Sensors and Materials, Vol. 31, No. 3 (2019) 845–858
MYU Tokyo

S & M 1819

Underwater Sensor Network Deployment Algorithm Using Density-based Spatial Clustering of Applications with Noise

Hui Wang,* Tingcheng Chang, Yexian Fan, and Zhiliang Li

Department of Computer Science Engineering, Ningde Normal University,
No. 1 Xueyuan Road, Ningde 352000, China

(Received September 30, 2018; accepted January 23, 2019)

Keywords: underwater sensor networks, DBSCAN, single or multiple hops, global search

Underwater sensor networks have received extensive attention owing to their promise for application to marine exploration, submarine navigation, and pollution monitoring. In an open ocean underwater environment, the targets to be monitored are undefined. Therefore, it is necessary to investigate how sensor nodes adjust their positions autonomously in accordance with the changes in the environment and targets to achieve optimal monitoring quality. We propose a fish-swarm-inspired underwater sensor network deployment algorithm using density-based spatial clustering of applications with noise (DBSCAN). Firstly, inspired by the operation mode of an artificial fish-swarm system, sensor nodes autonomously cover all the events by simulating physiological behaviors such as swarm, follow, and prey. Secondly, considering the complexity of the underwater environment and also to reduce the number of nodes participating in the movement and to avoid the blind movement of nodes, the DBSCAN model is introduced to achieve the sharing of sensed information among the nodes that communicate by the single-hop or multihop mode, thereby enhancing the global search ability of nodes. Finally, a large number of experiments are carried out to evaluate the performance of the proposed algorithm. The results show that the proposed algorithm can effectively solve the problem of underwater sensor deployment, and has the advantages of fast convergence and strong scalability.

1. Introduction

Underwater sensor networks are widely used for marine tactical reconnaissance, early warning of disasters, marine exploration, and submarine navigation, and therefore are attracting increasing attention.[1,2] By mounting sensors on an autonomous underwater vehicle or underwater mobile device, an underwater mobile sensor node is formulated to perform a task underwater. A reasonable and effective sensor node deployment can significantly reduce the time taken to construct a sensor network and quickly cover the target area. It can also prolong the network life and adapt to the changing topology through coordinated control.

In an underwater sensor deployment scheme, nodes are deployed in an underwater three-dimensional space to achieve a stereoscopic perception of the monitored area. The existing

*Corresponding author: e-mail: wangh0802@163.com
https://doi.org/10.18494/SAM.2019.2159

ISSN 0914-4935

three-dimensional underwater sensor deployment schemes can be divided into two types. The first type is for uniform coverage, that is, sensor nodes are uniformly deployed in the monitored area.[3–5] The second type is for nonuniform coverage, that is, nodes are nonuniformly deployed in accordance with the specific distribution of the monitored targets.[6–10] In the nonuniform coverage scheme, the concept of "event" is explicitly proposed. Thus, the purpose of node deployment is no longer to uniformly cover the monitored area, but to cover the events. Such node deployment is more practical and effective, and also conforms to the characteristics of the sparseness of underwater sensor networks. In this paper, we focus on sensor node deployment for nonuniform coverage in underwater three-dimensional space and the design of a distributed underwater sensor network node deployment algorithm based on an artificial fish swarm. Because of the characteristics of adaptive self-deployment and a distributed operation mode, the proposed algorithm is suitable for complex and harsh environments, such as mountains, oceans, and deep sea trenches. At the same time, owing to the limited energy of nodes and to avoid the blind movement of nodes, we propose a density-based spatial clustering of applications with noise (DBSCAN) algorithm to cluster the nodes. Satisfactory deployment results are achieved by combining the DBSCAN algorithm and the intelligent optimization algorithm of an artificial fish swarm.

The rest of this paper is organized as follows. In Sect. 2, we describe some related work. Problems are described and a model is established in Sect. 3 to help understand the algorithm presented in this paper. A solution for the problem is proposed in Sect. 4. In Sect. 5, the simulation experiments and analysis of results are presented. Finally, conclusions are given in Sect. 6.

2. Related Work

For node deployment with nonuniform coverages, many achievements have been made at home and abroad. Wang *et al.*[11] proposed an optimized artificial fish swarm algorithm (OAFSA), where the maximum coverage rate is obtained by simulating the prey and follow behaviors in artificial fish. The optimized OAFSA effectively improved the coverage performance compared with the traditional artificial fish algorithms, but the deployment in complex environments (such as those with obstacles) was not considered in this research. Lian *et al.*[12] introduced four weighting indices (i.e., effective coverage rate, high monitoring probability range ratio, spatial overlap ratio, and resource utilization) to adjust the fitness function to improve the effective coverage rate of the sensor network. However, the algorithm had slow convergence, and as the population size increased, the computational complexity of the algorithm increased exponentially. Aitsaadi *et al.*[13] considered the difference in the distribution density of events in the monitored waters and proposed an algorithm for nonuniform node deployment by the mesh representation method. This algorithm could help the network achieve effective monitoring for events, but it was difficult to obtain a higher network connectivity rate. To effectively cover the events in the monitored waters, Du *et al.*[14] proposed particle-swarm-inspired underwater sensor self-deployment (PSSD), in which particle swarm optimization and congestion degree control are combined to effectively solve the network coverage problem. The complexity was

low, the convergence speed was high, and an effective node distribution could be realized. However, this PSSD algorithm only considered the coverage of the network for events, and it was also difficult to obtain a high network connectivity rate. In addition, this kind of algorithm has some disadvantages, for example, too many nodes participate in the movement and nodes may move blindly. Because of the limited energy of nodes and high energy consumption in an underwater environment, the nodes will fail owing to the rapid consumption of energy, which shortens the life of the sensor network.

The above methods have achieved good results in different aspects of underwater sensor node deployment, but cannot completely meet the needs of underwater sensor network applications. The main problems are as follows. Most of the methods are centralized optimization methods, making them difficult to implement with distributed nodes; the nodes are deployed for certain events, and it is difficult to adaptively adjust node deployment to ensure a satisfactory monitoring quality for uncertain events in an open-ocean underwater environment or for events that dynamically change depending on external factors such as ocean currents. For the above problems, inspired by the operation mode of the fish swarm system, a distributed and achievable underwater sensor node deployment algorithm is proposed in this paper. By simulating fish swarm behaviors such as prey, the sensor nodes autonomously cover events. The combination of the proposed algorithm and the DBSCAN algorithm greatly improves the convergence speed, avoids the blind movement of nodes, and prolongs the life of the underwater sensor networks.

3. Background

3.1 Problem description

Assuming that N sensor nodes are deployed in the monitored area A and s_i represents the ith node in the network, then the set of underwater sensor nodes is $S = \{s_1, s_2, ..., s_N\}$. Assuming that a node, s_i, has the ability to sense, communicate, and move, then $B_i = \left\langle r_i^s, r_i^c, l_i \right\rangle$, where $r_i^s \geq 0$ is the sensing radius of s_i, $r_i^c \geq 0$ is the communication radius of s_i, and $l_i \geq 0$ is the maximum moving step length of the node. It can be seen that, in a homogeneous network, all nodes have the same attributes, namely, $r_i^s = r^s$, $r_i^c = r^c$, and $l_i \geq l$ ($1 \leq i \leq N$). The dynamic point of interest e is referred to as an event, and the sensor node can detect the event and communicate with the neighbor node to obtain state information (the number of covered events) of the neighbor node. The task of a node is to cover the event and collect information about the event.

3.2 Coverage probability model

It is assumed that the monitored area A is a three-dimensional cuboid space ($m \times n \times p$), and that the coverage model of each node is a sphere with a sensing radius of $r_i^s \geq 0$. The communication range is a sphere with a radius of $r_i^c \geq 0$. To ensure network connectivity, the communication radius is set to be greater than or equal to twice the sensing radius,[15] i.e.,

$r_i^c \geq r_i^s$. In the monitored area A, for $\forall e_i \in A$, if the Euclidean distance between the node s_j and the event e_i represented by $d(e_i, s_j)$ satisfies $\leq r_j^s$, that is, $d(e_i, s_j) \leq r_j^s$, then the event s_j is covered by the node s_j. $d(e_i, s_j)$ is given by

$$d\left(e_i, s_j\right) = \sqrt{\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2 + \left(z_i - z_j\right)^2}, \tag{1}$$

where (x_i, y_i, z_i) is the coordinate of the event e_i and (x_j, y_j, z_j) is the coordinate of the node s_j. In the monitored area A, for $\forall e_i \in A$, the probability of the node s_j covering the event e_i is given by[16]

$$p\left(e_i, s_j\right) = \begin{cases} 1, & d\left(e_i, s_j\right) \leq \hat{r} \\ e^{-\lambda\left[d\left(e_i, s_j\right) - \hat{r}\right]}, & \hat{r} < d\left(e_i, s_j\right) \leq r^s \\ 0, & d\left(e_i, s_j\right) > r^s \end{cases}, \tag{2}$$

where $\hat{r}$ is the radius of the self-confidence circle of the node; λ is the attenuation factor of sensing and is the physical property of the node. Then, in the monitored area A, the number of events covered by the node s_j is expressed as

$$N_e\left(s_j\right) = \sum_{e_i \in A} p\left(e_i, s_j\right). \tag{3}$$

When the event e_i is not within the sensing range of any node in the node set S, the event e_i can be considered to be uncovered for the entire node set S. Therefore, by using multiple sensor nodes to simultaneously detect the event e_i, the joint coverage probability of points is given by

$$P\left(e_i, S\right) = 1 - \prod_{i=1}^{N}\left(1 - p\left(e_i, s_j\right)\right). \tag{4}$$

4. Design of Fish-swarm-inspired Underwater Sensor Deployment Algorithm Based on DBSCAN

4.1 Basic fish swarm algorithm

An artificial fish swarm algorithm (AFSA) is an optimization algorithm that simulates the behavior of a fish swarm. Using the behaviors of a fish swarm, such as prey, cluster, and follow, the global optimal solution can be found rapidly. The AFSA is a kind of swarm intelligence stochastic optimization algorithm. Similarly to the particle swarm and ant colony algorithms, the AFSA also has a strong global search ability and is insensitive to initial value parameter selection, robust, and easy to operate.

Assuming that, in an n-dimension search space, there are N artificial fish that make up a swarm. The state of each artificial fish can be expressed as $X = (X_1, X_2, ..., X_N)$, where X_i (i = 1, 2, ..., N) is the variable to be optimized. The food concentration for the artificial fish at the current position is expressed as $Y = f(X)$, where Y is the objective function; the distance between two artificial fish is expressed as $d_{ij} = \left\| X_i - X_j \right\|$; *Visual* represents the sensing range of one artificial fish; *Step* is the step length of artificial fish movement; δ is the congestion degree factor; *tryNum* represents the maximum number of trials of an artificial fish for each prey.

The artificial fish uses its vision to sense the external environment, as shown in Fig. 1. X_i is the current position of the artificial fish; *Visual* (equivalent to the communication radius) is the visual distance; X_h is the visual position at a certain moment. If a position within the field of view is better than the current position, the artificFial fish will go further in this direction to the next position X_{next}; otherwise, it continues to wander within the field of view.

The process of a mobile node in a sensor network aimed at greater network coverage is similar to the follow and prey behaviors of artificial fish. The food concentration at the location of an artificial fish can be regarded as the network coverage under the current state. Usually, fish will stay in a place rich in food, so the global optimal behavior can be found by simulating the characteristics of fish, which is the basic idea of the AFSA. The basic behaviors of artificial fish are as follows

(1) Prey behavior: Fish swim randomly to search for food. Assuming that X_i is the current state of an artificial fish, a state X_j is randomly selected within its visual range and Y is the food concentration (objective function). The larger the value of *Visual*, the easier it is to find the convergent global extremum.

$$X_j = X_i + Visual \times Rand(\ ) \tag{5}$$

If $Y_i < Y_j$, the artificial fish will go further in this direction; otherwise, it randomly selects a state X_j and determines whether the condition is met. If the condition is still not satisfied after repeating it *tryNum* times, it moves one step at random. When the value of *tryNum* is small, the artificial fish can swim randomly so that it can escape the local extremum.

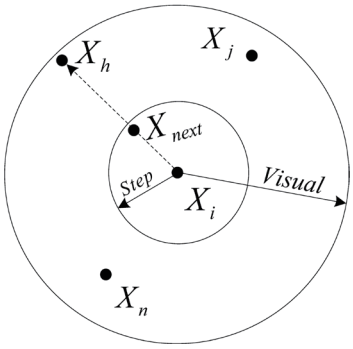

Fig. 1. Schematic diagram of the visual range of artificial fish.

$$X_i^{(t+1)} = X_i^{(t)} + Visual \times Rand(\ ) \tag{6}$$

(2) Cluster behavior: Fish naturally gather together during movement. Assume that X_i is the current state of artificial fish, X_c is the location of the swarm center, n_f is the number of companions in the current neighborhood ($d_{ij} < Visual$), and N is the total number of fish. The conditions $Y_c > Y_i$ and $n_f/N < \delta$ indicate that the swarm center has more food (high fitness function value) and is not very congested, then fish X_i will go further toward the center.

$$X_i^{(t+1)} = X_i^{(t)} + \frac{X_c - X_i^{(t)}}{\left\| X_c - X_i^{(t)} \right\|} \times Step \times Rand(\ ) \tag{7}$$

Otherwise, prey behavior is performed.

(3) Follow behavior: When one fish finds that food is abundant in the current place, other fish will quickly follow. Assuming that X_i is the current state of the artificial fish, it finds that a companion X_j is nearby ($d_{ij} < Visual$) where Y_j is large. If $Y_j > Y_i$ and $n_f/N < \delta$, it means that companion X_j has a higher food concentration (high fitness function value) and the surrounding environment is not very congested, so fish X_i will go further toward companion X_j.

$$X_i^{(t+1)} = X_i^{(t)} + \frac{X_j - X_i^{(t)}}{\left\| X_j - X_i^{(t)} \right\|} \times Step \times Rand(\ ) \tag{8}$$

Otherwise, prey behavior is performed.

It should be noted that the application of the cluster and follow behaviors of a fish swarm can make the node move closer to the fish with high food concentration. At the same time, in order to maximize the coverage rate, the fish swarm should maintain good formation, which can be controlled by the congestion degree factor δ and fish spacing.

4.2 Description of DBSCAN algorithm

The DBSCAN clustering algorithm can find clusters of arbitrary size and arbitrary shape, and effectively identify outliers,[17] and is insensitive to the order of input objects. Because of these advantages, the DBSCAN algorithm is widely used in many fields. For a d-dimensional data set Ω ($i = 1, 2, ..., d$), the definition in DBSCAN is as follows.[18]

Definition 1 E_{ps} neighborhood of data point ρ: a circular area with radius E_{ps} and centered on point ρ in the data set. The set of points contained in the area is denoted as $N_{E_{ps}}(\rho) = \left\{ q \in \Omega \middle| d(q,\rho) \le E_{ps} \right\}$.

Definition 2 Density of data point ρ: the number of data points in the E_{ps} neighborhood of data point ρ in data set Ω.

Definition 3 Core point: If the density of data point ρ is greater than or equal to *MinPts*, it is called the core point, where *MinPts* is a threshold specified by the user.

Definition 4 Boundary point: Data point q is not the core point, but it is in the E_{ps} neighborhood of another core point ρ; thus, it is called a boundary point.

Definition 5 Noise point: Other points in the data set that are neither core points nor boundary points are called noise points.

Definition 6 Directly density-reachable: Let data point q be the core point. As long as data point ρ belongs to the E_{ps} neighborhood of data point q, data point ρ is said to be directly density-reachable from data point q.

$$\rho \in N_{E_{ps}}(q) \tag{9}$$

$$\left|N_{E_{ps}}(q)\right| \geq MinPts \tag{10}$$

Definition 7 Density-reachable: If there is a data point chain $\rho_1, \rho_2, ..., \rho_n$, where $\rho_1 = \rho$, $\rho_n = q$, $\rho_i \in \Omega$, $1 \leq i \leq n$, and it is direct arrived density from ρ_i to ρ_{i+1}, then data point ρ is said to be density-reachable from data point q.

Definition 8 Density-connectable: In data set ρ, for a given E_{ps} and *MinPts*, if there is a point o capable of making data points ρ and q density-reachable from data point o, data point ρ and data point q are said to be density-connectable.

To facilitate the understanding of the above concepts, Fig. 2 shows the schematic diagram of the DBSCAN algorithm when $MinPts = 4$. The D point and other red points in the figure are the core points, because the area around these points within the E_{ps} radius contains at least 4 points (including the point itself). Because they are mutually reachable, they form a cluster. Point B and point C are not core points but can be reachable from point D (through other core points) and therefore belong to the cluster. Point G is neither a core point nor a directly reachable noise point.

For the given E_{ps} and *MinPts*, a summary of the flow of the DBSCAN algorithm is as follows. Select any undivided data object to determine whether it is a core data object, and if so, find all data objects that are density-reachable from it and mark these data objects as a

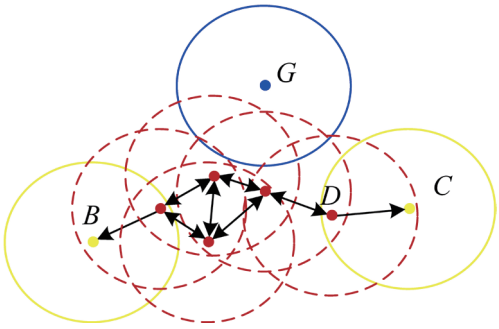

Fig. 2. (Color online) Schematic diagram of the DBSCAN algorithm when $MinPts = 4$.

cluster; if not, the noise judgment is performed on the data. If it is a noise point, it is marked. If not, the object is not processed. The above steps are repeated until all data objects are divided. According to repeated experiments, the parameter E_{ps} is equal to r^c in this research, and its role is to cluster the nodes that can realize data transmission by single or multiple hops in the underwater sensor networks. At the same time, *MinPts* is set to 2, indicating that the cluster should contain at least 2 nodes. Figure 3 shows the clustering results of 30 randomly deployed nodes when $E_{ps} = 50$ and $MinPts = 2$. As can be seen from the figure, 30 nodes are clustered into 6 categories, where the black nodes are noise points, i.e., isolated states. In addition, from *Cluster* #1 to *Cluster* #5, the data transmission between the nodes in each cluster can be realized in the single-hop or multihop mode.

4.3 Proposed algorithm

In underwater node deployment, a sensor node is equivalent to the artificial fish in AFSA inspired by the operation mode of an artificial fish swarm, and an event is equivalent to food. The process of sensor nodes going forward to events is equivalent to the process of artificial fish searching for food. However, considering that the single use of the fish swarming algorithm will cause too many nodes to participate in the movement and that the nodes are moving blindly, the nodes will fail rapidly owing to the rapid exhaustion of energy, which will shorten the life of the network. Therefore, faced with the complexity of an underwater environment, we propose a fish-swarm-inspired underwater sensor deployment algorithm based on DBSCAN, where the nodes are clustered in the process of artificial fish updating their states to realize the sharing of sensed information within the fish swarm. Therefore, the artificial fish are guided to migrate to a better state, thus speeding up the convergence of the optimized deployment of artificial fish.

Definition 9 Congestion degree factor: In the monitored area A, the allowed congestion degree at node s_i ($1 \le i \le n$) is defined as

$$\sigma(s_i) = \psi \times N_e(s_i), \tag{11}$$

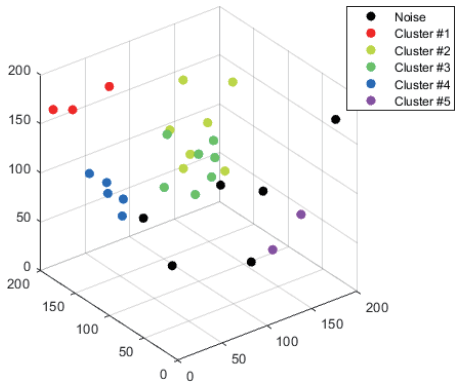

Fig. 3. (Color online) Clustering results of 30 randomly deployed nodes when $E_{ps} = 50$ and $MinPts = 2$.

where ψ is a constant indicating the expected coverage of a single event and $N_e(s_i)$ indicates the number of events covered by node s_i.

Initialization: n nodes are randomly scattered in the underwater monitored area A. The DBSCAN clustering algorithm is performed as described in Sect 4.2. Then, node s_i ($1 \le i \le n$) belongs to cluster $C_z(s_i)$, where $card(C_z(s_i))$ indicates that cluster $C_z(s_i)$ contains n nodes. Next, node s_i performs the following actions in accordance with its own state and those of neighbor nodes.

(1) Prey behavior: If $card(C_z(s_i)) = 0$, node s_i is in an isolated state, and prey behavior is performed. Assuming that the current position of the node is X_i, and within its maximum moving step length l, it randomly moves to a new position X_{next}.

$$X_{next} = X_i + rand(l) \cdot T \tag{12}$$

$rand(l)$ is a random number between 0 and l, and T is an arbitrary unit vector. If the number of events covered by node s_i increases, prey behavior is successful; otherwise, a new location is again randomly selected.

(2) Follow behavior: If $card(C_z(s_i)) > 1$, the optimal node s_{opt} is selected in the cluster $C_z(s_i)$. If node s_{opt} covers more events and is less congested, i.e., $N_e(s_{opt}) \ge N_e(s_i)$ and $N_{co}^{s}(s_{opt}) < \sigma(s_{opt})$, then node s_i goes further in the direction to the position of s_{opt}, i.e.,

$$X_{next} = X_i + l' \times \frac{X_{opt} - X_i}{\left\| X_{opt} - X_i \right\|}, \tag{13}$$

where X_i and X_{opt} represent the position vectors of s_i and s_{opt}, respectively. l' is the moving step length expressed as

$$l' = \begin{cases} \frac{1}{2} d\left(s_i, s_{opt}\right), & \frac{1}{2} d\left(s_i, s_{opt}\right) < l \\ l. & \text{otherwise} \end{cases} \tag{14}$$

If $N_e(s_i)$ increases, the follow is successful; otherwise, it fails.

(3) Cluster behavior: If $card(C_z(s_i)) > 2$, cluster behavior is performed. Firstly, determine the center locations s_c of the nodes in cluster $C_z(s_i)$.

$$X_c = \frac{1}{card\left(C_z\left(s_i\right)\right)} \sum_{s_k \in C_z(s_i)} X_k \tag{15}$$

At the same time, the average number of covered events of s_c is given by

$$N_e\left(s_c\right) = \frac{1}{card\left(C_z\left(s_i\right)\right)} \sum_{s_k \in C_z(s_i)} N_e\left(s_k\right), \tag{16}$$

and the number of neighbor nodes is given by

$$N_{co}^{s}\left(s_c\right)=\frac{1}{card\left(C_z\left(s_i\right)\right)}\sum_{s_k\in C_z\left(s_i\right)}N_{co}^{s}\left(s_k\right). \tag{17}$$

If there are more covered events and less congestion at s_c, i.e., $N_e(s_c) \geq N_e(s_i)$ and $N_{co}^{s}(s_c) < \sigma(s_c)$, then the node s_i goes further in the direction toward the position of s_c, i.e.,

$$X_{next}=X_i+l'\times\frac{X_c-X_i}{\left\|X_c-X_i\right\|}, \tag{18}$$

where X_i and X_{opt} represent the position vectors of s_i and s_{opt}, respectively. l' is the moving step length expressed as

$$l'=\begin{cases}\frac{1}{2}d\left(s_i,s_{opt}\right), & \frac{1}{2}d\left(s_i,s_{opt}\right)<l\\ l. & \text{otherwise}\end{cases} \tag{19}$$

If $N_e(s_i)$ increases, the cluster behavior is successful; otherwise, it fails.

On the basis of the above discussion, a complete fish-swarm-inspired underwater sensor deployment algorithm based on DBSCAN is presented, as shown in Algorithm 1.

Algorithm 1: Fish-swarm-inspired underwater sensor deployment algorithm based on DBSCAN

Initialize the maximum number of iterations N_{max} and the number of trials in the prey behavior N_{try};
Initialize the nodes $S = s_1, s_2, ..., s_n$;
while not (N_{max}**) do**
  for all $s_i \in S$ **do**
    Obtain the node cluster: $C_i \leftarrow dbscan(S, \varepsilon, \lambda)$;
    The number of events covered by node s_i: $N_e(s_i)$
    if $N_e(s_i) > 0$ **then**
      Move to the center of the covered event
    end if
    if $card(C_i(s_i)) > 2$ **then**
      Find optimal neighbor node s_{opt} in cluster $C_z(s_i)$ where node s_i is located;
      Find the center positions of all nodes in cluster $C_z(s_i)$ where node s_i is located;
      if $N_e(s_{opt}) \geq N_e(s_i)$ **and** $N_{co}^{s}(s_{opt}) < \sigma(s_{opt})$ **then**
        Node s_i moves to node s_{opt} and the follow behavior is performed;
      else if $N_e(s_c) \geq N_e(s_i)$ **and** $N_{co}^{s}(s_c) < \sigma(s_c)$ **then**
        Node s_i moves to node s_c and the cluster behavior is performed;
    else
      for $k \leftarrow 1$ to N_{try} **do**

```
                Prey behavior is performed;
                if N_e(s_i') > N_e(s_i) then break;
                end if
            end for
        else
            for k ← 1 to N_try do
                Prey behavior is performed;
                if N_e(s_i') > N_e(s_i) then break;
                end if
            end for
    end while
```

5. Simulation Experiment and Analysis

The PSSD algorithm is a typical nonuniform deployment algorithm for an underwater wireless sensor node network. To evaluate the performance of the proposed algorithm, the PSSD algorithm was used for comparison, in terms of network coverage rate, total moving distance of nodes, and running time. To eliminate the random effects of the experiment, 30 experiments were conducted and their averaged data were used. The parameter settings of the proposed algorithm are shown in Table 1.

The three-dimensional monitored area was 200 × 200 × 200 m^3; 40 target events were randomly distributed in the water, and the size and position were randomly configured; six sensor nodes were randomly deployed. The self-organized deployment of sensor nodes was achieved by the algorithm proposed in this paper. The results of running the algorithm are shown in Fig. 4, where (a) shows the initial state of node deployment and (b) shows the result of running the proposed algorithm. The blue stars indicate the event distribution, the red dots indicate node positions, and the sphere indicates the three-dimensional sensing range of the node. It can be seen that the proposed algorithm can track the event distribution well and achieve effective coverage of all events.

Figure 5 shows the average coverage rates of the proposed and PSSD algorithms with the number of iterations. In the figure, the abscissa represents the number of iterations and the ordinate represents the average coverage rate of 30 experiments. It can be seen that, under the same initial state, the proposed algorithm achieves not only a higher network coverage rate, but also a full coverage of events after a few iterations. Furthermore, the convergence is faster. Compared with the PSSD algorithm, the clustering of nodes by DBSCAN enables the proposed algorithm to overcome the disadvantage of the blindness of the random search by the traditional fish-swarm-inspired algorithm. Figure 6 shows the total moving distances of

Table 1
Parameter settings.

r^s	r^c	ε	λ	N_{try}	l	ψ
50 m	100 m	100 m	2	5	15	0.1

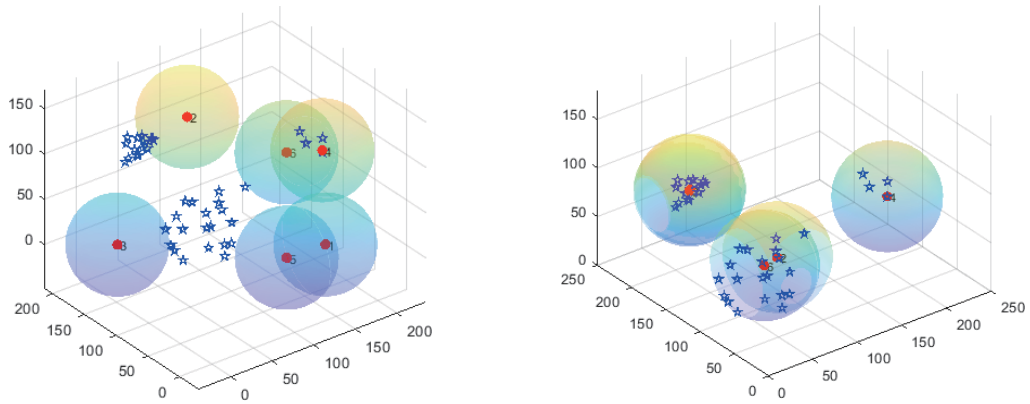

Fig. 4. (Color online) Result display of self-organized deployment. (a) Initial state and (b) result of running the proposed algorithm.

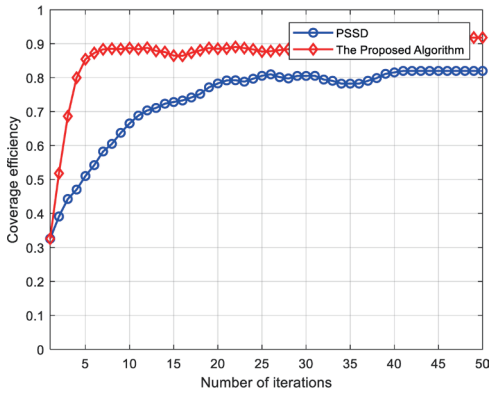

Fig. 5 (Color online) Comparison of average coverage rates.

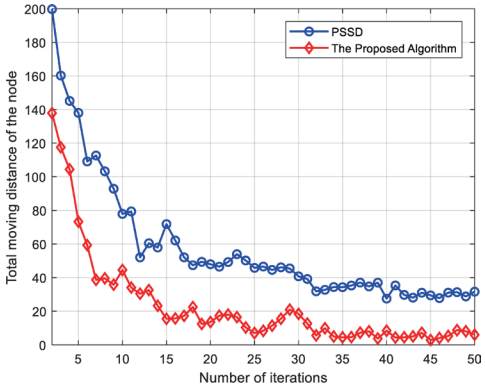

Fig. 6 (Color online) Comparison of total moving distances of nodes.

nodes for the proposed and PSSD algorithms with the number of iterations. It can be seen that, when the number of iterations is the same, the proposed algorithm can greatly reduce the total moving distance of nodes compared with the PSSD algorithm, thereby saving node energy and prolonging the network life. The main reason behind this phenomenon is that information sharing between nodes that can realize data transmission in the single or multihop mode is achieved with the help of the DBSCAN algorithm, thus improving the global sensing ability of the distributed fish swarm algorithm, avoiding blind movement, and reducing energy loss during node movement.

Above, the deployment of 40 target events and 6 sensor nodes was analyzed. However, to obtain accurate information, a large number of sensor nodes (far more than 6) are usually deployed in the monitored area. Therefore, below, we will consider the effect of the change in the number of nodes on the performance of the proposed algorithm.

Table 2
Average moving distance of nodes for the number of nodes from 6 to 30.

	Number of nodes				
	6	10	14	22	30
PSSD	75.2975	68.0178	59.5518	47.7318	27.9360
Proposed algorithm	56.7577	49.2103	47.3806	25.4953	13.4761

Table 3
Average running time for the number of nodes from 6 to 30.

	Number of nodes				
	6	10	14	22	30
PSSD	0.0996	0.1083	0.1279	0.1534	0.2147
Proposed algorithm	0.1634	0.2091	0.2788	0.3408	0.3706

Table 2 shows the average moving distance of nodes for the number of nodes from 6 to 30. It can be seen from Table 2 that, as the number of nodes increases, the average moving distance of nodes decreases to varying extents, but compared with the PSSD algorithm, the proposed algorithm shows a clearer decreasing trend. This is because as the number of nodes increases, the number of nodes participating in the sharing of sensed information through the single or multihop mode also increases, whereas the PSSD algorithm mainly makes decisions by sensing the information of nodes within the communication range. In other words, the proposed algorithm is better at sensing global information and then finding the location of the target event. As a result, fewer nodes participate in the movement, and the nodes are no longer moving blindly. Therefore, the total moving distance of the nodes is reduced.

Table 3 shows the average running time for the number of nodes from 6 to 30. It can be seen from the table that, as the number of nodes increases, the running times of the two algorithms are extended to varying extents. Compared with the PSSD algorithm, the proposed algorithm has a longer running time when the number of nodes is the same. This is mainly because each node must analyze the sensing state of the nodes in its cluster before making a decision and the processing time will increase as the number of nodes in the cluster increases. Therefore, in terms of energy consumption, the energy consumed by node movement is much greater than that by data transmission. Moreover, the running time of the proposed algorithm is within the allowable range. Therefore, the effectiveness of the algorithm proposed in this paper is verified.

6. Conclusions

We proposed a fish-swarm-inspired underwater sensor deployment algorithm based on DBSCAN. By simulating the three behaviors of a fish swarm (prey, follow, and cluster), nodes autonomously tend to the area where events are located to achieve the full coverage of all events. At the same time, to reduce the number of nodes participating in the movement and avoid the blind movement of nodes, DBSCAN clustering analysis is performed to achieve the sharing of sensed information among the nodes that can realize data transmission through the single-hop or multihop mode, thereby enhancing the global search ability of nodes.

A large number of experiments were carried out. The results showed that the proposed algorithm has strong tracking ability and can achieve a high network coverage rate. In addition, compared with the PSSD algorithm, the proposed algorithm has the following three advantages:(1) the node has strong global sensing ability and can track target events rapidly;(2) the disadvantages of too many nodes participating in the movement and the blind movement of nodes are avoided, which effectively solves the energy consumption problem caused by the excessive movement of the nodes;(3) it exhibits distributed deployment and strong scalability. Further efforts are expected to be made on solving the problem associated with the fully connected network.

Acknowledgments

This research was supported by Youth Project of Ningde Normal University (2017Q105, 2018Q103), The Education Department of Fujian Province under Project Number JT180596, Teaching Reform Project of the Ningde Teachers College (JG2018012) and project of the Fujian Provincial Natural Science Fund (2017I0016, 2017J01775).

References

1 I. F. Akyildiz, D. Pompili, and T. Melodia: Ad Hoc Networks **3** (2005) 257.
2 Z. W. Guo, H. J. Luo, F. Hong, M. Yang, and M. X. Ni: J. Comput. Res. Dev. **47** (2010) 377.
3 D. Pompili, T. Melodia, and I. F Akyildiz: Ad Hoc Networks **7** (2009) 778.
4 K. Akkaya and A. Newell: Comput. Commun. **32** (2009) 1233.
5 B. Zeng, D. H. Zhong, and L. Yao: Appl. Res. Comput. **27** (2010) 3926.
6 H. Wang, Y. M. Li, T. Z. Chang, S. M. Chang, and Y. X. Fan: Appl. Sci. **8** (2018) 1638.
7 P. Jiang, J. Liu, and F. Wu: Sensors **15** (2015) 29997.
8 H. Wang, Y. M. Li, T. Z. Chang, and S. M. Chang: Sensors **8** (2018) 2512.
9 P. Jiang, J. Liu, W. Feng J. Wang, and A. Xue: Sensors **16** (2016) 388.
10 H. Wang, Y. M. Li, and T. C. Chang: Microsyst. Technol. **6** (2018) 2803.
11 Y. Wang, H. Liao, and H. Hu: 2012 Int. Conf. Computer Science and Electronics Engineering (ICCSEE) (IEEE, 2012) 90.
12 X. Y. Lian, J. Zhang, C. Chen, and F. Deng: Proc. 10th World Congress on Intelligent Control and Automation (WCICA) (IEEE, 2012) 4395.
13 N. Aitsaadi, N. Achir, K. Boussetta, and G. Pujolle: Proc. IEEE Int. Conf. Oceans (2007) 1.
14 H. Du, X. Na, and Z. Rong: Sensors **14** (2014) 15262.
15 H. H. Zhang and J. C. Hou: Ad Hoc Sens. Wirel. Netw. **1** (2005) 89.
16 A. Ghosh and S. K. Das: Pervasive Mob. Comput. **4** (2008) 303.
17 J. Hou, H. Gao, and X. Li : IEEE Trans. Image Process. **7** (2016) 3182.
18 J. Shen, X. Hao, Z. Liang, Y. Liu, and W Wang: IEEE Trans. Image Process. **25** (2016) 5933.