Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 7842768, 12 pages
https://doi.org/10.1155/2020/7842768

Research Article

Multi-USV System Cooperative Underwater Target Search Based on Reinforcement Learning and Probability Map

Yuan Liu, Yan Peng, Min Wang, Jiajia Xie, and Rui Zhou

School of Mechatronic Engineering and Automation, Shanghai University, Shanghai 200444, China

Correspondence should be addressed to Min Wang; xmwangmin@shu.edu.cn

Received 23 November 2019; Revised 23 March 2020; Accepted 24 March 2020; Published 14 May 2020

Academic Editor: Javier Martinez Torres

Unmanned surface vehicle (USV) is a robotic system with autonomous planning, driving, and navigation capabilities. With the continuous development of applications, the missions faced by USV are becoming more and more complex, so it is difficult for a single USV to meet the mission requirements. Compared with a single USV, a multi-USV system has some outstanding advantages such as fewer perceptual constraints, larger operation ranges, and stronger operation capability. In the search mission about multiple stationary underwater targets by a multi-USV system in the environment with obstacles, we propose a novel cooperative search algorithm (CSBDRL) based on reinforcement learning (RL) method and probability map method. CSBDRL is composed of the environmental sense module and policy module, which are organized by the "divide and conquer" policy-based architecture. The environmental sense module focuses on providing environmental sense values by using the probability map method. The policy module focuses on learning the optimal policy by using RL method. In CSBDRL, the mission environment is modeled and the corresponding reward function is designed to effectively explore the environment and learning policies. We test CSBDRL in the simulation environment and compare it with other methods. The results prove that compared with other methods, CSBDRL makes the multi-USV system have a higher search efficiency, which can ensure targets are found more quickly and accurately while ensuring the USV avoids obstacles in time during the mission.

1. Introduction

Unmanned surface vehicle (USV) is a robot system with capabilities of autonomous planning, driving, and navigation [1]. It can be carried and deployed to mission area via shore-based transportation or large vessels to independently complete missions such as environmental monitoring [2–4], sampling [5, 6], search, communication [7, 8], harbor protection, and patrol [9–12]. Among these missions, search mission is one of the most suitable missions for USV. Generally speaking, mission areas need to be regulated according to the mission requirements and environmental conditions (such as wind, current, and obstacles). The mission areas are usually large, and a manned vessel needs to take a long time to search the whole areas, so the crew needs to stay on the vessel for a long time. As is known to all, long-term offshore operation is very difficult, inconvenient, and dangerous for people. However, due to its own characteristics, USV can perform a search mission autonomously for a long time in the mission area without manual intervention, which not only greatly improves the search efficiency, but also greatly reduces the work intensity and risk.

Nevertheless, as most USVs are smaller than manned vessels, the space of USV is small and it cannot be equipped with some high-power observation sensors, so the sensing range of USV is limited, which means that the search efficiency of USV per unit time is low. In addition, once USV has some unforeseen accidents in the process of mission (device failure, unavoidable obstacles, strong electromagnetic interference, etc.), the mission must be suspended, which is a negative influence for some urgent situations (such as searching for victims and antimine).

Although improving the performance of USV can solve the above problems to a certain extent, we have to admit that no matter how we improve the performance of USV; the

search efficiency of a single USV is far less than that of a multiple-USV system. Multiple-USV system can enhance the robustness and reliability of USV and provide different policies for various types of missions. In the system, the USV can transmit information to each other and adjust the mission plan at any time according to the process of mission. Once one of the USVs fails to continue working, other USVs can quickly replace it to ensure that the mission not interrupted during the execution.

Although the use of multiple-USV system in search missions is rare, scholars have conducted some research on multiagent search missions. Using a probability map to represent the search regions divided into units and update the probability map through Bayesian rules is a common search method now. Millet et al. [13] propose a distributed search algorithm which includes both map update and fusion procedure. Based on this algorithm, Hu et al. [14] design an algorithm for a cooperative target search mission called coverage control path planning. The main advantage of this method is that it can directly calculate the optimal direction of the agent at the next moment, which improves the search efficiency. However, this method has disadvantages, that is, the number of agents cannot be too small, it cannot escape obstacles, and it is easy to fall into local stability. Apart from this, recent studies have also explored some new methods for the problems of multiagent cooperative search [15, 16]. At the present stage, the control of the multiple-USV system still faces many challenges; therefore, multiple-USV system requires more advanced control methods to enhance the collaboration capability of USV.

In recent years, the continuous development of the RL method has provided new options for solving the problem of target search. The essence of RL is to learn the policy from the interaction between the agent and the environment. In order to overcome the problems of area coverage control, Adepegba et al. [17] combine RL algorithm with the Voronoi-based coverage control algorithm, which uses RL algorithm to approximate the control law. Zhao et al. [18] develop a flocking control framework based on an RL algorithm called DDPG which uses the centralized training and distributed execution framework; it differs from this paper in both the mission and training framework.

In this paper, a novel cooperative search learning algorithm (CSBDRL) based on RL method and probability map method is proposed to apply in the search mission for multiple stationary underwater targets by a multi-USV system in the environment with obstacles. The proposed algorithm is composed of the environmental sense module and policy module, which are organized by the "divide and conquer" policy-based architecture. The environmental sense module focuses on providing environmental sense values. The policy module focuses on how to learn the optimal policy. In CSBDRL, the mission environment is modeled and the corresponding reward function is designed to effectively explore the environment and learning policies. We test CSBDRL in the simulation environment and compare it with other algorithms. The results prove that, compared with other algorithms, CSBDRL makes the multi-USV system have a higher search efficiency, which can make sure targets are found more quickly and accurately while ensuring the USV avoids obstacles in time during the mission.

The rest of this paper is organized according to the structure below. The related background is reviewed in Section 2. In Sections 3–5, we introduce the architecture and each part of the algorithm in detail. The simulation is carried out in Section 6. Finally, the conclusion and future works are given in Section 7.

2. Background

2.1. Multi-USV Cooperation. Compared with a single USV, a multi-USV system can expand sensing range and have wider application in ocean observation, autonomous sampling, search, and other missions [19, 20]. The US Navy completes a mission of underwater mine clearance mission in Iraq using a multi-USV system. The system takes just 16 hours to complete the whole mission, which is originally planned to take 21 days. Throughout the mission, the search areas of the system reach $2.5 \times 10^6\,\mathrm{m}^2$, which successfully improves the mission efficiency and minimized casualties. Arrichiello et al. [21] develop two USVs for a tracking mission. Then, they add another four USVs to form a multi-USV system which has obstacle avoidance capability and follows international navigation traffic rules, greatly improving the mission accuracy. Joseph et al. [22–24] use four low-cost USVs to complete some missions such as port monitoring and environmental monitoring. The Portuguese Institute of Advanced Technology-Systems and Robotics (IST-ISR) [25, 26] carries out a large number of experiments on issues related to the motion control of multi-USV system under different scenarios, and the major research includes collision avoidance under dynamic environments, time-varying communication topology, and effectiveness of the control method.

2.2. RL Method. RL is a kind of learning algorithm that maps from environment state to action, and the goal is to make agent obtain the maximum accumulated reward in their interaction process with environment [27]. Markov decision process (MDP) can be used to model RL problems. MDP is usually defined as a four-tuple (S, A, f, R), where S is a set of all environmental states, and $s_t \in S$ indicates the states of the agent at time t. A is a set of actions that the agent can execute, and $a_t \in A$ indicates the action taken by the agent at time t. $f: S \times A \mapsto [0, 1]$ is the probability distribution function of state transition. $s_{t+1} \sim f(s_t, a_t)$ indicates the probability that the agent performs action a_t in the state s_t and moves to the next state s_{t+1}. $R: S \times A \mapsto \mathbb{R}$ is a reward function. $r_t \sim R(s_t, a_t)$ represents the value of immediate reward obtained by the agent when performing the action a_t in state s_t.

Q-learning algorithm [28] is one of the most widely used RL algorithms. It learns the optimal policy by calculating the state-action value $Q(s, a)$ of different actions taken by the agent in each state. In the Q-learning algorithm, iteration formula (1) is the core of the algorithm:

$$Q_{i+1}(s,a) = E_{S_l \sim S}\left[r + \gamma \max_{a_l} Q(s', a') \mid s, a\right]. \quad (1)$$

However, in practical problems, a large state space makes the calculation cost of formula (1) too high to solve the optimal policy. To solve this problem, Mnih et al. [29, 30] propose a deep Q-network (DQN) algorithm by combining convolution neural network with Q-learning algorithm. DQN algorithm has shown a comparable level to human players in solving complex problems that are similar to real environment, such as Atari 2600 games. In some uncomplicated nonstrategic games, DQN outperforms some experienced human players. On this basis, some improved DQN algorithms are proposed continuously. Van et al. [31] propose a deep double Q-network (DDQN) algorithm based on the double Q-learning algorithm [32]. Bellemare et al. [33] propose an advantage DQN algorithm based on the advantage learning [34]. The advantage DQN algorithm increases the difference between the optimal action value and the suboptimal action value to alleviate the evaluation error caused by selecting the action corresponding to the maximum Q value in the next state every time. Lakshminarayanan et al. [35] propose a dynamic frame skipping DQN (DFDQN), which uses dynamic frame skipping instead of the action that is repeating K times at each moment.

2.3. Probability Map Model. The probability map model uses graphs to represent the joint probability distribution of variables related to the model. Howard et al. first put forward the concept of probability map model. Later, through the efforts of Heckerman et al. [36], the probability map model is developed to a great extent. At present, probability map model is the major method to deal with uncertain data and knowledge in the field of artificial intelligence. When the probability map model is applied to search for targets, the whole search area is divided into cells and associates each cell with the probability or confidence level of the targets in the cells, thus forming the probability map of the whole area [37–40].

An online planning and control algorithm for cooperative search of UAVs is proposed in [41]. In this algorithm, each agent keeps a separate probability map for the whole region and updates the map according to the Dempster–Schafer theory. Millet et al. [42] develop a completely decentralized search algorithm that does not require complete connection. Every agent uses Bayesian rules to update their probability maps obtained through the observation and then fuse with their probability maps of the neighboring agents. Tian et al. [43] propose a cooperative search algorithm combining genetic algorithm (GA) and model predictive control (MPC) to solve the search problem in an uncertain environment. This algorithm uses a probability map to describe the uncertainty of the mission area, takes the gains of the information as the optimization target, and finally uses GA to solve the problem of the optimal control input.

3. Architecture

3.1. Control Architecture. Multi-USV system mainly has two control structures: centralized control and distributed control [44]. Centralized control targets the overall performance of the system and controls all USVs through a central node. Centralized control takes the overall performance of the system as the target value and controls all USVs through a central node. Compared with centralized control, distributed control has no central node, and each USV plans the next action according to its own state to maximize the reward function. Information exchange is used to compensate for the lack of observation capability of the single USV. According to the position distribution of USV in the system, this paper adopts the distributed control structure as shown in Figure 1, which enhances the robustness of the system and makes the system easy to expand.

3.2. Algorithm Architecture. CSBDRL is composed of environmental sense module and policy module, which are organized by the "divide and conquer" policy-based architecture. The environmental sense module focuses on providing environmental sense values which are transmitted to the policy module. The policy module focuses on how to use the environmental sense values that are obtained by the environmental sense module to learn the optimal policy. The architecture diagram of the CSBDRL is shown in Figure 2.

4. Environmental Sense Module

The responsibility of the environmental sense module is to generate the environmental sense values, which will directly affect whether the policy module can learn an effective policy. Each environmental sense value consists of four parts, including target information, collaborative information, obstacle information, and collision avoidance information (in the following discussion, we mainly focus on the policy of the USV in the search process, rather than the underlying motion control of the USV platform, so it is assumed that the USV is a particle in the mission).

4.1. Target Information. Target information is to model uncertain information in a distributed environment. To solve this problem, modeling method based on probability map is widely used. As the mission progresses, the probability map is also dynamically updated according to the corresponding update rules, and it can always reflect the latest USV's understanding of the mission situation, so that the USV can determine the next action based on the mission situation. When generating the probability map model, we assume that there are N USVs in a multi-USV system. Each USV moves in the mission area $A \in \mathbb{R}^2$ which is a rectangular area with length L and width W. As shown in Figure 3(a), we use the top left corner of **A** as the origin to create a coordinate system and then **A** is partitioned into cells. The coordinates of the center position of each cell are expressed as $g = (x, y), x \in \{1, 2, \ldots, L_x\}, y \in \{1, 2, \ldots w_y\}$ and total number of cells is g_j, $j\,\varepsilon\,\{1, 2, \ldots, L_x \times w_y\}$. $\theta_g = 1$ and $\theta_g = 0$ indicate the presence or absence of a target in the cell, respectively. The coordinate of USV_i in the mission area at time k can be described as $\alpha_{i,k}$. We model each cell as Bernoulli distribution, i.e., $\theta_g = 1$ with probability $P(\theta_g = 1)$

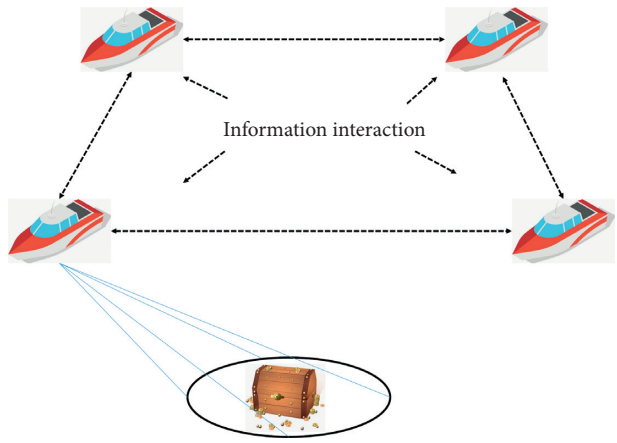

Figure 1: Multi-USV system distributed control architecture.

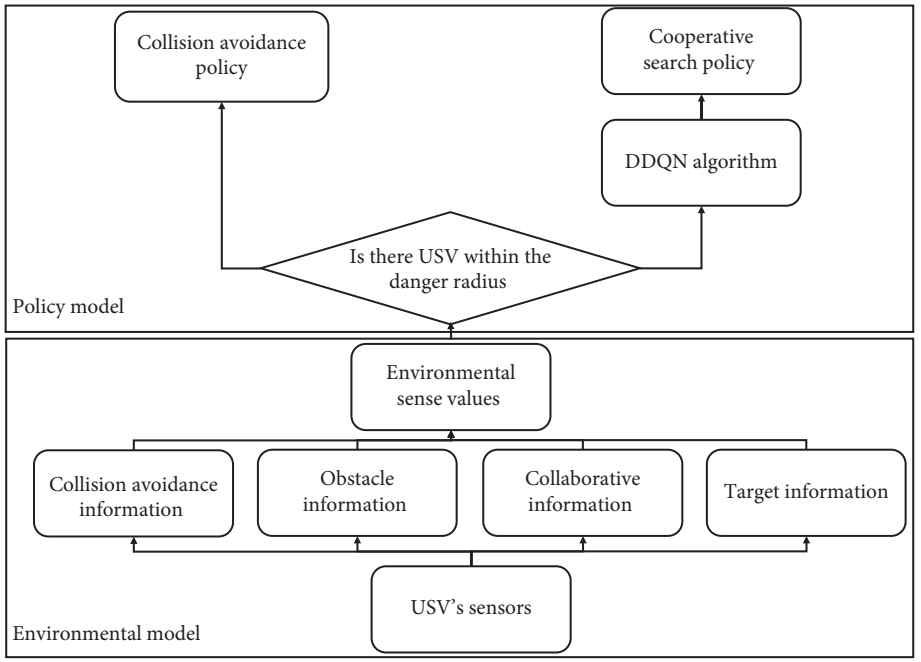

Figure 2: Architecture of CSBDRL.

and $\theta_g = 0$ with probability $1 - P(\theta_g = 1)$. Due to the limited observation capability of the USV, the USV_i at time k can only sample in the sensing region $s_{i,k}$ which is defined by sensing radius R_s, where $S_{i,k} = \{g \varepsilon A: \|g - \alpha_{i,k}\| \leq R_s\}$ and $\|*\|$ denotes the 2-norm for vectors. When the coordinates of the center position of a cell are located in $s_{i,k}$, it is considered to be completely within $s_{i,k}$. The sampling results of USV_i at time k for $s_{i,k}$ are represented by $z_{i,g,k}$, where $z_{i,g,k} = 1$ indicates that a target is detected and $z_{i,g,k} = 0$ indicates that no target is detected. Therefore, $P(z_{i,g,k} = 1 \mid \theta_g = 1) = d$ (detection probability) and $P(z_{i,g,k} = 1 \mid \theta_g = 0) = f$ (false alarm probability) are used to model the sampling process. In summary, each USV_i generates an individual probability map $\mathscr{P}_{i,k} \triangleq \{\mathscr{P}_{i,g,k} \in [0, 1]\}$, where $\mathscr{P}_{i,g,k}$ denotes the estimation of probability of a target existence within cell g by USV_i at time k. The commonly used method of updating the probability map by measurements is based on the Bayesian rule [13], which is given as follows (the initial value of $\mathscr{P}_{i,g,k}$ is set to 0.5 meaning there is no information and should be updated during the search process):

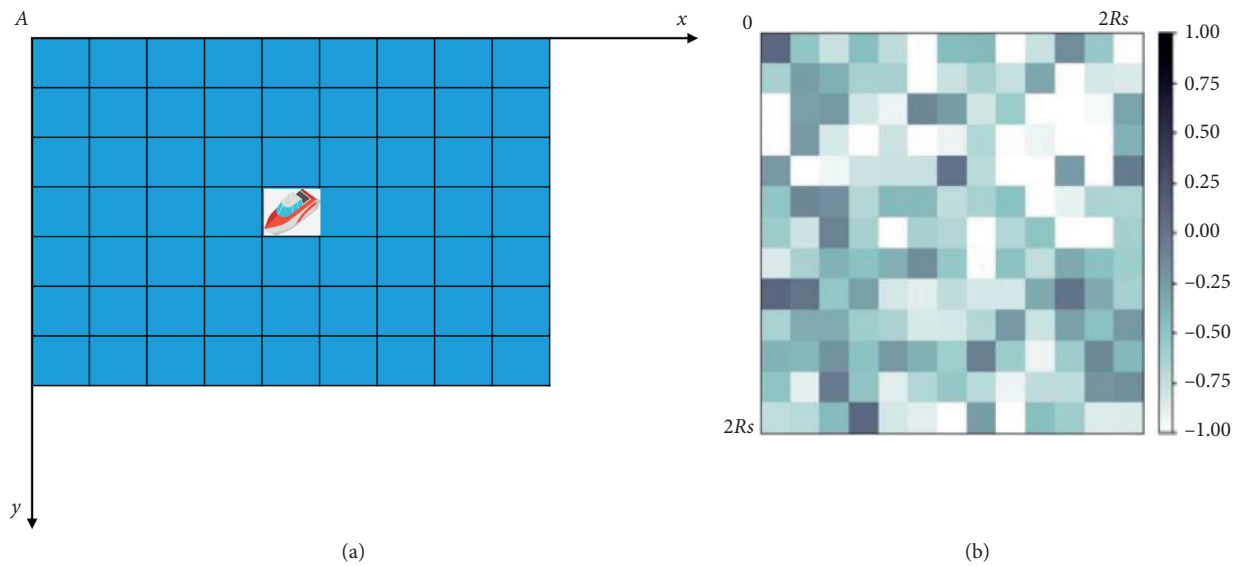

Figure 3: Target Information. (a) Coordinate system. (b) Local w-map centered on the coordinates of the USV.

$$\mathscr{P}_{i,g,k} = \frac{P\left(z_{i,g,k} \mid \theta_g = 1\right)\mathscr{P}_{i,g,k-1}}{P\left(z_{i,g,k} \mid \theta_g = 1\right)\mathscr{P}_{i,g,k-1} + P\left(z_{i,g,k} \mid \theta_g = 0\right)\left(1 - \mathscr{P}_{i,g,k-1}\right)}$$

$$= \begin{cases} \dfrac{d\mathscr{P}_{i,g,k-1}}{d\mathscr{P}_{i,g,k-1} + f\left(1 - \mathscr{P}_{i,g,k-1}\right)}, & \text{if } z_{i,g,k} = 1, \\ \dfrac{(1-d)\mathscr{P}_{i,g,k-1}}{(1-d)\mathscr{P}_{i,g,k-1} + (1-f)\left(1 - \mathscr{P}_{i,g,k-1}\right)}, & \text{if } z_{i,g,k} = 0, \\ \mathscr{P}_{i,g,k-1}, & \text{otherwise.} \end{cases} \tag{2}$$

When $0 < d < 1$ and $0 < f < 1$, then formula (2) becomes

$$\frac{1}{\mathscr{P}_{i,g,k}} - 1 = \begin{cases} \dfrac{d}{f}\left(\dfrac{1}{\mathscr{P}_{i,g,k-1}} - 1\right), & \text{if } z_{i,g,k} = 1, \\ \dfrac{(1-d)}{(1-f)}\left(\dfrac{1}{\mathscr{P}_{i,g,k-1}} - 1\right), & \text{if } z_{i,g,k} = 0, \\ \dfrac{1}{\mathscr{P}_{i,g,k-1}} - 1, & \text{otherwise.} \end{cases} \tag{3}$$

We use the a nonlinear transformation $T_{i,k}$ of $\mathscr{P}_{i,g,k}$ instead of $\mathscr{P}_{i,g,k}$:

$$T_{i,k} \triangleq \ln\left(\frac{1}{\mathscr{P}_{i,g,k}} - 1\right), \tag{4}$$

to facilitate the calculation more efficiently. Therefore, the probability map updating formula can be simplified as follows:

$$T_{i,g,k} = T_{i,g,k-1} + q_{i,g,k}, \tag{5}$$

where (d is detection probability and f is false alarm probability)

$$q_{i,g,k} = \begin{cases} \ln\left(\dfrac{f}{d}\right), & if\ z_{i,g,k} = 1, \\ \ln\left(\dfrac{1-f}{1-d}\right), & if\ z_{i,g,k} = 0. \end{cases} \tag{6}$$

A related proof of convergence is given in [14]. Compared with formula (2), formula (5) converts the nonlinear function into a linear function, which simplifies the calculations, and formula (4) can be recovered uniquely whenever needed. In order to make the USV obtain the global information in a short time, after the USV obtains the probability map, it uses the following formula to fuse the probability map with the neighbors:

$$\tilde{T}_{i,g,k} = \sum_{j=1}^{n} \alpha_{i,j,k} T_{i,g,k}, \tag{7}$$

where n is the number of neighbors, $\alpha_{i,g,k} = 1 - (|Z_{i,k}| - 1/n)$, $\alpha_{i,g,k} = (1/n)$ for $j \in Z_{i,k}\,(j \neq i)$ and $\alpha_{i,g,k} = 0$ for $j \notin Z_{i,k}$. The target information is a local w-map centered on the coordinates of the USV, denoted as $\check{T}_{i,k} = \{\check{T}_{i,g,k}: |x_g - x_{\alpha_{i,k}}| < R_s \text{ and } |y_g - y_{\alpha_{i,k}}| < R_s\}$, x_g and y_g denote the x-coordinate and y-coordinate, respectively. An illustration of $\check{T}_{i,k}$ is shown in Figure 3(b).

4.2. Collaborative Information. The purpose of collaboration information is to make the USV adjust the search area according to the positions of other USVs, thereby preventing multiple-USVs repeatedly searching the same area. USV's collaboration information is related to its communication capability, but this capability is limited, so it can only interact with other USVs within a communication radius R_c. The USVs that can interact with USV_i at time k are called neighbors which are defined as $Z_{i,k} = \{j: \alpha_{j,k} - \alpha_{i,k} < R_c\}$ (including USV_i). Collaborative information is represented by a c-map, which maps the coordinates of the neighbors within the scope of communication range to a matrix of size $2R_c \times 2R_c$, where R_c is a communication radius of the USV. The collaborative map of USV_i at time k is denoted as $C_{i,k} = \{c_{i,g,k} \in \mathbb{R}: \|g - \alpha_{i,k}\| < R_c\}$, where $c_{i,g,k}$ is the superposition of Gaussian distributions centered on the coordinates of the neighbors.

4.3. Obstacle Information. Obstacle information is to make the USV evade local obstacles (other USVs are not included) in the process of mission. Obstacle information is represented by an obstacle -map, which maps the coordinates of obstacles within the scope of USV's obstacle avoidance range into a matrix with size $2R_o \times 2R_o$, where R_o is the obstacle avoidance radius. The obstacle map of USV_i at time k is denoted as $O_{i,k} = \{o_{i,g,k} \in \mathbb{R}: \|\alpha_{i,k} - \text{Obstacle}\| < R_o\}$.

4.4. Collision Avoidance Information. Collision avoidance information is to make the USV evade other USVs in the process of the mission. Collision avoidance information is represented by a collision map, which maps the coordinates of USV within the scope of the USV's collision avoidance range into a matrix with size $2R_d \times 2R_d$, where R_d is the dangerous radius of collision. The collision map of USV_i at time k is denoted as $d_{i,k} = \{d_{i,g,k} \in \mathbb{R}: \|g - \alpha_{i,k}\| < R_d\}$.

5. Policy Module

The responsibility of the policy module is to learn the optimal policy. In the search mission, we must consider how to find a policy to improve the target search efficiency and accuracy of the multi-USV system based on USV's navigation safety. Therefore, a policy learned by the policy module consists of two parts: one is collision avoidance policy and the other is cooperative search policy.

5.1. Action Definition. Policy refers to the mapping from state to action, so we first define the USV's actions. The action range of the USV is discretized. For example, when the degree of discretization M is 8 and the maximum turning angle $\theta_{\max} = 45^\circ$, the action range of the USV can be expressed in Figure 4(a). Assuming at time k, the set of possible positions for USV_i is $\text{LocNext}_i(k)$ and the position of USV_i at time $k+1$ is $\text{Loc}_i(k+1)$; it must be satisfied $\text{Loc}_i(k+1) \in \text{LocNext}_i(k)$ due to the limitation of USV's maneuverability, as shown in Figure 4(b). Assuming that the moving direction of USV_i at time k is east, then $\text{LocNext}_i(k) = \{0, 1, 7\}$, and at this time, if the degree of discretization $M = 16$ and the maximum turning angle $\theta_{\max} = 45^\circ$, then the $\text{LocNext}_i(k) = \{0, 1, 2, 14, 15\}$.

5.2. Collision Avoidance. Collision avoidance policy refers to the USV's position transfer policy when the USV finds other USVs within the danger radius R_d. As shown in Figure 5, assuming that USV_i appears in the danger radius of USV_j at time k, where r_{ij} represents the orientation of USV_i relative to USV_j. If the azimuth of connection line between the USV_i and USV_j is closest to one of the azimuths of the action, r_{ij} is the relative direction of USV_i to USV_j. We stipulate that the collision avoidance policy is to choose the opposite relative direction or_{ij}. The calculation formula of or_{ij} is given by

$$\text{or}_{ij} = \begin{cases} r_{ij} + 2/M, & \text{for } 0 \le r_{ij} \le 2/M, \\ r_{ij} - 2/M, & \text{for } 2/M \le r_{ij} \le M. \end{cases} \tag{8}$$

At the same time, considering that the action of USV_i is restricted by the maximum turning angle $\theta_{\max}$, it may not be able to move directly according to the collision avoidance policy. Since the next transfer position of USV must meet the condition: $\text{Loc}_i(k+1) \in \text{LocNext}_i(k)$, we choose the position that minimizes $\|\text{Loc}_i(k+1) - \text{or}_{ij}\|$ as the next transfer position, where $\|*\|$ is calculated as follows:

$$\|x - y\| = \begin{cases} |x - y|, & \text{for } |x - y| \le 2/M, \\ M - |x - y|, & \text{for } |x - y| \ge 2/M. \end{cases} \tag{9}$$

When N USVs appear within the dangerous radius of USV_i at time k, the position $\text{Loc}_i(k+1)$ that minimizes the value of $\sum_{j=1}^{N} \|\text{Loc}_i(k+1) - \text{or}_{ij}\|$ is selected as the next transfer position, where the calculation rule of $\|*\|$ is determined by formula (9). Figure 5 is a schematic diagram of the collision avoidance policy.

5.3. Cooperative Search Policy. As shown in Figure 6, when no other USVs are detected within the R_d, the USV needs to select a search policy which is to cooperate with other USVs and reduce the global uncertainty to find the targets as quickly as possible. In order to learn the optimal cooperative search policy, we use RL algorithm. DDQN is a classic RL algorithm which is a value-based method and can be easily integrated into the environment.

The state space includes the state of each USV s_{ik}, which consists of target information $\check{T}_{i,k}$, collaborative information $C_{i,k}$, obstacle information $O_{i,k}$, and collision avoidance information $d_{i,k}$. The action space of the USV is {0, 1, 2, 3, 4, 5, 6, 7}, and each number represents an action that has been defined in Section 5.1. The reward function is defined in Section 5.4. The USV's states s_{ik} are input into DDQN

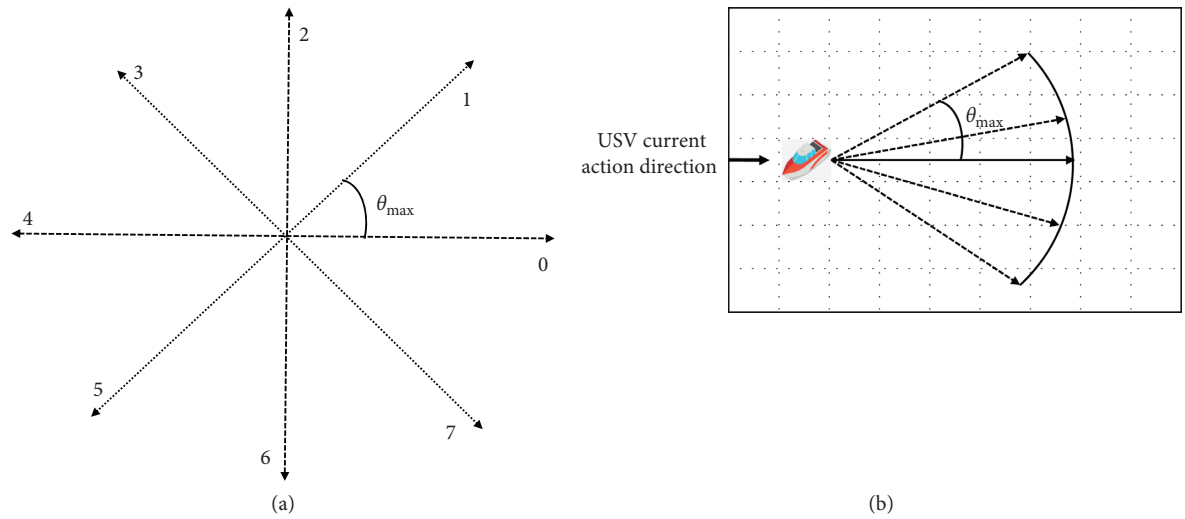

Figure 4: Action definition. (a) Discretized action. (b) The USV's possible positions at next time.

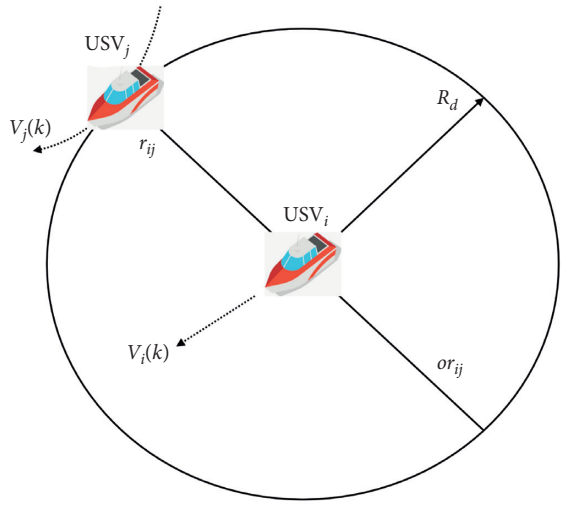

Figure 5: Collision avoidance policy of the USV.

algorithm and then choose the appropriate action a_{ik} to generate a large number of sample data $(o_{ik}, a_{ik}, r_{ik}, o_{ik+1})$ and store them in the replay buffer. When enough sample data are accumulated in the replay buffer, DDQN algorithm randomly extracts samples from the replay buffer, and these sample data are used to learn the cooperative policy. Algorithm 1 gives the training process of DDQN algorithm.

5.4. Reward Function. Reward function is the direct interface between the agent and environment. In the search mission, whether reward function can provide appropriate reward values to the USV according to the USV's state and behavior directly determines whether the algorithm can guide the USV to explore and learn efficiently. The reward function designed here consists of four parts, namely, target reward: $r^{\text{target}}(r^t)$, time consumption reward: $r^{\text{consumption}}(r^c)$, guiding reward: $r^{\text{guide}}(r^g)$, and obstacle reward: $r^{\text{obstacle}}(r^o)$. Therefore, the reward function of USV$_i$ at time k is represented as follows:

$$r_{i,k} = r^t_{i,k} + r^c_{i,k} + r^g_{i,k} + r^o_{i,k}. \tag{10}$$

5.4.1. Target Reward. r^t is a reward that can be obtained when the USV accurately determines the location of the targets. r^t can encourage the USV to discover as many targets as possible while ensuring a certain accuracy. Assuming that there are m targets in the mission area, we stipulate that when $p_{i,gk} \geq \overline{p}$, it indicates that the USV determines the location of a target, where $\overline{p}$ is a threshold. The USV can get a positive reward r^t when a target is found:

$$r^{t1}_{i,k} = \beta_1 \sum_{g \in \mathcal{G}} 1_{p_{i,g,k \geq \overline{p}} \text{ and } p_{i,g,k-1 < \overline{p}}}, \tag{11}$$

if the multi-USV system can correctly determine the location of all targets after an episode, each USV will receive an additional target reward r^{t2}:

$$r^{t2}_{i,k} = \beta_2 \prod_{g \in \{g \in A\,:\,\theta_g = 1\}} 1_{p_{i,g,k \geq \overline{p}}}. \tag{12}$$

β_1 and β_2 are weight coefficients of each part of the reward and can be set empirically as 0.5 and 5.0. Finally, the target reward of USV$_i$ at time k is composed as follows:

$$r^t = r^{t1}_{i,k} + r^{t2}_{i,k}. \tag{13}$$

5.4.2. Time Reward. In order to optimize efficiency, we set up a time reward r^c to encourage the USV to find the targets in a shorter time. r^c is designed to be a piecewise linear function:

$$r^c = \beta_3\left(\overline{k} - k_e + \gamma_1 \max(k_1 - k_e, 0) + \gamma_2 \max(k_2 - k_e, 0)\right), \tag{14}$$

where γ_1 and γ_2 are additional weight coefficients for the current segment and normally set to 1, k_e is the number of episode steps, k_1 and k_2 are preset segmentation points, and

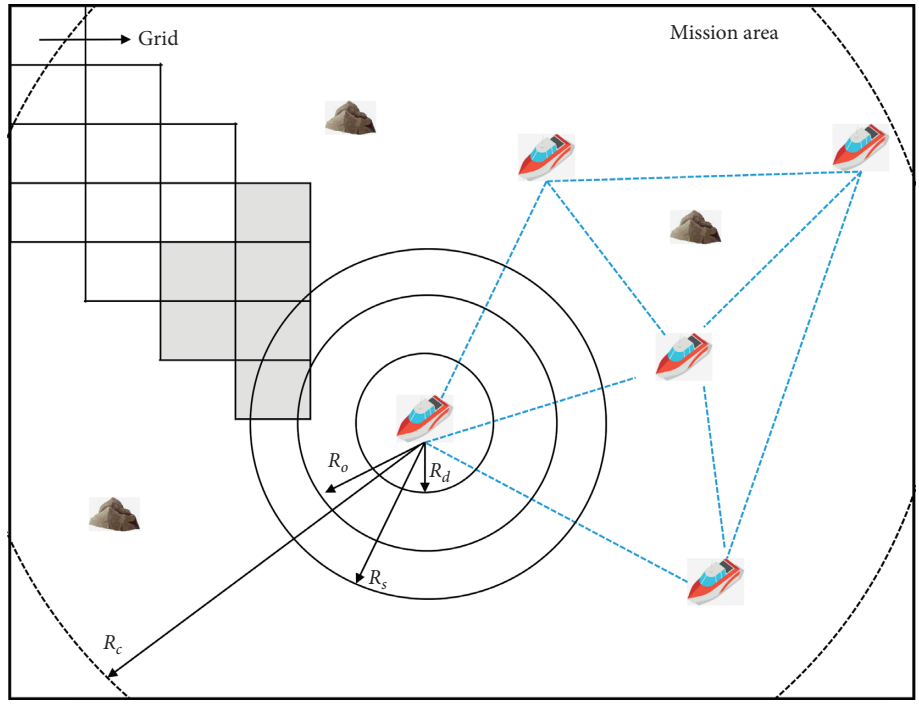

Figure 6: Cooperative search policy of the USV.

(1) Initialize Q-network Q^{θ_i} for the USV, replay buffer $\mathscr{D}$
(2) **for** episode = $1, \overline{e}$ **do**
(3) Initialize the environment, state $s_{i,0}$ and time k
(4) while not ($k \geq \overline{k}$ or targets are found) **do**
(5) for $i \in 1, N$ in parallel **do**
(6) Receive observation $o_{i,k} = \phi(s_{i,k})$
(7) Select action $a_{i,k}$ according to $Q^{\theta_i}(o_{i,k}, a)$
(8) Execute action $a_{i,k}$, receive reward $r_{i,k}$ and reach state $s_{i,k+1}$
(9) Get observation $o_{i,k+1} = \phi(s_{i,k+1})$
(10) Store transition $(o_{i,k}, a_{i,k}, r_{i,k}, o_{i,k+1})$ in $\mathscr{D}$ if USV_i is trained
(11) **end**
(12) Sample random minibatch of transitions (o_j, a_j, r_j, o_{j+1}) from $\mathscr{D}$
(13) Perform a gradient descent step on Q^{θ_i}
(14) Update time $k \longleftarrow k + 1$

Algorithm 1: Training process of the DDQN algorithm.

$\overline{k}$ is the preset maximum number of episode steps when an episode is forced to end. β_3 is a weight coefficient, which is 0.01. Obviously, time consumption reward is the same for all USVs in the system.

5.4.3. Guiding Reward. Since the rewards mentioned above are too scarce in the early stage of training to provide effective guidance for the USV, in order to make the USV have better exploration ability, we reduce the global uncertainty to make the reward as a dense reward. The global uncertainty is defined as $\eta_{i,k} = (1/L_X \times W_y) \sum_{g \in A} i, g, k$, where $\eta_{i,g,k} = e^{-\beta_\eta |T_{i,g,k}|}$, in which β_η is a constant coefficient that set as 2.0. $r^g_{i,k}$ is calculated as follows:

$$r^g_{i,k} = \beta_4 \frac{\eta_{i,k-1} - \eta_{i,k}}{\eta_{i,k-1}}, \tag{15}$$

where β_4 is a weight coefficient which is set as 1.0.

5.4.4. Obstacle Reward. Obstacle reward is used to guide the USV to avoid local obstacles. Different from the above rewards, the value of the obstacle reward is always negative. The size of the obstacle reward is related to the average distance between the USV and obstacles. When USV_i at time k does not detect any obstacle, the obstacle reward is 0. The obstacle reward is defined as follows:

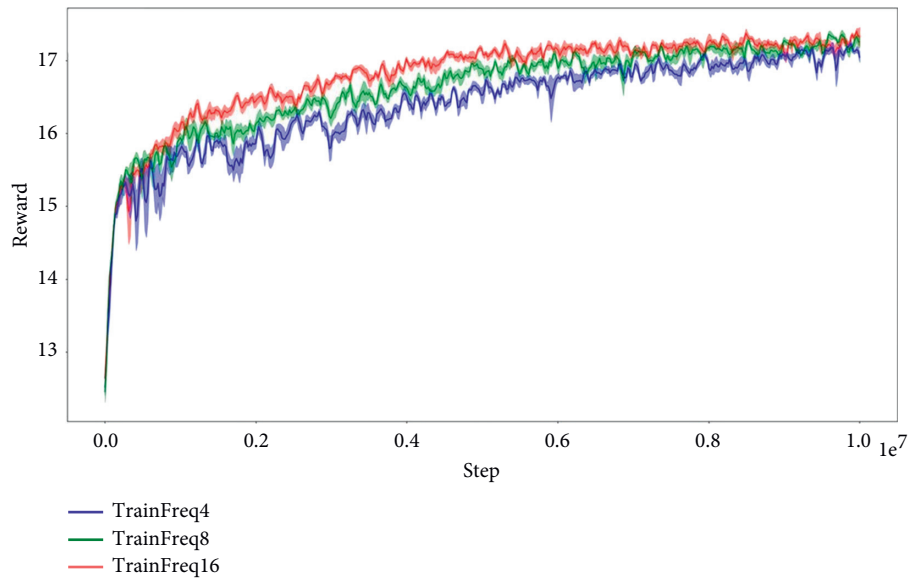

Figure 7: The USV's reward with different training frequencies during the training process.

$$r_{i,k}^{o} = -\frac{\text{distance}\left(\alpha_{i,k}, \text{obstacle}_n\right)}{\beta_\omega}, \tag{16}$$

where obstacle_n represents the coordinates of the nth obstacle, $\text{distance}(\alpha_{i,k}, \text{obstacle}) = |\alpha_{i,} \quad k\text{obstacle}_1| + |\alpha_{i,k}\text{obstacle}_1| + \cdots + |\alpha_{i,k}\text{obstacle}_n|/n$, and β_ω is a constant coefficient which is usually set as 100.

6. Simulation Test

We use Python 3.5 to build a simulation environment on the computer equipped with i7-8700 CPU and GTX 1080ti graphics card for tests. The number of USVs and obstacles can be adjusted according to the needs of the test. The sensing radius R_s is set to 7, the communication radius R_c is set to 9, the danger radius R_d is set to 5, and the obstacle detection radius R_o is set to 3. The detection probability p and false alarm probability q are set to 0.9 and 0.3, respectively. Initial coordinates of the USV and obstacles are random. The total number of training steps is $1e^7$. At the beginning of training, each USV in the system is sampling, moving, and fusing. Figure 7 is the reward of the USV with CSBDRL under different training frequencies during the training.

In test after training, we stipulate that when the system determines the location of all targets or any USV in the system colliding with any obstacle or the search time exceeds 500 steps (the search time is measured by the total number of steps of the multi-USV system), the one-round episode ends. Each test is performed in 500 episodes, and the results are measured by three parameters: search time of the system, search accuracy of the system (the number of episodes that the system correctly determines the location of all targets divided by the total number of episodes), and collision rate of the system (the number of episodes that the collision occurred divided by the total number of episodes). The test parameters are set as shown in Table 1. Figure 8 is a visual interface of the test process, where triangles are represented by USV, red rectangles are represented by obstacles, and black rectangles are represented by targets that are finally determined. As can be seen from Figure 8(b), the USV in the system can avoid obstacles during the process of searching.

Tests 1, 2, and 3 are used to test the impact of the number of USVs on the mission. It can be seen that when the number of obstacles is constant, the search time gradually decreases with the increase of the USVs, proving that CSBDRL effectively enables the USV to learn a cooperative search policy; thus, increasing the number of USVs can increase the search efficiency. In tests 1, 2, and 3, the search accuracy of the system is above 99%, and the collision rate is 0, which proves that the algorithm can effectively find targets in the mission area while ensuring the navigation safety.

Tests 4, 5, and 6 are used to test the impact of the obstacles on the system's capability of collision avoidance. It can be seen that when the number of USVs is constant, the search time gradually increases with the increase of obstacles because obstacles make the USV spend more time to evade. However, the increase of obstacles does not affect the search accuracy of the system. In tests 4, 5, and 6, the search accuracy is still higher than 99%. When the number of obstacles reaches 5, the USVs and obstacles collided, but the number of collisions throughout the test is only 0.2%.

In order to better reflect the performance of the CSBDRL, we compare the algorithm with random algorithm and coverage control algorithm. In the comparison test, there are no obstacles in the mission area because random algorithm and coverage control algorithm do not have the function of obstacle avoidance. We also do 500 episodes for every test. In terms of accuracy, as can be seen from Table 2, CSBDRL has the highest search accuracy and fastest search time, followed by coverage control algorithm, and random algorithm has the lowest search accuracy and slowest search time. When the number of targets increases, the search accuracy of CSBDRL does not change significantly, and the

Table 1: Performance test.

Setup				Search performance		
No. of test	No. of USVs	No. of targets	No. of obstacles	Search time	Search accuracy (%)	Collision rate (%)
Test 1	3	2	1	198	99.2	0.0
Test 2	4	2	1	180	99.3	0.0
Test 3	5	2	1	142	99.5	0.0
Test 4	5	3	2	155	99.5	0.0
Test 5	5	3	3	159	99.4	0.0
Test 6	5	3	4	166	99.5	0.2

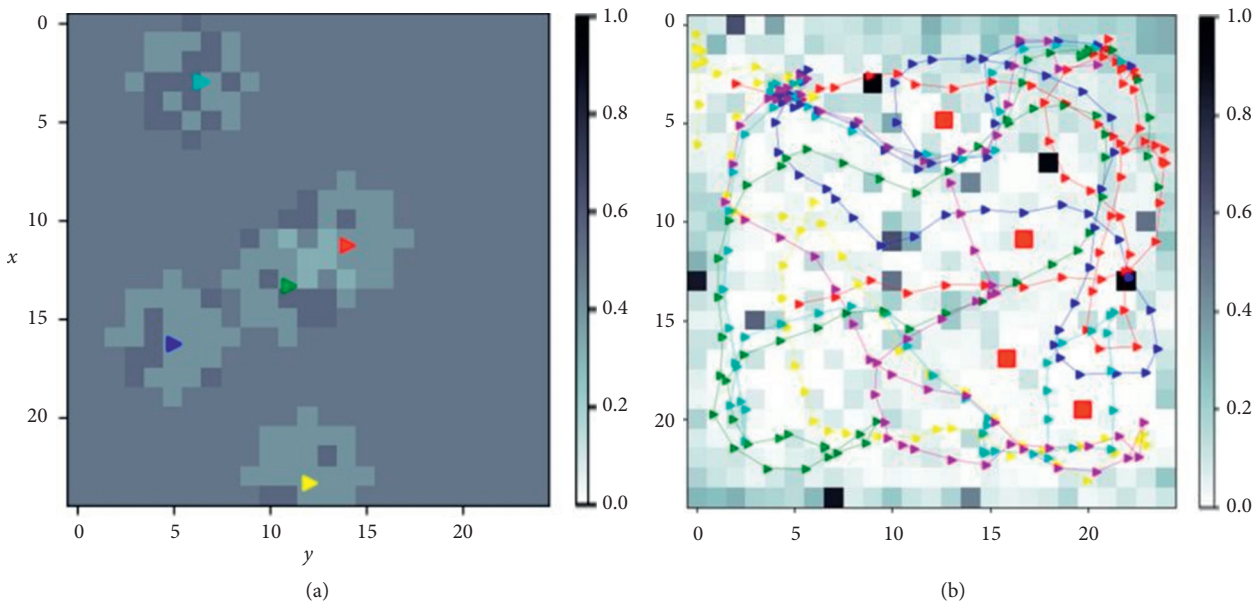

Figure 8: Visual interface of the test process: (a) change of probability map and (b) the USV's trajectory.

Table 2: Contrast test.

Setup			Search performance(search time and search accuracy)		
No. of USVs	No. of targets	No. of obstacles	CSBDRL	Coverage control	Random
5	2	0	131 (100%)	235 (98.3%)	410 (79.4%)
5	3	0	144 (99.6%)	243 (96.3%)	456 (73.3%)
5	4	0	161 (99.7%)	265 (94.0%)	790 (67.2%)

average search accuracy exceeds 99%. The search accuracy of the coverage control algorithm decreases, the performance of the random algorithm is poor, and the search accuracy decreases significantly. In addition, in order to compare the convergence performance, we measure the convergence rates of the uncertainty for the different algorithms. Figure 9 shows the decreasing curve of the uncertainty for the different algorithms. As can be seen from Figure 9, the uncertainty of CSBDRL has converged in 200 steps, which has a great advantage over the random algorithm and coverage control algorithm.

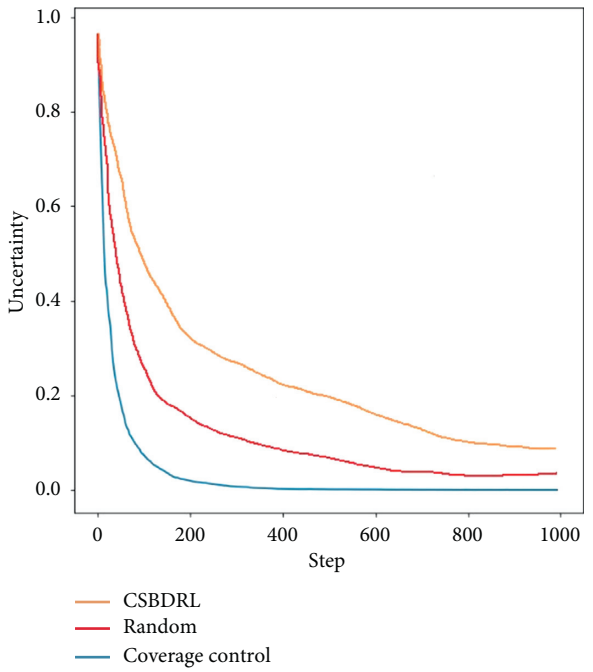

Figure 9: Uncertainty convergence rates of CSBDRL, coverage control, and random.

7. Conclusion

In this paper, we studied the cooperative underwater target search by a group of USVs. A novel cooperative search learning algorithm (CSBDRL) based on RL method and probability map method is proposed to apply in the search mission for multiple stationary underwater targets by a multi-USV system in the environment with obstacles. The algorithm uses the "divide and conquer" policy-based architecture. First, the environmental sense values of the system are produced by environmental sense model based on the probability map method, which is simplified to a linear update and fused. Then, based on these values, the policy module learns the corresponding policy using RL method. We test CSBDRL in the simulation environment and compare it with other algorithms. The results prove that CSBDRL makes the multi-USV system have better search efficiency, which can make sure the targets are found more quickly and accurately in the mission area while guaranteeing the USV's navigation safety in the system. In future research, we will consider the dynamic factors of the USV and the influence of environmental interference factors (such as current and wind) on the USV and improve the algorithm to enable the multi-USV system to search for dynamic targets.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (61403245).

References

[1] R. Tiron, "High-speed unmanned craft eyed for surveillance role," National Defense, Arlington, VA, USA, 2002.

[2] T. C. Furfaro, J. E. Dusek, and K. D. Von Ellenrieder, "Design, construction, and initial testing of an autonomous surface vehicle for riverine and coastal reconnaissance," *OCEANS 2009, MTS/IEEE Biloxi - Marine Technology for Our Future: Global and Local Challenges*, pp. 1–6, IEEE, Piscataway, NJ, USA, 2009.

[3] Anon: 2014, http://www.asvglobal.com/.

[4] W. Naeem, T. Xu, R. Sutton, and A. Tiano, "The design of a navigation, guidance, and control system for an unmanned surface vehicle for environmental monitoring," *Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment*, vol. 222, no. 2, pp. 67–79, 2008.

[5] M. Caccia, M. Bibuli, R. Bono, G. Bruzzone, G. Bruzzone, and E. Spirandelli, "Unmanned surface vehicle for coastal and protected waters applications: the Charlie project," *Marine Technology Society Journal*, vol. 41, no. 2, pp. 62–71, 2007.

[6] M. Caccia, "Modelling and identification of the Charlie2005 ASC," in *Proceedings of the 14th Mediterranean Conference on Control and Automation*, pp. 1–6, Ancona, Italy, June 2006.

[7] G. N. Roberts and R. Sutton, Eds., *Advances in Unmanned Marine Vehicles*, The Institution of Engineering and Technology, London, UK, 2006.

[8] P. Gomes, C. Silvestre, A. Pascoal, and R. Cunha, "A path-following controller for the DELFIMx autonomous surface craft," in *Proceedings of the 7th IFAC Manoeuvring and Control of Marine Craft*, Lisbon, Portugal, 2006.

[9] A. Motwani, "A survey of uninhabited surface vehicles," Technical report. MIDAS.SMSE.2012.TR.001, MIDAS, Mymouth, UK, 2012.

[10] J. Murray, "Sentry—an unmanned swimmer intercept system," QinetiQ North America Inc., Waltham, MA, USA, DTIC Document, 2008.

[11] C. R. Sonnenburg, *Modeling, identification, and control of an unmanned surface vehicle*, Ph.D. thesis, Virginia Polytechnic Institute and State University, Blacksburg, VA, USA, 2012.

[12] W.-R. Yang, C.-Y. Chen, C.-M. Hsu, C.-J. Tseng, and W.-C. Yang, "Multifunctional inshore survey platform with unmanned surface vehicles," *International Journal of Automation and Smart Technology*, vol. 1, no. 2, pp. 19–25, 2011.

[13] T. Millet, D. Casbeer, T. Mercker, and J. Bishop, "Multi-agent decentralized search of a probability map with communication constraints," in *Proceedings of the AIAA Guidance, Navigation, & Control Conference*, Boston, MA, USA, August 2013.

[14] J. Hu, L. Xie, K.-Y. Lum, and J. Xu, "Multiagent information fusion and cooperative control in target search," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 4, pp. 1223–1235, 2012.

[15] T. Bera, R. Bardhan, and S. Suresh, "An integrated decision and control theoretic solution to multi-agent co-operative search problems," 2017, https://arxiv.org/abs/1704.07158.

[16] M. Otte, M. Kuhlman, and D. Sofge, "Competitive target search with multi-agent teams: symmetric and asymmetric communication constraints," *Autonomous Robots*, vol. 42, no. 6, pp. 1207–1230, 2017.

[17] Adepegba, A. Adekunle, S. Miah, and D. Spinello, "Multi-agent area coverage control using reinforcement learning," in *Proceedings of the Twenty-Ninth International Flairs Conference*, Key Largo, FL, USA, May 2016.

[18] X. Zhao, L. Yang, Q. Pan, J. Hu, C. Zhao, and S. Liu, "Multi-vehicle flocking control with deep deterministic policy gradient method," in *Proceedings of the 2018 IEEE 14th International Conference on Control and Automation (ICCA)*, Singapore, November 2018.

[19] N. E. Leonard, D. A. Paley, R. E. Davis, D. M. Fratantoni, F. Lekien, and F. Zhang, "Coordinated control of an underwater glider fleet in an adaptive ocean sampling field experiment in Monterey Bay," *Journal of Field Robotics*, vol. 27, no. 6, pp. 718–740, 2010.

[20] S. Yoon and C. Qiao, "Cooperative search and survey using autonomous underwater vehicles (AUVs)," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 3, pp. 364–379, 2011.

[21] F. Arrichiello, J. Das, H. Heidarsson et al., "Multi-robot collaboration with range-limited communication: experiments with two underactuated ASVs," in *Proceedings of the Field & Service Robotics, Results of the International Conference, FSR*, Cambridge, Massachusetts, USA, July 2009.

[22] J. Curcio, J. Leonard, and A. Patrikalakis, "SCOUT—a low cost autonomous surface platform for research in cooperative

autonomy," in *Proceedings of the OCEANS 2005 MTS/IEEE*, pp. 725–729, Washington, DC, USA, September 2005.

[23] J. Curcio, T. Schneider, M. Benjamin, and A. Patrikalakis, "Autonomous surface craft provide flexibility to remote adaptive oceanomapic sampling and modeling," in *Proceedings of the OCEANS 2008 MTS/IEEE*, pp. 1–7, Quebec City, Canada, September 2008.

[24] T. Schneider, H. Schmidt, T. Pastore, and M. Benjamin, "Cooperative autonomy for contact investigation," in *Proceedings of the OCEANS 2010 MTS/IEEE*, pp. 1–7, Sydney, NSW, Australia, May 2010.

[25] A. P. Aguiar, J. Almeida, M. Bayat et al., "Cooperative control of multiple marine vehicles theoretical challenges and practical issues," *IFAC Proceedings Volumes*, vol. 42, no. 18, pp. 412–417, 2009.

[26] S. Carvalhosa, A. P. Aguiar, and A. Pascoal, "Cooperative motion control of multiple autonomous marine vehicles: collision avoidance in dynamic environments," *IFAC Proceedings Volumes*, vol. 43, no. 16, pp. 395–400, 2010.

[27] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA, USA, 1998.

[28] C. J. C. H. Watkins, "Learning from delayed rewards," *Robotics & Autonomous Systems*, vol. 15, no. 4, pp. 233–235, 1989.

[29] V. Mnih, K. Kavukcuoglu, D. Silver et al., "Playing Atari with deep reinforcement learning," in *Proceedings of the Workshops at the 26th Neural Information Processing Systems 2013*, pp. 201–220, Lake Tahoe, CA, USA, December 2013.

[30] V. Mnih, K. Kavukcuoglu, D. Silver et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[31] H. V. Van, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 2094–2100, Phoenix, AZ, USA, February 2016.

[32] H. V. Hasselt, "Double Q-learning," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 2613–2621, Vancouver, Canada, December 2010.

[33] M. G. Bellemare, G. Ostrovski, A. Guez et al., "Increasing the action gap: new operators for reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 1476–1483, Phoenix, AZ, USA, February 2016.

[34] L. C. Baird III, *Reinforcement Learning through Gradient Descent*, Carnegie Mellon University, Pittsburgh, PA, USA, 1999.

[35] A. S. Lakshminarayanan, S. Sharma, and B. Ravindran, "Dynamic frame skip deep Q network," in *Proceedings of the Workshops at the International Joint Conference on Artificial Intelligence*, New York, NY, USA, 2016.

[36] D. Heckerman, D. Geiger, and D. M. Chickering, "Learning Bayesian networks: the combination of knowledge and statistical data," *Machine Learning*, vol. 20, no. 3, pp. 197–243, 1995.

[37] M. Polycarpou, Y. Yang, and K. Passino, "A cooperative search framework for distributed agents," in *Proceedings of the 2001 IEEE International Symposium on Intelligent Control (ISIC)*, pp. 1–6, Mexico-City, Mexico, September 2001.

[38] L. Bertuccelli and J. How, "Robust UAV search for environments with imprecise probability maps," in *Proceedings of the 44th IEEE Conference on Decision and Control*, pp. 5680–5685, Seville, Spain, December 2005.

[39] Y. Yang, A. Minai, and M. Polycarpou, "Decentralized cooperative search by networked UAVs in an uncertain environment," in *Proceedings of the 2004 American Control Conference*, pp. 5558–5563, Boston, MA, USA, June–July 2004.

[40] L. Stone, *Theory of Optimal Search*, Academic Press, New York, NY, USA, 1975.

[41] Y. Yang, M. M. Polycarpou, and A. A. Minai, "Multi-UAV cooperative search using an opportunistic learning method," *Journal of Dynamic Systems, Measurement, and Control*, vol. 129, no. 5, pp. 716–728, September 2007.

[42] P. T. Millet, D. W. Casbeer, T. Mercker, and J. L. Bishop, "Multiagent decentralized search of a probability map with communication constraints," in *Proceedings of the AIAA Guidance, Navigation and Control Conference*, Toronto, Canada, August 2010.

[43] J. Tian, C. Yan, and L. C. Shen, "Cooperative search algorithm for multiple unmanned aerial vehicles in uncertain environment," *Journal of Electronics and Information*, vol. 29, no. 10, pp. 2325–2328, 2007.

[44] L. Tao, "Research on distributed task allocation and task coordination technology in multi-UCAV cooperative task control," National Defense University of Science and Technology, Changsha, China, 2006.