IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 26, 2019, accepted January 8, 2020, date of publication January 20, 2020, date of current version February 26, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2967225

A Distributed Multi-Agent Dynamic Area Coverage Algorithm Based on Reinforcement Learning

JIAN XIAO[1], GANG WANG[1], YING ZHANG[2], AND LEI CHENG[1]

[1]School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China
[2]School of Resources and Environment, University of Electronic Science and Technology of China, Chengdu 611731, China

Corresponding author: Ying Zhang (zhangyingxf@uestc.edu.cn)

This work was supported by the National Natural Science Foundation of China under Grant 61371182 and Grant 41301459.

ABSTRACT Dynamic area coverage is widely used in military and civil fields. Improving coverage efficiency is an important research direction for multi-agent dynamic area coverage. In this paper, we focus on the non-optimal coverage problem of free dynamic area coverage algorithms. We propose a distributed dynamic area coverage algorithm based on reinforcement learning and a γ-information map. The γ-information map can transform the continuous dynamic coverage process into a discrete γ point traversal process, while ensuring no-hole coverage. When agent communication covers the whole target area, agents can obtain the global optimal coverage strategy by learning the whole dynamic coverage process. In the event that communication does not cover the whole target area, agents can obtain a local optimal coverage strategy; in addition, agents can use the proposed algorithm to obtain a global optimal coverage path through off-line planning. Simulation results demonstrate that the required time for area coverage with the proposed algorithm is close to the optimal value, and the performance of the proposed algorithm is significantly better than the distributed anti-flocking Algorithms for dynamic area coverage.

INDEX TERMS Dynamic area coverage, multi-agent, reinforcement learning, optimal coverage.

I. INTRODUCTION

Dynamic area coverage has been widely used in target detection [1], monitoring [2] and searching [3], [4]. In the process of dynamic area coverage, agents are equipped with sensors to establish a mobile sensor network (MSN), and the agent can be controlled to achieve coverage of the target area with a dynamic coverage algorithm [5]–[7]. Compared with the traditional static area coverage method where sensor nodes cannot be rearranged easily once deployed [8], [9], the dynamic area coverage method has the characteristics of good flexibility. Moreover, when the range of the target area is large, the dynamic area coverage requires less sensors than the static area coverage.

For autonomous agents in multi-agent systems (MAS), we classify dynamic coverage control algorithms for multi-agents into two categories: 1) non-self-organizing control algorithms, and 2) self-organizing control algorithms. The non-self-organizing control algorithm for dynamic coverage reduces repeated coverage of the area through some constraint relationships between the agents to achieve efficient dynamic coverage of the target area [10]–[12]. However, this non-self-organizing method is not robust, and an agent that is out of work as a result of some emergency situation during the coverage process will result in an uncovered area [13]. The self-organizing control algorithm enables agents to have great autonomy, robustness and good flexibility [7], [14]–[16], and solves the problem where the target area cannot be covered completely in emergencies. In this paper, we focus on the self-organizing control algorithm.

The anti-flocking control algorithm is a classic self-organizing algorithm for dynamic area coverage. The concept of anti-flocking control was first introduced in [17]. Ganganath et al. proposed a distributed anti-flocking algorithm for dynamic area coverage based on the flocking algorithm proposed in [18] and information map [19], and improved the coverage efficiency of their proposed algorithm using a territorial marking inspired information map

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

in [13]. However, in the calculation of the target point with the anti-flocking control algorithm proposed in [13], agents only consider the current coverage status, and ignore the entire coverage process. Thus, the obtained target position may not be the globally optimal point for the coverage process. Because the target point calculated by the anti-flocking control algorithm is non-optimal, this will cause the agent to cover an area repeatedly, reducing the coverage efficiency and increasing the coverage time. Without doubt, a non-optimal solution can be provided by another self-organizing control algorithm for dynamic area coverage. Therefore, a feasible way to improve dynamic coverage efficiency involves selecting the best target position using a method that considers the whole coverage process.

At present, there exist some process optimization methods, including ant colony optimization (ACO) [20], particle swarm optimization (PCO) [21] and model predictive control (MPC) [22]. These methods can optimize the motion path of each agent in the MAS. However, these methods cannot take the motion state of their neighbors into account. Therefore, it is also difficult for them to determine the optimal dynamic coverage strategy in the MAS.

Reinforcement learning (RL) is a process learning algorithm that can learn the best behavior strategy for the entire process [23]. Agents can select the optimal action in different states according to their previous experience [24]. RL is widely used in various process control fields, including sensor networks [25], [26], path planning [27], [28] and robotics [29], [30]. Recently, the application of reinforcement learning to multi-agent collaborative control is increasing [31]–[34]. However, to our limited knowledge, reinforcement learning has not been applied to dynamic area coverage.

In this paper, we construct a MAS motion model based on the flocking algorithm proposed in [18]. We then propose a distributed self-organizing multi-agent dynamic area coverage algorithm based on RL, which transforms the area coverage problem into an optimal on-line planning problem for the agents' target points (γ points). To construct the RL model, we design a γ-information map by gridding the target area to record the coverage information for the γ points. Through learning in the coverage process with the proposed algorithm, agents can plan the best γ points, get the optimal coverage path, and cover the free area efficiently. In some extreme environments, it is difficult for communication to meet the real-time interaction requirements. We propose an off-line global optimal coverage path planning method based on our proposed algorithm that achieves off-line optimal planning not possible with other self-organizing dynamic coverage algorithms.

The main contributions of the paper can be summarized as follows: (1) A distributed multi-agent dynamic area coverage algorithm based on RL is proposed to obtain the global optimal or local optimal dynamic coverage strategy for the entire coverage process. (2) We present an off-line global optimal area coverage scheme based on the proposed algorithm. (3) A γ-information map is proposed and used to convert continuous area coverage problems into discrete γ point planning problems. Based on the γ-information map, a continuous-discrete hybrid control system is established.

The rest of the paper is organized as follows. In Section II, we state the problem formulation, including the system framework, the construction of the multi-agent motion model, and the design of the γ-information map. In Section III, we construct the state space and action space for the RL and define the reward function. In Section IV, the computational complexity of the proposed algorithm is analyzed. The simulation and evaluation process are introduced in Section V, and the simulation results and result analysis are described in Section VI. The conclusion is presented in Section VII.

II. PROBLEM STATEMENT

In this section, we describe the framework of a multi-agent dynamic area coverage system, construct a multi-agent motion model based on the flocking algorithm, and give the definition of the γ-information map.

A. SYSTEM FRAMEWORK

The multi-agent dynamic area coverage system constructed in this paper is a continuous-discrete hybrid control system, which is described in Fig. 1.

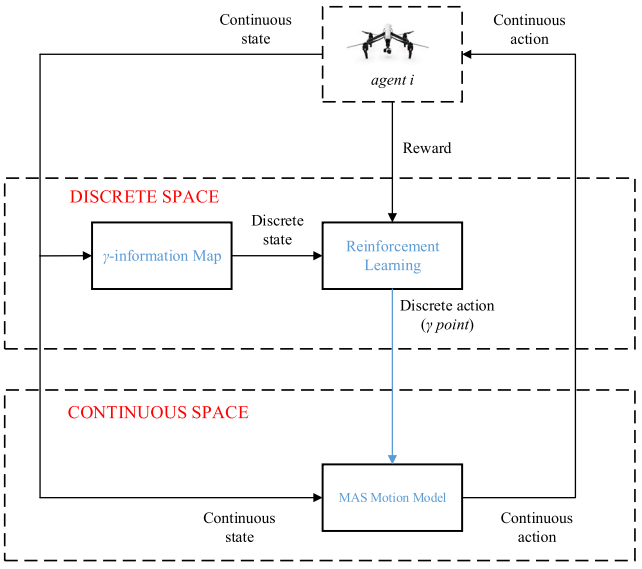


FIGURE 1. Hybrid control system for multi-agent dynamic area coverage.

In Fig. 1, The MAS motion control model transforms discrete action into continuous action, and controls agent i to move in a continuous space. γ-information map discretizes the continuous state of agent i, and agent i select the best γ point (a discrete action) based on reinforcement learning.

B. MAS MOTION MODEL

In this paper, we assume that each agent in the MAS satisfies the particle motion model. Let p_i, v_i and $u_i \in R^2$ denote the position, velocity and control input of the ith agent, respectively. Then agent i satisfies the following equation

of motion:

$$\begin{cases} \dot{p}_i = v_i, \\ \dot{v}_i = u_i, \quad i = 1, 2, \cdots, N \end{cases} \tag{1}$$

where N is the number of agents in the MAS, and the N agents constitute the set $V = \{1, 2, \cdots, N\}$.

We define the neighbor set of agent i based on the communication distance, r_c, as follows:

$$N_i = \{j \in V : ||p_j - p_i|| \leq r_c, j \neq i\}. \tag{2}$$

In the coverage process, each agent needs to plan its target position in the next moment according to its coverage state, and avoid collisions between agents. To achieve self-organizing control for dynamic coverage, we take the agent and the target position as α-agent and γ-agent in [18], respectively. Because there is only a repulsive force between agents, we need to redefine the formula for calculating the potential field force between α-agents in [18].

We define the potential field force between agent i and agent j as follows:

$$f_i^{\alpha} = \phi(||p_j - p_i||_{\sigma})\sigma(p_j - p_i) \tag{3}$$

where $\phi(z)$, $||z||_{\sigma}$ and $\sigma(z)$ are defined as follows:

$$\phi(z) = \rho_h(z/d)(\sigma(z - d_{\sigma}) - 1), \tag{4}$$

$$||z||_{\sigma} = \sqrt{1 + ||z||^2} - 1, \tag{5}$$

$$\sigma(z) = \frac{z}{\sqrt{1 + ||z||^2}}. \tag{6}$$

$||z||_{\sigma}$ is a map $R^2 \to R$, and $\sigma(z)$ is the gradient of $||z||_{\sigma}$. d in (4) is the avoidance distance, and d_{σ} is denoted as

$$d_{\sigma} = \sqrt{1 + d^2} - 1. \tag{7}$$

$\rho_h(z)$ in (4) is the bump function introduced in [18]

$$\rho_h(z) = \begin{cases} 1, & 0 \leq z < h, \\ \frac{1}{2}[1 + \cos(\pi \frac{z - h}{1 - h})], & h \leq z < 1, \\ 0, & otherwise, \end{cases} \tag{8}$$

where h is a constant, and $0 < h < 1$. $\rho_h(z)$ can map z to [0,1] smoothly.

Considering all the neighbors of agent i, we obtain the control quantity of avoidance given by

$$u_i^{\alpha} = c^{\alpha} \sum_{k \in N_i} \phi(||p_j - p_i||_{\sigma})\sigma(p_j - p_i), \tag{9}$$

where c^{α} is a positive constant.

Based on the PID control algorithm [35], we obtain the control quantity generated by γ-agent, represented as follows:

$$u_i^{\gamma} = -c_1^{\gamma}(p_i - p_r) - c_2^{\gamma} v_i, \tag{10}$$

where c_1^{γ} and c_2^{γ} are the proportional and differential control parameters in the PID algorithm, respectively, and p_r is the position of γ-agent.

We obtain the following expression for the control quantity from (9) and (10):

$$u_i = u_i^{\alpha} + u_i^{\gamma} \tag{11}$$

For ease of reference, the parameters and notations used in this paper are summarized in Table 1.

TABLE 1. Summary of parameters and notations.

Parameters	Notations
N	The Number of agents
N_T	The maximum training times
$m \times n$	Size of the target area
$k \times l$	Size of the γ-information map
α	Learning rate
λ	Discounting factor
w	Weight of the cooperative learning
h	Parameters of the bump function
r_{ref}	Standard reward value for the whole traversal process
T_{min}	Minimum traversal time in the ideal condition
r_s	perceived radius
r_c	Communication distance
ε	Permissible position error
c_r , c_1^T and c_2^T	Parameters of the reward function
c^{α} , c_1^{γ} and c_2^{γ}	Control Parameters of the MAS motion model
d	Avoidance distance

C. γ-INFORMATION MAP

To increase the instantaneous coverage area of agents and reduce the RL state, the target area is designed as a γ-map with traversal information. The γ-information map for agent i is defined as follows:

γ-Information Map: Assuming that the target area can be constructed as a rectangular region of $m \times n$, we divide the area into $k \times l$ small rectangular regions, and the center of each small rectangle can be regarded as the γ point, which is equivalent to γ- agent. All centers of small rectangles constitute γ point sets, $M_i(\gamma) = \{\gamma_{x,y}\}, x = 1, 2 \cdots k, y = 1, 2 \cdots l$. The traversal information of each element in $M_i(\gamma)$ is represented by $m_i(\gamma_{x,y})$, $\gamma_{x,y} \in M_i(\gamma)$. If $\gamma_{x,y}$ has been traversed, then $m_i(\gamma_{x,y}) = 1$, otherwise $m_i(\gamma_{x,y}) = 0$. All the $m_i(\gamma_{x,y})$, $\gamma_{x,y} \in M_i(\gamma)$ constitute the γ-information map represented by $M_i = \{\gamma_{x,y} \in M_i(\gamma) | m_i(\gamma_{x,y})\}$, and (x, y) and $m_i(\gamma_{x,y})$ represent the position and information value of $\gamma_{x,y}$ on the γ-information map.

k and l can be calculated from the following formula:

$$\begin{cases} k = \left\lceil \frac{n}{\sqrt{2} r_s} \right\rceil, \\ l = \left\lceil \frac{m}{\sqrt{2} r_s} \right\rceil, \end{cases} \tag{12}$$

where r_s is the perceived radius of agent i, and $\lceil z \rceil$ is the ceiling operation for z. Equation (12) guarantees that the small rectangular region where $\gamma_{x,y}$ is located is completely covered when agent i reaches $\gamma_{x,y}$. Fig. 2 shows an example of γ-information map traversal.

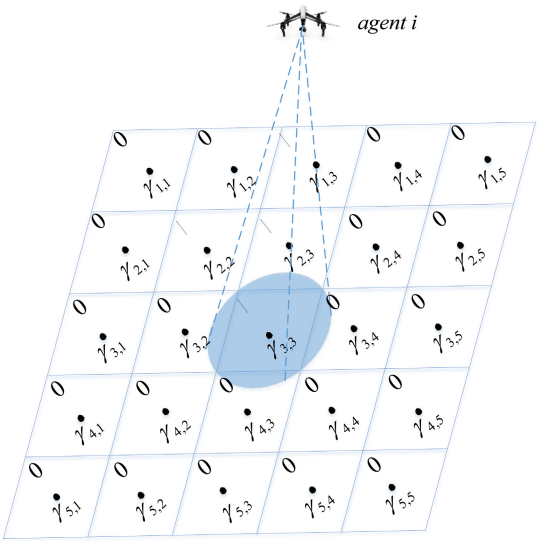


FIGURE 2. Example of agent traversing γ-information map. The value in the corner of each small rectangular region is the value of $m_i(\gamma_{x,y})$.

When communication between agents is established, the agent will fuse its γ-information map and its neighbor's γ-information map using the method given by the following equation:

$$m_i(\gamma_{x,y}) = \max_{s \in V}(m_s(\gamma_{x,y})). \tag{13}$$

Equation (13) guarantees that the traversal information for $\gamma_{x,y}$ is the latest, which is beneficial for the agent to select the γ points that are not traversed and improve the traversal efficiency of the MAS.

Therefore, the γ-information map of agent i has two functions. 1) It records the information on the γ points that have been traversed by agent i. 2) It records the information on the γ points that have been traversed by the neighbors of agent i through the information interaction between the agents. The proposed γ-information map allows us to convert dynamic area coverage into γ point traversal; thus we can achieve a transformation from a continuous coverage process to a discrete traversal process, which provides the conditions for the RL algorithm proposed in the following section.

III. REINFORCEMENT LEARNING

A. ASSUMPTIONS

Q-learning is a typical reinforcement learning algorithm that records the learned experience in a Q-value table, from where we can obtain the best action strategy. In the traversal process of the γ-information map, we use Q-learning to plan the γ points of agents, and we can get the best planning strategy for the γ points after learning. Thus agents can complete dynamic coverage of the target area efficiently. In applying Q-learning to dynamic area coverage, we make the following assumptions.

(1) Agents in MAS motion are on the same horizontal plane and satisfy the MAS motion model proposed in Section II.
(2) Agents have the same detection range to the ground, $r_s > 0$, and the communication distance r_c between agents is the same. When communication is established between agents, agents can share information on position, velocity, γ-information map, learning experience and so on.
(3) At the start of motion, communication is established between agents.

Assumption (3) constrains the initial position of the agent, which creates better initial conditions to improve Q-learning and improve the learning efficiency.

To illustrate the application of Q-learning to the traversal process of the γ-information map, we provide a symbolic definition in Q-learning. Let the current state, action and reward of agent i be s_i, a_i and r_i, respectively, and let the next state and action of agent i be s_i' and a_i', respectively. Next, we construct the state and action space, and give the calculation method for r_i and the Q-value and the criterion for action selection.

B. STATE

During the traversal process of the γ-information map, agent i obtains a fused γ-information map by interacting with its neighbors, then decides the next γ point according to its γ-information map and the target position of its neighbors. So, we define the state of agent i as follows:

$$s_i = [M_i, p_1^\gamma, p_2^\gamma, \cdots, p_i^\gamma \cdots, p_N^\gamma], \tag{14}$$

where p_i^γ is the position of the target γ point of agent i on the γ-information map, calculated as follows:

$$p_i^\gamma = f_{index}(\gamma_{x,y}) = (x, y), \tag{15}$$

where $f_{index}(\gamma_{x,y})$ is the index function. For example, if the target γ point of agent i is $\gamma_{x,y}$ then $p_i^\gamma = (x, y)$. In (14), if agent j is not adjacent to agent i, then $p_j^\gamma = (0, 0)$, otherwise p_j^γ can be calculated from (15).

C. ACTION SPACE

The action in Q-learning is expressed as the choice of the γ point. When agent i is in a state s_i, its optional γ point is the $\gamma_{x,y}$ determined by p_i^γ and eight γ points adjacent to $\gamma_{x,y}$. We use 1–9 to represent the nine γ points shown in Fig. 3. Therefore, the action space of agent i can be defined as:

$$A_i = \{1, 2, 3, 4, 5, 6, 7, 8, 9\} \tag{16}$$

When the agent is at the boundary of the γ-information map, the action space of the agent will be a subset of the action space represented by (16). For example, when the agent is located at the upper left corner of the γ-information map, then $A_i = \{1, 2, 8, 9\}$; when the agent is located at

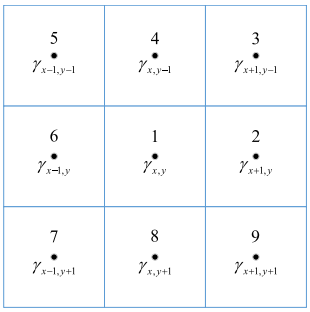


FIGURE 3. Nine optional actions. Each number corresponds to a γ point.

the lower boundary of the γ-information map, then $A_i = \{1, 2, 3, 4, 5, 6\}$. Fig. 4 shows these two situations. Other boundary cases can be inferred from the above two cases.

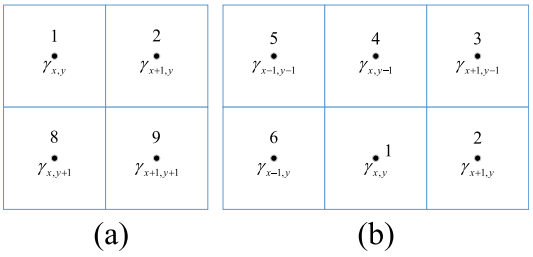


FIGURE 4. Action space in different cases. (a) Agent at the upper left corner of the γ-information map. (b) Agent at the lower boundary of the γ-information map.

When one of the following conditions is met, agent i will update its action.

(1) Agent i reaches the target γ point $\gamma_{x,y}$; the formula, $|p_i - \gamma_{x,y}| < \varepsilon$, holds, where ε is the permissible error.
(2) Agent i and agent j have the same γ point.

D. REWARD FUNCTION

We estimate the value of the selected action according to the traversal state of the γ-information map. The reward function is defined as follows:

$$r_i(s_i, a_i) = \begin{cases} 0, m_i(\gamma'_{x,y}) = 0, & a_i \in \{1, 2, 4, 6, 8\}, \\ -0.3, m_i(\gamma'_{x,y}) = 0, & a_i \in \{3, 5, 7, 9\}, \\ -0.2e^{c_r(k_r-1)}, & m_i(\gamma'_{x,y}) \neq 0, \\ R(T), m_i(\gamma_{x,y}) \neq 0, & \gamma_{x,y} \in M_i(\gamma), \end{cases} \tag{17}$$

where $\gamma'_{x,y}$ is the next γ point obtained by executing action a_i, $0 < c_r < 1$ is a constant, k_r is the number of repeated traversals, T is the time consumed during the process of γ-information map traversal or dynamic area coverage and $R(T)$ is defined as follows:

$$R(T) = \begin{cases} 0, & T > c_1^T T_{min}, \\ \dfrac{r_{ref}}{2}[1 + \cos(\dfrac{\pi(T - c_2^T T_{min})}{(c_1^T T_{min} - c_2^T T_{min})})], & \\ & c_2^T T_{min} < T \leq c_1^T T_{min}, \\ r_{ref}, & T \leq c_1^T T_{min}, \end{cases} \tag{18}$$

where c_1^T and c_2^T are constants and $c_1^T > c_2^T > 1$; r_{ref} is a standard reward value for the whole traversal process and T_{min} is the minimum traversal time in the ideal condition, calculated from the following formula:

$$T_{min} = \min\{\frac{(l-1)mk}{l|v_{max}|} + \frac{(k-1)n}{k|v_{max}|}, \frac{(k-1)nl}{k|v_{max}|} + \frac{(l-1)m}{l|v_{max}|}\}, \tag{19}$$

where $|v_{max}|$ is the magnitude of the maximum velocity of the agent. As the velocity change of the agent is not considered in the T_{min} calculation process, we have $T > T_{min}$.

Equation (17) shows that when $m_i(\gamma'_{x,y}) = 0$, the reward value of $a_i \in \{1, 2, 4, 6, 8\}$ is lower than the reward value of $a_i \in \{3, 5, 7, 9\}$. This is in consideration of the time cost. Assuming that the magnitude of the agent's velocity is constant, and each small rectangular region on the γ-information map is a square, the require time ratio of the two cases is $\sqrt{2} : 1$. However, the traversal effect of the action in both cases is the same, because both of them are traversing one γ point. When the agent repeatedly traverses the γ point, we should give a negative reward, and the more times the traversal is repeated, the greater the negative reward.

After agents traverse the whole γ-information map, we use $R(T)$ to reward the whole traversal process, and $R(T)$ is depicted in Fig. 5. The figure shows that when $c_2^T T_{min} < T < c_1^T T_{min}$, the smaller T, the greater the reward.

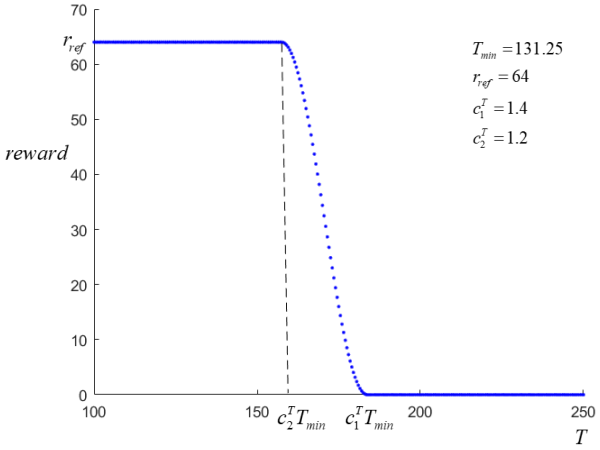


FIGURE 5. R(T) curve for Tmin = 131.25, rref = 64, c_2^T = 1.2, c_1^T = 1.4.

E. Q- VALUE UPDATE AND ACTION SELECTION

To accelerate learning for the MAS, we use the distributed cooperative Q-learning algorithm proposed in [33]. The Q-value table of agent i is updated as follows:

$$\xi_i^\kappa(s_i, a_i) = Q_i^\kappa(s_i, a_i) + \alpha[r_i^\kappa + \lambda \max_{a'_i \in A_i} Q_i^\kappa(s'_i, a'_i) - Q_i^\kappa(s_i, a_i)], \tag{20}$$

$$Q_i^{\kappa+1}(s_i, a_i) = w\xi_i^\kappa(s_i, a_i) + w\sum_{j=1}^{|N_i|} \xi_j^\kappa(s_i, a_i), \tag{21}$$

where α is the learning rate, λ is a discounting factor, $Q_i^{\kappa}(s_i, a_i)$ is the Q-value under s_i and a_i, w is the weight, which satisfies $0 \leq w \leq 1$, and the superscript κ denotes the number of iterations.

To further improve the learning efficiency of Q-learning, we restrict the agent's behavior space according to the γ-information map:

$$A_i' = \{\gamma_{x,y} \in A_i | m_i(\gamma_{x,y}) = 0\}, \tag{22}$$

where A_i' is a subset of A_i. The traversal information of all γ points in A_i' is the same, and their values are zero. For example, in Fig. 2, the optional action space of agent i is $A_i' = \{2, 3, 6, 7, 8, 9\}$. It can be inferred from (22) that an agent prefers to traverse the untraversed γ points around it, which shows that the traversal algorithm based on RL proposed in this paper can complete the whole area coverage even without training.

Obviously, when the γ points of the A_i are all traversed, A_i' will be an empty set. So, we divide the action selection of agent i into the following two cases based on whether A_i' is an empty set.

Case 1 ($A_i' \neq \emptyset$): In this case, typically, the next action selection in Q-learning is based on the principle of the maximum Q-value. The maximum Q-value selects an action as follows:

$$a_i' = \arg\max_{a_i \in A_i'} Q_i(s_i, a_i). \tag{23}$$

Case 2 ($A_i' = \emptyset$): In this case, we find the nearest untraversed γ point on the γ-information map by the following formula:

$$\gamma_{x_1,y_1} = \arg\min_{\substack{\gamma_{x,y} \in M_i(\gamma) \\ m_i(\gamma_{x,y}) \neq 1}} (||p_i^r - \gamma_{x,y}||). \tag{24}$$

We consider γ_{x_1,y_1} as the field source, and its attractive force to $\gamma_{x,y} \in A_i$ is inversely proportional to the square of the distance between them. Therefore, we can choose the best action according to the magnitude of the attractive force, as represented by the following formula:

$$a_i' = \arg\max_{\gamma_{x,y} \in A_i} (||r_{x_1,y_1} - \gamma_{x,y}||^2) \tag{25}$$

The proposed episodic procedure for the distributed traversal algorithm based on Q-learning (herein Q-Traversal) is shown in Algorithm 1.

Algorithm 1 Q-Traversal

```
Set Q-learning parameters α, λ, w, r_ref, etc.
Set MAS motion model parameters r_c, c^α, c_1^γ, c_2^γ, etc.
Set other parameters r_s, m, n, N, etc.
Create state space, behavior space and Q-value.
tables, and γ-information map.
Set the maximum training times N_T.
For each episode do
  Initialize the position and velocity of each agent.
  Initialize state s and action a randomly for agents.
  While (not meet episode end condition) do
    For each agent i
      Calculate u_i, v_i and p_i through the MAS
      motion model.
      Update the γ-information map using (13).
      Calculate the value of last state-action
      r(s_i, a_i) using (17).
      Calculate state s_i' using (14).
      Update the Q-value using (20) and (21).
      Calculate A_i' using (22).
      If A_i' ≠ ∅ do
        Select action a_i' using (23).
      Else do
        Select action a_i' using (25).
      End If
      Update the state and action: s_i = s_i', a_i = a_i'.
    End For
  End While
End For
```

IV. COMPUTATIONAL COMPLEXITY

The main difference among the different dynamic area coverage algorithms is the obtained way of γ points. After an agent is trained with Q-Traversal algorithm, the agent can select the best γ point by querying the trained Q-value table. The process of selecting the best γ point is represented by (23). For the traditional dynamic area coverage algorithm, most of them solve the next γ point according to the current coverage state. The computational complexity of dynamic area coverage algorithm is mainly determined by the solution process of γ point, and the computational complexity of the control model can be ignored.

The solution equations for γ point in the distributed anti-flocking algorithm proposed in [13] is expressed as follows:

$$\xi_i(x, t) = (t - m_i(x))(\rho + (1 - \rho)\lambda_i(x)), \tag{26}$$

$$\lambda_i(x) = e^{(-\sigma_1||p_i - x|| - \sigma_2||p_i^{\gamma} - x||)}, \tag{27}$$

$$p_i^{\gamma}(t+1) = \arg\max_{x \in \tilde{X}_i} \xi_i(x, t), \tag{28}$$

where $\tilde{X}_i$ is the two-dimensional position space of the target area. In applications, we usually convert the target area into a discrete grid map.

Assume that the size of the target area is $m \times n$ and the grid factor is $k_g (k_g > 0, k_g \in Z)$. Then the size of the grid map is $k_g m \times k_g n$, and the number of γ points is $k_g^2 mn$. In the process of solving γ point, (26) and (27) need to be executed cyclically $k_g^2 mn$ times. Equation (27) includes exponentiation and norm operation, and (28) is a process of finding the maximum value in a list of length $k_g^2 mn$.

According to (14), the size of state space of Q-Traversal algorithm can be obtained and denoted by $(kl)^N$. In the calculation process, M_i is regarded as a record of the historical position of agents, and the capacity of the state depends on the size of γ- information map and the number of agents. After all

the states are trained, the Q-value of the current state can be obtained by querying a list of length $(kl)^N$, and then the best γ point can be obtained by selecting the maximum Q-value of action space. Therefore, the computational complexity of Q-Traversal algorithm consists of querying a list and finding the maximum Q-value in action space. In fact, list query and maximum search are composed of multiple comparison operations. The number of comparisons is determined by the length of the list and the type of comparison data.

Based on the above analysis, the computational complexity of the anti-flocking algorithm and Q-Traversal algorithm are obtained, and summarized in Table 2.

TABLE 2. Computational complexity of Q-Traversal algorithm and the anti-flocking algorithm.

	anti-flocking algorithm	Q-Traversal algorithm
Addition/Subtraction	$7k_g{}^2mn$	0
Multiplication/ Division	$4k_g{}^2mn$	0
Exponentiation and Norm operation	$3k_g{}^2mn$	0
Comparison operation	$k_g{}^2mn$ -1	8+$O((kl)^N)$

When k_g is large, k, l, and N are small, the computational complexity of Q-Traversal algorithm will be less than the anti-flocking algorithm. Actually, in the training process of Q-Traversal algorithm, most states are not to be trained, so the length of the Q-value table is much less than $(kl)^N$, and the computational complexity of Q-Traversal algorithm is much less than the expression presented in Table 2.

V. SIMULATION AND EVALUATION PROCESS

In this section, we present the simulation process, including parameter settings and the end conditions for simulation, and then define evaluation parameters for the performance of the algorithm.

A. SIMULATION PROCESS

In the simulation experiment, we set $N = 3$, $r_s = 4.5$ m, $m = n = 50$ m, the simulation time step to 0.5 s and the initialization time of the MAS to 0.3 s. From (12), we obtained $k = l = 8$.

The motion parameters of the agent are shown in Table 3; the distributed cooperative Q-learning parameters are shown in Table 4.

When setting c_r, c_1^T and c_2^T, we need to consider r_c. When $r_c \geq 50$m, communication between agents can be established; to avoid falling into a local optimum, the values of $c_1^T - c_2^T$ and c_r should be set reasonably. When $r_c < 50$m, the smaller r_c, the lower the probability of establishing communication between agents, and the harder it is to obtain a globally optimal traversal strategy. To avoid the Q-Traversal algorithm not converging due to a failure to find a global optimal traversal strategy, the values of $c_1^T - c_2^T$ should be increased, and c_r should be decreased.

TABLE 3. Computational complexity.

Parameters	$\lvert v_{max} \rvert$	d	h	c^α	c_1^γ	c_2^γ
Value	1.0	8	0.1	0.5	0.5	0.8

TABLE 4. Q-learning parameters.

Parameters	α	λ	w	r_{ref}	ε	c_r	N_T
Value	1.0	0.8	0.5	64	0.2	1.0	10^6

In the simulations, we set the termination conditions for the whole training process. The training will be terminated in either of the following two scenarios.

(1) The number of trainings is more than N_T.

(2) $\Delta T_{AVE} \leq 1$, with ΔT_{AVE} defined as follows:

$$\Delta T_{AVE} = \frac{1}{500}\sum_{\kappa=\kappa'+1}^{\kappa'+500}(T^{\kappa+1} - T^{\kappa}), \quad \kappa' \in \{1, 2, \cdots, N_T - 500\} \tag{29}$$

where T^κ is the time for the κth train. Equation (29) shows that when the training time is stable, we assume that the algorithm has converged.

The stop criterion for each training episode for the learning algorithm includes the following two scenarios:

(1) $T^\kappa > 3T_{min}$, $T_{min} = 131.25$ s can be calculated from (19).

(2) $m(\gamma_{x,y}) \neq 0$, $\gamma_{x,y} \in M_1(\gamma)$ and $m(\gamma_{x,y})$ is the fused information map of three γ-information maps, which we can obtain from the following formula:

$$m(\gamma_{x,y}) = \max\{m_i(\gamma_{x,y})\}, \quad i = 1, 2, 3. \tag{30}$$

The first scenario shows that agents do not complete the coverage of the target area in time $3T_{min}$, and the second scenario shows that the agent has completed its coverage of the target area.

B. EVALUATION CRITERION

When the motion parameters of agents are determined, T can evaluate the performance of the coverage algorithm. We define the mean $\overline{T}$ and variance s^2 of the coverage time as performance indicators of the traversal algorithm.

$$\overline{T} = \frac{1}{100}\sum_{\kappa=1}^{100} T^\kappa, \tag{31}$$

$$s^2 = \frac{1}{100}\sum_{\kappa=1}^{100}(T^\kappa - \overline{T})^2. \tag{32}$$

$\overline{T}$ indicates the traversal efficiency of the traversal algorithm, and s^2 indicates the stability of T under different initial conditions.

In the simulation, we compare the coverage performance in free space between Q-Traversal and the anti-flocking algorithm proposed in [13] with $r_c = 60$ m, 50 m, 40 m, 30 m, 20 m and 10 m. In the process of comparison, we only limit its velocity in the anti-flocking algorithm, while other parameters remain unchanged.

VI. SIMULATION RESULTS AND ANALYSIS

In this section, first we give the coverage effect diagram for the anti-flocking algorithm and Q-Traversal algorithm with $r_c = 50$ m. Then we give the test result for the $\overline{T}$ and s^2 of the anti-flocking algorithm and Q-Traversal with different values of r_c. Finally, the results of the convergence experiment and robustness experiment are presented to verify the feasibility of Q-Traversal.

A. COVERAGE EFFECT

In the simulation, we set $r_c = 50$m, $c_r = 1.2$, $c_1^T = 1.18$ and $c_2^T = 1.12$. With these parameters, we obtained the coverage results for the anti-flocking, untrained and trained Q-Traversal algorithms as shown in Fig. 6 (a), (b) and (c), respectively.

Fig. 6 (a) shows that the anti-flocking algorithm can complete the coverage of the target area, but there are repeated coverages of a region in the coverage process, which is similar to the results shown in Fig. 6 (b). This existence of repeated coverage greatly reduces the coverage efficiency. From Fig. 6 (c), when training of the Q-Traversal algorithm is completed, the actions chosen by the agent from the trained Q-value table are mostly horizontal and vertical, which reduces the time cost of oblique motion. In addition, there are almost no crossovers between the trajectories of the agents, which avoids repeated coverage of the area. Therefore, the action strategy of the agent is globally optimal.

To study the coverage rate of the three coverage methods, we recorded the cumulative area coverage at different times and plotted Fig. 7 based on the recorded data. Fig. 7 depicts the relationship between cumulative coverage area and time in the process of coverage. The slope of the curve represents the coverage rate. Fig. 7 shows that the coverage rates of the three methods are not much different at the beginning of the coverage process, but when the time is greater than 80 s, the coverage rates of the anti-flocking and untrained Q-Traversal algorithms are significantly lower than the trained Q-Traversal algorithm. In addition, because there is no repeated coverage, the coverage rate of the trained Q-Traversal algorithm is more stable than the other two.

B. PERFORMANCE AT DIFFERENT COMMUNICATION DISTANCES

In the simulation, we set r_c to 60 m, 50 m, 40 m, 30 m, 20 m and 10 m, and set the values of , c_r, c_1^T and c_2^T based on the value of r_c. These parameters are shown in Table 5.

Fig. 8 gives the performance parameter $\overline{T}$ curves for the anti-flocking algorithm and Q-Traversal algorithm with $r_c = 10$ m, 20 m, 30 m, 50 m and 60 m.

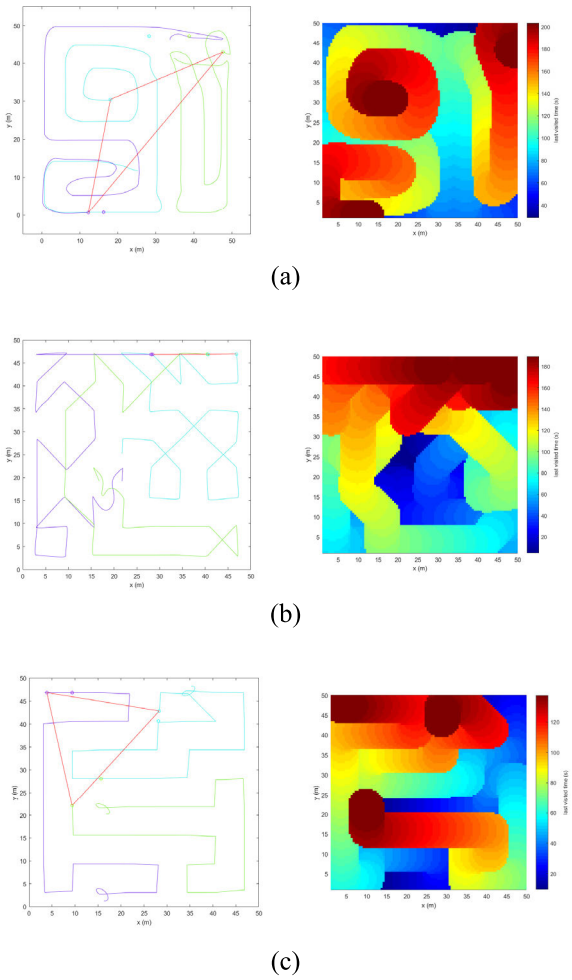


FIGURE 6. Coverage effect diagram. The motion trajectory map is on the left, and different colors represent different agents. Circles and hexagons denote agents and γ points, respectively. A connection between two agents is represented using a red-colored straight line. The coverage time map is on the right, and different colors represent different times. (a) Coverage results for the anti-flocking algorithm in [13]. (b) Coverage results for the untrained Q-Traversal algorithm. (c) Coverage results for the trained Q-Traversal algorithm.

TABLE 5. Value of c_r, c_1^T and c_2^T at different r_c.

r_c (m)	60	50	40	30	20	10
c_1^T	1.18	1.18	1.25	1.37	1.43	1.55
c_2^T	1.12	1.12	1.14	1.21	1.28	1.32
c_r	1.2	1.2	0.95	0.8	0.6	0.5

Fig. 8 shows that when $r_c \geq 20$ m, the $\overline{T}$ of the untrained Q-Traversal algorithm is roughly the same as that of the anti-flocking algorithm, but the $\overline{T}$ of the trained Q-Traversal algorithm is obviously smaller than that of the anti-flocking algorithm. In addition, with the decrease in r_c, the time for the anti-flocking algorithm will increase significantly, while the Q-Traversal algorithm is relatively stable. From the $\overline{T}$ curve for the Q-Traversal algorithm, it can be seen that with the

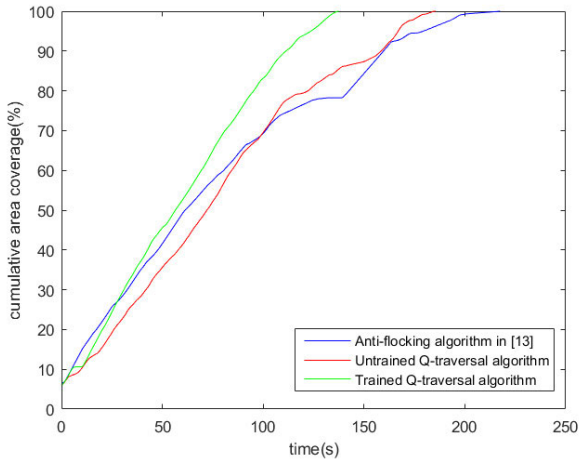


FIGURE 7. Cumulative area coverage of the MAS for the three different methods.

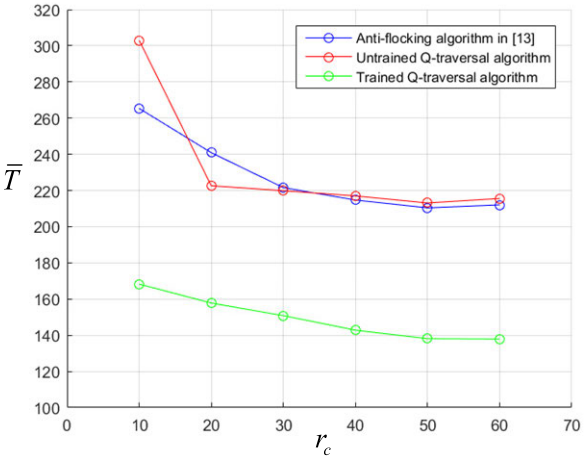


FIGURE 8. $\overline{T}$ of the MAS for the three different methods.

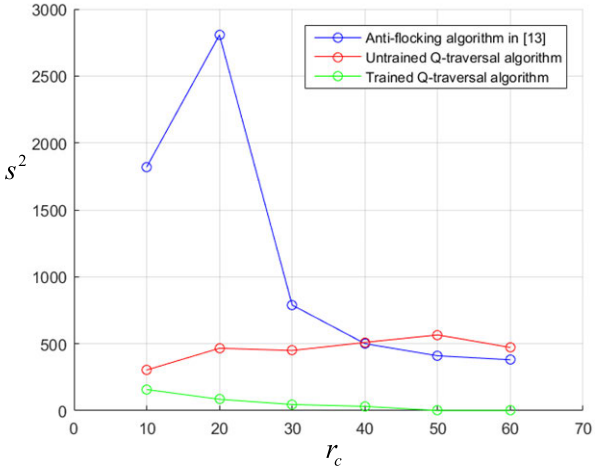


FIGURE 9. Value of s^2 of the MAS for the three different methods.

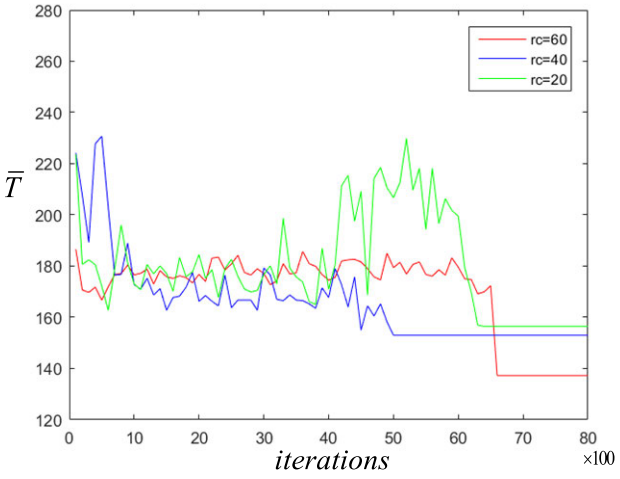


FIGURE 10. Convergence curves of $\overline{T}$ at different values of r_c.

decrease in r_c, $\overline{T}$ will increase slightly. This is due to the lack of communication between agents in some cases, meaning the γ-information map of agents cannot be shared, so the action strategy is the local optimal. When $r_c = 50$ m and 60 m, $\overline{T}$ is close to T_{min}, and the agent can obtain the global optimal action strategy.

Fig. 9 shows the s^2 curves for the three methods with different values of r_c. With different values of r_c, the s^2 of the Q-Traversal algorithm is obviously smaller than the s^2 of the anti-flocking algorithm. This shows that the Q-Traversal algorithm can reduce the impact of the different initial conditions of agents on the traversal effect; when $r_c \geq 50$ m, s^2 is close to 0, and the Q-Traversal algorithm can almost eliminate the impact.

C. CONVERGENCE OF Q-TRAVERSAL

In this experiment, we tested the convergence of Q-Traversal with $r_c = 20$ m, 40 m and 60 m. We set the initial positions of the three agents to (16.75, 19.25), (21.75, 25.25) and (21.75, 19.25), and the parameters of the Q-Traversal algorithm were the same as in Part B. In the experiment, we tested $\overline{T}$ during the training process, and the results of the simulations are given in Fig. 10.

The simulation results show that the Q-Traversal algorithm can achieve convergence at different communication distances. Training the Q-Traversal is a process of finding the optimal coverage strategy. When an optimal strategy under the parameters in Table 5 is found, the algorithm will converge quickly.

In addition, in the simulation process, we find: 1) When $r_c \geq 50$ m, the convergence value of $\overline{T}$ is close to T_{min}, which can be regarded as the global optimal value. 2) When $r_c < 50$ m, it is difficult for $\overline{T}$ to converge on the global optimal value, but it can converge on the local optimal value. 3) When c_2^T is set too small, the optimal coverage strategy is difficult or even impossible to find. When c_1^T is set too large, it is easy to fall into a sub-optimal strategy.

In practical application, when the communication distance is not enough or communication cannot be established, the MAS can also obtain the optimal dynamic coverage path from the off-line Q-Traversal algorithm. Thus, no matter what

situation the MAS is in, the Q-Traversal algorithm allows it to dynamically cover the target area in a globally optimal manner.

D. ROBUSTNESS OF Q-TRAVERSAL

In the robustness simulation, we simulated the situation in which an agent cannot work in the process of dynamic coverage with $r_c = 40$ m. In the simulation, we trained the Q-Traversal algorithm with $N = 2, 3$. We did not run the Q-Traversal algorithm for one agent when $T \geq 100$s, and the agent could be regarded as out of work. We gave the following two solutions for the two cases.

Case 1: Agents in the MAS can recognize that the agent is out of work. In this case, we let the agent query the Q-value table trained with $N = 2$, and we obtained a track of the area coverage shown in Fig. 11 (a).

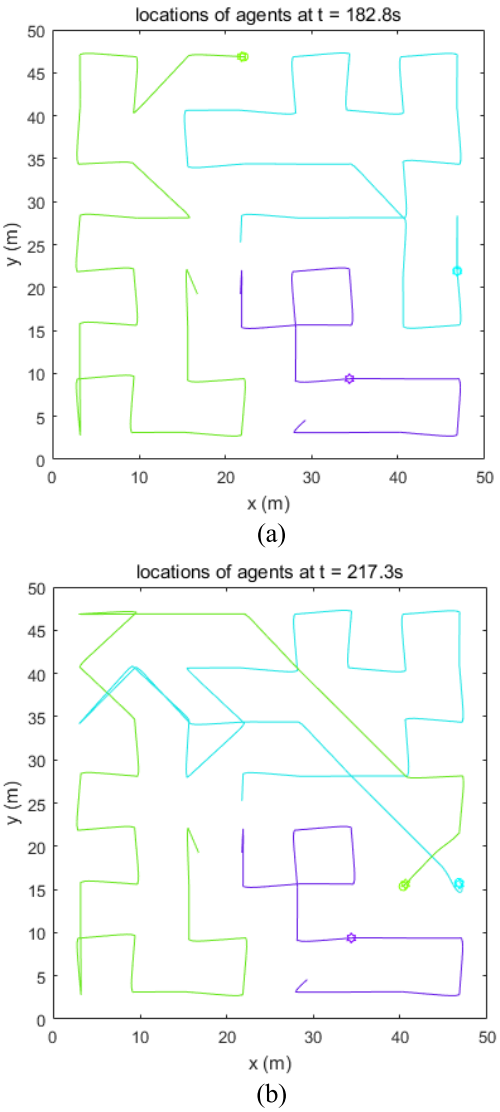


FIGURE 11. Track of area coverage. Different colors represent the coverage trajectories of different agents, where purple is an agent that is out of work when $T > 100$ s. Circles denote agents. (a) is the trajectory for case 1, and (b) is the trajectory for case 2.

Case 2: Agents in the MAS cannot recognize that the agent is out of work. In this case, nothing will be changed, and the track of the area coverage is shown in Fig. 11 (b).

We can see that the coverage effect in Fig. 11 (a) is significantly better than that in Fig. 11 (b), because we have trained the coverage strategy of the two agents.

In the experiment, we verified that the Q-traversal algorithm is robust in practical applications, demonstrating that the Q-traversal algorithm can also complete the task of dynamic coverage efficiently in the event of an emergency situation.

VII. CONCLUSION

In this paper, first we design a MAS motion model based on a flocking algorithm. Then, we construct a γ-information map by gridding the target area. Finally, we propose a free area Q-Traversal algorithm based on the γ-information map and distributed cooperative Q-learning algorithm.

The simulation demonstrated that the MAS can cover the whole target area with the untrained Q-Traversal algorithm; the effects of the untrained Q-Traversal and anti-flocking algorithms were almost the same. When communication between agents can cover the target area, the agent can obtain a global optimal coverage strategy after training with the Q-Traversal algorithm, and the coverage time is close to the minimum time under ideal conditions. When communication cannot cover the whole target area, the agent can obtain a local optimal coverage strategy after training with the Q-Traversal algorithm, and the coverage time is less than with the anti-flocking or untrained Q-Traversal algorithms. In addition, simulation results show that the Q-Traversal algorithm can effectively reduce any instability in the coverage time caused by different initial conditions and is very robust.

In the event that the communication distance is insufficient and online planning is difficult, we can also use the Q-Traversal algorithm for off-line global path planning, providing another global optimal coverage method for area coverage. In these extreme cases, the Q-Traversal algorithm still has great practical value.

REFERENCES

[1] G. Xing, X. Wang, Y. Zhang, C. Lu, R. Pless, and C. Gill, "Integrated coverage and connectivity configuration for energy conservation in sensor networks," *ACM Trans. Sen. Netw. (TOSN)*, vol. 1, no. 1, pp. 36–72, Aug. 2005.

[2] X. Cao, J. Chen, Y. Zhang, and Y. Sun, "Development of an integrated wireless sensor network micro-environmental monitoring system," *ISA Trans.*, vol. 47, no. 3, pp. 247–255, Jul. 2008.

[3] F. Yang, X. Ji, C. Yang, J. Li, and B. Li, "Cooperative search of UAV swarm based on improved ant colony algorithm in uncertain environment," in *Proc. IEEE Int. Conf. Unmanned Syst. (ICUS)*, Beijing, China, Oct. 2017, pp. 231–236.

[4] W. Meng, Z. He, R. Su, P. K. Yadav, R. Teo, and L. Xie, "Decentralized multi-UAV flight autonomy for moving convoys search and track," *IEEE Trans. Control Syst. Technol.*, vol. 25, no. 4, pp. 1480–1487, Jul. 2017.

[5] G. Avellar, G. Pereira, L. Pimenta, and P. Iscold, "Multi-UAV routing for area coverage and remote sensing with minimum time," *Sensors*, vol. 15, no. 11, pp. 27783–27803, Nov. 2015.

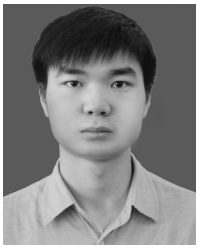
[6] C. Zhai and Y. Hong, "Decentralized sweep coverage algorithm for multi-agent systems with workload uncertainties," *Automatica*, vol. 49, no. 7, pp. 2154–2159, Jul. 2013.

[7] N. Ganganath, C.-T. Cheng, and C. K. Tse, "Distributed anti-flocking control for mobile surveillance systems," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Lisbon, Portugal, May 2015, pp. 1726–1729.
[8] M. Thiene, Z. S. Khodaei, and M. H. Aliabadi, "Optimal sensor placement for maximum area coverage (MAC) for damage localization in composite structures," *Smart Mater. Struct.*, vol. 25, no. 9, Sep. 2016, Art. no. 095037.
[9] V. Mallardo, M. H. Aliabadi, and Z. S. Khodaei, "Optimal sensor positioning for impact localization in smart composite panels," *J. Intell. Mater. Syst. Struct.*, vol. 24, no. 5, pp. 559–573, Mar. 2013.
[10] Q. Wang and J. Huang, "A geometric method for improving coverage in Sensor Networks," in *Proc. Int. Conf. Syst. Informat.*, May 2012, pp. 1111–1115.
[11] A. Agarwal and S. R. Sahoo, "Target centric area coverage control using formation in a multi-agent system," in *Proc. Indian Control Conf. (ICC)*, Guwahati, India, Jan. 2017.
[12] Y.-Q. Miao, A. Khamis, and M. Kamel, "Coordinated motion control of mobile sensors in surveillance systems," in *Proc. 3rd Int. Conf. Signals, Circuits Syst. (SCS)*, Nov. 2009, pp. 1–6.
[13] N. Ganganath, W. Yuan, C.-T. Cheng, T. Fernando, and H. H. C. Iu, "Territorial marking for improved area coverage in anti-flocking-controlled mobile sensor networks," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Florence, Italy, May 2018.
[14] H. Su, X. Wang, and Z. Lin, "Flocking of multi-agents with a virtual leader," *IEEE Trans. Autom. Control*, vol. 54, no. 2, pp. 293–307, Feb. 2009.
[15] N. Ganganath, C.-T. Cheng, C. K. Tse, and X. Wang, "Cluster-based informed agents selection for flocking with a virtual leader," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2015, pp. 2692–2695.
[16] S. H. Semnani and O. A. Basir, "Semi-flocking algorithm for motion control of mobile sensors in large-scale surveillance systems," *IEEE Trans. Cybern.*, vol. 45, no. 1, pp. 129–137, Jan. 2015.
[17] Y.-Q. Miao, A. Khamis, and M. S. Kamel, "Applying anti-flocking model in mobile surveillance systems," in *Proc. IEEE Int. Conf. Auton. Intell. Syst. (AIS)*, Povoa de Varzim, Portugal, Jun. 2010, pp. 1–6.
[18] R. Olfati-Saber, "Flocking for multi-agent dynamic systems: Algorithms and theory," *IEEE Trans. Autom. Control*, vol. 51, no. 3, pp. 401–420, Mar. 2006.
[19] N. Ganganath, C.-T. Cheng, and C. K. Tse, "Distributed antiflocking algorithms for dynamic coverage of mobile sensor networks," *IEEE Trans. Ind. Informat.*, vol. 12, no. 5, pp. 1795–1805, Oct. 2016.
[20] Y. Gao, J. Wang, W. Wu, A. Sangaiah, and S.-J. Lim, "A hybrid method for mobile agent moving trajectory scheduling using ACO and PSO in WSNs," *Sensors*, vol. 19, no. 3, p. 575, Jan. 2019.
[21] Y. Mu, B. Li, D. An, and Y. Wei, "Three-dimensional route planning based on the beetle swarm optimization algorithm," *IEEE Access*, vol. 7, pp. 117804–117813, 2019.
[22] J. Zhan and X. Li, "Flocking of multi-agent systems via model predictive control based on position-only measurements," *IEEE Trans. Ind. Informat.*, vol. 9, no. 1, pp. 377–385, Feb. 2013.
[23] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 1998.
[24] C. J. C. H. Watkins and P. Dayan, "Technical note Q-learning," *Mach. Learn.*, vol. 8, no. 3, pp. 279–292, 1992.
[25] S. Kosunalp, "MAC protocols for energy harvesting wireless sensor networks: Survey," *ETRI J.*, vol. 37, no. 4, pp. 804–812, Aug. 2015.
[26] M. Qiao, H. Zhao, L. Zhou, C. Zhu, and S. Huang, "Topology-transparent scheduling based on reinforcement learning in self-organized wireless networks," *IEEE Access*, vol. 6, pp. 20221–20230, 2018.
[27] A. Konar, I. G. Chakraborty, S. J. Singh, L. C. Jain, and A. K. Nagar, "A deterministic improved Q-learning for path planning of a mobile robot," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 43, no. 5, pp. 1141–1153, Sep. 2013.
[28] Q. Wang, H. Liu, K. Gao, and L. Zhang, "Improved multi-agent reinforcement learning for path planning-based crowd simulation," *IEEE Access*, vol. 7, pp. 73841–73855, 2019.
[29] F. Li, Q. Jiang, S. Zhang, M. Wei, and R. Song, "Robot skill acquisition in assembly process using deep reinforcement learning," *Neurocomputing*, vol. 345, pp. 92–102, Jun. 2019.
[30] E. S. Low, P. Ong, and K. C. Cheah, "Solving the optimal path planning of a mobile robot using improved Q-learning," *Robot. Auton. Syst.*, vol. 115, pp. 143–161, May 2019.
[31] S.-M. Hung and S. N. Givigi, "A Q-learning approach to flocking with UAVs in a stochastic environment," *IEEE Trans. Cybern.*, vol. 47, no. 1, pp. 186–197, Jan. 2017.
[32] H. Shi, S. Yang, K.-S. Hwang, J. Chen, M. Hu, and H. Zhang, "A sample aggregation approach to experiences replay of dyna-Q learning," *IEEE Access*, vol. 6, pp. 37173–37184, 2018.
[33] H. M. La, R. Lim, and W. Sheng, "Multirobot cooperative learning for predator avoidance," *IEEE Trans. Control Syst. Technol.*, vol. 23, no. 1, pp. 52–63, Jan. 2015.
[34] G. Zhang, Y. Li, X. Xu, and H. Dai, "Efficient training techniques for multi-agent reinforcement learning in combat tasks," *IEEE Access*, vol. 7, pp. 109301–109310, 2019.
[35] K. Heong Ang, G. Chong, and Y. Li, "PID control system analysis, design, and technology," *IEEE Trans. Control Syst. Technol.*, vol. 13, no. 4, pp. 559–576, Jul. 2005.

JIAN XIAO received the B.E. degree in electronic information science and technology from the Physics and Electrical Engineering College, Jishou University, Hunan, China, in 2016. He is currently pursuing the M.E. degree in circuits and systems with the University of Electronic Science and Technology. His research interests include adaptive control, intelligent control, and distributed control.

GANG WANG received the B.E. degree in communication engineering and the Ph.D. degree in biomedical engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 1999 and 2008, respectively. He is currently an Associate Professor with the School of Information and Communication Engineering, University of Electronic Science and Technology of China. His current research interests include distributed signal processing and intelligent systems.

YING ZHANG received the Ph.D. degree in detection technology and automatic equipment from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2009. She is currently an Associate Professor with the School of Resources and Environment, UESTC. Her research interests include remote sensing, hyperspectral image processing, and intelligent systems.



LEI CHENG received the B.E. degree in electronic information engineering from the University of South China, in 2018. He is currently pursuing the M.E. degree in electronic and communication engineering with the University of Electronic Science and Technology of China. His research interests include switched systems and machine learning.