left, front, and right, action a_4 will be executed and the obstacle avoidance rule will be established after three shocks.

Table 2: State-behavior selection relationship.

Obstacle	Goal	Forward	Turn left	Turn right	Back
P	P_C_G	a_1	a_2	a_3	a_4
P_C_O	P_C_G	a_1		a_3	a_4
P_B_O	P_C_G		a_2	a_3	a_4
P_A_O	P_C_G	a_1	a_2		a_4
P	P_B_G	a_1	a_2	a_3	a_4
P_C_O	P_B_G	a_1		a_3	a_4
P_B_O	P_B_G		a_2	a_3	a_4
P_A_O	P_B_G	a_1	a_2		a_4
P	P_A_G	a_1	a_2	a_3	a_4
P_C_O	P_A_G	a_1		a_3	a_4
P_B_O	P_A_G		a_2	a_3	a_4
P_A_O	P_A_G	a_1	a_2		a_4

4.3. Retreat and Avoid Obstacles with Three Shocks. In the process of exploration and learning, industrial wheeled robots may enter a deadlock state of motion behavior in some operating environments. This state makes the robot unable to avoid the U-shaped obstacle in time and repeatedly swing left and right in its detection area. Aiming at the adverse effect of the trap on the efficiency of robot exploration and learning, a three-oscillation obstacle avoidance algorithm is designed and added to the path planning algorithm with Q-BP learning. This allows robot to quickly judge and stay away from deadlock traps when making behavior choices.

The steps of the obstacle avoidance algorithm after three shocks are as follows: (1) The robot detects and records the obstacles in the current state s_t. (2) When the values of the three sensors are all less than $D_{\min}$, randomly perform a left turn action a_2 or a right turn action a_3 and reach the new state s_{t+1}. (3) Detect the obstacle distribution in the state s_{t+1}. If the detection value of the left sensor is less than or equal to the right sensor, perform action a_3. If the detection value of the right sensor is less than or equal to the left sensor, perform action a2. At this time, state s_{t+2} is reached. (4) Repeat action a_2 or a_3 three times to reach state s_{t+3} and record the value of each sensor at this time. (5) Execute action a4 in state s_{t+3} and reach state s_{t+4}. (6) If the detection value of each sensor of the robot in state s_{t+4} is still less than $D_{\min}$, it will directly return to the starting point to re-explore and learn.

4.4. Reward Function. The previously set environment state information provides the input data information needed to train the BP neural network for the algorithm designed in this section. According to the state-behavior relationship selection mechanism in Table 2, actions are performed so

that the robot changes from the current state s_t to the next state s_{t+1}. In the process of state transition, the robot's behavior needs to be reasonably evaluated, and quantitative indicators that affect obstacle avoidance and approaching targets are given. The design of the reward function follows the principle that the closer the obstacle is, the greater the negative reward, and the farther away obstacle is, the greater positive reward. Robot motion is continuously evaluated.

The distance relation function is

$$r = \log_{D_{\max}}(d_{or} + 0.01). \tag{7}$$

The robot obstacle avoidance reward selection function R_3 is

$$R_3 = \begin{cases} 0, & d_{or} \geq D_{\max}, \\ -\left|r_{t+1} - r_t\right|, & d_{or} < D_{\max} \text{ and } r_{t+1} > r_t, \\ \left|r_{t+1} - r_t\right|, & d_{or} < D_{\max} \text{ and } r_{t+1} \leq r_t, \end{cases} \tag{8}$$

where d_{or} is the distance between obstacle and robot.

The distance function between robot and target is expressed as

$$d_{rg} = \sqrt{\left(x_g - x_r\right)^2 + \left(y_g - y_r\right)^2}. \tag{9}$$

Reward design for robots approaching targets:

$$R_4 = \begin{cases} 0, & d_{rg}(t+1) = d_{rg}(t), \\ \left|d_{rg}(t+1) - d_{rg}(t)\right|, & d_{rg}(t+1) > d_{rg}(t), \\ -\left|d_{rg}(t+1) - d_{rg}(t)\right|, & d_{rg}(t+1) < d_{rg}(t). \end{cases} \tag{10}$$

The total reward function is

$$R_5 = (R_3 + R_4). \tag{11}$$

4.5. Design of Q Value Function Prediction with BP. BP neural network has a nonlinear mapping capability that can approximate any continuous function, and the Q value prediction algorithm with BP network is aimed at mapping capability. Use collected state-action pairs and the action evaluation Q value function to train the BP neural network so that it learns Q value function prediction model and predicts new Q value data through the model.

The prediction model with Q-BP value function designed is a BP neural network composed of three layers of neuron. The environment state variables perceived by robot are used as the input, and the value of the dimension of the state vector is 5. The output layer is Q values corresponding to each action. The model is illustrated in Figure 2.

The training steps of neural network are as follows: (1) Obtain input data. Robot utilizes a reinforcement Q learning to obtain sensor's perception feature vector S and normalize it as input sample of BP neural network Q value prediction model. The motion behavior of robot is constrained by the state-behavior relationship table in Table 2 and the established three shocks and obstacle avoidance rules. After the robot performs one action, the current state behavior is given a real Q value evaluation. Save feature quantity S and the expected Q value, add a sample training data to input data set. (2) Adjust BP network parameters. Input obtained training data into the input layer of BP network, adjust the weights and thresholds, and minimize the error between the expected Q value and the predicted Q value. (3) Judge whether the BP neural network converges, and use the sum of the squares of the errors of multiple trials as the evaluation function. When value of evaluation function is less than given precise value, BP network is judged to converge. (4) Save the weight and threshold information of each connection layer of the BP neural network.

5. Experiments and Discussion

5.1. Evaluation on Q-CM. The learning rate and discount factor when choosing path planning are 0.7 and 0.3, respectively, and the simulation results of path planning are obtained. Figure 3 illustrates the simulation results of path planning of robot after learning in unknown simple obstacle environment. The learning situation of the robot is shown in Table 3.

In the unknown simple obstacle grid environment, the obstacles similar to the maze are added to form a mixed complex obstacle environment. The path planning result is illustrated in Figure 4. The settings for other parameters used in the Q-CM algorithm remain unchanged. The learning situation of the wheeled robot is shown in Table 4.

Obviously, different times of experiments were performed under the same experimental setting. As the number of experiments increases, the average path length decreases, indicating that there are still a few states that have not been learned. After 50 experiments, the average number of learning is basically stable. At this time, all the states from start point to end point have been learned, and Q matrix converges. If more experiments are performed, the average path length will no longer change.

5.2. Evaluation on Reinforced Q Learning. Industrial wheeled robots carry out path planning experiments in a different number of discrete or continuous obstacle environments to verify the effectiveness of algorithm and observe the effect of path planning. Figure 5 illustrates the path planning results of robot's general obstacle environment with reinforcement Q learning.

The wheeled robot can plan the path between start point and end point arbitrarily set in map, and learn a safe and collision-free better path. The path learned did not reach the shortest. The possible reason is that the number of learning times is small, and individual states cannot be learned. In addition, since the Q table is discrete, the enhanced Q learning algorithm has limited ability to learn the state of continuous obstacles.

5.3. Evaluation on Q-BP Learning Algorithm. Use the best state-action pair generated by the reinforcement Q learning to train the BP neural network and predict the Q value. The wheeled robot executes the action corresponding to the

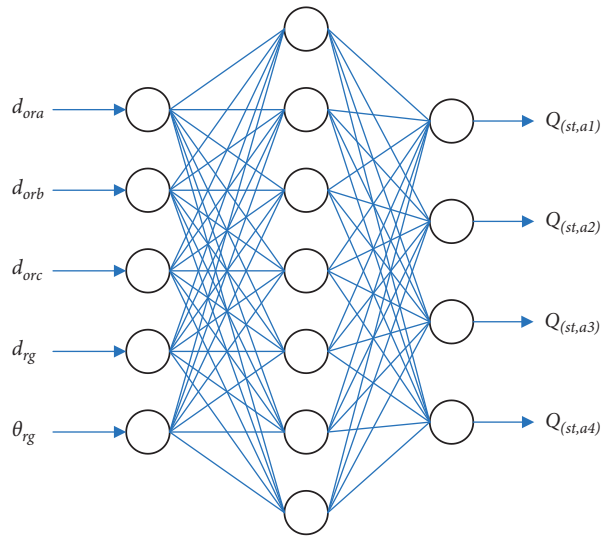


Figure 2: Q-BP value prediction model.

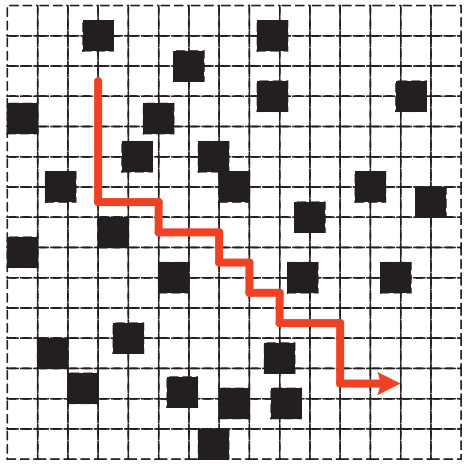


Figure 3: Path planning in unknown simple environment.

Table 3: Test result of Q-CM learning algorithm under simple environment.

Number of trials	Average learning number	Average path length
10	673	48
20	691	46
30	657	42
40	682	42
50	651	40
60	651	40
70	652	40
80	651	40

largest Q value among four outputs of BP network, and performs path planning in discrete or continuous obstacle maps. The purpose of the experiment is to verify the effectiveness of Q-BP algorithm and test the robot's obstacle avoidance ability against U-shaped obstacles. Figure 6 shows the path planning results.

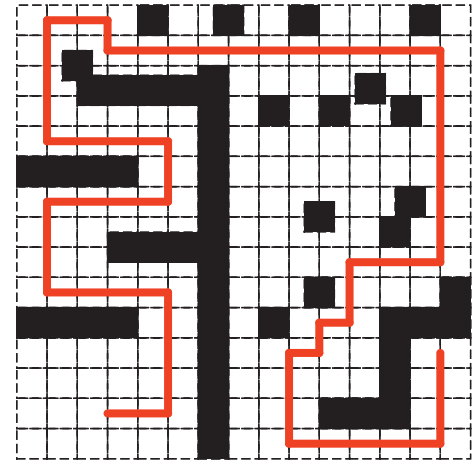


Figure 4: Path planning under complex and mixed environment.

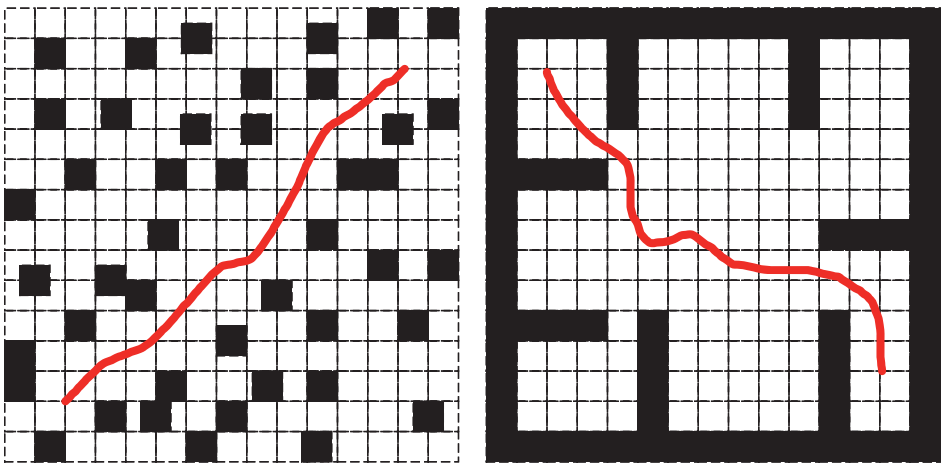
Table 4: Test result of Q-CM learning under complex and mixed environment.

Number of trials	Average learning number	Average path length
10	1877	198
20	1904	184
30	1953	182
40	1930	148
50	2031	134
60	2032	134
70	2032	134
80	2032	134

Industrial wheeled robots can complete path planning tasks in both discrete obstacle environments and continuous obstacle environments, and the path is better, indicating that the robot path planning scheme with Q-BP algorithm is effective. The wheeled robot can plan path between starting point and ending point arbitrarily set in the map, and learn a safe and collision-free better path. While the experiments in this section verify the effectiveness of the Q-BP algorithm, it also conducts U-shaped obstacle avoidance experiments on it. Figure 7 shows the result of U-shaped obstacle environment path planning of wheeled robot with Q-BP learning algorithm.

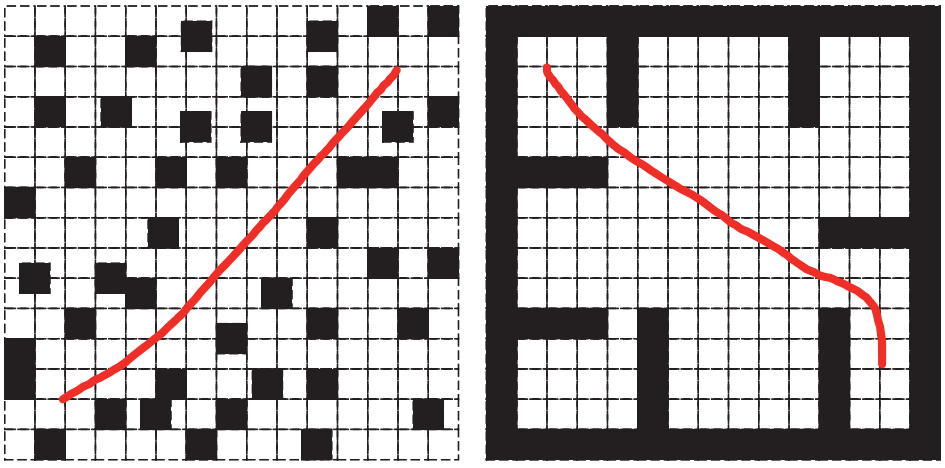
The wheeled robot uses Q-BP algorithm to complete obstacle avoidance and path planning tasks in the U-shaped obstacle environment. In addition to learning the optimal state-behavior, the Q-BP algorithm also has generalized self-learning capabilities. It can also generalize the states that are not included. Therefore, the robot U-shaped obstacle avoidance based on the Q-BP algorithm is smooth. A safe and collision-free shorter path from start point to end point can be planned as required, and the effect is good.

In addition, look up the Q table algorithm and the Q-BP algorithm through comparison. The experimental results show that the wheeled robot can complete the path planning behavior by using the two path planning methods, and find a short, collision-free path from the start point to the end point. However, the path length planned using the Q-BP model is shorter than that of looking up the Q table. In



(a) (b)

Figure 5: Path planning based on reinforcement Q learning in common obstacle.



(a) (b)

Figure 6: Path planning based on Q-BP learning in common obstacle.

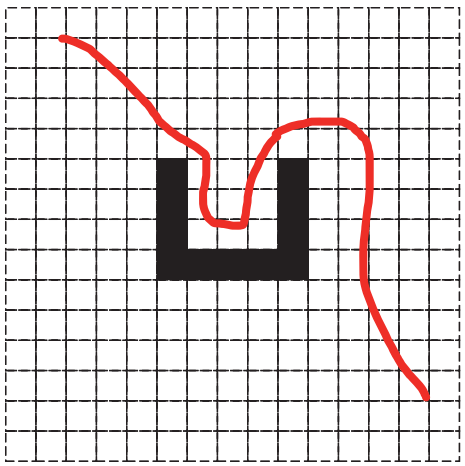


Figure 7: Path planning based on Q-BP learning in U-shaped obstacle.

particular, the continuous generalization ability of the BP network can be used to continue learning, which can achieve better operating results than learning in a discrete state environment.

6. Conclusion

This paper mainly studies the path planning algorithm of industrial wheeled robot based on Q-CM learning and Q-BP learning under wireless communication. The main research results are as follows: (1) based on the grid map environment, a path planning algorithm for industrial wheeled robots with Q-CM learning is proposed. First, according to robot path planning requirements, design the robot state and motion vector, and design the reward function. Secondly, for the robot, use the reinforced Q learning algorithm to solve issue of precise obstacle avoidance. According to the principle of coordinate matching between robot and obstacle, the CM obstacle avoidance method is designed to realize the precise obstacle avoidance function of robot path planning. (2) Based on the map environment of free space and the generalization ability of BP neural network, a path planning algorithm for industrial wheeled robots with Q-BP learning is proposed. First, according to the robot path planning requirements, design the map and the robot state. Aiming at the flexibility of motion behavior selection, the state-action selection strategy is designed to improve the flexibility of the robot's steering and the smoothness of the planned path. Secondly, in view of the slow convergence speed of the reinforcement *Q* learning algorithm, a dynamic reward function is formulated. Then, for the problem that the discrete Q table is not applicable to a large state space, the BP neural network is introduced. The BP neural network is trained using the best state-action pairs collected after the convergence of the enhanced Q learning algorithm, and the BP neural network with generalization ability is used to predict the Q value and complete the path planning task to improve the efficiency and accuracy of path planning. Finally, aiming at the problem of special U-shaped obstacle avoidance, a three-time shock retreat obstacle avoidance strategy was designed to make the robot quickly escape the deadlock trap and improve the robot's obstacle avoidance ability.

Data Availability

The datasets used during the current study are available from the corresponding author on reasonable request.

Conflicts of Interest

The author declares that there are no conflicts of interest.

Acknowledgments

This work was supported by the construction of the labor education security system of Jilin Higher Education Research Fund (JGJX2021D393).

References

[1] P. K. Das and P. K. Jena, "Multi-robot path planning using improved particle swarm optimization algorithm through novel evolutionary operators," *Applied Soft Computing*, vol. 92, Article ID 106312, 2020.

[2] Y. Yang, L. Juntao, and P. Lingling, "Multi-robot path planning based on a deep reinforcement learning DQN algorithm," *CAAI Transactions on Intelligence Technology*, vol. 5, no. 3, pp. 177–183, 2020.

[3] D. Zhang, X. You, S. Liu, and H. Pan, "Dynamic multi-role adaptive collaborative ant colony optimization for robot path planning," *IEEE Access*, vol. 8, Article ID 129958, 2020.

[4] 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.

[5] Q. Li, F. Gama, A. Ribeiro, and A. Prorok, "Graph Neural Networks for Decentralized Multi-Robot Path Planning," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Article ID 11785, Las Vegas, NV, USA, January 2020.

[6] F. Xu, H. Li, C.-M. Pun et al., "A new global best guided artificial bee colony algorithm with application in robot path planning," *Applied Soft Computing*, vol. 88, Article ID 106037, 2020.

[7] M. Dirik, A. F. Kocamaz, and O. Castillo, "Global path planning and path-following for wheeled mobile robot using a novel control structure based on a vision sensor," *International Journal of Fuzzy Systems*, vol. 22, no. 6, pp. 1880–1891, 2020.

[8] J. Pei, K. Zhong, J. Li, J. Xu, and X. Wang, "ECNN: evaluating a cluster-neural network model for city innovation capability," *Neural Computing and Applications*, pp. 1–13, 2021.

[9] K. Yu, X.-f. Liang, M.-z. Li et al., "USV path planning method with velocity variation and global optimisation based on AIS service platform," *Ocean Engineering*, vol. 236, Article ID 109560, 2021.

[10] M. Indri, F. Sibona, P. D. C. Cheng, and C. Possieri, "Online supervised global path planning for AMRs with human-obstacle avoidance," in *Proceedings of the IEEE international conference on emerging technologies and factory automation*, vol. 1, pp. 1473–1479, Vienna, Austria, September 2020.

[11] J.-H. Jung and D.-H. Kim, "Local path planning of a mobile robot using a novel grid-based potential method," *International Journal of Fuzzy Logic and Intelligent Systems*, vol. 20, no. 1, pp. 26–34, 2020.

[12] T. A. Teli and M. A. Wani, "A fuzzy based local minima avoidance path planning in autonomous robots," *International Journal on Information Technology*, vol. 13, no. 1, pp. 33–40, 2021.

[13] G. Liu, C. Li, T. Gao, and X. He, "Double BP Q-learning algorithm for local path planning of mobile robot," *Journal of Computer and Communications*, vol. 9, no. 6, pp. 138–157, 2021.

[14] K. Wyrąbkiewicz, T. Tarczewski, and Ł Niewiara, "Local path planning for autonomous mobile robot based on apf-bug algorithm with ground quality indicator," *Advanced, Contemporary Control*, pp. 979–990, 2020.

[15] M. Kobayashi and N. Motoi, "Local path planning method based on virtual manipulators and dynamic window approach for a wheeled mobile robot," in *Proceedings of the IEEE/SICE International Symposium on System Integration*, pp. 499–504, Iwaki, Fukushima, Japan, January 2021.

[16] L. Zhou and W. Li, "Adaptive artificial potential field approach for obstacle avoidance path planning," in *Proceedings of the International Symposium on Computational Intelligence*

and Design, vol. 2, no. 2, pp. 429–432, Hangzhou, China, December 2014.

[17] F. Bounini, D. Gingras, H. Pollart, and D. Gruyer, "Modified artificial potential field method for online path planning applications," in *Proceedings of the IEEE Intelligent Vehicles Symposium*, pp. 180–185, Los Angeles, CA, USA, June 2017.

[18] S. J. Fusic, P. Ramkumar, and K. Hariharan, "Path Planning of Robot Using Modified Dijkstra Algorithm," in *Proceedings of the National Power Engineering Conference*, pp. 1–5, Madurai, India, March 2018.

[19] H. Hliwa, M. Daoud, N. Abdulrahman, and A. Bassam, "Optimal path planning of mobile robot using hybrid tabu search-firefly algorithm," *International Journal of Computer Science Trends and Technology*, vol. 6, no. 6, pp. 7–15, 2018.

[20] H. Zhang, Y. Wang, J. Zheng, and J. Yu, "Path planning of industrial robot based on improved RRT algorithm in complex environments," *IEEE Access*, vol. 6, Article ID 53296, 2018.

[21] J. Cao, "Robot global path planning based on an improved ant colony algorithm," *Journal of Computer and Communications*, vol. 04, no. 02, pp. 11–19, 2016.

[22] P.-W. Tsai, T.-T. Nguyen, and T.-K. Dao, "Robot path planning optimization based on multiobjective grey wolf optimizer," *Advances in Intelligent Systems and Computing*, pp. 166–173, 2016.

[23] M. Brand and X. H. Yu, "Autonomous robot path optimization using firefly algorithm," in *Proceedings of the International Conference on Machine Learning and Cybernetics*, vol. 3, pp. 1028–1032, Tianjin, China, July 2013.

[24] M. Saraswathi, G. B. Murali, and B. B. V. L. Deepak, "Optimal path planning of mobile robot using hybrid cuckoo search-bat algorithm," *Procedia Computer Science*, vol. 133, pp. 510–517, 2018.

[25] J. Lee and D.-W. Kim, "An effective initialization method for genetic algorithm-based robot path planning using a directed acyclic graph," *Information Sciences*, vol. 332, pp. 1–18, 2016.

[26] H.-Y. Lee, H. Shin, and J. Chae, "Path planning for mobile agents using a genetic algorithm with a direction guided factor," *Electronics*, vol. 7, no. 10, p. 212, 2018.

[27] M. Li, C. Wang, Z. Chen, X. Lu, M. Wu, and P. Hou, "Path Planning of Mobile Robot Based on Genetic Algorithm and Gene Rearrangement," in *Proceedings of the Chinese Automation Congress*, pp. 6999–7004, Jinan, China, October 2017.

[28] M. Elhoseny, A. Shehab, and X. Yuan, "Optimizing robot path in dynamic environments using Genetic Algorithm and Bezier Curve," *Journal of Intelligent and Fuzzy Systems*, vol. 33, no. 4, pp. 2305–2316, 2017.

[29] Z. Han, D. Wang, F. Liu, and Z. Zhao, "Multi-AGV path planning with double-path constraints by using an improved genetic algorithm," *PLoS One*, vol. 12, no. 7, Article ID 0181747, 2017.

[30] E. Masehian and D. Sedighizadeh, "Multi-objective robot motion planning using a particle swarm optimization model," *Journal of Zhejiang University - Science C*, vol. 11, no. 8, pp. 607–619, 2010.

[31] N. Supakar and A. Senthil, "PSO obstacle avoidance algorithm for robot in unknown environment," in *Proceedings of the International conference on communication and computer vision*, pp. 1–7, Coimbatore, India, December 2013.

[32] B. Tang, Z. Zhanxia, and J. Luo, "A convergence-guaranteed particle swarm optimization method for mobile robot global path planning," *Assembly Automation*, vol. 37, no. 1, pp. 114–129, 2017.

[33] H. S. Dewang, P. K. Mohanty, and S. Kundu, "A robust path planning for mobile robot using smart particle swarm optimization," *Procedia Computer Science*, vol. 133, pp. 290–297, 2018.