IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 16 January 2023, accepted 5 March 2023, date of publication 9 March 2023, date of current version 29 March 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3255007

RESEARCH ARTICLE

Predator-Prey Reward Based *Q*-Learning Coverage Path Planning for Mobile Robot

MEIYAN ZHANG[1], WENYU CAI[2], AND LINGFENG PANG[2]
[1]College of Electrical Engineering, Zhejiang University of Water Resources and Electric Power, Hangzhou 310018, China
[2]College of Electronics and Information, Hangzhou Dianzi University, Hangzhou 310018, China

Corresponding author: Wenyu Cai (dreampp2000@163.com)

This work was supported in part by the Natural Science Foundation of Zhejiang Province under Grant LZ22F010004 and Grant LZJWY22E090001, in part by the National Natural Science Foundation of China under Grant 62271179 and Grant 61871163, in part by the National Innovation and Entrepreneurship Training Program for College Students under Grant 202111841031, and in part by the Scientific and Technological Innovation Activity Plan for College Students in Zhejiang Province (New Talent Plan).

ABSTRACT Coverage Path Planning (CPP in short) is a basic problem for mobile robot when facing a variety of applications. *Q*-Learning based coverage path planning algorithms are beginning to be explored recently. To overcome the problem of traditional *Q*-Learning of easily falling into local optimum, in this paper, the new-type reward functions originating from Predator-Prey model are introduced into traditional *Q*-Learning based CPP solution, which introduces a comprehensive reward function that incorporates three rewards including Predation Avoidance Reward Function, Smoothness Reward Function and Boundary Reward Function. In addition, the influence of weighting parameters on the total reward function is discussed. Extensive simulation results and practical experiments verify that the proposed Predator-Prey reward based *Q*-Learning Coverage Path Planning (PP-*Q*-Learning based CPP in short) has better performance than traditional BCD and *Q*-Learning based CPP in terms of repetition ratio and turns number.

INDEX TERMS Coverage path planning, predator-prey model, reinforcement learning, *Q*-learning algorithm, mobile robot.

I. INTRODUCTION

With the rapid development of computer and automatic control technology, mobile robots have been applied to industrial manufacturing, medical services, logistics sorting and other fields. As one of the important research directions in the field of mobile robots, Coverage path planning (CPP) [1], [2] has received much focus from researchers due to its great applications in many fields, such as air cleaning robot, exploration robot, demining robot, lawn mower and automatic harvester, so it has recently become an increasingly popular research topic. As we know, researches on coverage path planning play an important role in improving the working efficiency for mobile robots. Traditionally, for coverage path planning, it is required to generate appropriate paths that mobile robot can visit all points in the target area completely. Moreover, mobile robot needs to fill the region without overlapping paths and repetition of paths. However, in complex environments, enabling mobile robots to perform coverage path planning is still a challenging problem.

As far as we know, the existing CPP solutions can be divided into two categories: classic methods and heuristic methods [1], [2]. In the field of classic methods, the simple random walk algorithm (RW) [3], as a random covering algorithm, does not need to know the detailed environment, but it is hard to ensure high coverage and low repetition ratio. As a special random motion considering Levy flight [4], coverage is improved by using variable step size with gradient. The Boustrophedon Cellular Decomposition (BCD) algorithm [5] decomposes free space into simple, non-overlapping cells, and each unit is covered by ox plow path. The famous BCD method can be applied to various scenarios flexibly, but its inter-regional path planning often determines the repetition ratio. In the online BCD [6], critical points of each region are recorded by proximity searching algorithm, and inter-regional planning is realized by A* algorithm.

The associate editor coordinating the review of this manuscript and approving it for publication was Yangmin Li.

Reinforcement learning is used to learn inter-regional path planning and reduces the repetition ratio or original BCD [7]. Spinning Tree Coverage (STC) [8] is often used for small-scale maps, where the workspace is subdivided into a finite sequence of disjoint cells using either a cell decomposition based approach or a grid-based approach, which constructs a graph spanning tree in the corresponding giant cell and divides it into four subunits, where the size of corresponding cell is equal to the size of mobile robot. The STC algorithm uses tree traversal algorithm to find optimal paths so that the robot can cover every unoccupied unit. Unfortunately, STC algorithm has lower repetition ratio, but higher turns number. The single-robot STC algorithm is extended to multi-robot with region partition algorithm [9]. Huang et al. [10] use quadtree algorithm to decompose regions and STC algorithm to reduce coverage time and repetition ratio in sub-regions.

In recent years, some heuristic algorithms are widely used to solve CPP problems. Biological Inspired Neural Network (BINN) based CPP algorithm [11] calculates the activity of each grid in the grid map, and mobile robot will design specific path according to the activity level. An improved BINN based CPP algorithm [12] is proposed to overcome the dead zone problem through backtracking technology and neural network, thus improving the computational efficiency of activity value. However, these CPP algorithms based on BINN are easy to fall into the dead zone. A predator-hunting coverage algorithm based on the relationship between the starting point and the prey is proposed in [13]. The predator-prey model sets the starting point in the environment, and the prey, namely the robot, would be rewarded for escaping from the starting point and performing relevant behaviors, so as to achieve full-coverage path planning. Genetic algorithm (GA) is a kind of meta-heuristic random algorithm based on population, inspired by the natural law of biological genetics and the survival and reproduction of the fittest, so as to solve the search problem [14]. Genetic algorithm based CPP algorithms have good global search ability, but they require high computation time due to the large search space and poor stability [15]. Multi-objective genetic algorithm with dynamic programming is proposed to improve the speed of convergence to optimal value [16]. Particle swarm optimization (PSO) is a meta-heuristic algorithm based on biological social behavior patterns, involving clustering of natural populations [17]. In the field of coverage path planning, the famous PSO algorithm has global search ability in the initial stage, but tends to trap into local optimal value in later search process, and the convergence speed is slow. Couceiro et al. [18] divide groups into several small cooperative groups (subgroups), which provides the ability to evade local optimal solutions based on reward and punishment mechanisms. Ant colony algorithm (ACO) is a probabilistic technology that simulates the behavior of ants and the process of searching for food to solve complex optimization problems [19]. ACO based CPP algorithms have the advantages of strong robustness and parallel operation, but they are easy to trap into local optimum and the convergence speed is slow too. In [20], pheromone update rules is provided to avoid falling into local minima. A novel multi-agent coverage path planning algorithm is proposed in [21] inspired by the social behaviors in the biological world. To avoid falling into the local optimum, a cooperation mechanism is designed to improve the system adaptability.

Reinforcement Learning [22] is also a kind of heuristic algorithm, which has been widely applied in the field of robots, but is still in its infancy in solving CPP problems. A full-coverage path planning algorithm based on Q-Learning is proposed in [23], which optimizes coverage paths using raster graphs. Robot reward and punishment mechanism in DQN (Deep Q-Network) is proposed in [24] for full-coverage of UAV. Jin et al. [25] use reinforcement learning to achieve coverage of three-dimensional object representation, and demonstrate that ϵ-greedy strategy is better than pure greedy strategy. Zhang et al. [26] propose a multilevel humanlike motion planning approach is proposed for indoor mobile robots. Moreover, a new velocity-adjustable trajectory planning algorithm is put forward which is provably complete and time optimal considering multiple constraints from both the robot and the environment. In [27], an optimization based approach is proposed to obtain an optimal and robust path planning solution by assigning a potential function for each individual obstacle. In addition, there is obvious problem with the existing methods: there are few related literature with outdoor experiments and most of them focus on pure simulations.

At present, many experts believe that CPP solution based on Reinforcement Learning can obtain better performance. However, reward function related designs for Reinforcement Learning CPP methods are relatively simple. Although the Q-Learning based CPP algorithm can obtain optimal trajectory, it does not take into account the unsatisfactory results of repetition ratio and turns number. In this paper, an improved Q-Learning coverage path planning algorithm with new-designed comprehensive reward function is proposed. The mobile robot is guided to complete coverage by introducing the far from the starting point prize, the linear reward and the covering behavior reward to make the robot walk along a straight line as much as possible and reduce the repetition ratio. Simulation and experimental results show that the improved Q-Learning coverage path planning algorithm can complete the coverage task perfectly, and it has lower repetition ratio and fewer rotations than BCD and Q-Learning-based coverage path planning algorithms.

The main contributions of this paper mainly include following three-folders:

(1) Comprehensive Predator-Prey reward based Q-Learning coverage path planning algorithm is introduced in this paper, which considers multiple factors such as scene features, obstacles, and mobile robot. Different weighting factors associated with the smoothness reward and the boundary reward are simulated and analyzed to verify their impacts on the results.

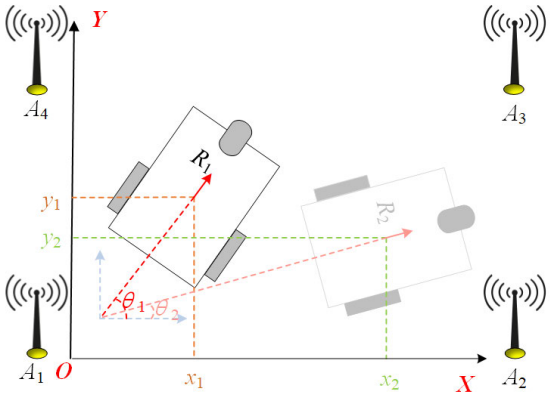

FIGURE 1. UWB positioning based mobile robot.

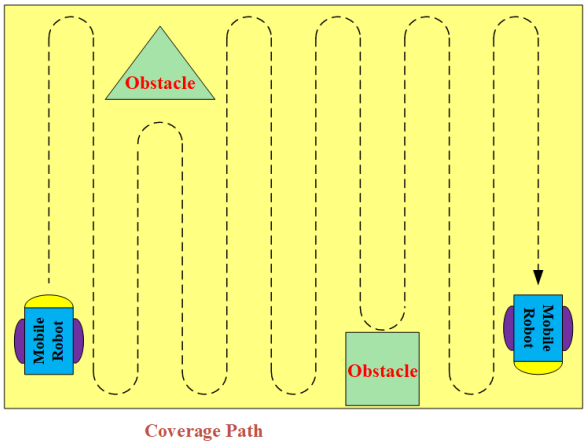

FIGURE 2. The principle of CPP problem.

(2) As unexpected stationary obstacles make the coverage problem challenging, we adopt Reinforcement Learning method to obtain optimal full coverage path on the basis of unexpected stationary obstacles.

(3) Different to pure software simulation, this paper also tests the proposed CPP algorithm on the mower platform to verify the actual performance.

The rest of this paper is organized as follows. Section II describes problem formulation of coverage path planning based on reinforcement learning. Then, Predator-Prey Reward based Q-Learning coverage path planning algorithm is introduced in Section III. Simulation results are presented in Section IV under static and dynamic environments. Finally, the conclusion remarks are given in Section V.

II. PROBLEM FORMULATION

This section defines some assumptions and definitions needed to address the CPP problem. The mobile robot is a two-wheeled robot and its motion obeys a nonholonomic constraint with velocity vector. In this paper, the robot is assumed to obtain its location using UWB (Ultra wide Band) positioning method [28]. The UWB positioning based coverage path planning is described in Figure 1, where four UWB stations are distributed at the corners of coverage region, and one UWB label is equipped in the mobile robot to output its location.

As an essential for robotic tasks, Coverage Path Planning of mobile robot is to find a motion path for the robot to pass over all points in a given region, which is illustrated in Figure 2. As we know, solving traditional CPP with minimal cost problem that are subject to unexpected changes is challenging, this is because: (1) mobile robot is initially unaware of the obstacles or the changes to coverage area; (2) mobile robot is not only expected to achieve complete coverage and unexpected changes, but also needs to do so with minimal cost.

For the convenience of description, the following assumption is proposed in this paper.

Assumption 1: In the region of $W \times L$, the two-wheeled mobile robot with the center $O(t) = (x, y)$ and motion radius r obeys a nonholonomic constraint with velocity vector expressed as $v_r(t)$. The position and velocity of mobile robot are a function of time since the mobile robot is continuously moving.

The Predator-Prey based CPP (PPCPP) approach [14] is inspired by the predator-prey behavior, which is developed that is efficient to respond to changes in real-time while aiming to achieve complete coverage with minimal cost. As we know, the concept of PPCPP has been investigated for coverage path planning problem [29], where the prey represents the coverage spot of robot's end-effector tool, and the predator is a virtual stationary point that the prey continually maximizes its distance between them while covering the target areas. As a result, this approach accounts for improving the path length and smoothness by rewarding the prey to continue its motion in a straight direction and covering the boundary as much as possible. In addition, the method can learn from prior environmental information and use the learned parameters to perform adaptive local planning in real time. In short, the above Predator-Prey model is very suitable for traditional CPP.

As we know, unlike stationary obstacles, unexpected dynamic obstacles further complicate the problem since the robot needs to avoid the obstacles and efficiently re-plan the trajectory while still aiming to achieve complete coverage with minimal cost. The CPP problem addressed in this article involves unexpected changes occurring during the real-time deployment process, and we adopt Reinforcement Learning method to obtain optimal full coverage path on the basis of specific reward function from Predator-Prey model.

III. PREDATOR-PREY REWARD BASED Q-LEARNING COVERAGE PATH PLANNING

A. Q-LEARNING BASED COVERAGE PATH PLANNING

Reinforcement learning is one kind of machine learning technologies, which includes five basic elements: agent,

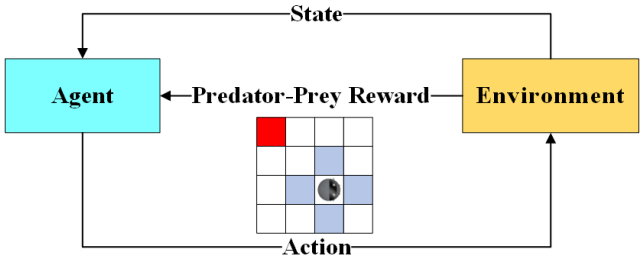

FIGURE 3. *Q*-Learning principle.

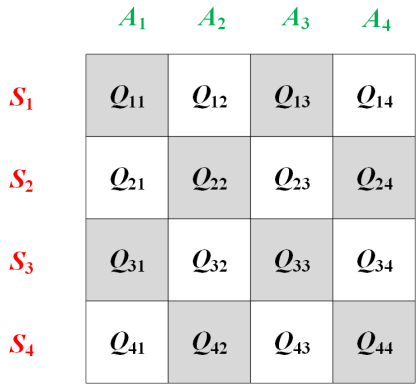

FIGURE 4. *Q*-table example.

environment, state, action and reward. The relationship between them is described in Figure 3, where the agent obtains learning information and updates model parameters according to its own state and receives rewards from the environment for performing actions, so as to maximize benefits [30].

Q-Learning based Reinforcement Learning algorithm will establish a Q-table $Q(S, A)$, where S represents the state and A represents the action. In the traditional CPP problem, S represents the position grid of mobile robot and A represents different actions of mobile robot, such as four motion directions: forward, backward, turn left and turn right. Figure 4 shows a simple Q-table initialized by a 2×2 raster map.

In addition, ϵ-Greedy strategy is introduced, i.e. the random action strategy is implemented with probability ϵ, and the optimization strategy is implemented under the condition of $1 - \epsilon$. The main flowchart of Q-Learning based CPP approach is described in Algorithm 1.

At the beginning of learning process, all elements in Q-table are set to 0 and updated by Equation (1) during iterative learning.

$$Q(S, A) = R(S, A) + \tau \times Max(Q(S^*, ;) \tag{1}$$

where $R(S, A)$ denotes the reward brought by performing the action A under state S, τ represents the decay rate, $Max(Q(S^*, ;))$ represents the largest Q-value in the previous state S^*. $R(S, A)$ is defined as Equation (2). R_{finish} will be obtained if and only if the mobile robot completes total

Algorithm 1 Q-Learning based CPP Problem

Initialize $R(S, A)$ for all S,A
Initialize $Q(S, A) = 0$ for all S,A
#Q-Learning Training
Repeat
Repeat
S = start_point;
ϵ = random();
if $\epsilon < \epsilon_0$
A = random(A);
else
A = argmax($Q(S, ;)$)
end
S^* = Perform(A);
$Q(S, A) = R(S, A) + \tau \times max(Q(S^*, ;))$
$S = S^*$;
Until finish CPP
Until the number of episodes is reached;
#Q-Learning based Coverage Path Planning
S = start_point;
Repeat
S = start_point;
A = argmax($Q(S^*, ;)$)
S^* = Perform(A);
Until finish CPP

coverage path planning, i.e.,

$$R(S, A) = \begin{cases} R_{finish}, \text{if CPP finish} \\ 0, \text{otherwise.} \end{cases} \tag{2}$$

Although Q-Learning based full coverage path planning algorithm can solve the problem perfectly, it is easy to cause too many turns number and high repetition ratio in the environment with many obstacles or large maps. As far as we know, selecting the appropriate return function can improve the performance of the algorithm. Therefore, this paper discusses how to set appropriate reward function to improve the effect of coverage path planning.

B. PREDATOR-PREY REWARD

As a typical Cyber-Physical interactive system between mobile robot and environment, Predator-Prey model is inspired by the concepts of foraging and risk of predation in predator-prey relation, which is described as Figure 5. Φ, $\boldsymbol{o}_k$ and $\boldsymbol{o}_j$ denote the predator, the current prey target and target gird j in the grid set $\boldsymbol{O}$, respectively. Hence, the newly defined reward function is discussed as follows.

1) TOTAL REWARD

A total reward function R is designed as a heuristic for the prey to select its next best move at each step $k (k = 1, 2, \ldots, n_k)$ where ideally n_k is equal to $n^{\boldsymbol{O}}$, which is the total number of targets in the set $\boldsymbol{O}$. Thus, at step k, the prey evaluates R for all neighbors and moves to the neighbor

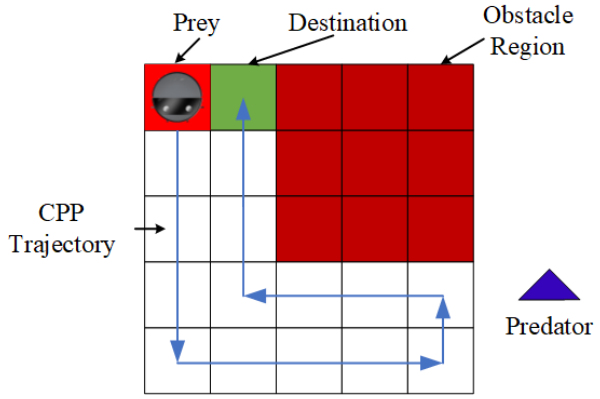

FIGURE 5. Predator-prey model.

with maximal reward. The derived function R comprises of three rewards: (1) Predation Avoidance Reward Function $R^d(\boldsymbol{o}_j)$ for maximizing the distance to the predator; (2) Smoothness Reward Function $R^s(\boldsymbol{o}_j)$ for continuing motion in a straight direction; (3) Boundary Reward Function $R^b(\boldsymbol{o}_j)$ for covering the boundary [14]. The total reward for moving to an uncovered neighbor $\boldsymbol{o}_j \in \boldsymbol{O}$ is the sum of all the rewards previously stated, i.e.,

$$R(\boldsymbol{o}_j) = R^d(\boldsymbol{o}_j) + \lambda_s R^s(\boldsymbol{o}_j) + \lambda_b R^b(\boldsymbol{o}_j) \tag{3}$$

where λ_b and λ_s are the weighting factors associated with the smoothness reward and the boundary reward, respectively. The weighting factors govern the extent to which each reward is emphasized by the prey when deciding on the next movement.

2) PREDATION AVOIDANCE REWARD FUNCTION

Clearly, the prey will maximize its reward by moving toward a neighbor that is uncovered (not yet covered) and that has the farthest distance from the predator at each step. The predation avoidance reward function for the prey moving to the j-th neighbor $\boldsymbol{o}_j$ is formulated as,

$$R^d(\boldsymbol{o}_j) = \frac{D(\boldsymbol{o}_j) - D_{\min}(\boldsymbol{o}_k)}{D_{\max}(\boldsymbol{o}_k) - D_{\min}(\boldsymbol{o}_k)} \tag{4}$$

where $D(\boldsymbol{o}_j) = \| \boldsymbol{o}_j - \Phi \|$ gives distance from $\boldsymbol{o}_j$ to the predator Φ, $D_{\max}(\boldsymbol{o}_k) = max_j \| \boldsymbol{o}_j - \Phi \|$ gives the maximum distance from one of the neighbors of the current prey target to the predator, and similarly, $D_{\min}(\boldsymbol{o}_k) = min_j \| \boldsymbol{o}_j - \Phi \|$ gives the minimum distance.

3) SMOOTHNESS REWARD FUNCTION

For mobile robots' applications, having a path that has more straight lines (fewer turns) can be beneficial, e.g., mobile robots that consume more energy or time due to frequent turns. Hence, the smoothness reward function is formulated as follows:

$$R^s(\boldsymbol{o}_j) = \frac{\angle\, \boldsymbol{o}_{k-1}\boldsymbol{o}_k\boldsymbol{o}_j}{180^\circ} \tag{5}$$

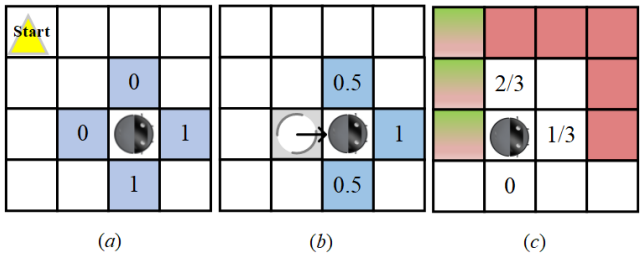

FIGURE 6. Three examples for different reward functions.

where $R^s(\boldsymbol{o}_j) \in (0, 1]$ is the reward associated with the j-th neighbor $\boldsymbol{o}_j$ of the current prey target $\boldsymbol{o}_k$ due to the angle $\angle \boldsymbol{o}_{k-1}\boldsymbol{o}_k\boldsymbol{o}_j \in (0^\circ, 180^\circ]$ which is the angle between the vectors $(\boldsymbol{o}_{k-1} - \boldsymbol{o}_k)$ and $(\boldsymbol{o}_k - \boldsymbol{o}_j)$, and $\boldsymbol{o}_{k-1}$ is the target covered by the prey at the previous step $(k-1)$.

4) BOUNDARY REWARD FUNCTION

The boundary reward function is formulated as follows:

$$R^b(\boldsymbol{o}_j) = \frac{n^{N_{\max}} - n^N(\boldsymbol{o}_j)}{n^{N_{\max}}} \tag{6}$$

where $R_b(\boldsymbol{o}_j) \in [0, 1]$ is the reward associated with the j-th neighbor o_j of current prey target $\boldsymbol{o}_k$ and $n^N(\boldsymbol{o}_j)$ calculates the number of uncovered neighbors of the target $\boldsymbol{o}_j$, $n^{N_{max}}$ is the maximum possible number of neighbors for a target.

Detailed settings of these parameters used in our algorithm refer to Ref [14]. Three examples for three different reward functions are shown in Figure 6. As Figure 6(a), prediction avoidance reward values of four optional grids are 0, 0, 1, 1 according to the principle of keeping away from the starting point as far as possible. Two grids with reward value 1 are much far away than two grids with reward value 0. Smoothness reward value will be greater than that of turning any angle if mobile robot goes straight. In Figure 6(b), it is obvious that the straight grid has larger smoothness reward 1. As Figure 6(c), the smaller the number of uncovered neighbor grids of certain target, the higher the boundary reward value. Therefore, the upper grid has larger boundary reward value $\frac{2}{3}$.

C. TOTAL ALGORITHM FLOW

To summarize, the proposed Predator-Prey reward based Q-Learning CPP method for mobile robot is described as Figure 7.

IV. RESULTS AND DISCUSSION

In this section, Matlab platform is used to conduct simulations. Three different CPP solutions including BCD, Q-Learning CPP, PP-Q-Learning CPP are introduced to compare their performance. The famous BCD (Boustrophedon Cellular Decomposition) is an exact cellular decomposition approach, where each cell in the boustrophedon is covered with simple back and forth motions. BCD based CPP algorithm will decompose the map into several regions, and

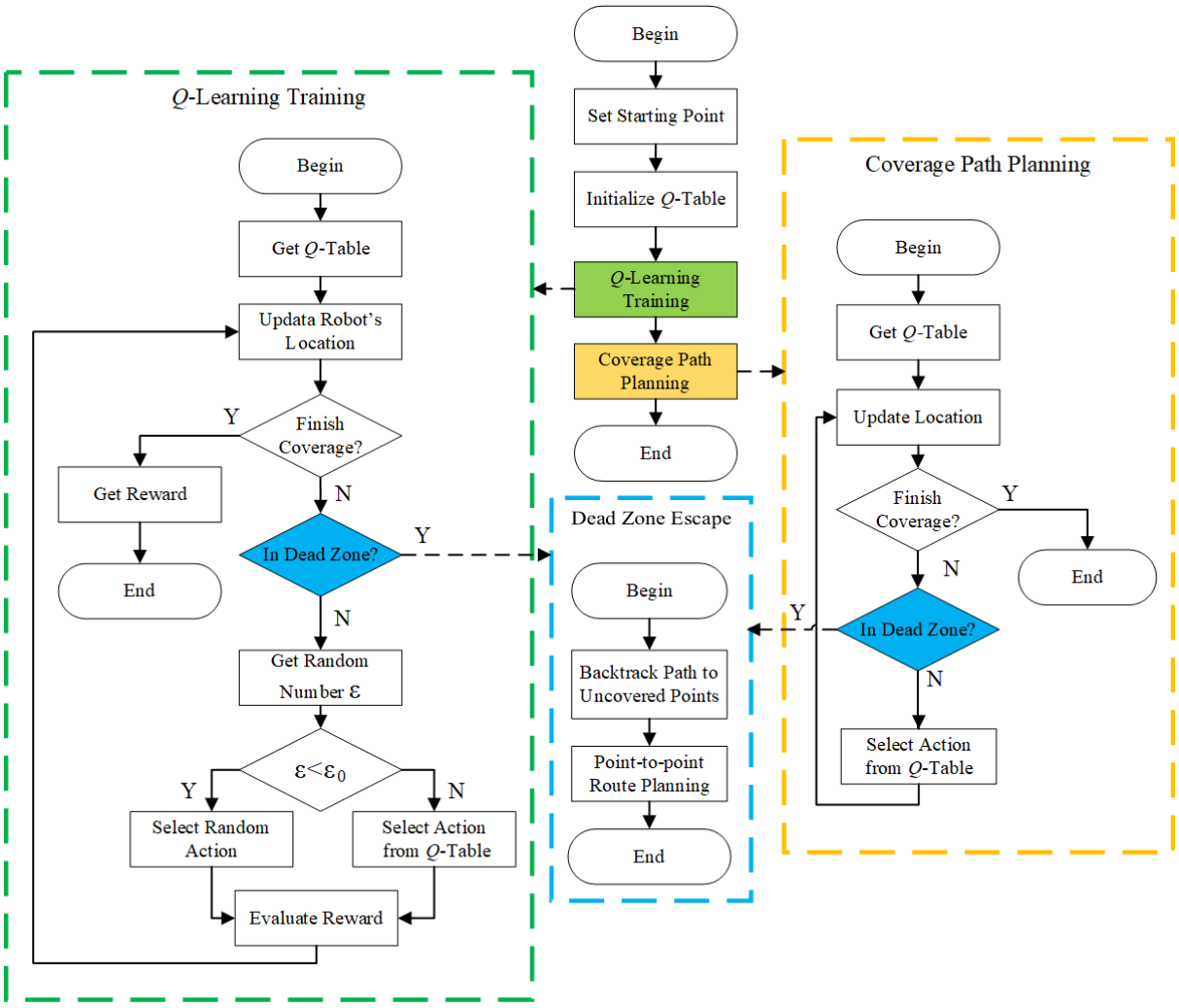

FIGURE 7. The flowchart of total algorithm.

cover each sub region according to the cattle plowing path. The sub regions are connected with each other in the order of depth first traversal. Two main coverage cases with region size 10 × 10 and 20 × 22 are applied to validate the proposed approach. Due to the long training time of deep learning, this paper selects two smaller scenes for algorithm verification.

Since the coverage ratios of these three CPP algorithms are close to 1, another two metrics including repetition ratio R_r and turns number N_t are applied to measure performance. The turns number is defined as the number of turns in the total full-coverage path. The repetition ratio R_r is defined as following equation:

$$R_r = \frac{N_a - N_g}{N_a} \tag{7}$$

where N_a and N_g denote the number of actually passed grids and the total number of girds, respectively. Lower repetition ratio and turns number should be achieved with better coverage path planning approach.

TABLE 1. Parameter.

Parameter	Value
$W \times L$	$10 \times 10/20 \times 22$
λ_b	0-0.8
λ_s	0-0.8
ε_0	0.1
τ	0.9
r	1

A. SIMULATION SETUP

The main simulation parameters are listed in Table 1.

B. COVERAGE PATH PLANNING RESULTS

Firstly, full-coverage path planning trajectories are compared in details. In the first case with size of 10 × 10, the CPP trajectories derived from BCD, Q-Learning based CPP, and PP-Q-Learning based CPP are demonstrated in Figure 8, Figure 9 and Figure 10, respectively. It is obvious that the full coverage path of our algorithm is better than that of BCD and original Q-Learning based CPP algorithms.

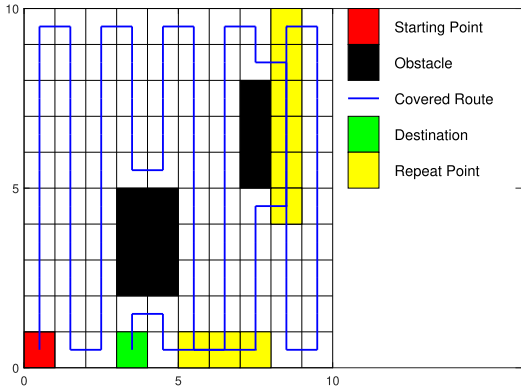

FIGURE 8. Coverage trajectory with BCD based CPP (10 × 10).

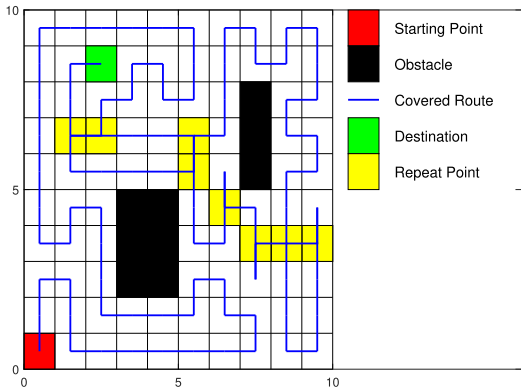

FIGURE 9. Coverage trajectory with *Q*-Learning based CPP (10 × 10).

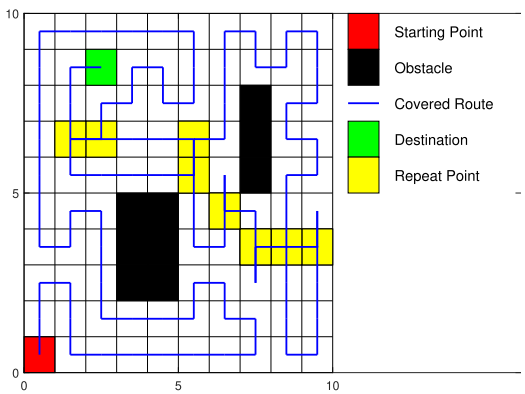

FIGURE 10. Coverage trajectory with PP-*Q*-Learning based CPP (10 × 10).

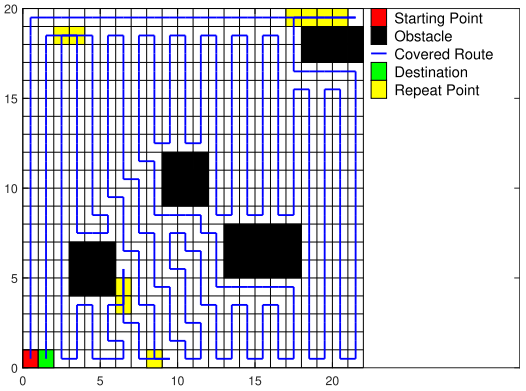

FIGURE 11. Coverage trajectory with BCD based CPP (20 × 22).

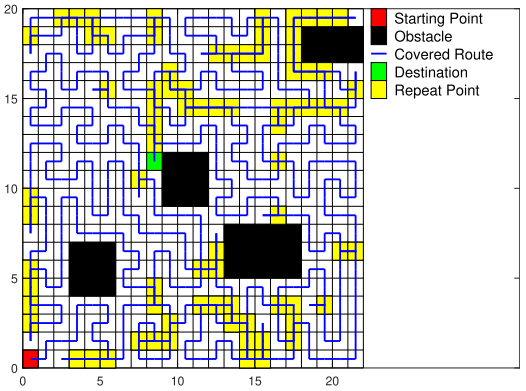

FIGURE 12. Coverage trajectory with *Q*-Learning based CPP (20 × 22).

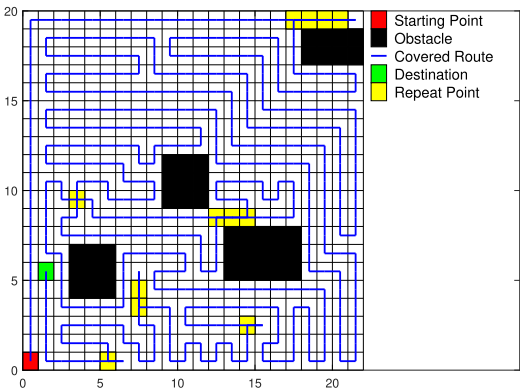

FIGURE 13. Coverage trajectory with PP-*Q*-Learning based CPP (20 × 22).

Similarly, in the other case with size of 20 × 22, the CPP trajectories derived from BCD, *Q*-Learning based CPP, and PP-*Q*-Learning based CPP are described in Figure 11, Figure 12 and Figure 13, respectively. As a result, extensive simulation results in different scenarios verify the superiority of the proposed algorithm.

Furthermore, repetition ratio and turns number metrics are calculated to compare performance quantitatively. As Figure 14, the repetition ratio and turns ation number are significantly reduced compared to BCD and *Q*-Learning based CPP algorithm. In other words, the proposed PP-*Q*-Learning based CPP algorithm outperforms traditional BCD and latest Deep Learning method. The repetition ratio and turns number will improve more than 50%. In conclusion, these numerical results verify the effectiveness of proposed approach.

C. PARAMETER COMPARISON RESULTS

The proposed reward function has two parameters λ_s and λ_b, and so we will discuss how to determine these parameters.

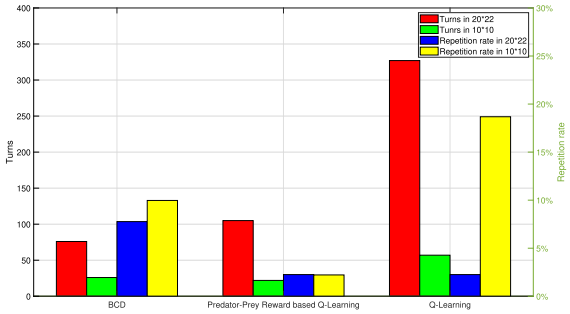

FIGURE 14. Comparison of repetition ratio and turns number.

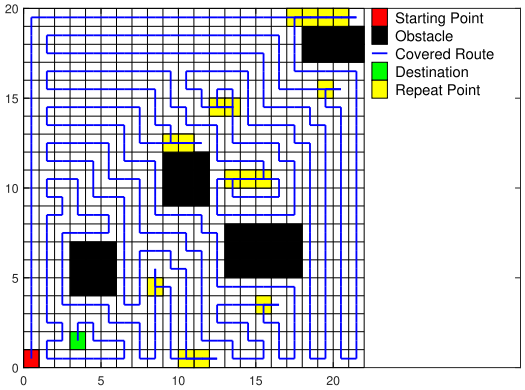

FIGURE 15. CPP results with $\lambda_s = 0$ and $\lambda_b = 0$.

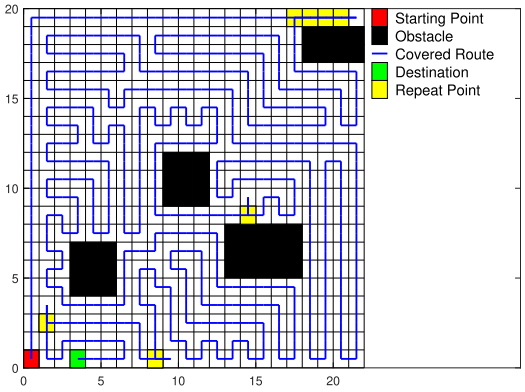

FIGURE 16. CPP results with $\lambda_s = 0$ and $\lambda_b = 0.4$.

A case study is presented in this section to validate the usage of each reward function and demonstrate how the parameters affect the performance. Different parameter combinations of λ_s and λ_b are used to simulate in the other case with size of 20 × 22, and the final trajectory results under typical parameter combinations are derived as Figure 15 to Figure 20.

To clarify the influence of parameters on path results, the repetition ratios under different parameters are compared in Figure 21. Furthermore, these numerical results are listed in Table 2.

Similarly, the turn numbers under different parameters are compared in Figure 22. For the sake of clarity, these numerical results are listed in Table 3.

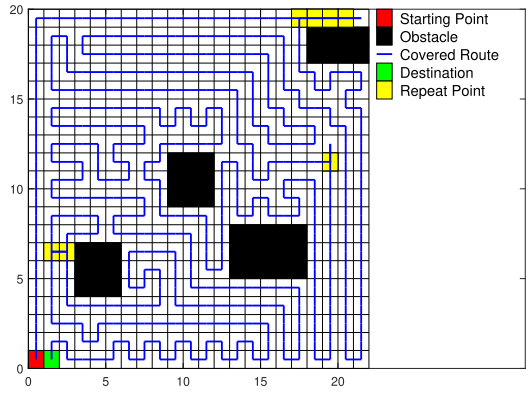

FIGURE 17. CPP results with $\lambda_s = 0.2$ and $\lambda_b = 0.6$.

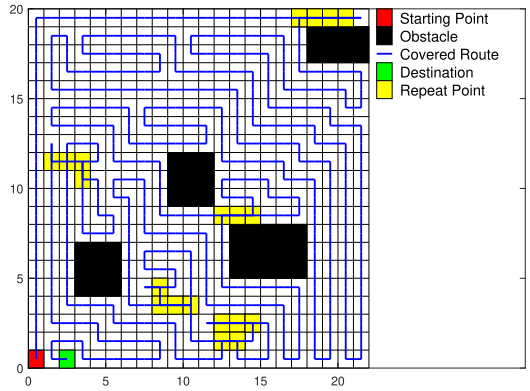

FIGURE 18. CPP results with $\lambda_s = 0.6$ and $\lambda_b = 0$.

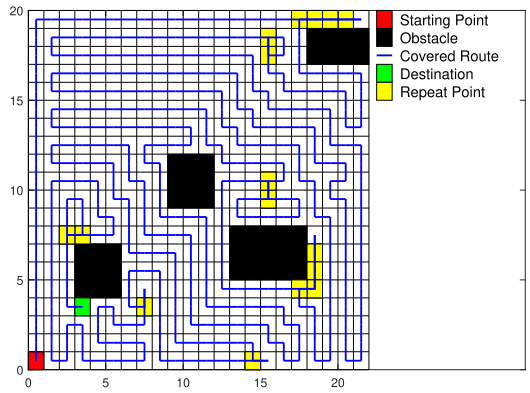

FIGURE 19. CPP results with $\lambda_s = 0.6$ and $\lambda_b = 0.2$.

TABLE 2. Repetition ratio under different weight parameters.

λ_s/λ_b	λ_s=0.0	λ_s=0.2	λ_s=0.4	λ_s=0.6	λ_s=0.8
λ_b=0.0	13.78%	8.02%	10.28%	10.28%	10.28%
λ_b=0.2	3.01%	4.01%	8.27%	8.02%	9.27%
λ_b=0.4	2.51%	3.01%	6.02%	9.27%	8.52%
λ_b=0.6	2.51%	3.51%	3.51%	2.26%	7.02%
λ_b=0.8	2.51%	3.51%	4.01%	3.76%	7.03%

From the above simulation results, it is evidently that the effect of proposed algorithm is different under different parameters. Obviously, two weighting parameters λ_b and λ_s

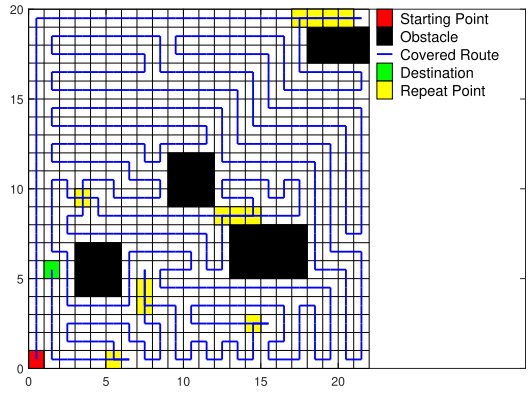

FIGURE 20. CPP results with $\lambda_s = 0.6$ and $\lambda_b = 0.6$.

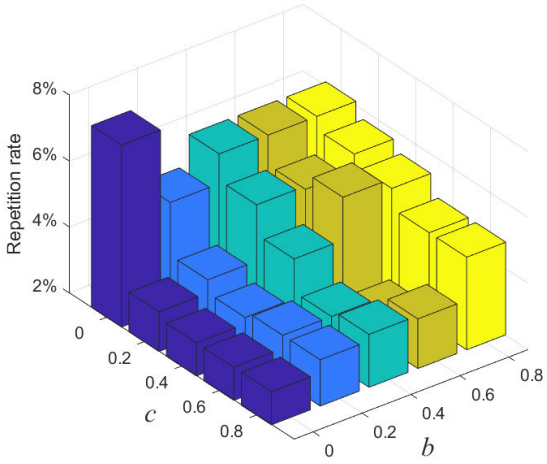

FIGURE 21. The relationship between repetition ratio and weight parameters.

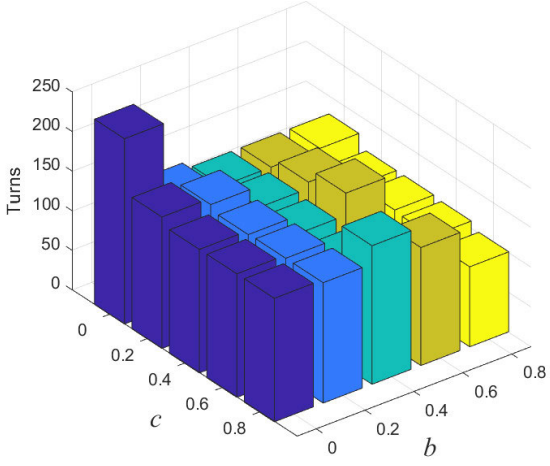

FIGURE 22. The relationship between turns number and weight parameters.

have irreplaceable influence on the CPP results. In conclusion, by comparing the results under the different parameters, it can provide a guidance for the selection of weighting parameters.

TABLE 3. Turns number under different parameters.

λ_s/λ_b	λ_s=0.0	λ_s=0.2	λ_s=0.4	λ_s=0.6	λ_s=0.8
λ_b=0.0	234	136	127	127	127
λ_b=0.2	166	158	131	139	111
λ_b=0.4	156	151	128	156	111
λ_b=0.6	156	152	120	105	109
λ_b=0.8	156	152	175	149	101

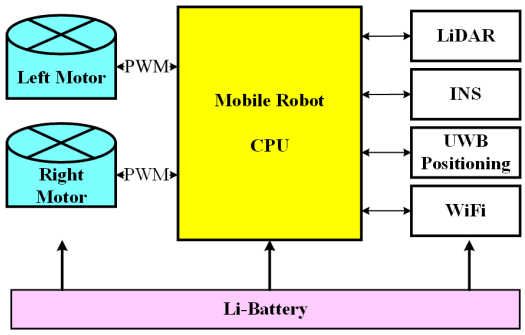

FIGURE 23. Hardware framework.

FIGURE 24. Mobile robot.

FIGURE 25. Experimental scenario.

D. PRACTICAL EXPERIMENT RESULTS

Finally, the proposed Predator-Prey reward based Q-Learning coverage path planning algorithm is realized on the self-designed lawn mower, which is two wheeled mobile robot with obstacle avoidance ability. The hardware architecture and actual photo of the lawn mower are exhibited in Figure 23 and Figure 24, respectively. The self-designed lawn mower is equipped with LiDAR, INS, UWB label and WiFi module, which is ideal for performance validation.

With the experimental scene as Figure 25 in the school playground, the actual full-coverage path is displayed in Figure 26. It is obvious that the error between actual trajectory

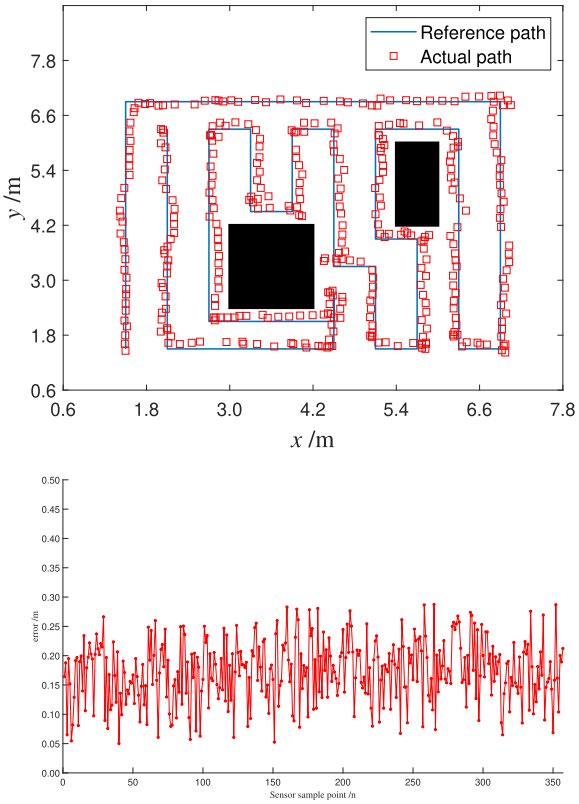

FIGURE 26. Practical trajectory results.

and planned trajectory is comparatively small. Therefore, the practical trajectory and error results verify that the actual performance of proposed approach.

V. CONCLUSION

The objective of coverage path planning is to ensure persistent coverage of target area for mobile robots. To reduce repetition ratio and turns number during coverage path planning, this paper introduces the Predator-Prey model into *Q*-Learning based CPP problem. Three rewards including predation avoidance reward function, smoothness reward function and boundary reward function are combined to determine optimization strategy. Extensive simulation results and practical experiments verify that the performance of proposed PP-*Q*-Learning based coverage path planning algorithm is better than that of traditional BCD and *Q*-Learning based CPP methods. In our future work, actual performance of the proposed algorithm will be examined in the presence of mobile obstacles.

ACKNOWLEDGMENT

The authors would like to thank the anonymous referees for their constructive comments and valuable suggestions, which help them to improve the quality and presentation of the article greatly.

REFERENCES

[1] E. Galceran and M. Carreras, "A survey on coverage path planning for robotics," *Robot. Auton. Syst.*, vol. 61, no. 12, pp. 1258–1276, 2013.
[2] C. S. Tan, R. Mohd-Mokhtar, and M. R. Arshad, "A comprehensive review of coverage path planning in robotics using classical and heuristic algorithms," *IEEE Access*, vol. 9, pp. 119310–119342, 2021.
[3] F. Bartumeus, M. G. E. Da Luz, G. M. Viswanathan, and J. Catalan, "Animal search strategies: A quantitative random-walk," *Ecology*, vol. 86, no. 11, pp. 3078–3087, Nov. 2005.
[4] A. Schroeder, S. Ramakrishnan, M. Kumar, and B. Trease, "Efficient spatial coverage by a robot swarm based on an ant foraging model and the Lévy distribution," *Swarm Intell.*, vol. 11, no. 1, pp. 39–69, Mar. 2017.
[5] H. Choset and P. Pignon, "Coverage path planning: The boustrophedon cellular decomposition," in *Field and Service Robotics*. London, U.K.: Springer, 1998, pp. 203–209.
[6] A. Khan, I. Noreen, H. Ryu, N. L. Doh, and Z. Habib, "Online complete coverage path planning using two-way proximity search," *Intell. Service Robot.*, vol. 10, no. 3, pp. 229–240, Jul. 2017.
[7] P. T. Kyaw, A. Paing, T. T. Thu, R. E. Mohan, A. Vu 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.
[8] Y. Gabriely and E. Rimon, "Spanning-tree based coverage of continuous areas by a mobile robot," *Ann. Math. Artif. Intell.*, vol. 31, nos. 1–4, pp. 77–98, 2001.
[9] A. C. Kapoutsis, S. A. Chatzichristofis, and E. B. Kosmatopoulos, "DARP: Divide areas algorithm for optimal multi-robot coverage path planning," *J. Intell. Robotic Syst.*, vol. 86, nos. 3–4, pp. 663–680, Jun. 2017.
[10] X. Huang, M. Sun, H. Zhou, and S. Liu, "A multi-robot coverage path planning algorithm for the environment with multiple land cover types," *IEEE Access*, vol. 8, pp. 198101–198117, 2020.
[11] C. Luo and S. X. Yang, "A bioinspired neural network for real-time concurrent map building and complete coverage robot navigation in unknown environments," *IEEE Trans. Neural Netw.*, vol. 19, no. 7, pp. 1279–1298, Jul. 2008.
[12] A. Singha, A. K. Ray, and A. B. Samaddar, "Neural dynamics-based complete coverage of grid environment by mobile robots," in *Proc. Int. Conf. Frontiers Comput. Syst.* (Advances in Intelligent Systems and Computing), vol. 1255, D. Bhattacharjee, D. K. Kole, N. Dey, S. Basu, and D. Plewczynski, Eds. Singapore: Springer, 2021, pp. 411–421.
[13] M. Hassan and D. Liu, "PPCPP: A predator-prey-based approach to adaptive coverage path planning," *IEEE Trans. Robot.*, vol. 36, no. 1, pp. 284–301, Feb. 2020.
[14] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis With Applications to Biology, Control, and Artificial Intelligence*. Cambridge, U.K.: MIT Press, 1992.
[15] L. Qiu, "Research on a hierarchical cooperative algorithm based on genetic algorithm and particle swarm optimization," in *Computational Intelligence and Intelligent Systems* (Communications in Computer and Information Science), vol. 874, K. Li, W. Li, Z. Chen, and Y. Liu, Eds. Singapore: Springer, 2018, pp. 16–25.
[16] M. G. Sadek, A. E. Mohamed, and A. M. El-Garhy, "Augmenting multi-objective genetic algorithm and dynamic programming for online coverage path planning," in *Proc. 13th Int. Conf. Comput. Eng. Syst. (ICCES)*, Cairo, Egypt, Dec. 2018, pp. 475–480.
[17] J. Kennedy, "The particle swarm: Social adaptation of knowledge," in *Proc. IEEE Int. Conf. Evol. Comput.*, Indianapolis, IN, USA, Apr. 1997, pp. 303–308.
[18] M. S. Couceiro, R. P. Rocha, and N. M. F. Ferreira, "A novel multi-robot exploration approach based on particle swarm optimization algorithms," in *Proc. IEEE Int. Symp. Saf., Secur., Rescue Robot. (SSRR)*, Kyoto, Japan, Nov. 2011, pp. 327–332.
[19] M. Dorigo, M. Birattari, and T. Stutzle, *Ant Colony Optimization*. Cambridge, MA, USA: MIT Press, 2004.
[20] W. Zhang, X. Gong, G. Han, and Y. Zhao, "An improved ant colony algorithm for path planning in one scenic area with many spots," *IEEE Access*, vol. 5, pp. 13260–13269, 2017.
[21] L. Jiao, Z. Peng, L. Xi, S. Ding, and J. Cui, "Multi-agent coverage path planning via proximity interaction and cooperation," *IEEE Sensors J.*, vol. 22, no. 6, pp. 6196–6207, Mar. 2022, doi: 10.1109/JSEN.2022.3150098.
[22] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. Cambridge. MA, USA: MIT Press, 2018.
[23] L. Piardi, J. Lima, A. I. Pereira, and P. Costa, "Coverage path planning optimization based on Q-learning algorithm," in *Proc. AIP Conf.*, Rhodes, Greece, 2019, pp. 1–4.

[24] J. Dong, "Area coverage path planning of UAV based on deep reinforcement learning," *Ind. Control Comput., China*, vol. 34, no. 5, pp. 80–82, 2021.
[25] W. Jing, C. F. Goh, M. Rajaraman, F. Gao, S. Park, Y. Liu, and K. Shimada, "A computational framework for automatic online path generation of robotic inspection tasks via coverage planning and reinforcement learning," *IEEE Access*, vol. 6, pp. 54854–54864, 2018.
[26] X. Zhang, J. Wang, Y. Fang, H. Gao, and J. Yuan, "Multi-level human-like motion planning for mobile robots in complex indoor environments," *IEEE Trans. Automat. Sci. Eng.*, vol. 16, no. 3, pp. 1244–1258, Jul. 2019, doi: 10.1109/TASE.2018.2880245.
[27] F. Bayat, S. Najafinia, and M. Aliyari, "Mobile robots path planning: Electrostatic potential field approach," *Exp. Syst. Appl.*, vol. 100, pp. 68–78, Jun. 2018, doi: 10.1016/j.eswa.2018.01.050.
[28] C. Tian, "Design and research on path planning of intelligent mower based on UWB positioning," M.S. thesis, Dept. Mech. Eng., Southeast Univ., Nanjing, China, 2018.
[29] M. Hassan, D. Mustafic, and D. Liu, "Dec-PPCPP: A decentralized predator-prey-based approach to adaptive coverage path planning amid moving obstacles," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Las Vegas, NV, USA, Oct. 2020, pp. 11732–11739.
[30] W. Chen, X. Qiu, T. Cai, H.-N. Dai, Z. Zheng, and Y. Zhang, "Deep reinforcement learning for Internet of Things: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 3, pp. 1659–1692, 3rd Quart., 2021, doi: 10.1109/COMST.2021.3073036.

MEIYAN ZHANG was born in Sanmen, Taizhou, Zhejiang, China, in 1983. She received the B.S. and master's degrees from the Zhejiang University of Technology, Hangzhou, China, in 2006 and 2009, respectively. She joined the Postgraduate Program with the Institute of Automation, Zhejiang University of Technology. From 2016 to 2017, she was a Visiting Research Scholar with the Missouri University of Science and Technology, Rolla, MO, USA. She is currently with the College of Electrical Engineering, Zhejiang University of Water Resources and Electric Power, as an Associate Professor. In recent years, she has published more than 20 articles on wireless sensor networks, and holds more than ten patents. Her research interests include wireless communication, wireless sensor networks, and multimedia sensor networks.

WENYU CAI was born in Cixi, Ningbo, Zhejiang, China, in 1979. He received the B.S. and Ph.D. degrees from Zhejiang University, Hangzhou, China, in 2002 and 2007, respectively. He joined the Successive Postgraduate and Doctoral Program in physical electronics with Zhejiang University. From 2016 to 2017, he was a Visiting Research Scholar with the Missouri University of Science and Technology, Rolla, MO, USA. He is currently with the College of Electronics and Information, Hangzhou Dianzi University, as an Associate Professor. In recent years, he has published more than 40 articles on wireless sensor networks, and holds more than 20 patents. His research interests include wireless communication, wireless sensor networks, and underwater sensor networks. He served as a reviewer for more than ten journals.

LINGFENG PANG was born in Tiantai, Taizhou, Zhejiang, China, in 1997. He received the B.S. degree from Ningbo Tech University, Ningbo, China, in 2020. He is currently pursuing the degree in electronic science and technology with Hangzhou Dianzi University, Hangzhou, China. His current research interest includes coverage path planning.