Sensors **2014**, *14*, 15262-15281; doi:10.3390/s140815262

OPEN ACCESS

sensors

ISSN 1424-8220

www.mdpi.com/journal/sensors

Article

Particle Swarm Inspired Underwater Sensor Self-Deployment

Huazheng Du [1], Na Xia [1,*] and Rong Zheng [2]

[1] School of Computer and Information, Hefei University of Technology, Hefei 230009, China; E-Mail: huazheng2616@163.com

[2] Department of Computing and Software, McMaster University, Hamilton L8S4K1, Canada; E-Mail: rzheng@mcmaster.ca

* Author to whom correspondence should be addressed; E-Mail: xiananawo@hfut.edu.cn; Tel.: +86-551-290-1394; Fax: +86-551-290-1392.

Received: 9 June 2014; in revised form: 27 July 2014 / Accepted: 8 August 2014 / Published: 19 August 2014

Abstract: Underwater sensor networks (UWSNs) can be applied in sea resource reconnaissance, pollution monitoring and assistant navigation, *etc.*, and have become a hot research field in wireless sensor networks. In open and complicated underwater environments, targets (events) tend to be highly dynamic and uncertain. It is important to deploy sensors to cover potential events in an optimal manner. In this paper, the underwater sensor deployment problem and its performance evaluation metrics are introduced. Furthermore, a particle swarm inspired sensor self-deployment algorithm is presented. By simulating the flying behavior of particles and introducing crowd control, the proposed algorithm can drive sensors to cover almost all the events, and make the distribution of sensors match that of events. Through extensive simulations, we demonstrate that it can solve the underwater sensor deployment problem effectively, with fast convergence rate, and amiable to distributed implementation.

Keywords: underwater sensor networks; sensor deployment; coverage efficiency; particle swarm; crowd factor; water flow field

1. Introduction

Underwater Sensor Networks (UWSNs) are underwater monitoring network systems consisting of sensor nodes with computing and acoustic communication abilities. Due to their important applications

in sea resource reconnaissance, pollution monitoring and navigation assistance, UWSNs have attracted much attention from government agencies and research institutions, and have become a hot area in sensor network research [1–3]. In recent years, many aspects of UWSNs such as underwater communication [4,5], sensor deployment and self-organization [6,7], data routing [8,9], and localization and tracking [10–12] have been investigated. Among these, sensor deployment is a key research topic since it not only determines the monitoring quality of the networks, but also acts as the foundation of the network organization, protocol design and application deployment.

Existing underwater sensor deployment methods fall into three categories: Sea-bottom, Sea-surface and Sea-column sensor deployment briefly discussed as follows:

(1) *Sea-bottom sensor deployment*. In these methods, sensors are anchored to the sea bottom to form a two-dimensional monitoring network. The monitoring area is divided into triangle or square grids in which sensors are deployed. The goal is to use as few sensors as possible to cover the monitoring area [6,13]. Sea-bottom sensor deployment is similar to land sensor placement with regards to research objectives and methods, and does not truly reflect the characteristics of UWSNs. In most applications, we need to collect the three-dimensional information of an underwater environment. As a result, sea-bottom sensor deployment is insufficient.

(2) *Sea-surface sensor deployment*. In this category, sensors (gateways or data collectors) are deployed on the sea surface, and collect data from underwater sensor nodes. Sea-surface sensor deployment is generally formulated as an optimization problem. In [14–19], the positions of sensors are computed by an Integer Linear Program (ILP), greedy algorithm and aided geometrical algorithm, so that the number of sensors can be reduced, and the network lifetime can be prolonged. But this line of work mostly still assumes two-dimensional sensor deployment.

(3) *Sea-column sensor deployment*. In this category, sensors are deployed in the three-dimensional underwater space. Existing sea-column sensor deployment methods can be further divided into two classes:

The first class is *uniform coverage requirement sensor deployment.* Sensors are uniformly deployed in the monitoring space. In 2006, Pompili *et al.* [6,7] pioneered the research in three-dimensional underwater sensor deployment, and proposed the Bottom-Grid algorithm. The basic idea is to start from a sea-bottom triangle-grid sensor deployment, and adjust the depth of sensors to form a three-dimensional sensor deployment. The goal is to use as few sensors as possible to cover the 3D monitoring space seamlessly. In this algorithm, global information is needed for adjusting the depth of sensors, and thus the algorithm has to operate in a manner. In 2009, Akkaya *et al*. [20] proposed a Self-deployment algorithm by adjusting the depth of sensors continuously to reduce the coverage overlap between adjacent sensors so as to improve the total coverage in the monitoring space. In [21–23], the authors discussed the network restoration method. When a coverage hole appears in the network, new sensors or redundant sensors will be sent to the position to fix the hole, so as to maintain the coverage and connectivity of the network.

The second class is *non-uniform coverage requirement sensor deployment.* Sensors are non-uniformly deployed in the monitoring space according to the distribution of targets; In 2007, Aitsaadi *et al*. [24] proposed the differentiated deployment algorithm (DDA) for lake water monitoring. Sensors are deployed in the lake according to the distribution of pollutant using a mesh line

representation. This algorithm utilizes a centralized optimization method and is only suitable when the environment and targets are static. In 2008, Koutsougeras *et al*. [25] presented the Self Organizing Maps (SOM) method. Attracted by events (targets), sensors can move to areas with high density of events. Though the method was not originally designed for terrestrial sensor networks, it is expected to be applicable to UWSNs as well. In 2010, Golen *et al*. [26] proposed a scheme to estimate the probability of events in each underwater subregion, and compute the number of sensors that should be allocated in each subregion. This method also suffered from the problem of centralized implementation and high computation complexity.

In non-uniform coverage requirement sensor deployments, the concept of "event" (target) is generally well defined. The objective of sensor deployment is to cover the events and make the distribution of sensors match that of the events. This kind of research is close to practice, and reflects the sparsity characteristic of UWSNs. However, existing methods fall short in the following aspects:

(1) Existing methods all reply on centralized optimization methods, and are difficult to realize in practice;

(2) Most of the methods are designed for static environments. For dynamic environments with uncertain events, these methods cannot be used to adjust the positions of sensors to guarantee desired monitoring quality;

(3) These methods have not taken into account the influence of the water flow;

(4) The performance evaluation metrics of the event-driven underwater sensor deployment are not fully quantifiable.

To address these problems, we study the problem of three-dimensional underwater sensor deployment with the non-uniform coverage requirement. Inspired by particle swarm systems, and we propose PSSD (particle swarm inspired underwater sensor deployment), a distributedly realizable underwater sensor deployment algorithm. By simulating the flying behavior of particles and introducing crowd control, PSSD can drive the sensors to positions with high density of events, and avoid over crowding simultaneously. The design and simulation of PSSD utilize the Lagrange flow model, and are evaluated using an information theoretical metric. The rest of the paper is organized as follows: in Section 2, the underwater sensor deployment problem and its performance metrics are formally defined. In Section 3, the PSSD algorithm is presented in detail. An extensive evaluation is provided in Section 4 with our conclusions in Section 5.

2. Preliminaries

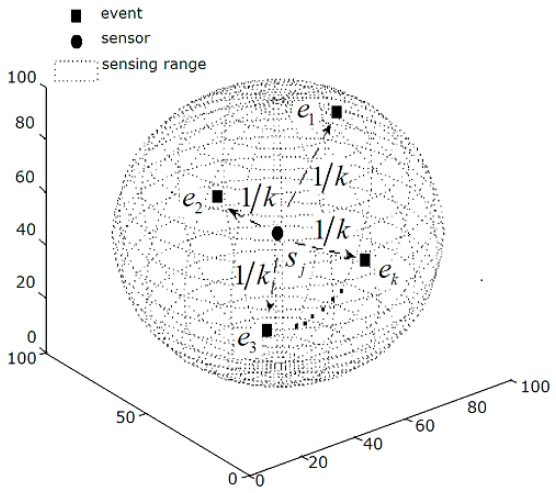
2.1. Underwater Sensor Deployment Problem

Suppose an underwater monitoring space A. Denote a dynamic event by e, and the set of events as $E=\{e_1,e_2,\cdots,e_m\}$, $(e_i\in A, i=1,2,\cdots,m)$. Let the set of sensors be $S=\{s_1,s_2,\cdots,s_n\}$, and each sensor s_j $(1\le j\le n)$ has the ability of sensing, communication and moving. Denote the attribute vector of s_j as $\mathbf{B}_j=\langle r_j^s, r_j^c, l_j, \mathbf{p}_j\rangle$, where $r_j^s\ge 0$, $r_j^c\ge 0$, $l_j\ge 0$ describe the sensing radius, communication radius, and the maximal moving range of s_j, respectively and $\mathbf{p}_j$ is the current position of s_j. All sensors have the same attributes except for position in a homogeneous network, namely, $r_j^s=r^s$, $r_j^c=r^c$,

$l_j = l$ $(1 \le j \le n)$. Sensors can detect events and communicate with their adjacent sensors to exchange information (the number of covered events). The task of sensors is to cover events and collect relevant information.

Definition 1: Coverage. Consider an event at location $e_i \in A$, if $d(e_i, s_j) \le r_j^s$, we call e_i is covered by s_j. Here $d(e_i, s_j)$ is the Euclidean distance between e_i and s_j. If $e_1, e_2, \cdots, e_k$ are all covered by s_j (Figure 1), s_j is "divided" equally among all events, and each event shares $1/k$ of sensor j.

Figure 1. Sensor s_j covering k events.



Definition 2: Coverage degree. Given a collection of sensor placement, the number of sensors shared by event e_i is the coverage degree of e_i. It can be computed as follows:

$$D_A(e_i) = \sum_{s_j \in S} \frac{\mathrm{I}\left\{d(e_i, s_j) \le r_j^s\right\}}{\sum_{e_u \in E} \mathrm{I}\left\{d(e_u, s_j) \le r_j^s\right\}} \tag{1}$$

where $\mathrm{I}(\cdot)$ is an indicator function. $\sum_{e_u \in E} \mathrm{I}\left\{d(e_u, s_j) \le r_j^s\right\}$ is thus the number of events that sensor s_j covers. The physical meaning of $D_A(e_i)$ is the total shares of coverage of event e_i among all sensors.

The underwater sensor deployment problem is to place sensors in the underwater monitoring space A to cover as many events as possible, and make the coverage degree of every event as equal as possible. In other words, the sensor distribution matches the event distribution.

2.2. Performance Metrics

The underwater sensor deployment problem is to place sensors to cover events, and make the coverage degree of every event as equal as possible. So, we need a metric to evaluate how equal the coverage degree of every event is.

Then, we borrow the concept of "entropy" from Information Theory. As we know, the closer the probabilities of q outcomes are, the larger the entropy is, and when the probabilities of q outcomes are equal, the entropy reaches the maximum value $\log q$. So, "entropy" is suitable as the metric to evaluate

how equal the coverage degree of every event is. Furthermore, because the maximum value of entropy is known, it is easy to be normalized for evaluation. Normalizing $D_A(e_i)$, we have:

$$D'_A\left(e_i\right) = \frac{D_A(e_i)}{\sum_{e_u \in E} D_A(e_u)} \tag{2}$$

Clearly, $D'_A(e_i) \in [0,1]$. When $D_A(e_i) = c$, namely all events have the same normalized coverage degree, $D'_A(e_i) = \frac{1}{m}$.

Definition 3: Coverage entropy of the event set. It measures the uniformity of the coverage degree of events, and can be computed as follows:

$$H_A(E) = -\sum_{e_i \in E} D'_A(e_i) \log D'_A(e_i) \tag{3}$$

Clearly, $H_A(E) \le m$. When the number of sensors is large, $H_A(E)$ remains the same when only a subset of sensors covers the events equally. Therefore, we further introduce a penalty factor associated with the percentage of sensors covering events in the final definition of coverage efficiency as follows.

Definition 4: Coverage efficiency of the event set. It evaluates the overall performance of sensor deployment, and is computed as:

$$\eta(E) = \alpha \frac{H_A(E)}{\log m} + \beta \frac{\hat{n}}{n} \tag{4}$$

where $\alpha, \beta \in [0,1]$, and $\alpha + \beta = 1$, $\hat{n}$ is the number of sensors covering events. In what follows, we prove some properties of coverage efficiency.

Lemma 1: Coverage efficiency $\eta(E)$ of event set E will be maximized when all sensors cover some events and all the coverage degree of events are equal.

The proof of Lemma 1 is in Appendix.

Lemma 2: Coverage efficiency of the events set $\eta(E)$ will increase when the number of sensors covering events is fixed, and the coverage degree of events becomes equal.

The proof of Lemma 2 is in Appendix.

From on Lemmas 1 and 2, we observe that $\eta(E)$ is a suitable metric to characterize event-driven underwater sensor deployment.

2.3. Water Flow Field

Underwater environments are complicated and dynamic. Water flow and vortex may introduce disturbance to events and sensor measurements. We introduce the water flow field model discussed in [27] to simulate the underwater environment and test the sensor deployment algorithms. For an incompressible fluid, it can be described by a stream function ψ, from which the two components of the divergenceless velocity field $\mathbf{u} \equiv (u, v)$ are computed as:

$$u = -\frac{\partial \psi}{\partial y}; \quad v = \frac{\partial \psi}{\partial x}, \tag{5}$$

where u is the zonal (eastward) component of the velocity field and v is the meridional (northward) one. The trajectory of a Lagrange device that moves with the current is the solution of the following system of Hamiltonian ordinary differential equations:

$$\dot{x} = -\partial_y \psi(x, y, t), \quad \dot{y} = -\partial_x \psi(x, y, t) \tag{6}$$

The water flow jet model is given by:

$$\psi(x,y,t) = -\tanh\left[\frac{y - B(t)\sin(k(x - ct))}{\sqrt{1 + k^2 B^2(t)\cos^2(k(x - ct))}}\right] \tag{7}$$

where $B(t) = Av + \varepsilon\cos(\omega t)$. The flow induces a net mass transport along the current, and a vigorous chaotic mixing across the current in a wide range of parameters. The parameter k sets the number of meanders in the unit length; c is the phase speed with which they drift downstream; the time-dependent function B modulates the width of the meanders: Av determines the average meander width, ε is the amplitude of the modulation, and ω is its frequency. Water flow leads to the movement of events, and can also be utilized for sensors to move so as to conserve energy.

3. PSSD (Particle Swarm Inspired Underwater Sensor Deployment) Algorithm

In this paper, a particle swarm inspired underwater sensor deployment algorithm (PSSD) is proposed to solve the underwater sensor deployment problem. Particle swarm optimization (PSO) is a population based stochastic optimization technique developed by Eberhart and Kennedy [28,29], to simulate the social behavior of bird flocking or fish schooling. In PSO, the potential solutions are called particles. All particles have velocities that direct the flight of the particles. Particles fly across the problem space to search for the optimal position (solution). PSO is initialized with a group of random particles and then searches for the optima by following two "best" solutions to date. One is the best position that has been achieved so far by the particle itself. This is called "pbest". The other is the best position obtained so far by any particle in the population. This is a global best and called "gbest". In the past several years, PSO has been successfully applied in many research and application areas [30,31], and its parameters setting has been well discussed [32,33].

PSO is a centralized intelligent searching method. Inspired by the operation mechanism of particle swarm systems, we present a distributedly realizable underwater sensor deployment algorithm. Sensors correspond to the particles of PSO. Sensors moving and covering events is similar to particles searching for solutions.

Denote the number of events covered by s_j:

$$N_{event}(s_j) = \sum_{e_k \in E} \mathrm{I}\left\{d(s_j, e_k) \le r_j^s\right\} \tag{8}$$

If an event is in the sensing range of a sensor, the sensor will detect the event and obtain its approximate location. Furthermore, the sensor will share this information with its adjacent sensors.

Definition 5: Allowed crowd factor. Let the allowed crowd factor of s_j in monitoring space A be:

$$\delta(s_j) = \bar{D} \times N_{event}(s_j) \tag{9}$$

where $\bar{D}$ denotes the expected coverage degree for each event. It can be set by experience. Another reasonable way is to set $\bar{D}$ as the average value *n/m* for *n* sensors and *m* events.

The number of sensors in the coverage of s_j is $N_{near}(s_j)$.

The set of the adjacent sensors of s_j is:

$$K(s_j) = \left\{ s_k \middle| d(s_j, s_k) \le r_j^c, k = 1, 2, \cdots, n \right\} \tag{10}$$

So, the number of the adjacent sensors of s_j is:

$$N_{neighbor}(s_j) = card(K(s_j)) \tag{11}$$

Where card(.) is to compute the number of the elements in a set.

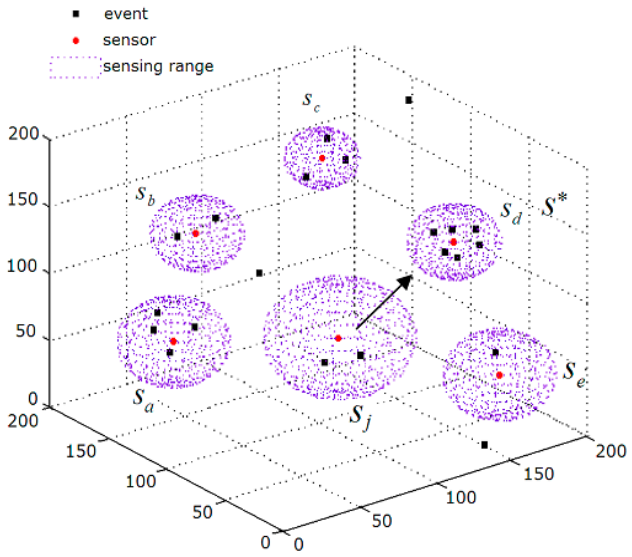
Initialization: randomly scatter *n* sensors in the underwater monitoring space A. Sensor s_j will execute the following operations according to the state of itself and its adjacent sensors.

I. If $N_{neighbor}(s_j) > 0$, which means s_j has some adjacent sensors, s_j will follow the "gbest".

Find the best adjacent sensor $s^* = \arg \max_{s_k \in K(s_j)} \{N_{event}(s_k)\}$.

If s^* is covering more events than s_j, and the position of s^* is not crowded, that is, $N_{event}(s^*) \ge N_{event}(s_j)$, and $N_{near}(s^*) < \delta(s^*)$, then s_j will move toward s^*. It is depicted in Figure 2.

Figure 2. s_j moves toward s^* (the red points are sensors; the purple spheres represent the sensing space (coverage) of sensors; black dots are events.).



The expected velocity of s_j is as follows:

$$\mathbf{v}_j^*(t+1) = \mathbf{x}^* - \mathbf{x}_j(t) \tag{12}$$

$$\mathbf{v}_j^*(t+1) = \begin{cases} l_j \dfrac{\mathbf{v}_j^*(t+1)}{\left\|\mathbf{v}_j^*(t+1)\right\|} & if \ \left\|\mathbf{v}_j^*(t+1)\right\| > l_j, \\ \mathbf{v}_j^*(t+1) & otherwise \end{cases} \tag{13}$$

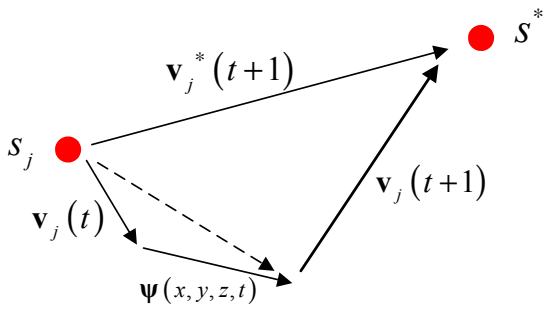
where $\mathbf{x}^*$ is the position of s^*; $\mathbf{x}_j(t)$ is the current position of s_j; t is the iteration number of the algorithm. The expected velocity $\mathbf{v}_j^*(t+1)$ is restricted by the maximal moving range l_j, and the adjustment of velocity is given in Equation (13).

Since the movement of the sensor will be affected by its inertia and the water flow, s_j should either overcome or take advantage of them in moving. Therefore, the actual velocity of s_j is computed according to Equation (14) as illustrated in Figure 3.

$$\mathbf{v}_j(t+1) = \mathbf{v}_j^*(t+1) - \mathbf{v}_j(t) - \boldsymbol{\psi}(x,y,z,t) \tag{14}$$

where $\mathbf{v}_j(t)$ is the current velocity of s_j. It represents the sensor's inertia. $\mathbf{v}_j(t+1)$ is the actual velocity of s_j. $\boldsymbol{\psi}(x,y,z,t)$ is the local water flow field (the water flow velocity vector can be measured by acoustic Doppler velocity measurement ADCP) detected by s_j.

Figure 3. the computing for the actual velocity.



Then, the position of s_j will be updated as follows:

$$\mathbf{x}_j(t+1) = \mathbf{x}_j(t) + \mathbf{v}_j(t+1) \tag{15}$$

This sensor movement strategy not only inherits the flying behavior of particle swarm, but also takes into account the influence of water flow to conserve energy.

After the movement, if $N_{event}(s_j)$ increases, the moving behavior is successful, otherwise the sensor will move back to its original position.

II. If $N_{neighbor}(s_j) = 0$, or equivalently, s_j has no adjacent nodes, s_j will follow the "pbest". "pbest" is the best position that has been achieved so far by s_j itself. It is recorded by s_j.

II.(a) Find the best recorded position $\hat{\mathbf{x}}_j$. If $\hat{\mathbf{x}}_j$ is not the current position of s_j, *i.e.*, $\hat{\mathbf{x}}_j \neq \mathbf{x}_j(t)$, s_j will move toward $\hat{\mathbf{x}}_j$. The expected velocity is computed as:

$$\mathbf{v}_j^{*}(t+1) = \hat{\mathbf{x}}_j - \mathbf{x}_j(t) \tag{16}$$

Equation (13) can also be applied to adjust the velocity. The actual velocity and the updated sensor position follow Equations (14) and (15).

II.(b) If $\hat{\mathbf{x}}_j$ is the current position of s_j, *i.e.*, $\hat{\mathbf{x}}_j = \mathbf{x}_j(t)$, s_j will move randomly.

After the movement, if $N_{event}(s_j)$ increases, the moving behavior is successful, otherwise the sensor will move back to its original position.

Based on the description above, we present the complete particle swarm inspired underwater sensor deployment algorithm in Algorithm 1.

Algorithm 1. Particle Swarm Inspired Underwater Sensor Deployment Algorithm.

Input: sensing range r^s, communication range r^c, the maximal moving range l, the maximal iteration number I.

Output: the positions of sensors in the underwater monitoring space

1. $S \leftarrow$ Randomly deploy sensors in the monitoring space;
2. **for** *step* ← 1 to I **do**
3. $N_{event}(s_j) \leftarrow detect1(s_j)$ /* detect the number of the events covered by s_j itself */;
4. $N_{neighbor}(s_j) \leftarrow detect2(s_j)$ /* detect the number of the adjacent sensors */;
5. $N_{near}(s_j) \leftarrow detect3(s_j)$ /* detect the number of the near sensors */;
6. **if** $N_{neighbor}(s_j) > 0$ **then**
7. find the best adjacent sensor s*;
8. **if** $N_{event}(s^*) > N_{event}(s_j)$ and $N_{near}(s^*) < \delta(s^*)$ **then** /* follow the gbest */;
9. move towards s*;
10. **end**
11. **else** find the best position of s_j during its moving ($\hat{\mathbf{x}}_j$);
12. **if** $\hat{\mathbf{x}}_j$! = $\mathbf{x}_j(t)$ **then** /*follow the pbest */;
13. move towards $\hat{\mathbf{x}}_j$;
14. **else**
15. move randomly;
16. **end**
17. **end**
18. **end**

---

4. Performance Evaluation

In order to evaluate the performance of the proposed algorithm PSSD, we conduct several rounds of Monte Carlo simulations using Matlab. We define the attribute and moving strategy of the sensors in the program. Because the existing event-driven sensor deployment algorithms are all centralized optimization methods while our algorithm is realized in a distributed manner, a comparison between them is not very meaningful. However, for completeness, we still compare the SOM (self organizing maps) algorithm in [25] with PSSD. As a centralized method, SOM can utilize the global information

to search for the solution, so the quality of its solution is guaranteed. In the comparison experiments, if PSSD does better than SOM, the validity of PSSD will be verified.

The parameter settings of the simulations are given in Table 1. The evaluation metrics for both algorithms include $\eta(E)$ defined in Section 2.2, and the convergence speed. In $\eta(E)$, $\alpha = \beta = 0.5$.

Table 1. Parameter settings.

Experiment parameters			**Algorithm parameters**	
r^s (m)	r^c (m)	l (m)	$\overline{D}$	I
sensing radius	communication radius	the maximal moving range	the expected coverage degree for each event	Max number of iterations
40	80	15	1	50

4.1. Static Environments

In a 200 × 200 × 200 sea-column monitoring space, three sets of experiments are conducted:

Experiment Ⅰ: 40 events are uniformly distributed. A total of six sensors are deployed.

Experiment Ⅱ: 40 events are non-uniformly distributed following a T-shape. A total of six sensors are deployed.

Experiment Ⅲ: 40 events are non-uniformly distributed following a line. A total of six sensors are deployed.

Both SOM and PSSD are applied to all scenarios. Figures 4–6 show the results. Black dots denote events, and the spheres represent the sensing space (coverage) of sensors. At the center of each sphere is a sensor. It is clear that with PSSD, all sensors cover some events, and the distribution of sensors matches that of events. In contrast, with SOM, a few sensors cannot cover any events, and the distribution of sensors does not match that of events. Figure 7 shows the evolution of $\eta(E)$ with both algorithms in three sets of experiments. It can be seen that PSSD reaches higher coverage efficiency than SOM. Furthermore, the sensors require fewer steps to reach good position, which indicates the fast convergence speed of PSSD. Also note that PSSD is a distributedly realizable algorithm, while SOM is a centralized algorithm.

Figure 4. Events randomly distributed. (**a**) solution of SOM (self organizing maps); (**b**) solution of PSSD (particle swarm inspired underwater sensor deployment).

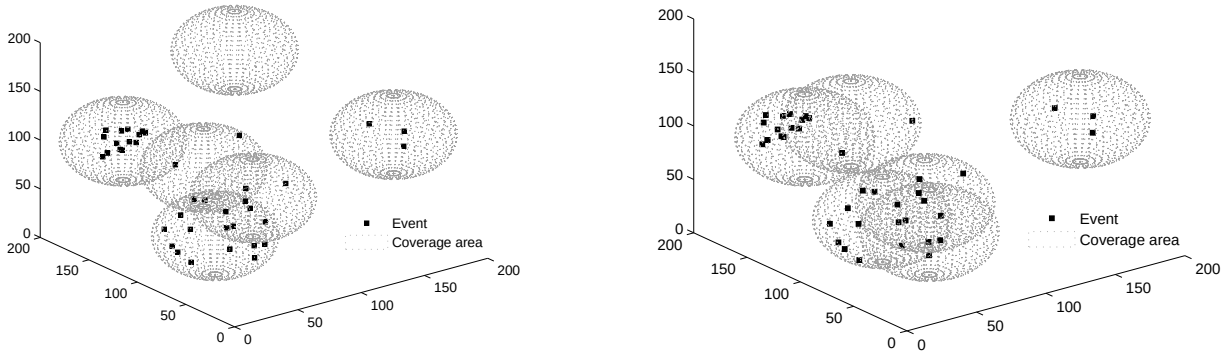
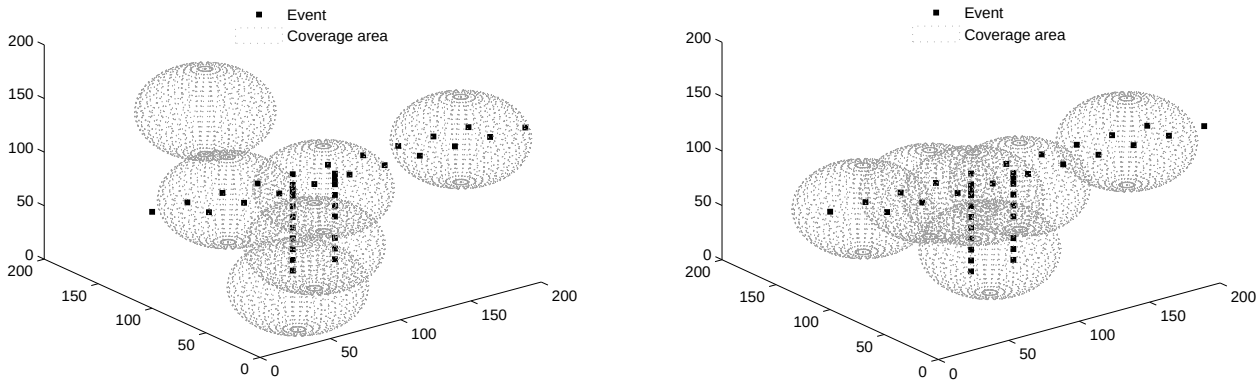
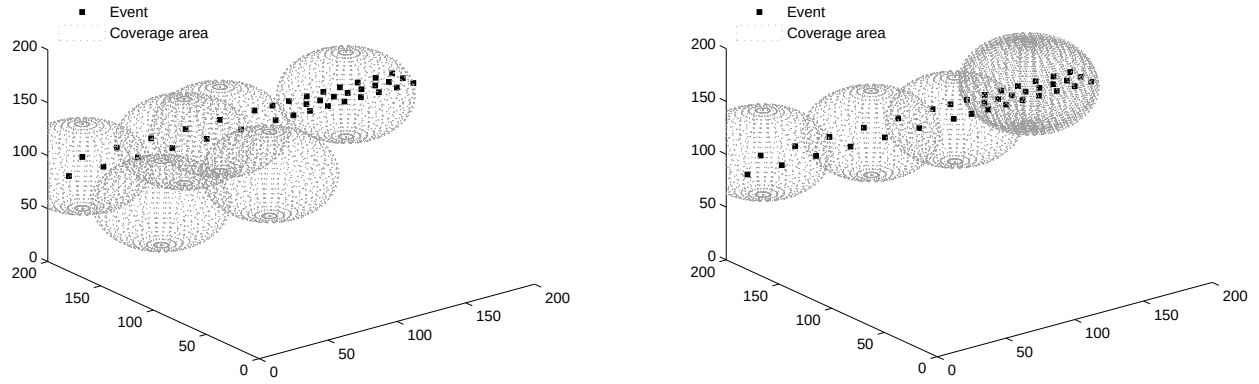


Figure 5. Events distributed non-uniformly in T-shape. (**a**) solution of SOM; (**b**) solution of PSSD.



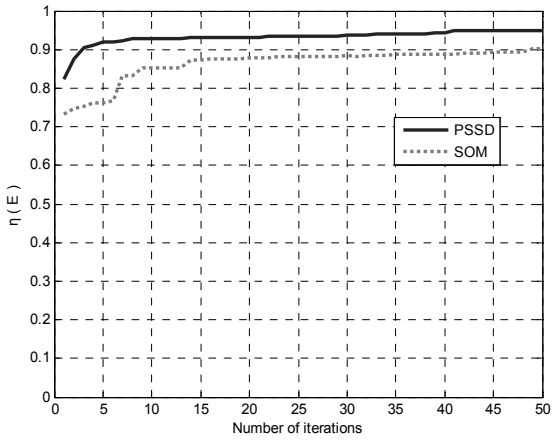
(**a**) (**b**)

Figure 6. Events distributed non-uniformly along a line. (**a**) solution of SOM; (**b**) solution of PSSD.



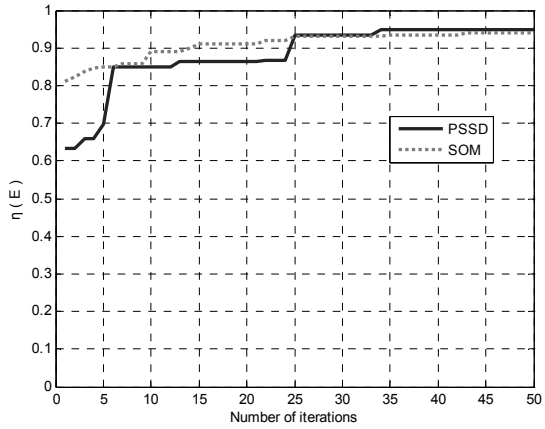
(**a**) (**b**)

Figure 7. The evolving of $\eta(E)$ of the two algorithms in three sets of experiments. (**a**) Experiment Ⅰ; (**b**) Experiment Ⅱ; (**c**) Experiment Ⅲ.

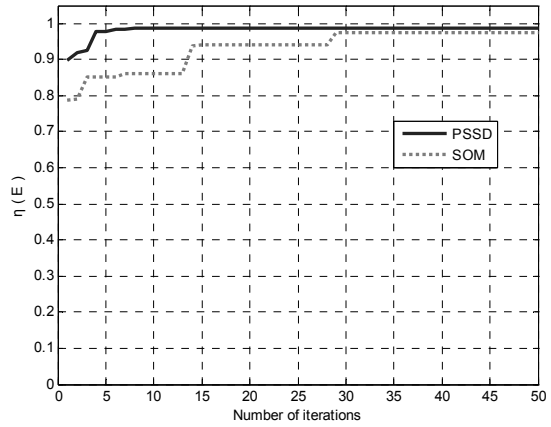


(**a**)

Figure 7. *Cont.*



(**b**)



(**c**)

Table 2 summarized the results for static environments. In each set of experiments, PSSD algorithm runs 20 times to determine the optimal and average coverage efficiency denoted by $\eta^*(E)$ and $\overline{\eta}(E)$, respectively. The average running time is estimated as the time for each sensor to move 50 steps measured on a desktop PC with Intel Core2 CPU@ 2.00 GHz, 1G memory. Again, we observe that PSSD achieves higher coverage efficiency than SOM (1 is the best possible value) at very low computation costs. As deterministic method, SOM just runs once.

Table 2. Summary of Results in Static Experiments.

Experiment set	**Ⅰ: Randomly distributed**		**Ⅱ: T-shape non-uniformly distributed**		**Ⅲ: Line-shape non-uniformly distributed**	
algorithms	PSSD	SOM	PSSD	SOM	PSSD	SOM
the optimal coverage efficiency $\eta^*(E)$	0.9707	0.905	0.9708	0.945	0.9860	0.976
the average coverage efficiency $\overline{\eta}(E)$	0.9634	0.905	0.9535	0.945	0.9847	0.976
average running time (ms)	0.0921	0.024	0.1003	0.063	0.1012	0.055

4.2. Dynamic Environments

Dynamic simulations are conducted in a special water flow environment. Figure. 8 illustrates the water flow environment and the established grids. Figure 9 depicts the distribution of water flow velocity. The water flow velocity is generated using the model described in Section 2.3, and the model parameters are given in Table 3. The update period T for sensors in PSSD is 1 s.

Initially, 16 events are distributed non-uniformly along a line in the water flow environment (Figure 10). The black dots denote the events. Six sensors are deployed randomly in the field. The red circle represents the sensing area of a sensor. Figures 10–13 show the distribution of the events and sensors at four time points during the execution of the algorithm. Evidently, the sensors swarm toward and eventually cover the events, adjusting their positions with the moving events, and maintaining the optimal coverage over time. Figure 14 shows the changes in the coverage efficiency $\eta(E)$ over time. It is clear that after time t2, the sensor deployment becomes stable with $\eta(E)$ fluctuates between 0.8 and 0.9, achieving high monitoring quality.

Table 3. Parameters of the water flow field.

Parameter	k	c	Av	ε	ω
value	$2\pi/7.5$	0.12	1.2	0.3	0.4

Figure 8. Water flow environment and established grids.

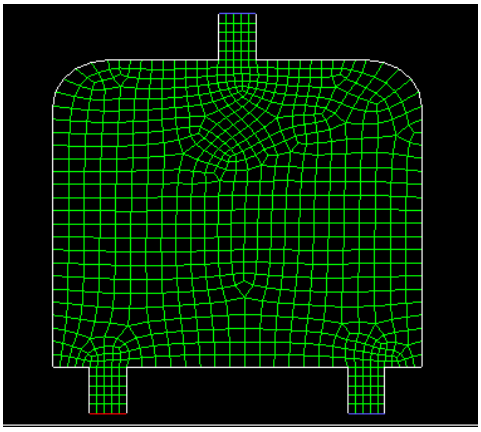


Figure 9. The distribution of water flow velocity.

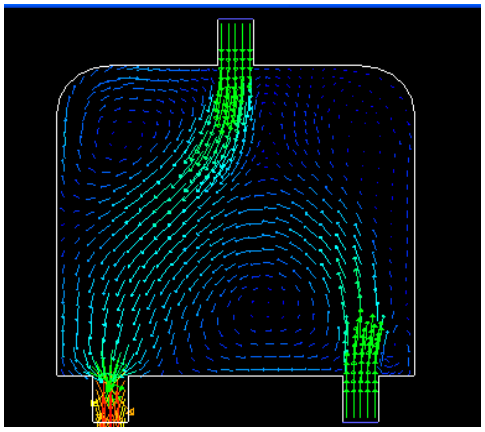


Figure 10. The distribution of events and sensors at t1.

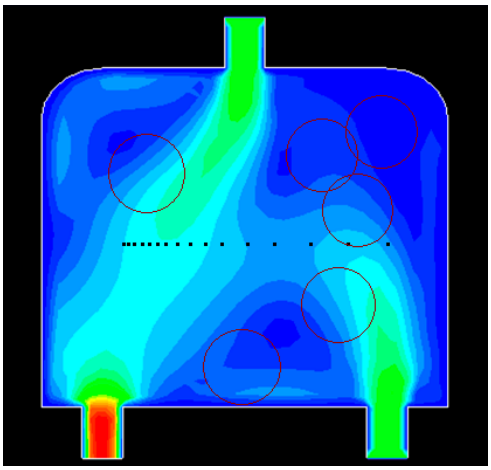


Figure 11. The distribution of events and sensors at t2.

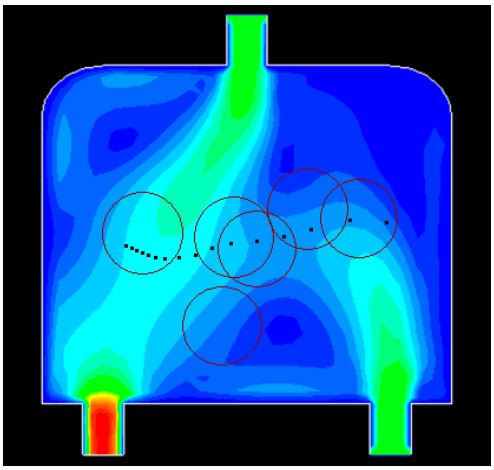
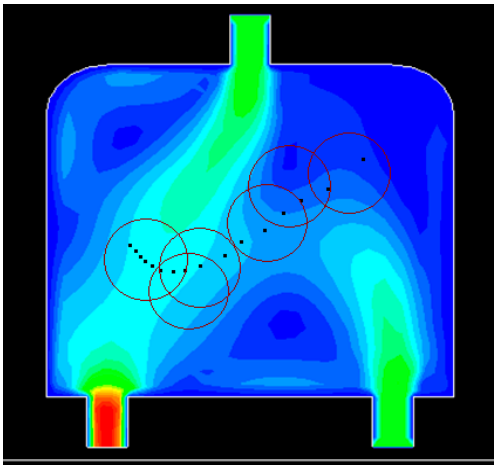


Figure 12. The distribution of events and sensors at t3.



Because the proposed moving strategy can overcome or use the inertia and the water flow to move, as described in Equation (14), the moving efficiency of the sensor is improved. Furthermore, sensors can save energy. Another experiment is conducted to show this point. We modify Equation (14) as $\mathbf{v}_j(t+1) = \mathbf{v}_j^*(t+1)$. The dynamic simulation experiment result is shown in Figure 15. It shows that

the sensors spend more time on moving to cover the events. As a result, more energy is consumed. Even if in the stable stage, $\eta(E)$ is around 0.8. The monitoring quality decreased.

Figure 13. The distribution of events and sensors at t4.

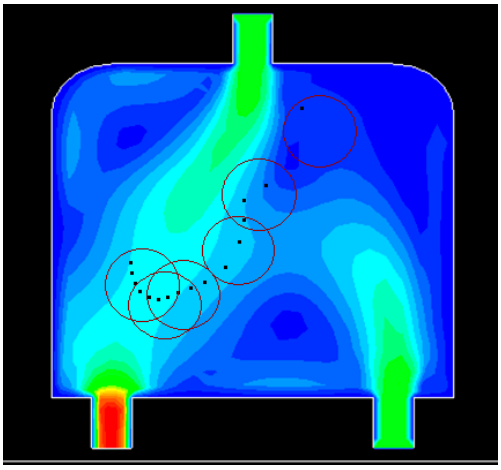


Figure 14. The varying of the coverage efficiency $\eta(E)$.

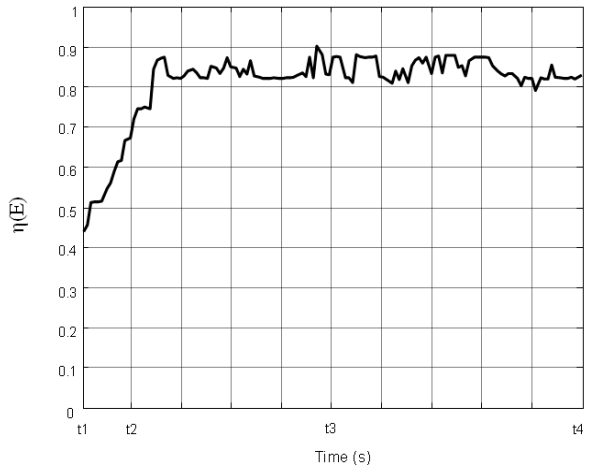
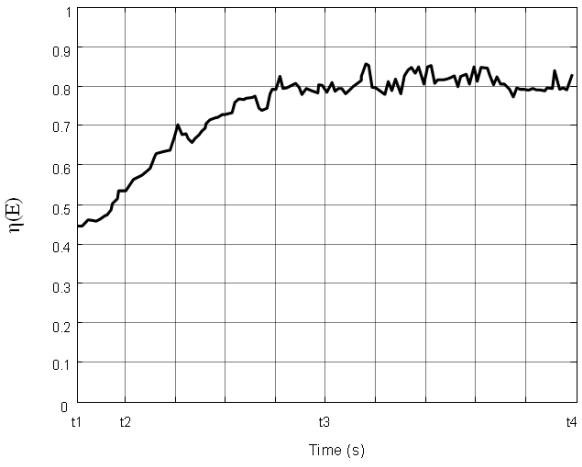
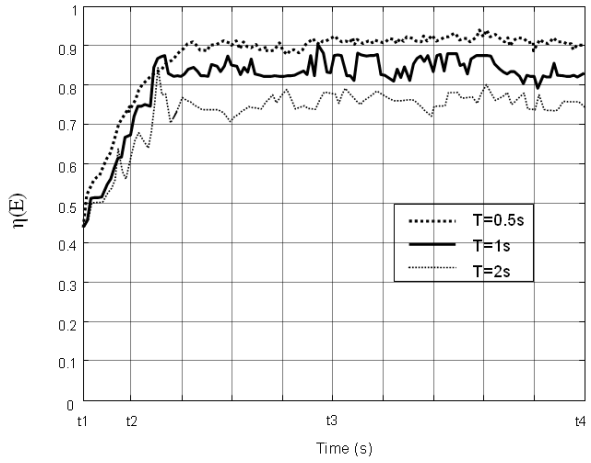


Figure 15. The varying of the coverage efficiency $\eta(E)$ in the comparison experiment.



Impact of the update period: We conduct three sets of dynamic simulations with different parameters T (0.5 s, 1 s, and 2 s). The experimental results are shown in Figure 16. It can be seen that the smaller T is, the more efficiently the sensors move, and the better coverage efficiency the network achieves. However, if T is too small, the sensors will collect the environment information (covered events, adjacent sensors, and near sensors) more frequently, and need to update the velocity frequently. Thus, a smaller T leads to higher communication overhead as well as higher computing cost. An optimal choice of T would need to account for the velocity of the water flow, sensor mobility, communication latency and remaining energy. This will part of our future work.

Figure 16. The varying of the coverage efficiency $\eta(E)$ with different parameter T.



5. Conclusions

In this paper, we investigated the problem of event-driven underwater sensor deployment:

(1) We introduced a novel performance evaluation metric, termed coverage efficiency of the events set, which not only reflects the total number of the sensors covering events, but also quantifies the matching between distributions of sensors and events.

(2) Inspired by particle swarm system, we propose PSSD, a distributedly realizable underwater sensor deployment algorithm. By simulating the flying behavior of the particles (following "gbest" and "pbest") and introducing a crowd control factor, PSSD can drive sensors to swarm to and cover the dynamic events adaptively, and make the distribution of sensors match that of events.

(3) Simulations under both static and dynamic setting show that the algorithm can solve the underwater sensor deployment problem effectively, with fast convergence rate, and amiable to distributed implementation.

As an ongoing objective, we plan to rigorously prove the convergence of PSSD, and conduct experiments in realistic underwater environments.

Appendix:

Lemma 1: Coverage efficiency $\eta(E)$ of event set E will be maximized when all sensors cover some events and all the coverage degrees of events are equal.

Proof: Suppose stochastic vector $\mathbf{Y}=\frac{1}{\mathbf{D}'_A}$, that is $y_i=\frac{1}{D'_A(e_i)}$. $\log \mathbf{Y}$ is a convex function on the set of positive real numbers, so:

$$E[\log \mathbf{Y}] \le \log(E[\mathbf{Y}])$$

$$\sum_{i=1}^{m} D'_A(e_i)\log y_i \le \log \sum_{i=1}^{m} D'_A(e_i)\, y_i$$

$$\sum_{i=1}^{m} D'_A(e_i)\log \frac{1}{D'_A(e_i)} \le \log \sum_{i=1}^{m} D'_A(e_i)\frac{1}{D'_A(e_i)} = \log m$$

$$H_A(E) \le \log m$$

If $D'_A(e_i)=1/m$, which means all the coverage degree of events are equal:

$$H_A(E)=\sum_{i=1}^{m} D'_A(e_i)\log m = \log m$$

Furthermore, if all sensors cover events, that is $\hat{n}=n$, then

$$\eta(E) = \alpha\frac{H_A(E)}{\log m}+\beta\frac{\hat{n}}{n}=\alpha+\beta=1 \text{ (the maximal value)} \square$$

Lemma 2: Coverage efficiency of the events set $\eta(E)$ will increase when the number of sensors covering events is fixed, and the coverage degree of events becomes equal.

Proof: Suppose $D'_A(e_1), D'_A(e_2), \cdots, D'_A(e_m)$ are the coverage degree of $e_1, e_2, ..., e_m$, and $D'_A(e_i) > D'_A(e_j)$, $i,j=1,...,m$. If $D''_A(e_i)=D'_A(e_i)-\varepsilon$, $D''_A(e_j)=D'_A(e_j)+\varepsilon$, where $0<2\varepsilon<\left(D'_A(e_i)-D'_A(e_j)\right)$, and the rest coverage degrees of events don't change, the increment of $\eta(E)$ will be:

$$\begin{aligned}\eta'(E)-\eta(E) &= \frac{\alpha}{\log m}\left(H'_A(E)-H_A(E)\right)\\ &= \frac{\alpha}{\log m}\left\{\left[-D''_A(e_i)\log D''_A(e_i)-D''_A(e_j)\log D''_A(e_j)\right]\right.\\ &\qquad \left.+\left[D'_A(e_i)\log D'_A(e_i)+D'_A(e_j)\log D'_A(e_j)\right]\right\}\\ &= \frac{\alpha}{\log m}\left\{\left[D'_A(e_i)\log D'_A(e_i)+D'_A(e_j)\log D'_A(e_j)\right]\right.\\ &\qquad \left.-\left[(D'_A(e_i)-\varepsilon)\log(D'_A(e_i)-\varepsilon)+(D'_A(e_j)+\varepsilon)\log(D'_A(e_j)+\varepsilon)\right]\right\}\end{aligned}$$

Let $f(x)=(D'_A(e_i)-x)\log(D'_A(e_i)-x)+(D'_A(e_j)+x)\log(D'_A(e_j)+x)$, then:

$$\eta'(E)-\eta(E)=\frac{\alpha}{\log m}(f(0)-f(\varepsilon))$$

$$f'(x)=\log\frac{D'_A(e_j)+x}{D'_A(e_i)-x},$$

when $x\in\left(0,\frac{D'_A(e_i)-D'_A(e_j)}{2}\right)$, $f'(x)<0$, which means $f(x)$ is a decreasing function in this interval, then:

$$\eta'(E)-\eta(E)=\frac{\alpha}{\log m}(f(0)-f(\varepsilon))>0$$

It indicates that $\eta(E)$ is increased when the coverage degree of events becomes equal. $\square$

From Lemmas 1 and 2, we observe that $\eta(E)$ is a suitable metric to characterize event-driven underwater sensor deployment.

Acknowledgments

This work is funded by the National Natural Science Fund of China under Grants 61100211 and 61003307, the Program for New Century Excellent Talents in University of China under Grant NCET-13-0768, and the Anhui Province Science Fund for Distinguished Young Scholars of China under Grant 1408085J05.

Author Contributions

Huazheng Du and Na Xia conceived and designed the experiments; Huazheng Du performed the experiments; Na Xia and Rong Zheng analyzed the data; Huazheng Du wrote the paper; Rong Zheng contributed to the writing.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Akyildiz, I.F.; Pompili, D.; Melodia, T. Underwater Acoustic Sensor Networks: Research Challenges. *Ad Hoc Netw.* **2005**, *3*, 257–279.
2. Munir, M.F.; Xu, H.; Filali, F. Underwater Acoustic Sensor Networking Using Passive Phase Conjugation. In Proceedings of IEEE ICC 2008, Beijing, China, 19–23 May 2008; pp. 2426–2430.
3. Guo, Z.W.; Luo, H.J.; Hong, F.; Yang, M.; Ni, M.X. Current Progress and Research Issues in Underwater Sensor Networks. *J. Comput. Res Dev.* **2010**, *47*, 377–389.
4. Berger, C.R.; Zhou, S.L.; Willett, P.; Liu, L.B. Stratification Effect Compensation for Improved Underwater Acoustic Ranging. *IEEE Trans. Signal Process.* **2008**, *56*, 3779–3783.
5. King, P.; Venkatesan, R.; Li, C. An Improved Communications Model for Underwater Sensor Networks. In Proceedings of IEEE GLOBECOM 2008, New Orleans, LO, USA, 30 November–4 December 2008; pp. 1–6.
6. Pompili, D.; Melodia, T.; Akyildiz, I.F. Deployment Analysis in Underwater Acoustic Wireless Sensor Networks. In Proceedings of 1st ACM International Workshop on Underwater Networks, Los Angeles, CA, USA, 25 September 2006; pp. 48–55.
7. Pompili, D.; Melodia, T.; Akyildiz, I.F. Three-Dimensional and Two-Dimensional Deployment Analysis for Underwater Acoustic Sensor Networks. *Ad Hoc Netw.* **2009**, *7*, 778–790.
8. Pompili, D.; Melodia, T.; Akyildiz, I.F. Distributed Routing Algorithms for Underwater Acoustic Sensor Networks. *IEEE Trans. Wirel. Commun.* **2010**, *9*, 2934–2944.
9. Lee, U.C.; Wang, P.; Noh, Y.T.; Vieira, F.L.M.; Gerla, M.; Cui, J.H. Pressure Routing for Underwater Sensor Networks. In Proceedings of IEEE INFOCOM 2010, San Diego, CA, USA, 14–19 March 2010; pp. 1–9.

10. Tan, X.; Li, J. Cooperative Positioning in Underwater Sensor Networks. *IEEE Trans. Signal Process.* **2010**, *58*, 5860–5871.
11. Teymorian, A.Y.; Cheng, W.; Ma, L.; Cheng, X.; Lu, X.; Lu, Z. 3D Underwater Sensor Network Localization. *IEEE Trans. Mobile Comput.* **2009**, *8*, 1610–1621.
12. Cheng, W.; Teymorian, A.Y.; Ma, L.; Cheng, X.; Lu, X.; Lu, Z. Underwater Localization in Sparse 3D Acoustic Sensor Networks. In Proceedings of IEEE INFOCOM 2008, Phoenix, AZ, USA, 13–18 April 2008; pp. 236–240.
13. Huang, Y.; Liang, W.; Yu, H.B. A Deployment Strategy for Effective Coverage in Underwater Sensor Networks. *J. Electron. Inf. Tech.* **2009**, *31*, 1035–1039.
14. Ibrahim, S.; Cui, J.H.; Ammar, R. Surface-Level Gateway Deployment for Underwater Sensor Networks. In Proceedings of 2007 Military Communications Conference, Orlando, FL, USA, 29–31 October 2007; pp. 1–7.
15. Ibrahim, S.; Cui, J.H.; Ammar, R. Efficient Surface Gateway Deployment for Underwater Sensor Networks. In Proceedings of 2008 IEEE Symposium on Computers and Communications, Marrakech, Morocco, 6–9 July 2008; pp. 1177–1182.
16. Ibrahim, S.; Ammar, R.; Cui, J.H. Geometry-assisted Gateway Deployment for Underwater Sensor networks. In Proceedings of 2009 IEEE Symposium on Computers and Communications, Sousse, Tunisia, 5–8 July 2009; pp. 932–937.
17. Ibrahim, S.; Ammar, R.; Cui, J.H. Surface Gateway Placement Strategy for Maximizing Underwater Sensor Network Lifetime. In Proceedings of 2010 IEEE Symposium on Computers and Communications, Riccione, Italy, 22–25 June 2010; pp. 342–346.
18. Alsalih, W.; Akl, S.; Hassanein, H. Placement of Multiple Mobile Data Collectors in Underwater Acoustic Sensor Networks. *Wirel. Commun. Mob. Comput.* **2008**, *8*, 1011–1022.
19. Alsalih, W.; Hassanein, H.; Akl, S. Delay Constrained Placement of Mobile Data Collectors in Underwater Acoustic Sensor Networks. In Proceedings of 33rd IEEE Conference on Local Computer Networks, Montreal, QC, Canada, 14–17 October 2008; pp. 91–97.
20. Akkaya, K.; Newell, A. Self-deployment of Sensors for Maximized Coverage in Underwater Acoustic Sensor Networks. *Comput. Commun.* **2009**, *32*, 1233–1244.
21. Liu, B.; Ren, F.Y.; Lin, C.; Yang, Y.; Zeng, R.; Wen, H. The Redeployment Issue in Underwater Sensor Networks. In Proceedings of IEEE GLOBECOM 2008, New Orleans, LO, USA, 30 November–4 December 2008; pp. 1–6.
22. Domingo, M.C. Optimal Placement of Wireless Nodes in Underwater Wireless Sensor Networks with Shadow Zones. In Proceedings of Wireless Days 2009, Paris, France, 15–17 December 2009; pp. 1–6.
23. Zeng, B.; Zhong, D.H.; Yao, L. Research of Underwater Mobile Sensor Network Algorithm Based on Water Flow. *Appl. Res. Comput.* **2010**, *27*, 3926–3928.
24. Aitsaadi, N.; Achirt, N.; Boussettat, K.; Pujolle, G. Differentiated Underwater Sensor Network Deployment. In Proceedings of IEEE OCEANS 2007, Aberdeen, Scotland, UK, 18–21 June 2007; pp. 1–6.
25. Koutsougeras, C.; Liu, Y.; Zheng, R. Event-driven Sensor Deployment Using Self Organizing Maps. *Int. J. Sens. Netw.* **2008**, *3*, 142–151.

26. Golen, E.F.; Mishra, S.; Shenoy, N. An Underwater Sensor Allocation Scheme for a Range Dependent Environment. *Comput. Netw.* **2010**, *54*, 404–415.
27. Caruso, A.; Paparella, F.; Vieira, L.F.M.; Erol, M.; Gerla, M. The Meandering Current Mobility Model and its Impact on Underwater Mobile Sensor Networks. In Proceedings of IEEE INFOCOM 2008, Phoenix, AZ, USA, 13–18 April 2008; pp. 771–775.
28. Kennedy, J.; Eberhart, R. Particle Swarm Optimization. In Proceedings of IEEE International Conference on Neural Networks, Washington, DC, USA, 27 November–1 December 1995; pp. 1942–1948.
29. Eberhart, R.; Kenedy, J. A new optimizer using particle swarm theory. In Proceedings of 6th International Symposium on Micro Machine and Human Science, Nagoya, Japan, 4–6 October 1995; pp. 39–43.
30. Xia, N.; Han, D.; Zhang, G.F.; Jiang, J; Vu, K. Study on Attitude Determination Based on Discrete Particle Swarm Optimization. *Sci. China Technol. Sci.* **2010**, *53*, 3397–3403.
31. Du, H.Z.; Xia, N.; Jiang, J.G.; Xu, L.N.; Zheng, R. A Monte Carlo Enhanced PSO Algorithm for Optimal QoM in Multi-Channel Wireless Networks. *J. Comput. Sci. Technol.* **2013**, *28*, 553–563.
32. Samal, N.R.; Konar, A.; Das, S.; Abraham, A. A closed loop stability analysis and parameter selection of the Particle Swarm Optimization dynamics for faster convergence. In Proceedings of the IEEE Congress on Evolutionary Computation, Singapore City, Singapore, 25–28 September 2007; pp. 1769–1776.
33. Ghosh, S.; Das, S.; Kundu, D.; Suresh, K.; Abraham, A. Inter-particle communication and search-dynamics of lbest particle swarm optimizers: An analysis. *Inf. Sci.* **2012**, *182*, 156–168.