Article

Multi-AUVs Cooperative Target Search Based on Autonomous Cooperative Search Learning Algorithm

Yuan Liu [†] , Min Wang [†] , Zhou Su, Jun Luo, Shaorong Xie, Yan Peng , Huayan Pu ^{*}, Jiajia Xie and Rui Zhou

School of Mechatronic Engineering and Automation, Shanghai University, Shanghai 200444, China; liuyuanji@shu.edu.cn (Y.L.); xmwangmin@shu.edu.cn (M.W.); zhousuieee@126.com (Z.S.); luojun@shu.edu.cn (J.L.); srxie@shu.edu.cn (S.X.); pengyan@shu.edu.cn (Y.P.); xiejia610@163.com (J.X.); zr901122@sina.com (R.Z.)

^{*} Correspondence: phygood_2001@shu.edu.cn; Tel.: +86-021-6613-6396

[†] Yuan Liu and Min Wang contributed equally to this work.

Received: 10 September 2020; Accepted: 23 October 2020; Published: 26 October 2020

Abstract: As a new type of marine unmanned intelligent equipment, autonomous underwater vehicle (AUV) has been widely used in the field of ocean observation, maritime rescue, mine countermeasures, intelligence reconnaissance, etc. Especially in the underwater search mission, the technical advantages of AUV are particularly obvious. However, limited operational capability and sophisticated mission environments are also difficulties faced by AUV. To make better use of AUV in the search mission, we establish the DMACSS (distributed multi-AUVs collaborative search system) and propose the ACSLA (autonomous collaborative search learning algorithm) integrated into the DMACSS. Compared with the previous system, DMACSS adopts a distributed control structure to improve the system robustness and combines an information fusion mechanism and a time stamp mechanism, making each AUV in the system able to exchange and fuse information during the mission. ACSLA is an adaptive learning algorithm trained by the RL (Reinforcement learning) method with a tailored design of state information, reward function, and training framework, which can give the system optimal search path in real-time according to the environment. We test DMACSS and ACSLA in the simulation test. The test results demonstrate that the DMACSS runs stably, the search accuracy and efficiency of ACSLA outperform other search methods, thus better realizing the cooperation between AUVs, making the DMACSS find the target more accurately and faster.

Keywords: target search; AUV; multi-AUVs system; reinforcement learning

1. Introduction

With the development of technology and oceanic applications, AUV (autonomous underwater vehicle) has played an important role in marine applications. Compared with HOV (human occupied vehicle) and ROV (remotely operated vehicle), AUV is an unmanned agent without cables and can accomplish the work independently, safely and efficiently [1]. Relying on these advantages, AUV has been widely used in minefield search, reconnaissance, and anti-submarine, marine exploration, marine rescue, marine observation [2–5], etc. However, limited operational capability and sophisticated mission environments make a single AUV unable to meet the requirements of high efficiency and large-scale missions. MAS (multi-AUVs system) provides a new method to overcome these difficulties due to the great efficiency and high reliability brought by the space–time distribution and redundant configuration [6]. Compared with a single AUV, MAS has the following characteristics: (1) distribution, including spatial distribution and functional distribution. Spatial distribution is reflected in the fact that AUVs can be distributed in different areas for the operation to improve operational efficiency.

The functional distribution shows that AUVs can carry different sensors and actuators to complete sophisticated missions through cooperation [7]. (2) Redundancy: MAS is redundant in quantity. When an AUV cannot work anymore, other AUVs in the MAS can replace it, ensuring that the mission is not interrupted [8]. The above characteristics make MAS show great advantages in applicability, economy, robustness, scalability, and is particularly beneficial to the larger missions such as underwater target search. The characteristic of the underwater target search is that the mission areas are large, and the targets are randomly distributed within the mission area, which requires the AUV to search the whole area in the shortest time to determine the location of the target [9]. Most MAS adopt centralized control structure. Although this control structure is relatively simple, once the central node fails, the entire system will be paralyzed [10,11]. Then, most search methods at this stage are pre-planning algorithms. This kind of algorithm first rasterizes the mission area and then makes the AUV scan all the grids according to a pre-planned path [12]. Although these search algorithms can accurately locate the target, they cannot produce effective cooperation behavior between AUVs, which will reduce mission efficiency. This shortcoming is unacceptable for some urgent search missions, such as shipwrecked ship searching and locating, anti-mine, and anti-submarine. In addition, limited detection, communication, endurance capability, and changing water depth will cause unexpected situations for AUVs during the mission. The pre-planning algorithm cannot adjust the search strategy based on the real-time situation [13]. Hence, for the current problem, we establish the DMACSS (distributed multi-AUVs collaborative search system) and propose the ACSLA (autonomous collaborative search learning algorithm) that is integrated into the DMACSS. This article mainly has the following contributions:

1. DMACSS and system modeling: we establish the DMACSS and build the dynamic detection and update model of DMACSS with the sophisticated environment (changing depth of water, randomly distributed targets, the uncertainty of the sensors, limited communication capability) based on the probability map model. Using this model, the search environment and the entire search process of the DMACSS are reasonably abstracted. The basis for the rationality of modeling is also provided.

2. Information communication and fusion: a local information fusion mechanism is proposed, which can use the information collected by each AUV in DMACSS to make up for the lack of perception ability of a single AUV. The information fusion mechanism combined with specially designed time stamp mechanism, that can make the information fusion process more effectively under limited communication condition, improve the search efficiency and reduce the error rate.

3. ACSLA: we propose ACSLA and integrate it with DMACSS so that AUVs in the system can get cooperative search strategies. The ACSLA is trained based on RL algorithms [14,15], it has specially designed state information, reward functions, and a new distributed training framework called SASF (single asynchronous sharing framework) to make the training process more stable and easy to converge, which is essential for search performance of DMACSS.

2. Background

In many studies, the problem of multi-agent cooperative search has become a research hotspot in academia and industry and receives widespread attention. Although some researches do not directly use AUV as the research object, it also provides a basis and reference for research on multi-AUVs.

Rajnarayan, D.G.; et al. [16] discuss the application of cooperation theory in the search mission of the multi-agents and use Radner's decentralized cooperation theory to make the optimality of cooperation between agents the same as the global optimality. The CS (centralized strategy) and DCS (distributed collaboration strategy) are all derived. Wang, X.; et al. [17] assign multiple sensors to a set of discrete search units to find hidden targets. To solve the sensor's uncertainty in the detection process, interference is added to the traditional discrete search formula to establish a new mathematical model. Finally, the greedy algorithm is used to optimize and solve this model. Hong, S.P.; et al. [18] combine the Markov chain with the concept of minimizing the undiscovered probability and propose a fast hybrid heuristic intelligent algorithm. Experimental results show that this heuristic

algorithm can complete the search path decision in a short time. Singh [19] et al. propose a hybrid framework for guidance and navigation of swarm of unmanned surface vehicles (USVs) by combining the key characteristics of formation control and cooperative motion planning. Under this framework, a combination of offline planning and online planning is applied to the marine environment. In order to enable the USV to avoid dynamic obstacles, based on the USV maneuvering response time, using the A* algorithm based on offline optimal path planning and safe distance constraints, Mina [20] and others propose a general multi-USV navigation, guidance, and control framework. Thi, H.A.L.; et al. [21] study a hierarchical search planning model, which divides the search area space into several subspaces, and then conducts the second search planning in the search subspace. This secondary search planning method makes the entire search process efficient and precise.

Specific to AUV, Healey, et al. [22] studied the problem of complete coverage in the cooperative search process of multi-AUVs. In order to ensure complete coverage of the search area in the event of AUV loss in the formation, an effective cooperative strategy was designed and tested. Welling, et al. [23] used the multi-AUVs system to perform the cooperative search and target clear missions. They discussed the mission assignment problem and compared the two assignment strategies based on the closest distance and fuzzy logic from a time-consuming perspective. Shafer, et al. [24] studied the multi-AUVs cooperative adaptive search behavior under sophisticated environments and proposed a cooperative strategy that can make the multi-AUVs system achieve multiple missions in parallel. In addition to theoretical research, some multi-AUVs systems have already been applied in practical missions. Since REMUS (Remote Environmental Monitoring Underwater System) underwater robots have played an important role in minefield detection operations during the Iraq War, the ONR (Office of Naval Research) continuously funded several scientific research institutions to develop unmanned underwater systems. The MIT (Massachusetts Institute of Technology) carried out a research project called GOATS (Generic Oceanographic Array Technology System). The GOATS project used multi-AUVs equipped with underwater acoustic equipment to form a mobile underwater detection network which was used to search for mines in coastal waters [25–28]. Based on the GOATS project, the research team consisting of NURC (National Undersea Research Center) and MIT launched a project called Generic Littoral Interoperable Network Technology (GLINT) in 2008. The multi-AUVs system in the project is equipped with various sensors to complete the missions of automatic detection, positioning and tracking of specific targets [29]. In Europe, from 2012 to 2015, relevant research institutes in Italy, Estonia, the United Kingdom, Spain, and Turkey jointly launched a research project called ARROWS (Archaeological Robot System for the World's Seas) [30], which aimed to use multi-AUVs systems to improve the submarine scanning efficiency and research on the mission allocation strategy [31] and underwater communication [32]. At this stage, most MACSS systems adopt a centralized control structure and use a pre-planned method without considering real-time motion characteristics of AUV. To this end, we established the DMACSS system based on the distributed control structure and proposed the ACSLA algorithm that can realize real-time planning of search paths to adapt to complex environments.

3. Modeling

In order to establish DMACSS, we must reasonably simplify and abstract the mission environment according to the actual mission situation and effectively model the mission environment, each part of the system, and the system working process.

3.1. Environment Model

In order to effectively search for targets in the mission environment, DMACSS must keep updating the environment state in terms of limited target information. To this end, the probability map model is used. As we all know, the probability map model is used to model the uncertainty of the mission environment. We assume that the target is on the seafloor. Since the AUV navigates in the water, the complex terrain of the seabed has no effect on the navigation of the AUV, so we project the AUV onto a flat region,

as shown in Figure 1. First, we establish an inertial coordinate system, where α is the coordinate origin, and the positive direction of the x axis points to the east. The mission area of $A \in \mathbb{R}^2$ is divided into $L_x \times L_y$ grids, each grid is called a target area g , $g = (m, n)$, $m \in \{1, 2, 3 \dots, L_x\}$, $n \in \{1, 2, 3 \dots, L_y\}$. Let $\theta_g = 1$ denote that there is a target in the g , otherwise $\theta_g = 0$ indicates that there is no target in the g . Similarly, $\mathcal{P}_{g,k}$ can be expressed as the probability of target existence in the grid g at time k , $\mathcal{P}_{g,k} \in [0, 1]$. Where $\mathcal{P}_{g,k} = 1$ indicates that there must be a target in the g , and $\mathcal{P}_{g,k} = 0$ indicates that there must be no target in the g . Before the start of the mission, each grid has an a priori initial probability, $\mathcal{P}_g = 0.5$. Set the threshold $\mathcal{P}^+$ and $\mathcal{P}^-$ as the upper and lower limits of the $\mathcal{P}_{g,k}$, that is, $\mathcal{P}_{g,k} > \mathcal{P}^+$ means that there is a target in the g . $\mathcal{P}_{g,k} < \mathcal{P}^-$ means that there is no target in the g . The coordinates of each AUV are expressed as $\mu_{i,k} = [p_{i,k}^T, h_{i,k}]^T \in \mathbb{R}^3$ ($i = 1, 2, \dots, N$). $p_{i,k}^T$ are the coordinates of the AUV projection on the A . $h_{i,k}$ is the depth where the AUV is located, T is the conversion operation, N is the total number of AUVs. The kinematics model of AUV is shown in Appendix A.

Figure 1. (a) Autonomous underwater vehicle's (AUV) projection on the region; (b) search area model.

3.2. Sensor Model

At this stage, sonar is the most commonly used sensor for AUV to obtain underwater targets and other environmental information. The active sonar system and passive sonar system are two commonly used sonar systems. In the target search mission, an active sonar system is more suitable as the sensor of the system. Since the detection accuracy of the sonar system will directly affect the update of the probability graph model, in this article, we use sonar as a modeling object and focus on the relationship between the sonar accuracy and the mission environment.

The working principle of the active sonar system makes it susceptible to non-linear interference problems caused by the water media or other external factors. In addition, obstacles between the sonar and the target will also affect the accuracy of sonar. Formula (1) shows the forward-looking sonar detection model after adding constraints of nonlinear noise and obstacle:

$$I_{i,target} = \begin{cases} 0 & d_{i,target} > D_{max} \\ 0 & d_{i,target} < D_{min} \\ 0 & \text{There is an obstacle between sonar and target} \\ h_{target} + d_{i,target} \zeta D_{min} < d_{i,target} < D_{max} \end{cases} \quad (1)$$

Among them, $I_{i,target}$ represents the target information collected by the AUV_i , D_{max} and D_{min} are the maximum and minimum detection distance of the sonar, h_{target} represents the sonar detection

function under noise-free conditions, and $d_{i,target}$ represents the distance between the target and the AUV_i, ζ is nonlinear interference. Formula (1) indicates that when the $d_{i,target}$ not in $[D_{min}, D_{max}]$, or there exist the obstacle between the sonar and target, the target information cannot be obtained. ζ has an important impact on the correctness of the $I_{i,target}$. The larger value of the ζ , the lower the correctness of $I_{i,target}$. We set the probability of AUV_i detecting the correct $I_{i,target}$ as $p_{i,d}$, and the probability of AUV_i detecting the wrong $I_{i,target}$ as $p_{i,f}$:

$$p_{i,d} = P(O_{i,g,k} = 1 | \theta_g = 1) \quad 1 - p_{i,d} = p(O_{i,g,k} = 0 | \theta_g = 1) \quad (2)$$

$$p_{i,f} = P(O_{i,g,k} = 1 | \theta_g = 0) \quad 1 - p_{i,f} = p(O_{i,g,k} = 0 | \theta_g = 0) \quad (3)$$

As shown in Formulas (2) and (3), $p_{i,d}$ is the probability that the AUV_i detects the target when there is a target in the grid g . $p_{i,f}$ is the probability that the AUV_i detects the target when there is not a target in the grid g . $O_{i,g,k}$ is the observation value of g by AUV_i at time k . It can be seen from Formula (1) that the value of the ζ is related to the $d_{i,target}$, so we can get Formulas (4) and (5):

$$p_{i,d} = f_1(d_{i,target}) \quad (4)$$

$$\begin{cases} \widetilde{p_{i,d}} & d_{i,target} < D_{min} \\ f_1(x) = K_1 e^{-K_1(d_{i,target}-D_{min})^2} & D_{min} < d_{i,target} < D_{max} \\ 0.5 & d_{i,target} > D_{max} \end{cases}$$

where $f_1'(d_{i,target}) < 0$ for $d_{i,target} \in (D_{min}, D_{max})$ and $1 > f_1(D_{min}) = \widetilde{p_{i,d}} > f_1(D_{max}) = 0.5$.

$$p_{i,f} = f_2(d_{i,target}) \quad (5)$$

$$\begin{cases} \widehat{p_{i,f}} & d_{i,target} < D_{min} \\ f_2(x) = K_1 e^{K_2(d_{i,target}-D_{min})^2} & D_{min} < d_{i,target} < D_{max} \\ 0.5 & d_{i,target} > D_{max} \end{cases}$$

where $f_2'(d_{i,target}) > 0$ for $d_{i,target} \in (D_{min}, D_{max})$ and $0 < f_2(D_{min}) = \widehat{p_{i,f}} < f_2(D_{max}) = 0.5$.

Remark 1. Due to the sophisticated marine environment (for example, the temperature of seawater, salinity, and ocean currents etc.), the model may change with different locations. In this article, we do not consider this situation, so the sensor model does not change in the whole mission area.

When the targets are located on the plane of the seabed, from Figure 2, we can get the relationship between the $d_{i,target}$, $h_{i,k}$ and H (the water depth at its location) as shown in Formula (6):

$$d_{i,target} = \frac{(H - h_{i,k})}{\cos(\frac{\beta}{2})} \quad (6)$$

where β is the opening angle of the sonar.

Figure 2. Sonar model.

3.3. Environment Update Model

As the mission continues, the system's perception of the environment is constantly changing, so we need to update the environment model based on the latest detection results. $\mathcal{P}_{i,g,k}$ represents the probability that the grid g has a target at time k which is detected by AUV _{i} . Let m_g represent the number of times the g is searched, then we can build Formula (7):

$$\mathcal{P}_{i,g,k} = \begin{cases} 1 & \text{if } m_g \rightarrow \infty \text{ and } \theta_g = 1 \\ 0 & \text{if } m_g \rightarrow \infty \text{ and } \theta_g = 0 \end{cases} \quad (7)$$

Formula (7) indicates that when there is a target in the g and the number of times that the g is searched tends to infinity, $\mathcal{P}_{i,g,k} \rightarrow 1$, that is, there must be a target in the g . When there is no target in the g and the number of times that the g is searched tends to infinity, $\mathcal{P}_{i,g,k} \rightarrow 0$, that is, there must be no target in the g . Next, we update $\mathcal{P}_{i,g,k}$. According to the Bayesian empirical formula, the update formula of $\mathcal{P}_{i,g,k}$ is obtained:

$$\mathcal{P}_{i,g,k} = \frac{p(O_{i,g,k}|\theta_g=1)\mathcal{P}_{i,g,k-1}}{p(O_{i,g,k}|\theta_g=1)\mathcal{P}_{i,g,k-1} + p(O_{i,g,k}|\theta_g=0)(1-\mathcal{P}_{i,g,k-1})} = \begin{cases} \frac{p_{i,d}\mathcal{P}_{i,g,k-1}}{p_{i,d}\mathcal{P}_{i,g,k-1} + p_{i,f}(1-\mathcal{P}_{i,g,k-1})}, & \text{if } O_{i,g,k} = 1 \\ \frac{(1-p_{i,d})\mathcal{P}_{i,g,k-1}}{(1-p_{i,d})\mathcal{P}_{i,g,k-1} + (1-p_{i,f})(1-\mathcal{P}_{i,g,k-1})}, & \text{if } O_{i,g,k} = 0 \\ \mathcal{P}_{i,g,k-1} & \text{otherwise} \end{cases} \quad (8)$$

where $\mathcal{P}_{i,g,k-1}$ represents its a priori probability. Convert Formula (8) from nonlinear to linear form based on the Formula (9) to facilitate the calculation for more efficient:

$$\mathcal{L}_{i,k} \triangleq \ln n \left(\frac{1}{\mathcal{P}_{i,g,k}} - 1 \right) \quad (9)$$

The simplified update formula of probability map is shown in Formula (10):

$$\mathcal{L}_{i,g,k} = \mathcal{L}_{i,g,k-1} + q_{i,g,k} \quad (10)$$

$$q_{i,g,k} = \begin{cases} \ln \left(\frac{p_{i,f}}{p_{i,d}} \right), & \text{if } O_{i,g,k} = 1 \\ \ln \left(\frac{1-p_{i,f}}{1-p_{i,d}} \right), & \text{if } O_{i,g,k} = 0 \end{cases} \quad (11)$$

According to Formula (10), it can be proved that Formula (7) still holds and the proof process is shown in Appendix B. In Appendix C, we deduce the relationship between $p_{i,d}$, $p_{i,f}$, and the convergence rate of the probability map.

3.4. Search Information Fusions

Due to the limitation of sensor performance, each AUV can only observe a limited area, so it can only update the probability map of the area within the observation radius R_s , and because the basic probability map model does not integrate the state information of other AUVs, each AUV cannot grasp the global probability map information. This will negatively affect the search speed of the system. Therefore, to compensate for the limited detection capability of each AUV, the probability map model must be improved by building an information fusion mechanism that makes each AUV in the system faster converge to the same probability map reflecting the target location.

The communication between AUVs is the basis for building the information fusion mechanism. Same as the observation capability of AUV, the communication capability of AUV is also restricted by equipment and environment, the AUV_i can only communicate with AUVs within a communication radius R_c . Therefore, we use the $N_{i,k}$ represent the neighbor of the AUV_i at time k :

$$N_{i,k} = \{j | \{i, j\} \in \|\mu_{i,k} - \mu_{j,k}\| \leq R_c\} \cup \{i\} \quad (12)$$

According to the number of AUVs in the $N_{i,k}$, we can divide $N_{i,k}$ into different levels as $d_{i,k} = |N_{i,k}|$. Using $d_{i,k}$, we can calculate the fusion matrix w : $w_{i,i,k} = 1 - ((d_{i,k} - 1)/N)$, $w_{i,j,k} = (1/N)$ for $j \in N_{i,k}$ ($j \neq i$) and $w_{i,j,k} = 0$ for $j \notin N_{i,k}$. AUV_i searches at time k and stores $\mathcal{L}_{i,g,k}$, then transmit $\mathcal{L}_{i,g,k}$ to neighbors and use Formula (13) to update $\mathcal{L}_{i,g,k}$:

$$\mathcal{L}_{i,g,k} = \mathcal{L}_{i,g,k-1} + \sum_{j \in N_{i,k}} q_{j,g,k} \quad (13)$$

Next, through the $w_{i,j,k}$, the updated $\mathcal{L}_{i,g,k}$ is merged with the $\mathcal{L}_{j,g,k}$ of other AUVs, as shown in Formula (14):

$$\mathcal{H}_{i,g,k} = \sum_{j \in N_{i,k}} w_{i,j,k} \mathcal{L}_{j,g,k} \quad (14)$$

For the entire system, we first define the parameters for the DMACSS:

$$Y_{g,k} \triangleq [\mathcal{H}_{1,g,k}, \mathcal{H}_{2,g,k}, \dots, \mathcal{H}_{N,g,k}]^T \quad (15)$$

$$\Phi_{g,k} \triangleq [q_{1,g,k}, q_{2,g,k}, \dots, q_{N,g,k}]^T \quad (16)$$

$$[w_k]_{i,j} = w_{i,j,k} \quad (17)$$

Then, we can obtain the update rules of information fusion mechanism as shown in Formula (18):

$$Y_{g,k} = \left(\prod_{t=1}^k w_t \right) Y_{g,0} + \sum_{l=1}^k \left(\prod_{t=l}^k w_t \right) \Phi_{g,k} \quad (18)$$

3.5. Time Stamp

In order to improve the efficiency of information fusion, in addition to the information fusion mechanism, we also propose a time stamp mechanism. Specifically, when each AUV fuses the probability map, it is also necessary to transmit a timestamp map. The timestamp can prevent the AUV from repeatedly fusing some information about the determined search grid, and improve search efficiency. Set the timestamp as $t_{i,g}$, which means that the AUV_i has updated the latest probability search map of the grid g . We establish three rules for time stamp mechanism:

(1) When the AUV_i detects the grid at the current time, the update of $\mathcal{P}_{i,g,k}$ comes from its detection behavior. At this time, the current grid g and the timestamp $t_{i,g}$ are updated to the current time k as $t_{i,g,k}$.

(2) When the observation area of the AUV in the system overlaps, the AUV_i 's probability map update comes from the information fusion within its communication range, update the timestamp $t_{i,g}$, which is the timestamp closest to the current AUV_i 's position.

(3) When the AUV fuses information, it not only transmits the probability map information but also needs to interact with the information of the timestamp. When the AUV encounters different timestamps, information fusion will happen.

4. Search Method

In this section, we will introduce the search method that is integrated with DMACSS, so that each AUV in DMACSS can get a cooperative control strategy that can make the AUV obtain the best search trajectory based on the value of real-time detection and maximize system search capability.

Through the tailored design of state information, reward function, and training framework, ACSLA is proposed for the multi-AUVs cooperative target search. In order to compare the effects of different RL algorithms (the RL algorithm is introduced in Appendix D) when training ACSLA, we used the deep Q-network (DQN) algorithm based on value iteration and the deep deterministic policy gradient algorithm (DDPG) algorithm based on policy gradient to train ACSLA, respectively. Below we will describe in detail state information, reward function, and training framework.

4.1. State Information

For the agent, the essence of the strategy is a mapping from state to action, therefore the state information must be able to fully reflect the state of the agent at each step so that the agent can choose the correct action. Using the model we built in the previous chapter, the state information of each AUV in the DMACSS include two pieces of information, one is the target information $\mathcal{H}_{i,k}$ and another is cooperation information $C_{i,k}$:

$$S_{i,k} = \{\mathcal{H}_{i,k}, C_{i,k}\} \quad (19)$$

$\mathcal{H}_{i,k}$ represents the target information which is the probability map of the whole area obtained by $\mathcal{H}_{i,g,k}$:

$$\mathcal{H}_{i,k} = \sum_{g \in \{1,2,\dots,L_x \times L_y\}} \mathcal{H}_{i,g,k} \quad (20)$$

$C_{i,k}$ represents the coordinate information of other AUVs in the $2R_c$ of AUV_i . The role of this information is to make the AUVs have the ability to the cooperative. However, we cannot directly input AUV's coordinate information as cooperation information into the ACSLA, because the coordinate information will be interfered with by the probability map information, the ACSLA cannot extract the corresponding feature information. To this end, we have processed the coordinate information accordingly, that is, converting the coordinate information into a cooperation map of size $2R_c \times 2R_c$:

$$C_{i,k} = \sum_{g \in \|g - \mu_{i,k}\| \leq 2R_c} c_{i,g,k} \quad (21)$$

where $c_{i,g,k}$ is the Gaussian distribution centered on the coordinates of the neighboring of AUV_i . Using the branch structure of the convolutional neural network, the features of the two parts of state information can be extracted separately for the agent to learn.

4.2. Reward Function

The reward function is the core part of the ACSLA. Appropriate reward function can guide agents to learn appropriate strategies, without making the strategy unable to converge or fall into a local optimum. To design a reasonable reward function, we must first clarify the goal of the mission: (1) accurately locate all the targets in the mission area; (2) under the premise of accurately finding all the targets, the search time can be reduced as soon as possible. Among the two indicators, the former reflects the accuracy of the system, and the latter reflects the efficiency of the system. Our reward functions are designed according to these two goals.

4.2.1. Target Reward

We stipulate that when AUV_i finds a target during the search process (only when $\mathcal{P}_{i,g,k} > \overline{\mathcal{P}}$, it is considered that the AUV_i has found the target accurately at grid g , otherwise, it is considered that

there is no target found at grid g), the AUV_i will get a reward. The reward function of the initial target reward $r_{i,k}^{t1}$ is shown in Formula (22):

$$r_{i,k}^{t1} = \sum_{g \in A} 1_{\mathcal{P}_{i,g,k} \geq \bar{\mathcal{P}} \text{ and } \mathcal{P}_{i,g,k-1} < \bar{\mathcal{P}}} \quad (22)$$

When there are multiple targets in the search area, and the system finds all the location of the targets, each AUV in the DMACSS will receive a final target reward r^{t2} :

$$r_{i,k}^{t2} = \prod_{g \in \{g \in A: \theta_g = 1\}} 1_{\mathcal{P}_{i,g,k} \geq \bar{\mathcal{P}}} \quad (23)$$

If the system does not find all the targets, it will be punished accordingly, that is, the system will receive a certain negative reward r^{t3} . The penalty value is related to the number of missed targets. The greater number of the missing target, the greater the penalty value. r^{t3} is shown in Formula (24):

$$r_{i,k}^{t3} = \sum_{g \in A: \theta_g = 1} 1 - \sum_{g \in A} 1_{\mathcal{P}_{i,g,k} \geq \bar{\mathcal{P}} \text{ and } \mathcal{P}_{i,g,k-1} < \bar{\mathcal{P}}} \quad (24)$$

Finally, the target reward of AUV_i at time k is composed as:

$$r^t = \beta_1 r_{i,k}^{t1} + \beta_2 r_{i,k}^{t2} + \beta_3 r_{i,k}^{t3} \quad (25)$$

where $\beta_1, \beta_2, \beta_3$, are weight coefficients.

4.2.2. Dispersed Reward

In order to expand the search range, the AUV should be dispersed as much as possible in the search process to avoid the occurrence of multiple AUVs concentrated in a small area, resulting in a waste of search resources. To this end, we set up a dispersed reward $r_{i,k}^d$. The reward is based on the number of AUVs within the AUV_i 's communication radius to give the AUV_i 's different rewards. The larger the AUV is distributed in the search area, the more dispersed rewards the AUV gets. The distribution information of AUV is obtained by $C_{i,k}$:

$$r_{i,k}^d = \beta_4 \frac{1}{C_{i,k}} \quad (26)$$

In addition, in order to avoid collisions between AUVs, when the distance between AUV_i and other AUVs is less than the obstacle avoidance radius R_a , the AUV_i will receive a larger negative reward $r_{i,k}^{da}$:

$$r_{i,k}^{da} = \begin{cases} -\beta_5 \|\mu_{i,k} - \mu_{j,k}\|^2, & \exists j \|\mu_{i,k} - \mu_{j,k}\| \leq R_o, j \in N_{i,k} (j \neq i) \\ 0, & \text{other} \end{cases} \quad (27)$$

where β_4, β_5 are weight coefficients.

4.2.3. Time Consumption Reward

The time-consuming reward $r_{i,k}^c$ is to evaluate strategy from the perspective of time. $r_{i,k}^c$ is designed in the form of a piecewise function to enable the AUV to obtain corresponding time consumption rewards in different stages of training:

$$r_{i,k}^c = \beta_6 (k_{max} - k_e + \gamma_1 \max(k_1 - k_e, 0) + \gamma_2 \max(k_2 - k_e, 0)) \quad (28)$$

We divide the $r_{i,k}^c$ obtained during the search process into three parts. $k_{max} - k_e$ represents the consumption rewards obtained for the entire segment of the episode. $\gamma_1 \max(k_1 - k_e, 0)$ represents the consumption rewards of the first segment of the episode, $k_2 - k_e$ represents the consumption rewards of the second segment of the episode, where γ_1 and γ_2 are the weight coefficients of different segments, k_{max} is the preset maximum number of episode steps. k_e is the number of the episode step of

AUV_i, k_1, k_2 are the preset segmentation points of the step. $r_{i,k}^c$ is an overall reward, so it is the same for every AUV in the system.

4.2.4. Sparse Reward

In the initial stage of training, the AUV cannot obtain sufficient rewards to learn the search strategy because the AUV cannot find the target immediately. Therefore, to make the AUV have better learning ability in the early stage of training and make the learned strategy converge faster, we set up sparse rewards $r_{i,k}^s$:

$$r_{i,k}^s = \beta_7 \frac{\eta_{i,k-1} - \eta_{i,k}}{\eta_{i,k-1}} \quad (29)$$

where β_4 is the weight coefficient. Formula (29) rewards the AUV according to the uncertainty of each g in the search area. The uncertainty is defined as: $\eta_{i,k} \triangleq \frac{1}{L_x \times L_y} \sum_{g \in A} \eta_{i,g,k}$, where $\eta_{i,g,k} = e^{-\beta_\eta |\mathcal{L}_{i,g,k}|}$, β_η is a constant. $\eta_{i,k}$ will continue to decrease as the search progresses, so it will not affect the later stage of the training.

Finally, the reward function of AUV_i at time k is composed as:

$$r_{i,k} = r^t + r_{i,k}^d + r_{i,k}^{da} + r_{i,k}^c + r_{i,k}^s \quad (30)$$

4.3. Training Framework

An unstable training process is a common problem in MARL (multi-agent reinforcement learning). This is because each agent is part of the environment, so for any agent, the training environment is always changing. To overcome this problem, we design a new distributed training framework called SASF (single asynchronous sharing framework) for DMACSS. The schematic diagram of the SASF is shown in Figure 3. When we train the search strategy of AUV_i , other AUVs in the system use the same strategy τ_i^θ as AUV_i to sample the environment to provide AUV_i with information fusion data, but we only update τ_i^θ in real-time, the strategies used by other AUVs unchanged over a period of steps, so that the training environment can be relatively stable for some time. After a certain number of training steps, AUV_i shares the updated strategy with other AUVs, so that the entire system can improve the search capability together. Compared with the traditional training framework, our training framework can make the training process converge stably, and will not fall into the local optimal trap during the training process. Whether it is a value-based iterative or a gradient-based RL method can use this framework to train ACSLA. The training process of ACSLA for AUV_i is given in Algorithm 1 (trained by DQN) and Algorithm 2 (trained by DDPG). However, this training structure also has a significant shortcoming, that is, because the experience of other AUVs is not fully utilized, it takes long training time to achieve satisfactory results.

Figure 3. Training framework of the distributed multi-AUVs collaborative search system (DMACSS).

Algorithm 1 Training Process of the DQN

Initialize Q-network Q^θ and T-network $Q^{\theta'}$ with $\theta' = \theta$ for the each AUV, replay buffer $\mathcal{D}$, discount factor γ , number of samples in minibatch m , number of episodes $\bar{e}$, the maximum episode step T

For $episode = 1, \bar{e}$ **do**

Initialize sequence $s_{i,k}$ and preprocessed sequence $o_{i,k} = \phi(s_{i,k})$

For $k = 1, T$ **do**

select a random action $a_{i,k}$ with probability ε

Otherwise select action a_k according to $a_{i,k} = \operatorname{argmax}_a Q^\theta(o_{i,k}, a)$

Execute action $a_{i,k}$, receive reward $r_{i,k}$ and get next state $s_{i,k+1}$

Get observation $o_{k+1} = \phi(s_{i,k+1})$

Store transition $(o_{i,k}, a_{i,k}, r_{i,k}, o_{i,k+1})$ in $\mathcal{D}$

Sample random minibatch of transitions $(o_{i,j}, a_{i,j}, r_{i,j}, o_{i,j+1})$ from $\mathcal{D}$

$$y_{i,j} = \begin{cases} r_{i,j} & \text{if episode terminates at step } j+1 \\ r_{i,j} + \gamma \max_{a'} Q^{\theta'}(o_{i,k+1}, a') & \text{otherwise} \end{cases}$$

Perform a gradient descent step on Q^θ

$$(y_{i,j} - Q^\theta(o_{i,j}, a_{i,j}))^2$$

Every C steps reset $Q^{\theta'} = Q^\theta$

End for

End for

Algorithm 2 Training Process of the DDPG

Initialize Actor-network $A(s|\theta^A)$ and critic network $C(s, a|w^C)$ for each AUV with weights θ^A, w^C . Initialize target network A' and C' with weights $\theta^{A'} \leftarrow \theta^A, w^{C'} \leftarrow w^C$. Replay buffer $\mathcal{D}$, discount factor γ , update factor τ , number of samples in minibatch m , w update frequency f , random noise function $\mathcal{N}$, number of episodes $\bar{e}$, the maximum episode step T

for $episode = 1, \bar{e}$ **do**

Initialize the environment, state s_0 and time k

while not ($k \geq \bar{k}$ or targets are found) **do**

for $i \in 1, N$ **in parallel do**

Receive observation $o_{i,k} = \phi(s_{i,k})$

for $k = 1, T$ **do**

Select action $a_{i,k}$ according to $a_{i,k} = A(o_{i,k}|\theta^A) + \mathcal{N}$

Execute action $a_{i,k}$, receive reward $r_{i,k}$ and get next state $s_{i,k+1}$

Get observation $o_{i,k+1} = \phi(s_{i,k+1})$

Store transition $(o_{i,k}, a_{i,k}, r_{i,k}, o_{i,k+1})$ in $\mathcal{D}$

m transitions $(o_{i,j}, a_{i,j}, r_{i,j}, o_{i,j+1})$ are sampled from $\mathcal{D}$

Calculate the target-Q value $y_{i,j}$:

$$y_j = \begin{cases} r_j & \text{if } k \geq T \\ r_{i,j} + \gamma C'(o_{i,k+1}, A'(o_{i,k+1}|\theta^{A'}))|w^{C'} & \text{if } k < T \end{cases}$$

Update w^C :

$$\frac{1}{m} \sum_{j=1}^m (y_{i,j} - C(s, a|w^C))^2$$

Update θ^A :

$$\frac{1}{m} \sum_{j=1}^m \nabla_a C(s, a|w^C) \Big|_{s=s_{i,j}, a=A(s_{i,j})} \nabla_{\theta^A} A(s|w^C) \Big|_{s_{i,j}}$$

Update w^C and θ^A if $k \% f = 1$:

$$w^{C'} \leftarrow \tau w^C + (1 - \tau) w^{C'}$$

$$\theta^{A'} \leftarrow \tau \theta^A + (1 - \tau) \theta^{A'}$$

end for

end for

5. Simulation Test

In all simulations, the size of the entire surveillance region is set to $[0, 25] \times [0, 25]m^2$. The initial position of each AUV is relatively close, ensuring that all AUVs are in a state where they can communicate with each other. The speed of each AUV is uniform and the maximum steering angle of each AUV is θ . We set the $R_s = 6$, $R_c = 8$, $R_d = 2$. The depth change of the search area is shown in Figure 4.

Figure 4. Changes in the depth of the search area.

We use Google's TensorFlow [33] to build a simulation environment based on python. In this article, in order to improve the calculation accuracy of ACSLA, we use a neural network structure called dueling-network which is shown in Figure 5a–c. Dueling-network structure can reduce the calculation error of the algorithm. In terms of parameter update of the neural network, the optimizer is the Adam optimization method [34], the learning rate is set to 2.5×10^{-5} . The replay buffer only stores 10,000 data, the batch size is set to 40, that is, 40 samples are required for one training. The exploration strategy in training adopts the ϵ -greedy method, in which the probability of exploration is selected as 0.2. Figure 5d shows the reward changes of the ACSLA that are trained by different RL methods during the training process.

Environmental information entropy as an evaluation index of algorithm performance. Environmental information entropy can reflect the convergence speed of the probability map, and its calculation formula is as follows:

$$H(k) = - \sum_{g=1}^{L_x \times L_y} (1 - \overline{\mathcal{P}}_{g,k}) \ln(1 - \overline{\mathcal{P}}_{g,k}) \quad (31)$$

$$\overline{\mathcal{P}}_{g,k} = \frac{\sum_{i=1}^N (\mathcal{P}_{i,g,k})}{N} \quad (32)$$

Figure 5. (a) The structure of the actor-network. (b) The structure of the critic network. (c) The structure of the Q-network. (d) Reward change of the ACSLA during the training process.

Scenario 1: This scenario is used to compare the effects when ACSLA is trained by different RL methods (DQN and DDPG algorithms). To better reflect the changes in the probability map and the target distribution in the test, we use the three-dimensional histogram to show the probability map. It can be seen from Figure 6a,b that the final probability maps obtained by the ACSLA (trained by DQN and DDPG algorithms) can accurately reflect the target distribution. Figure 6c is the change graph of the environmental information entropy, it can be seen from Figure 6c that ACSLA trained by the DDPG algorithm can make the probability map converge faster. This is because DDPG adopts the Actor–Critic architecture, an actor-network is used to fit the strategy function and directly output actions. DQN is the algorithm based on the value function, which outputs the Q-value of the action instead of directly outputting the action. The agent also selects the corresponding action according to the Q-value of the action. Therefore, under the same mission environment, the convergence speed of DDPG is faster than DQN, especially when the action space is large, this is more obvious.

(a)

(b)

(c)

Figure 6. (a) The probability map of the deep Q-network (DQN). (b) The probability map of the deep deterministic policy gradient algorithm (DDPG). (c) Information entropy of the DQN and DDPG.

Scenario 2: This test is mainly to compare the effect of different numbers of AUVs on ACSLA. First, we ensure that other parameters in DMACSS remain unchanged, and set the number of AUVs in the system to 3, 4, 5, 6, and 7, respectively. The ACSLA is trained by the DDPG. It can be seen from the test results in Figure 7, when the number of AUVs increases from 3 to 4, the convergence speed of the probability map has been significantly improved. When the number of AUVs increases from 4 to 6, the convergence speed of the probability map does not change significantly. But it is worth noting that when the number of AUVs is increased from 6 to 7, the convergence speed of the probability map has a significant decrease. This is because, although the increasing number of AUVs enhances the system's ability to search for the area, this makes the calculation of the system also significantly increase, which will affect the convergence speed of the algorithm.

Figure 7. Changes in information entropy with different numbers of AUVs.

Scenario 3: In this test, we mainly verify the performance of ACSLA through comparative tests. Here we select some classic search methods such as the random algorithm and coverage control algorithm as the comparison algorithm of ACSLA. In the test, other parameters remain unchanged, the number of AUVs is 4 and the number of targets increases sequentially. The ACSLA is trained by the DDPG. Each algorithm performs 500 complete episodes, using average search accuracy and average search steps to evaluate the performance of different search methods. The test results are shown in Table 1 and Figure 8. It can be seen from Table 1 and Figure 8 that the system using the ACSLA algorithm has the best search performance regardless of the number of targets. The average convergence speed of the ACSLA in the test is 145 steps, and the average search accuracy reaches 99.77%. Compared with the other two algorithms, it has greater advantages in search accuracy and convergence speed.

Table 1. Comparative tests.

Setup		Search Time (Search Accuracy)		
AUVs	Targets	ACSLA	Coverage Control	Random
4	2	131 (100%)	235 (98.3%)	410 (79.4%)
4	3	144 (99.6%)	243 (96.3%)	456 (73.3%)
4	4	161 (99.7%)	265 (94.0%)	790 (67.2%)

Figure 8. Changes in information entropy of the different search methods.

6. Conclusions

In this paper, we establish the DMACSS and propose the ACSLA that is integrated into the DMACSS. Compared with the previous system, DMACSS adopts a distributed control structure to improve the system robustness, and combines an information fusion mechanism and a time stamp mechanism, so that each AUV in the system can exchange and fuse information during the mission, improving the operating efficiency of the system. ACSLA is an adaptive learning algorithm trained by the RL method with a tailored design of state information, reward function, and training framework. We test DMACSS in simulation experiments and compared ACSLA with other cooperative search methods. The test results show that DMACSS runs stably and the search accuracy and efficiency of ACSLA are higher than other search methods, meaning it can better realize the cooperation between AUVs, make DMACSS more accurate and faster to find the target. At the same time, our research still has some problems and shortcomings. DMACSS lacks the ability to search for dynamic targets, and the ACSLA requires a long training time to achieve stable performance. In addition, the fact that the system cannot perform tasks in sophisticated environments (obstacles and other interference) is also a problem we need to solve. We hope that we can gradually resolve these problems in future research.

Author Contributions: Conceptualization, Y.L. and S.X.; methodology, M.W. and Z.S.; software, Y.L. and Z.S.; validation, Y.P.; formal analysis, H.P. and Z.S.; investigation, Y.L. and M.W.; resources, M.W.; data curation, J.L.; writing—original draft preparation, Y.L. and J.X.; writing—review and editing, Y.L. and M.W.; visualization, J.L. and R.Z.; supervision, S.X.; project administration, Y.P.; funding acquisition, H.P. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Natural Science Foundation of China (917481116) and (61922053).

Conflicts of Interest: The authors declare there are no conflicts of interest regarding the publication of this paper.

Appendix A

The kinematics model determines the movement ability of each AUV during the mission. When establishing the AUV's kinematics model, the inertial coordinate system $E\xi\eta\zeta$ and the hull coordinate system $Oxyz$ are usually used to analyze the movement of the AUV. As shown in Figure A1, the pose variables of the AUV are described in the $E\xi\eta\zeta$, and the speed variables of the AUV are described in the $Oxyz$. With the help of the conversion relationship between the two coordinate systems, the position variable is calculated through the speed variable.

When the E (the origin of the $E\xi\eta\zeta$) coincides with the O (the origin of the $Oxyz$), the rotation is expanded in the order of $\psi \rightarrow \varphi \rightarrow \theta$ (ψ, φ, θ are the transverse inclination, longitudinal inclination and

bow angle of $Oxyz$ relative to $E\xi\eta\zeta$ according to Euler's theorem. After three rotations, the coordinate vector in the $Oxyz$ can coincide with the $E\xi\eta\zeta$. Then a certain position vector of the AUV in the $E\xi\eta\zeta$ is recorded as $z = (x \ y \ z)^T$, and the coordinate mark under the $E\xi\eta\zeta$ is $(\xi \ \eta \ \zeta)^T$. According to the principle of coordinate system conversion, the following coordinate conversion relationship can be obtained:

$$\begin{bmatrix} \xi \\ \eta \\ \zeta \end{bmatrix} = S \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad (\text{A1})$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = S^{-1} \begin{bmatrix} \xi \\ \eta \\ \zeta \end{bmatrix} = S^T \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad (\text{A2})$$

where S is the conversion matrix. Similarly, if in the $E\xi\eta\zeta$, the angular velocity of each direction can be denoted as $\dot{\lambda} = (\dot{\varphi}, \dot{\theta}, \dot{\psi})^T$, then in the $Oxyz$, it will be denoted as $\Omega = (p \ q \ r)^T$, then the resulting conversion relationship is as follows:

$$\dot{\lambda} = C\Omega \quad (\text{A3})$$

where C is the conversion matrix. In combination with the above, the AUV's position vector is recorded as $\eta = [\xi, \eta, \zeta, \varphi, \theta, \psi]$, and the velocity vector is recorded as $v = [u, v, w, p, q, r]^T$ (u, v, w are the sway, insult, and heave of the AUV in the $Oxyz$), therefore, the vector form of the AUV motion model can be obtained:

$$\dot{\eta} = J(\eta)v \quad (\text{A4})$$

where $J(\eta)$ is expressed as follows:

$$J(\eta) = \begin{bmatrix} S & 0_{3 \times 3} \\ 0_{3 \times 3} & C \end{bmatrix} \quad (\text{A5})$$

After expanding the kinematics vector, the following conversion form can be obtained:

$$\left\{ \begin{array}{l} \dot{\xi} = u \cos \psi \cos \theta + v(\cos \psi \sin \theta \sin \varphi - \sin \psi \cos \varphi) + w(\cos \psi \sin \theta \cos \varphi + \sin \psi \sin \varphi) \\ \dot{\eta} = u \sin \psi \cos \theta + v(\sin \psi \sin \theta \sin \varphi + \cos \psi \cos \varphi) + w(\sin \psi \sin \theta \cos \varphi - \cos \psi \sin \varphi) \\ \dot{\xi} = -u \sin \theta + v \cos \theta \sin \varphi + w \cos \theta \cos \varphi \\ \dot{\varphi} = p + q \sin \varphi \tan \theta + r \cos \varphi \tan \theta \\ \dot{\theta} = q \cos \varphi - r \sin \varphi \\ \dot{\psi} = q \sin \theta / \cos \theta + r \cos \varphi \cos \theta \end{array} \right. \quad (\text{A6})$$

Since this article aims to propose an effective multi-AUVs target collaborative search method, we ignore the influence of roll q and pitch p on the execution of the target search state of the AUV. In addition, the AUVs in the system are all operating at a fixed depth, and the depth is not changed during the operation, so the movement in the ζ direction is ignored. The final kinematics model of the AUV can be simplified as:

$$\left\{ \begin{array}{l} \dot{\xi} = u \cos \psi - v \sin \psi \\ \dot{\eta} = u \sin \psi + v \cos \psi \end{array} \right. \quad (\text{A7})$$

Figure A1. AUV's hull coordinate system and dynamic coordinate system.

Appendix B

Suppose that the total number of detections of AUV_i on grid g is set to $m_{i,g}$. $O_{i,g,k}$ is the observation value of g by AUV_i at time k . Set the number of times $O_{i,g,k} = 1$ to $a_{i,g}$ and $O_{i,g,k} = 0$ to $m_{i,g} - a_{i,g}$. Put the above result into Formula (10) to get Formula (A8):

$$\mathcal{L}_{i,g,k} = \mathcal{L}_{i,g,0} + a_{i,g} \ln \frac{p_{i,f}}{p_{i,d}} + (m_{i,g} - a_{i,g}) \ln \frac{1 - p_{i,f}}{1 - p_{i,d}} \quad (\text{A8})$$

If there is a target in g , the AUV_i 's observation result is a random variable that obeys the binomial distribution: $p_{i,d} = P(O_{i,g,k} = 1 | \theta_g = 1)$, $1 - p_{i,d} = P(O_{i,g,k} = 0 | \theta_g = 1)$. Knowing that each detection process is independent and identically distributed, we can obtain Formula (A9) according to the law of large numbers:

$$\frac{a_{i,g}}{m_{i,g}} \rightarrow p_{i,d} \quad (\text{A9})$$

among them $m_{i,g} \rightarrow \infty$, divide both sides of Formula (A8) by $m_{i,g}$, to get Formula (A10):

$$\frac{\mathcal{L}_{i,g,k}}{m_{i,g}} = \frac{\mathcal{L}_{i,g,0}}{m_{i,g}} + \frac{a_{i,g}}{m_{i,g}} \ln \frac{p_{i,f}}{p_{i,d}} + \left(1 - \frac{a_{i,g}}{m_{i,g}}\right) \ln \frac{1 - p_{i,f}}{1 - p_{i,d}} \quad (\text{A10})$$

among them:

$$p_d \ln \frac{p_{i,f}}{p_{i,d}} + (1 - p_{i,d}) \ln \frac{1 - p_{i,f}}{1 - p_{i,d}} < 0 \quad (\text{A11})$$

i.e., $\mathcal{L}_{i,g,k} \rightarrow -\infty$, according to the linear change formula, $\mathcal{P}_{i,g,k} \rightarrow 1$.

If there is not a target in g , i.e., $\theta_g = 0$: $p_{i,f} = P(O_{i,g,k} = 1 | \theta_g = 0)$, $1 - p_{i,f} = P(O_{i,g,k} = 0 | \theta_g = 0)$. Knowing that each detection process is independent and identically distributed, we can obtain Formula (A12) according to the law of large numbers:

$$\frac{a_{i,g}}{m_{i,g}} \rightarrow p_{i,f} \quad (\text{A12})$$

among them: $m_{i,g} \rightarrow \infty$. Divide both sides of Formula (A8) by $m_{i,g}$, to get Formula (A13):

$$\frac{\mathcal{L}_{i,g,k}}{m_{i,g}} = \frac{\mathcal{L}_{i,g,0}}{m_{i,g}} + \frac{a_{i,g}}{m_{i,g}} \ln \frac{p_{i,f}}{p_{i,d}} + \left(1 - \frac{a_{i,g}}{m_{i,g}}\right) \ln \frac{1 - p_{i,f}}{1 - p_{i,d}} \quad (\text{A13})$$

among them:

$$p_{i,d} \ln \frac{p_{i,f}}{p_{i,d}} + (1 - p_{i,d}) \ln \frac{1 - p_{i,f}}{1 - p_{i,d}} > 0 \quad (\text{A14})$$

i.e., $\mathcal{L}_{i,g,k} \rightarrow +\infty$, according to the linear change formula, $\mathcal{P}_{i,g,k} \rightarrow 0$.

Appendix C

Since the search process follows an independent binomial distribution, then in m observations, the probability of the existence of the target in k detection results is expressed by Formula (A15):

$$\mathcal{P}_m^k = C_m^k p^k (1-p)^{m-k} \quad (\text{A15})$$

where p is the probability of target existence. By updating the model, substitute Formula (8) to get the $\mathcal{P}_g^m$ that all detection results are existence in the grid g :

$$\mathcal{P}_g^m = \frac{(p_{i,d})^m \mathcal{P}_g^0}{(p_{i,d})^m \mathcal{P}_g^0 + (p_{i,f})^m (1 - \mathcal{P}_g^0)} \quad (\text{A16})$$

Given a hyperparameter: B^+ , when $P_g^m > T^+$, we can get:

$$\frac{(p_{i,d})^m \mathcal{P}_g^0}{(p_{i,d})^m \mathcal{P}_g^0 + (p_{i,f})^m (1 - \mathcal{P}_g^0)} > T^+ \quad (\text{A17})$$

Obtained by transformation:

$$m_{min}^+ > \frac{\log\left(\frac{\mathcal{P}_g^0(1-T^+)}{B^+(1-\mathcal{P}_g^0)}\right)}{\log \frac{p_{i,f}}{p_{i,d}}} \quad (\text{A18})$$

Similarly, assuming that the total number of search times is m , the number of search results that existence the target is x and the number of search results without the targets is y . According to the binomial distribution theorem, the mean value of the number of times that the result existence the target is $x = mp_{i,d}$ and the mean value of the number of times that the result without the target is $y = m - mp_{i,d}$. In the grid g , the probability that the detection results of y times are all without targets is shown in Formula (A19):

$$\mathcal{P}_g^y = \frac{(1-p_{i,d})^y \mathcal{P}_g^0}{(1-p_{i,d})^y \mathcal{P}_g^0 + (1-p_{i,d})^y (1 - \mathcal{P}_g^0)} \quad (\text{A19})$$

Therefore, the target existence probability after m search times is given in Formula (A20):

$$\mathcal{P}_g^m = \frac{p_{i,d}^x (1-p_{i,d})^y \mathcal{P}_g^0}{(1-p_{i,d})^y p_{i,d}^x \mathcal{P}_g^0 + (1-p_{i,d})^y p_{i,f}^x (1 - \mathcal{P}_g^0)} \quad (\text{A20})$$

Substituting $x = mp_{i,d}$, $y = m - mp_{i,d}$ into Formula (A20), satisfying $P_g^m > T^+$, we can get Formula (A21):

$$m_{avg}^+ > \frac{\log\left(\frac{\mathcal{P}_g^0(1-T^+)}{T^+(1-\mathcal{P}_g^0)}\right)}{(1-p_{i,d}) \log \frac{1-p_{i,f}}{1-p_{i,d}} + p_{i,d} \log \frac{p_{i,f}}{p_{i,d}}} \quad (\text{A21})$$

When there is no target in grid g , that is, $\mathcal{P}_g^m < T^-$, we can get m_{min}^- , m_{avg}^- :

$$m_{min}^- > \frac{\log\left(\frac{\mathcal{P}_g^0(1-T^-)}{T^-(1-\mathcal{P}_g^0)}\right)}{\log \frac{1-p_{i,f}}{1-p_{i,d}}} \quad (\text{A22})$$

$$m_{avg}^- > \frac{\log\left(\frac{\mathcal{P}_g^0(1-T^-)}{T^-(1-\mathcal{P}_g^0)}\right)}{(1-p_{i,f})\log\frac{1-p_{i,f}}{1-p_{i,d}} + p_{i,f}\log\frac{p_{i,f}}{p_{i,d}}} \quad (A23)$$

Through the above derivation, we can find that the target probability map is related to the $p_{i,d}$ and $p_{i,f}$ when it converges (reaching the upper of the T^+ and the lower of the T^-). The larger $p_{i,d}$, the smaller the $p_{i,f}$, the smaller $m_{avg}^+, m_{min}^+, m_{min}^-, m_{avg}^-$ can be obtained, that is, the smaller the number of search times needed to determine the state of the grid. These results show that the performance of the sensor will directly affect the search speed of the system.

Appendix D

ACSLA is trained by the RL method. When using the RL method, the agent needs to continuously interact with the environment and obtain corresponding rewards from the exploration process to learn the optimal strategy. The sample obtained by the agent interacting with the environment is (s, a, s', r) , (s : state, a : action, s' : next state, r : reward) which is called the experience fragment. These experience fragments are stored in the experience pool of the RL method and are randomly selected during learning to train the agent to learn the optimal strategy.

Appendix D.1 Value Iteration RL Method

RL methods are generally divided into two types, one based on the value iteration and the other based on the policy gradient. The core idea of value iteration is to solve dynamic programming problems. When solving the optimal value of the dynamic programming problem, the sub-problem of the optimal problem needs to be solved first, and then the optimal solution is finally obtained through iteration. Deep Q-network (DQN) is a classic value-based iterative RL method based on the Qlearning algorithm. In order to solve the dimensional explosion problem of the Q function, the Q function is fitted using a deep neural network [35]. DQN uses a deep neural network to approximate the value function, that is, the input of the neural network is state s and the output is $Q(s, a)$. After calculating the value function through the neural network, DQN uses the ϵ -greedy strategy to output the action a . DQN considers: $r(s, a) + \gamma \max_{a'} Q(s', a')$ as the objective function and uses Q-network to fit it. The objective function is shown in Formula (A24):

$$y_j = r_j + r(s, a) + \gamma \max_{a'} \hat{Q}(s', a', \varphi) \quad (A24)$$

The loss function is set as follows:

$$L(\varphi) = \frac{1}{2} \|y_j - \hat{Q}(s', a', \varphi)\|_2^2 \quad (A25)$$

where φ is the parameter of the Q-network. DQN has two characteristics. The first one is experience replay. The algorithm stores agent's experiences (s, a, s', r) in the replay buffer. When training the Q-network, the samples are obtained by random sampling from the replay buffer. The second is to use a fixed network for training. Two neural networks are used for the training update. One network is not directly updated, called the target network, and the other network is normally updated, called the evaluation network. The parameters of the evaluation network are copied to the target network for update after a certain amount of training [36].

Appendix D.2 Policy Gradient RL Method

Unlike value iterative RL methods, the strategy gradient RL methods are to use strategy π_θ to sample the environment to obtain sequence $\tau = \{s_1, a_1, r_1, s_2, a_2, r_2, \dots, s_T\}$, where θ is a strategy parameter [37]. The probability of generating the sequence τ is:

$$P_\theta(s_1, a_1, r_1, s_2, a_2, r_2, \dots, s_T) = p(s_1) \sum_{t=1}^T \pi(a_t|s_t) p(s_{t+1}|s_t, a_t) \quad (\text{A26})$$

Therefore, the reward generated by τ is reflected in the distribution of strategy parameters. By maximizing the expected reward, the θ can be calculated inversely:

$$\theta^* = \arg \max_{\theta} \mathbb{E}_{\tau \sim p_\theta(s,a)} \left[\sum_t r_t(s, a) \right] \quad (\text{A27})$$

The gradient value of the objective function concerning the θ :

$$\nabla_\theta J_\theta = \mathbb{E}_{\tau \sim \pi_\theta(\tau)} \left[\left(\sum_{t=1}^T \nabla_\theta \log \pi_\theta(a_t|s_t) \right) \left(\sum_{t=1}^T r_t(s_t, a_t) \right) \right] \quad (\text{A28})$$

Since the goal of RL methods is to maximize rewards, the gradient ascent algorithm is used to update the θ . The update formula of θ is shown as follow:

$$\theta \leftarrow \theta + \alpha \nabla_\theta J_\theta \quad (\text{A29})$$

The deep deterministic policy gradient algorithm (DDPG) [38] is the most commonly used gradient-based reinforcement learning algorithm. The actor-critic algorithm framework based on deterministic action strategy is used and the deterministic strategy method of deterministic policy gradient (DPG) is adopted in the actor part.

References

1. Cashmore, M.; Fox, M.; Long, D.; Magazzeni, D.; Ridder, B. Artificial Intelligence Planning for AUV Mission Control. *IFAC-PapersOnLine* **2015**, *48*, 262–267. [CrossRef]
2. Blidberg, D.R. Autonomous underwater vehicles: Current activities and research opportunities. In *Intelligent Autonomous Systems 2, An International Conference*; IOS Press: Amsterdam, The Netherlands, 1989.
3. Ridao, P.; Carreras, M.; Ribas, D.; Sanz, P.J.; Oliver, G. Intervention AUVs: The next challenge. *Annu. Rev. Control* **2015**, *40*, 227–241. [CrossRef]
4. Fiorelli, E.; Leonard, N.E. Adaptive Sampling Using Feedback Control of an Autonomous Underwater Glider Fleet. In *Proceedings of the 13th International Symposium on Unmanned Untethered Submersible Technology*, Durham, NH, UK, 1–3 January 2003; pp. 415–422.
5. Healey, A.J. A neural network approach to failure diagnostics for underwater vehicles. In *Proceedings of the Symposium on Autonomous Underwater Vehicle Technology*, Washington, DC, USA, 2–3 June 1992; IEEE: Piscataway, NJ, USA, 1992.
6. Sotzing, C.C.; Lane, D.M. Improving the Coordination Efficiency of Limited Communication Multi-AUVs Operations using a Multi-Agent Architecture. *J. Field Robot.* **2010**, *27*, 412–429. [CrossRef]
7. Hausman, K.; Mueller, J.; Hariharan, A.; Ayanian, N.; Sukhatme, G.S. Cooperative multi-robot control for target tracking with onboard sensing. *Int. J. Robot. Res.* **2015**, *34*, 1660–1677. [CrossRef]
8. Yoon, S.; Qiao, C. Cooperative Search and Survey Using Autonomous Underwater Vehicles (AUVs). *IEEE Trans. Parallel Distrib. Syst.* **2011**, *22*, 364–379. [CrossRef]
9. Yang, Y.; Polycarpou, M.M.; Minai, A.A. Multi-UAV Cooperative Search Using an Opportunistic Learning Method. *J. Dyn. Syst. Meas. Control* **2007**, *129*, 716–728. [CrossRef]

10. Wenjing, C.; Shenghong, X. Comparison of multi-UAV cooperation architectures. In Proceedings of the International Conference on Information Management, Chengdu, China, 21–23 April 2017; IEEE: Piscataway, NJ, USA, 2017.
11. Messac, A.; Caswell, R.; Henderson, T. Control-structure integrated design—Centralized vs decentralized control. In Proceedings of the Aerospace Design Conference, Irvine, CA, USA, 3–6 February 1992.
12. Yang, S.; Luo, C. A Neural Network Approach to Complete Coverage Path Planning. *IEEE Trans. Syst. Man Cybern. Part. B Cybern.* **2004**, *341*, 718–725. [[CrossRef](#)] [[PubMed](#)]
13. Maza, I.; Ollero, A.; Casado, E.; Scarlatti, D. Classification of Multi-UAV Architectures. In *Handbook of Unmanned Aerial Vehicles*; Springer: Berlin, Germany, 2015.
14. Adepegba, A.A.; Miah, S.; Spinello, D. Multi-agent area coverage control using reinforcement learning. In Proceedings of the Twenty-Ninth International Flairs Conference, Key Largo, FL, USA, 16–18 May 2016.
15. Silver, D.; Sutton, R.S.; Müller, M. Reinforcement learning of local shape in the game of go. In Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyder-Abad, India, 6–12 January 2007; pp. 1053–1058.
16. Rajnarayan, D.G.; Ghose, D. Multiple agent team theoretic decision-making for searching unknown environments. In Proceedings of the 42nd IEEE International Conference on Decision and Control, Maui, HI, USA, 9–12 December 2003; IEEE: Piscataway, NJ, USA, 2004; Volume 3, pp. 2543–2548.
17. Wang, X. Bayesian Ranking and Selection Models for Discrete Network Design Problems with Uncertainties and Multiple Environmental Objectives. Ph.D. Thesis, Cornell University, Ithaca, NY, USA, 2014.
18. Hong, S.-P.; Cho, S.-J.; Park, M.-J.; Lee, M.G. Optimal search-relocation trade-off in Markovian-target searching. *Comput. Oper. Res.* **2009**, *36*, 2097–2104. [[CrossRef](#)]
19. Singh, Y.; Bibuli, M.; Zereik, E.; Sharma, S.; Khan, A.; Sutton, R. A Novel Double Layered Hybrid Multi-Robot Framework for Guidance and Navigation of Unmanned Surface Vehicles in a Practical Maritime Environment. *J. Mar. Sci. Eng.* **2020**, *8*, 624. [[CrossRef](#)]
20. Mina, T.; Singh, Y.; Min, B.C. Maneuvering Ability-Based Weighted Potential Field Framework for Multi-USV Navigation, Guidance, and Control. *Mar. Technol. Soc. J.* **2020**, *54*, 40–58. [[CrossRef](#)]
21. Thi, H.A.L.; Nguyen, D.M.; Tao, P.D. *A DC Programming Approach for Planning a Multisensor Multizone Search for a Target.*; Elsevier Science Ltd.: Amsterdam, The Netherlands, 2014.
22. Healey, A.J. Application of Formation Control for Multi-Vehicle Robotic Mine Sweeping. In Proceedings of the 40th Conference on Decision and Control, Orlando, FL, USA, 4–7 December 2001; IEEE: Piscataway, NJ, USA, 2001; pp. 1497–1502.
23. Welling, D.M.; Edwards, D.B. Multiple Autonomous Underwater Crawler Control for Mine Reacquisition. In Proceedings of the 2005 International Mechanical Engineering Congress and Exposition, Orlando, FL, USA, 5–11 November 2005; ASME: New York, NY, USA, 2005; pp. 25–34.
24. Shafer, A.J.; Benjamin, M.R.; Leonard, J.J.; Curcio, J. *Autonomous Cooperation of Heterogeneous Platforms for Sea-Based Search Tasks*; OCEANS: Hanoi, Vietnam; IEEE: Piscataway, NJ, USA, 2008.
25. Gooding, T.R. A Framework for Evaluating Advanced Search Concepts for Mauv Mine Countermeasures. Master's Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2001; pp. 45–76.
26. Edwards, J.R. *Real-Time Classification of Buried Targets with Teams of Unmanned Vehicles*; OCEANS: Mississippi, MS, USA, 2002.
27. Liu, T.C.; Schmidt, H. *AUV-Based Seabed Target. Detection and Tracking*; OCEANS: Mississippi, MS, USA; IEEE: Piscataway, NJ, USA, 2002; pp. 474–478.
28. Bovio, E.; Cecchi, D.; Baralli, F. Autonomous Underwater Vehicles for Scientific and Naval Operations. *Annu. Rev. Control* **2006**, *30*, 117–130. [[CrossRef](#)]
29. Schneider, T.; Schmidt, H. Unified Command and Control for Heterogeneous Marine Sensing Networks. *J. Field Robot.* **2010**, *27*, 876–889. [[CrossRef](#)]
30. Allotta, B.; Costanzi, R.; Magrini, M.; Monni, N.; Moroni, D. Towards a Robust System Helping Underwater Archaeologists through the Acquisition of Geo-Referenced Optical and Acoustic Data. In Proceedings of the 10th International Conference on Computer Vision Systems, Copenhagen, Denmark, 6–9 July 2015; Springer: Berlin/Heidelberg, Germany, 2015; pp. 253–262.
31. Tsiogkas, N.; Frost, G.; Monni, N.; Lane, D. *Facilitating Multi-AUVs Collaboration for Marine Archaeology*; OCEANS: Genova, Italy; IEEE: Piscataway, NJ, USA, 2015; pp. 385–391.

32. Maurelli, F.; Saigol, Z.; Insaurrealde, C.C.; Petillot, Y.R.; Lane, D.M. Marine World Representation and Acoustic Communication: Challenges for Multi-Robot Collaboration. In *Autonomous Underwater Vehicles*; IEEE/OES: Southampton, UK; IEEE: Piscataway, NJ, USA, 2012; pp. 212–219.
33. Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Zhang, X. TensorFlow: A System for Large-Scale Machine Learning. 2016. Available online: <https://www.researchgate.net/publication/303657108>. (accessed on 26 October 2020).
34. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
35. Balduzzi, D.; Ghifary, M. Compatible value gradients for reinforcement learning of continuous deep policies. *arXiv* **2015**, arXiv:1509.03005.
36. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Gendron-Bellemare, M.; Graves, A.; Riedmiller, M.; Fidjeland, A.; Ostrovski, G.; et al. Shane Legg & Demis Hassabis. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533. [PubMed]
37. Munos, R. Policy Gradient in Continuous Time. *J. Mach. Learn. Res.* **2006**, *7*, 771–791.
38. Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Wierstra, D. Continuous control with deep reinforcement learning. *arXiv* **2015**, arXiv:1509.02971.

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).