A Distributed Algorithm for Constructing Underwater Strong k-Barrier Coverage

Weiqiang Shen[1], Chuanlin Zhang[1](✉), Jinglun Shi[2], Xiaona Zhang[3], and Zhihong Wang[1]

[1] College of Information Science and Technology, Jinan University, Guangzhou 510632, China
weiqshen@gmail.com, tclzhang@jnu.edu.cn, horace@139.com

[2] School of Electronic and Information Engineering, South China University of Technology, Guangzhou 510641, China
shijl@scut.edu.cn

[3] School of Computer Science and Engineering, South China University of Technology, Guangzhou 510641, China
zxn0304@qq.com

Abstract. Sensor barrier coverage has been recognized as an appropriate coverage model for intrusion detection, and many achievements have been obtained in two-dimensional (2D) terrestrial wireless sensor networks. However, the achievements based on 2D assumption cannot be directly applied in three-dimensional (3D) application scenarios, e.g., underwater wireless sensor networks. In this paper, we aim to devise a distributed algorithm for constructing maximum level underwater strong k-barrier coverage with mobile sensors in 3D underwater environment. Considering that an underwater strong k-barrier coverage is constituted with k underwater strong 1-barrier coverage which is referred to as layer in this work, we first derive the optimal positions of the sensors in each layer, then we propose a distributed algorithm for constructing maximum level underwater strong k-barrier coverage with available mobile sensors layer by layer from left to right in 3D underwater environment. Simulation results show that the proposed algorithm outperforms the optimal centralized approach (i.e., Hungarian algorithm) in terms of duration and achieves performance close to Hungarian algorithm with respect to several performance metrics.

Keywords: Wireless sensor networks · Underwater wireless sensor networks · Underwater sensor barrier coverage · Three-dimensional underwater strong k-barrier coverage

1 Introduction

Wireless sensor networks (WSNs) have many real life applications in environmental monitoring, battlefield surveillance and intrusion detection, etc. As an

B. Li et al. (Eds.): IoTaaS 2018, LNICST 271, pp. 179–194, 2019.
https://doi.org/10.1007/978-3-030-14657-3_18

important problem in WSNs, barrier coverage is garnering more and more attention in recent years [1,17–19]. Compared with the area coverage problem, barrier coverage does not necessarily cover every point of the monitored region, but rather only needs to detect intruders that cross the border [12]. Therefore, it is more cost-efficient for large-scale deployment of wireless sensors, and has been widely employed in practical security related applications, e.g., international border surveillance, intrusion detection and critical infrastructure protection.

Most existing works on barrier coverage assume that sensors are deployed in 2D long thin belt region, where a barrier is a chain of sensors from one end of the deployment region to the other end with overlapping sensing zones of adjacent sensors. This assumption is reasonable in 2D terrestrial WSNs where the height of the network is usually negligible as compared to its length and width. However, the 2D assumption may not be appropriate when considering WSNs in 3D application scenarios, e.g., underwater wireless sensor networks (UWSNs), where the sensors are finally distributed over 3D underwater environment. As technology advances, efforts are currently underway to extend sensor barrier coverage to underwater application scenarios. For example, underwater sensor barrier has been considered for detecting submarine intrusion in marine environment [3].

In 3D underwater application scenarios[1], the sensors composing UWSNs are distributed at different depths in underwater environment. In this case, a sensor barrier is not a chain of sensors from one end of the deployment region to the other end with overlapping sensing zones of adjacent sensors any more. Instead, a sensor barrier deployed in underwater environment should be a set of sensors with overlapping sensing zones of adjacent sensors that covers an entire (curly) surface that cuts across the 3D underwater space [3]. In practical underwater environment, as the existence of sudden sensor failures and water current which may led to that a sensor deviates from its desired position, a single underwater sensor barrier usually fails to provide adequate service quality. Hence, many real life underwater applications require k-barrier coverage to guarantee their service quality. The notion of k-barrier coverage is first defined in [11], the authors introduced two types of k-barrier coverage including weak barrier coverage, which guarantees to detect intruders moving along congruent paths, and strong barrier coverage, which guarantees to detect intruders no matter what crossing paths they choose.

In this paper, we aim to devise a distributed algorithm for constructing maximum level underwater strong k-barrier coverage with available mobile sensors in underwater environment, to thwart the intruders crossing the monitored 3D underwater environment. Considering that an underwater strong k-barrier coverage is constituted with k underwater strong 1-barrier coverage which is referred to as layer in this work, we first derive the optimal positions of the sensors in each layer, then we propose a distributed algorithm for constructing maximum level underwater strong k-barrier coverage with available mobile sensors layer by layer from left to right in 3D underwater environment. Simulation results show

[1] In this paper, we only consider 3D underwater application scenarios where the sensors are deployed in 3D underwater environment.

that the proposed algorithm outperforms the optimal centralized approach (i.e., Hungarian algorithm) in terms of duration and achieves performance close to Hungarian algorithm with respect to several performance metrics.

The rest of the paper is organized as follows. Next section reviews the related work. In Sect. 3, we explain the network model and provide the problem statement. In Sect. 4, we propose a distributed deployment algorithm for constructing maximum level underwater strong k-barrier coverage with available mobile sensors in underwater environment. Section 5 evaluates the performance of the proposed algorithm through extensive simulations, and finally, Sect. 6 concludes the paper.

2 Related Work

As an important problem in WSNs, barrier coverage has been extensively studied in the past decades. Most existing works consider 2D barrier coverage in terrestrial WSNs, while only recently 3D barrier coverage in UWSNs has been studied. In the following, we review the works on 2D and 3D sensor barrier coverage.

2.1 2D Sensor Barrier Coverage

The concept of barrier coverage was first appeared in [7] in the context of many-robot systems. In [11], Kumar et al. developed theoretical foundations for laying barriers of wireless sensors. They defined two types of barrier coverage including weak barrier coverage, which guarantees to detect intruders moving along congruent paths, and strong barrier coverage, which guarantees to detect intruders no matter what crossing paths they choose. Chen et al. [4] introduced the concept of "quality of barrier coverage" and proposed an effective way to measure it. Fan et al. [6] studied the coverage of a line interval with a set of wireless sensors with adjustable coverage ranges. Liu et al. [14] studied the strong barrier coverage of a randomly-deployed sensor network on a long irregular strip region. They showed that in a rectangular area of width ω and length ℓ with the relation $\omega = \Omega(\log \ell)$, if the sensor density reaches a certain value, then there exist, with high probability, multiple disjoint sensor barriers across the entire length of the area such that intruders cannot cross the area undetected; On the other hand, if $\omega = o(\log \ell)$, then with high probability there is a crossing path not covered by any sensor regardless of the sensor density. He et al. [9] presented a condition under which line-based deployment is suboptimal, and proposed a new deployment approach named curve-based deployment. Wang et al. [18] explored the effects of location errors on barrier coverage on a 2D plane by considering two scenarios (i.e. only stationary nodes have location errors, stationary and mobile nodes both have location errors), and proposed a fault-tolerant weighted barrier graph to model the barrier coverage formation problem.

With the advances of technology, sensor mobility has been incorporated into sensor deployment framework [20], which offers more flexibility for designing more efficient sensor deployment strategies to solve coverage problem in WSNs.

Li and Shen [13] studied the 2D MinMax barrier coverage problem of moving n sensors in a 2D plane to form a barrier coverage while minimizing the maximum sensor movement for the sake of balancing battery power consumption. Dobrev et al. [5] studied three optimization problems related to the movement of sensors to achieve weak barrier coverage, i.e., minimizing the number of sensors moved, minimizing the average distance moved by the sensors, and minimizing the maximum distance moved by the sensors. Silvestri and Goss [17] proposed an original algorithm called MobiBar, which has the capability of constructing k-barrier coverage in WSNs, self-reconfiguration and self-healing. Ban et al. [2] considered k-barrier coverage problem in 2D wireless sensor networks, and devised an approximation algorithm called AHGB to construct 1-barrier efficiently. Furthermore, based on AHGB, a Divide-and Conquer algorithm was proposed to construct k-barrier coverage for large scale WSNs. Saipulla et al. [16] explored the fundamental limits of sensor mobility on barrier coverage, and presented a sensor mobility scheme that constructs the maximum number of sensor barriers with the minimum sensor moving distance. Li and Shen [12] studied the 2D MinMax problem of barrier coverage in which the barrier is a line segment in a 2D plane and the sensors are initially resided on this plane.

2.2 3D Sensor Barrier Coverage

There are only a handful of works that have considered 3D sensor barrier coverage. The most related works to ours was presented in [3], the authors considered constructing underwater sensor barriers to thwart illegal intrusion of submarines, and devised a centralized approach and a decentralized approach to achieve this goal, respectively. However, both the centralized and decentralized approaches require one or several sensor nodes as leader to collect and propagate sensor positions. In this case, single point failure problem may not be avoid. In this work, we aim to devise a fully distributed deployment algorithm, which does not require any leader nodes to collect and propagate sensor positions, for constructing maximum level underwater strong k-barrier coverage with available mobile sensors in underwater environment, and hope to provide insights into further researches in 3D wireless sensor networks.

3 Network Model and Problem Statement

3.1 Network Model

We model the underwater deployment region as a cuboid of size $l \times w \times h$, where l, w, and h denote the length, the width, and the height of the cuboid, respectively. For the sake of clearness, we assume that the cuboid is located in a 3D Cartesian coordinate system with origin at the center point of the cuboid. Initially, n sensors are deployed in the cuboid where the sensor positions can follow any type of distribution, such as uniform distribution, Poisson distribution and normal distribution, as shown in Fig. 1. We assume that all the deployed sensors have the same sensing radius r_s and communication radius r_c, where $r_c \geq 2r_s$. Similarly to most previous works in underwater sensor deployment, we assume that all the deployed sensors are able to identify their current locations, and each sensor is able to relocate itself from its initial position to another specified position at a maximum speed of V_{max} m/s in underwater environment. For simplicity, we assume an ideal 0/1 sphere sensing model that an object within (outside) a sensor's sensing sphere is detected by the sensor with probability one (zero).

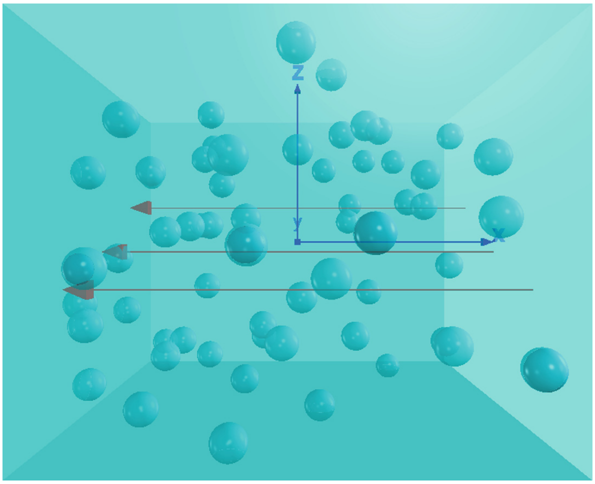

Fig. 1. Initially the sensors are deployed in the cuboid. Without loss of generality, we assume that an intruder's traversing path is a continuously moving trajectory starting at the cuboid's right face and ending at the opposite face.

Without loss of generality, we assume that the illegal intruders to be detected by underwater sensor barrier move along the direction of cuboid length, as shown in Fig. 1, the intruder's traversing path, i.e., the red lines, is a continuously moving trajectory starting at the cuboid's right face and ending at the opposite face.

3.2 Problem Statement

According to the aforementioned assumptions, the underwater deployment region is modeled as a cuboid of size $l \times w \times h$, where l, w and h denote the length, the width and the height of the cuboid, respectively. The position of a point in the cuboid is denoted by coordinates (x, y, z), where x, y, and z are the x-coordinate, the y-coordinate and the z-coordinate of this point, respectively. We suppose that n mobile sensors $S = \{s_1, s_2, ..., s_n\}$ is deployed in the cuboid with initial positions $p_1, p_2, ..., p_n$. Let (x_i, y_i, z_i) denote the coordinates of position p_i of sensor s_i.

The goal of our work is to devise a distributed deployment algorithm to drive the sensors to move to desired positions, and thus provide maximum level underwater strong k-barrier coverage, as shown in Fig. 2. Our problem is how to devise such a distributed deployment algorithm to achieve our goal.

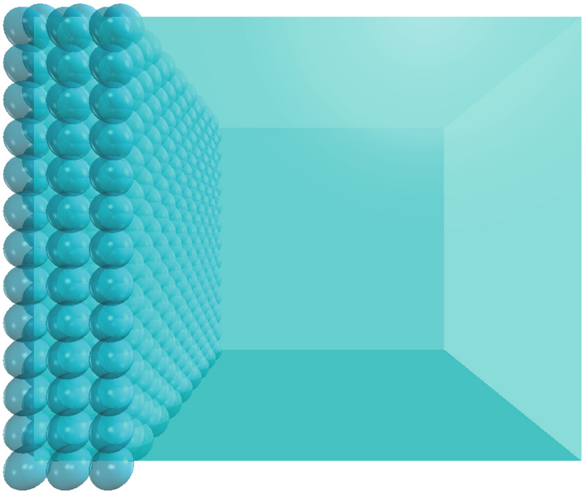

Fig. 2. In 3D UWSNs, an underwater strong k-barrier coverage is constituted with k underwater strong 1-barrier coverage (In this example, k = 3).

4 A Distributed Algorithm

In this section, we propose a distributed algorithm for constructing maximum level underwater strong k-barrier coverage with available mobile sensors.

4.1 Main Ideas

As showed in Fig. 2, an underwater strong k-barrier coverage is constituted with k underwater strong 1-barrier coverage. For simplicity, we refer to an underwater strong 1-barrier coverage as a layer in this work, and from left to right, we enumerate the sensor barriers b0, b1, b2,..., where b0 is base-layer, and the indexes of the other layers increase with the distance from b0. The main idea of the proposed algorithm is to construct underwater strong k-barrier coverage layer by layer from left to right.

For the sake of clearness, we introduce two important concepts. One is constructed-layer, on which no sensor has adjacent vacant positions resided on the same layer as itself, in other words, this layer has been constructed. The other is constructing-layer, on which there are still some vacant positions to be occupied by sensors, in other words, this layer is being constructed. Since underwater strong k-barrier coverage is to be constructed layer by layer, there is only one constructing-layer at the same time during the construction process. For example, base-layer b_0 is the constructing-layer at the beginning, and after the construction of b_0 is completed, b_0 becomes a constructed-layer and layer b_1 is the constructing-layer, and so on. Furthermore, we refer to the sensors resided on constructed-layer and constructing-layer as fixed sensors, while the sensors not already resided on constructed-layers or constructing-layer are referred to as movable sensors. A fixed sensor can not move any more, but a movable sensor can move freely in the underwater space according to its local information. In order to make each movable sensor move orderly to avoid collision as much as possible, and reduce the complexity of the proposed algorithm to ensure that the proposed algorithm performs efficiently, we set a movable sensor's movement route as follows. If a movable sensor s_i with initial position (x_s, y_s, z_s) intends to move to position (x_e, y_e, z_e), it first moves to position (x_s, y_e, z_e) by means of parallel moving, which means that a sensor moves on a plane parallel to the base-layer (i.e., the sensor's x-coordinate will not be changed in the moving process), then moves to position (x_e, y_e, z_e) by means of vertical moving, which means that a sensor moves along a straight line perpendicular to the base-layer. Nevertheless, in rare circumstances, a sensor moving by means of parallel moving may still collide with other sensors moving by means of parallel or vertical moving. Fortunately, the sensors can avoid collision by existing approaches [8], hence we will not discuss the collision avoidance among sensors in deeper in this work.

4.2 The Optimal Final Positions of Sensors in Each Layer

In order to construct maximum level underwater strong k-barrier coverage, we need to minimize the number of sensors required in each layer. Practically, as shown in Fig. 2, if we project a layer onto the cuboid's left face, the sensors on this layer will completely cover the cuboid's left face which is a rectangle of size $w \times h$, where w and h are the width and height of the cuboid, respectively. Hence, for each layer, the minimum number of sensors is equal to the minimum number of hexagons to completely cover a rectangle of size $w \times h$, as shown in Fig. 3. In the context of our work, for each layer, once we place a sensor in the center of each regular hexagon, the sensing range of all sensors will completely cover the rectangle and thus provides strong 1-barrier coverage. According to the above analysis, in each layer, the optimal final positions of sensors are the center points of regular hexagons whose x-coordinates equal to the layer's x-coordinate. In the following, we derive y, z-coordinates of each center point of regular hexagon.

As shown in Fig. 3, we first obtain the y-coordinate of each column via Eq. (1), where i denotes the i-th column.

$$f_y(w, r, i) = \begin{cases} \frac{r}{2}, & \text{i} = 0 \\ \frac{r}{2} + \frac{i \times r}{2}, & \frac{r}{2} + \frac{(f_c(w,r)-1)\times r}{2} < w \\ w, & \frac{r}{2} + \frac{(f_c(w,r)-1)\times r}{2} \geq w. \end{cases} \tag{1}$$

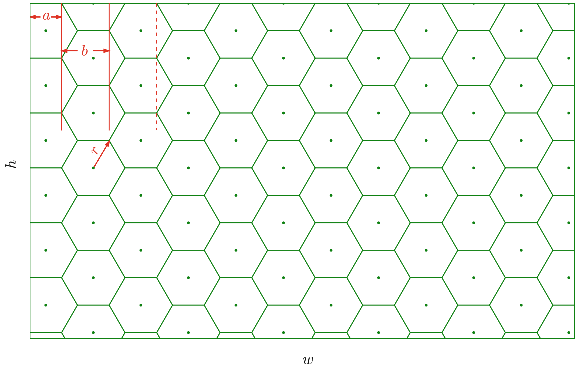

Fig. 3. A rectangle that is completely covered by minimum regular hexagons.

Then we get the z-coordinate of each point row by row. For odd-number columns, h denotes the cuboid height, r denotes the sensing radius, j denotes the j-th row, and $0 \leq j < \lceil \frac{h}{r \times \sqrt{3}} \rceil$, we have

$$f_z(w, r, j) = h - \frac{r \times \sqrt{3}}{2} - j \times r \times \sqrt{3}. \tag{2}$$

For even-number columns, h denotes the cuboid height, r denotes sensor's sensing radius, j denotes the j-th row, and $0 \leq j < \lceil \frac{h - \frac{r \times \sqrt{3}}{2}}{r \times \sqrt{3}} \rceil + 1$, we have

$$f_z(w, r, j) = h - j \times r \times \sqrt{3}. \tag{3}$$

Combining Eqs. (2) and (3), we have

$$f_z(w, r, j) = \begin{cases} h - \frac{r \times \sqrt{3}}{2} - j \times r \times \sqrt{3}, & \text{odd-number columns} \\ h - j \times r \times \sqrt{3}, & \text{even-number columns.} \end{cases} \tag{4}$$

Finally, by combining the layer's x-coordinate x, we obtain the optimal final positions $(x, f_y(w, r, i), f_z(w, r, j))$ of sensors in each layer.

4.3 A Distributed Algorithm for Constructing Underwater Strong k-Barrier Coverage

The proposed algorithm provides the interleaved execution of three main activities, namely vertical movement, vacant position processing and parallel movement. Figure 4 shows the flowchart of these activities.

Before starting the execution of the three main activities, each sensor performs an initial movement aiming to move to the closest centerline, which is perpendicular to the base-layer and pass through the hexagon center point of each layer, as shown in Fig. 5.

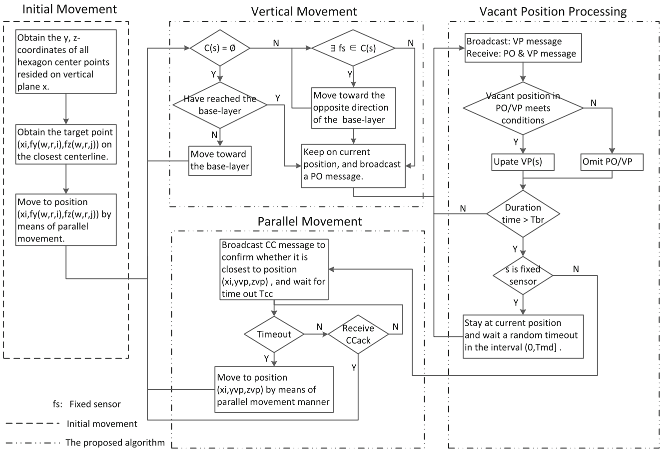

Fig. 4. Flowchart of the initial movement and the proposed algorithm executed by a sensor s with initial position (x, y, z).

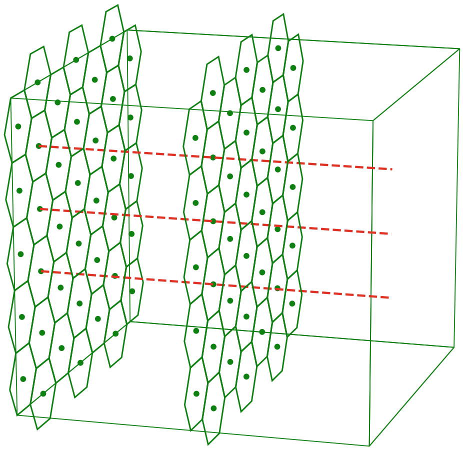

Fig. 5. The centerlines are perpendicular to the base-layer and pass through the hexagon center points in each layer. As shown in this figure, the red dash lines are centerlines. (Color figure online)

Initial Movement. In order to make each movable sensor move orderly in underwater environment, and thus ensure that the proposed algorithm performs effectively and efficiently, each sensor initially moves to the centerline closest to itself by means of parallel moving. It is worth noting that after the initial movement, all sensors are on the centerlines, and by executing the proposed algorithm, each sensor only moves along the centerline or moves to another centerline by means of parallel moving. In the following, we describe how to obtain the centerline closest to a sensor s_i.

For a sensor s_i with initial position (x_i, y_i, z_i), it first obtains the position list of the hexagon center points via Eqs. (1) and (4). Then it calculates the distance between sensor s_i and each hexagon center point $(x_i, f_y(w, r, i), f_z(w, r, j))$. Obviously, a minimum distance between sensor s_i and point $(x_i, f_y(w, r, i'), f_z(w, r, j'))$ will be obtained, the minimum distance is the distance between sensor s_i and the centerline closest to it. Thus, we obtain the centerline closest to sensor s_i:

$$\begin{cases} y = f_y(w, r, i'), \\ z = f_z(w, r, j'). \end{cases} \tag{5}$$

Vertical Movement. The vertical movement activity is twofold. On one hand, in order to increase the connectivity of the network, each movable sensor moves toward base-layer by means of vertical moving if it has no sensor in radio proximity which is resided on the same centerline and closer to base-layer than itself. The movement is stopped as soon as such a sensor is found or the base-layer is reached. On the other hand, to ensure that all sensors can finally move their desired positions, each movable sensor with a distance less than $2r_s$ from a fixed sensor moves toward the opposite direction of the base-layer until the distance not less than $2r_s$. By vertical movement, a movable sensor connects with other sensors to form a network component and thus prevent it becoming an isolated sensor. Moreover, sensors in the same network component can communicate with each others by means of a multi-hop manner. The vertical movement is based on the following described protocol.

For a sensor s_i, we refer to $C(s_i)$ as a set of sensors in the sensor's radio proximity and resided on the same centerline and closer to base-layer with respect to itself. If $C(s_i) = \emptyset$, it moves toward the base-layer along the centerline until $C(s_i) \neq \emptyset$ or it reaches the base-layer. If otherwise $C(s_i) \neq \emptyset$, it checks whether $C(s_i)$ contains fixed sensors. If $C(s_i)$ contains fixed sensors, it moves toward the opposite direction of the base-layer until $C(s_i)$ does not contain fixed sensor any more. If otherwise $C(s_i)$ does not contain fixed sensors, it keeps on its current position. It is worth noting that, moving a sensor s_i to the opposite direction of the base-layer when it finds that $C(s_i)$ contains fixed sensors, ensures that all sensors can finally move to their desired positions even if they are all initially placed on base-layer.

Finally, if sensor s_i has reached the constructing-layer, it broadcasts a PO (position occupation) message, which contains sensor ID, timestamp and its current position information, in the network to remind other sensors to remove this position from their vacant position queues, if their queue contain this position. And then, it gives start to the vacant position processing. If otherwise, it starts the vacant position processing without broadcasting a PO message.

Vacant Position Processing. There are three aspects in vacant position processing. First, to notify the other sensors of the vacant positions resided on the constructing-layer, each sensor periodically detects and broadcasts its adjacent vacant positions resided on the same layer as itself in the network. Second, each sensor receives and processes the vacant positions broadcasted by other sensors. It is worth noting that, in order to manage the vacant positions resided on the constructing-layer, each sensor maintains a vacant position queue, where the vacant positions are sorted by the distance between vacant position and the sensor in an ascending order. Finally, by leveraging the data cached in vacant position queue, a sensor makes its movement decision. The detailed protocol of the vacant position processing is described as follows.

For a sensor s_i with a vacant position queue $VP(s_i)$ (initially $VP(s_i) = \emptyset$), it first broadcasts a VP (Vacant Position) message containing sensor ID, timestamp and its adjacent vacant positions resided on the same layer as itself in the network, then it waits for the expiration of its timeout T_{rcv} to receive VP message and PO message broadcasted by other sensors. Notice that, we assume that the proposed algorithm is implemented over a communication protocol stack which handles possible transmission errors and message losses by means of timeout and retransmission mechanisms, and each sensor keeps track of the VP messages and PO messages sent so far to avoid multiple retransmissions and multiple processing by checking the unique message mark consisting of sensor ID and time stamp. When sensor s_i receives a VP message, it checks the cardinality of $VP(s_i)$. If $|VP(s_i)| = 0$, the vacant positions are put into $VP(s_i)$. If otherwise, it compares the x-coordinate of vacant position with that of one sensor in $VP(s_i)$, if larger, these vacant positions are omitted; if equal, these vacant positions are inserted into $VP(s_i)$; if smaller, it sets $VP(s_i) = \emptyset$ and puts these vacant positions into $VP(s_i)$. Similarly, when sensor s_i receives a PO message containing position p (p is a vacant position before being occupied), it removes p from its vacant position queue if p is in the queue.

By means of the above protocol, each sensor maintains a vacant position queue, where vacant positions are all resided on the constructing-layer[2] and sorted by the distance between vacant position and the sensor in an ascending order.

After the expiration of time T_{rcv}, sensor s_i determines whether it is a movable sensor or a fixed sensor by the following protocol. Sensor s_i first obtains the constructing-layer's x-coordinate by returning the first element's x-coordinate from its vacant position queue, then it compares the constructing-layer's x-coordinate with its, if larger, sensor s_i is a fixed sensor resided on constructed-layer; if equal, sensor s_i is a fixed sensor resided on the constructing-layer; if smaller, sensor s_i is a movable sensor. Then, according to this determination, sensor s_i makes its movement decision as follows.

If sensor s_i is a fixed sensor, it keeps its current position and waits a random timeout in the interval $(0, T_{md}]$ to restart the vacant position processing activity. If otherwise sensor s_i is a movable sensor, it terminates this activity and starts the parallel movement activity.

[2] The vacant positions resided on constructing-layer have the smallest x-coordinate, consequently they are finally cached in vacant position queue according to the above protocol.

Parallel Movement. In this activity, a movable sensor first confirms whether it is closest to the vacant position at the head of its vacant position queue. If it is the closest one, it moves to this vacant position by means of parallel moving then starts the vertical movement activity. If otherwise, it terminates this activity and gives start to the vertical movement activity. The approach to find the closest vacant position is described as follows.

For a movable sensor s_i with coordinates (x_i, y_i, z_i), it broadcasts a CC (Confirm the Closest) message containing sensor ID, timestamp, the vacant position vp with coordinates (x_{vp}, y_{vp}, z_{vp}) and the distance between sensor s_i and the vacant position vp, and then waits for the expiration of its timeout T_{cc}. When the other sensors receive the CC message, they check whether they are closer to position vp than sensor s_i, if closer, they broadcast a CCack message containing CC's original content. If otherwise, they omit this CC message. If, while waiting for the expiration of its timeout T_{cc}, sensor s_i receives a CCack message, it terminates this parallel movement activity and starts vertical movement activity. After the expiration of a timeout T_{cc}, sensor s_i moves to position (x_i, y_{vp}, z_{vp}) by means of parallel moving. Once sensor s_i reaches position (x_i, y_{vp}, z_{vp}), it gives start to the vertical movement. Similar to the vacant position processing activity, we assume that each sensor keeps track of the CC/CCack messages sent so far to avoid multiple retransmissions and multiple processing by checking the unique message mark consisting of sensor ID and time stamp.

5 Performance Evaluation

In this section, we evaluate the performance of the proposed algorithm by comparing with Hungarian algorithm [10] which obtains the optimal total movement distance of all sensors. The simulation parameters are described as follows. We conduct the experiments in a Python-based simulator, each data point of our experiment results is an average value of the data collected by running the experiments 100 times.

Considering that the average ocean depth is 3795 m [15], and some commercial magnetic sensor can detect submarines at distances of several hundred meters [3]. We assume that the underwater space where the underwater strong k-barrier coverage to be constructed is a cuboid of length $l = 4200$ m, width $w = 4000$ m, and height $h = 3800$ m, respectively. We set r_s and r_c to 200 m and 420 m, respectively.

As far as we know, the proposed algorithm is the first fully distributed algorithm for constructing maximum level underwater strong k-barrier coverage with available mobile sensors. To evaluate the performance of the proposed algorithm, we compare it to a classic centralized approach, namely Hungarian algorithm [10], which minimizes the total movement distance of all sensors. We assume that all sensors are initially randomly deployed throughout the underwater environment. The number of available sensors ranges from 100 to 800 with an increment of 100 each time.Three performance metrics are considered in the simulation experiments, namely maximum movement distance of all sensors, total movement distance of all sensors and duration of the construction of underwater strong k-barrier coverage.

Figure 6 shows the maximum movement distance of all sensors achieved by the proposed algorithm and Hungarian algorithm. Both algorithms show a gentle decreasing behavior of the maximum movement distance as the number of sensors increases. This is because more sensors locate closer to their final positions as the number of sensors increases. Hungarian algorithm achieves about 35% lower maximum movement distance due to the fact that, under Hungarian algorithm each sensor moves from its initial position to its final position along a straight line, while most of sensors in the proposed algorithm move to their final positions by means of vertical or parallel moving according to their movement decisions, which makes them move along zigzag route.

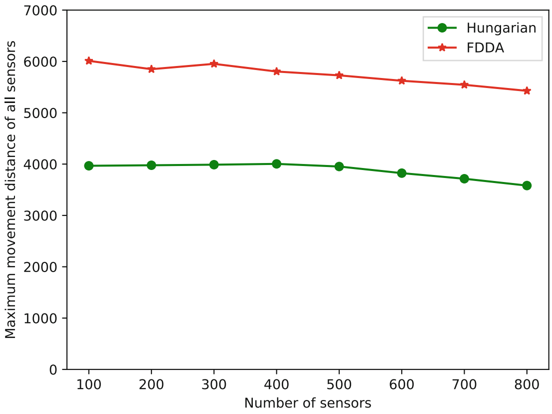

Fig. 6. Maximum movement distance vs number of sensors.

For the total movement distance of all sensors, although the Hungarian algorithm is the optimal solution that achieves the minimum total movement distance, the proposed algorithm also presents a sub-optimal results, as shown in Fig. 7, the optimal total movement distance of Hungarian algorithm is only 28% lower than that of the proposed algorithm.

The duration depicts the length of the time that the construction of the underwater strong k-barrier coverage continues. For Hungarian algorithm, the duration includes the computing time consumed by the central unit to assign a final position per sensor, and movement time consumed by all sensors to move from their initial positions to their final positions along a straight line. While for the proposed algorithm, the duration mainly includes the movement time consumed by all sensors to move from their initial positions to their final positions along a zigzag route. As shown in Fig. 8, the duration of Hungarian algorithm increases sharply as the number of sensors increases, while the increase of the duration of the proposed algorithm is relatively flat. This is because the optimal

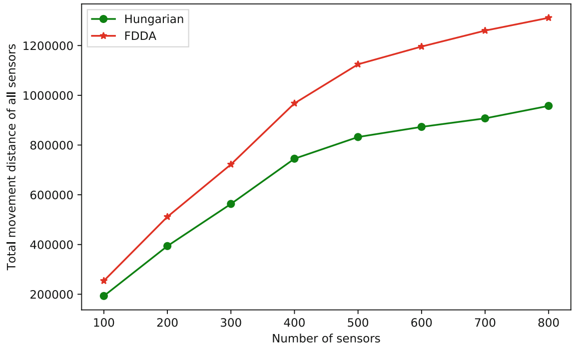

Fig. 7. Total movement distance vs number of sensors.

computational complexity of Hungarian algorithm is $O(n^3)$, while for the proposed algorithm, each sensor in three main activities only needs to communicate with at most $(n-1)$ sensors at each step, hence the total time complexity of the proposed algorithm is $O(n)$. It is worth noting that, the duration of Hungarian algorithm is a little less than that of the proposed algorithm when the number of sensors is less than 500, this is because, in this case, the central unit of the Hungarian algorithm only consumes relatively little time, while the maximum movement distance of the proposed algorithm is larger than that of Hungarian algorithm, which results in more time needed for the proposed algorithm to move the sensors to their desired positions. In general, the proposed algorithm outperforms Hungarian algorithm in terms of duration.

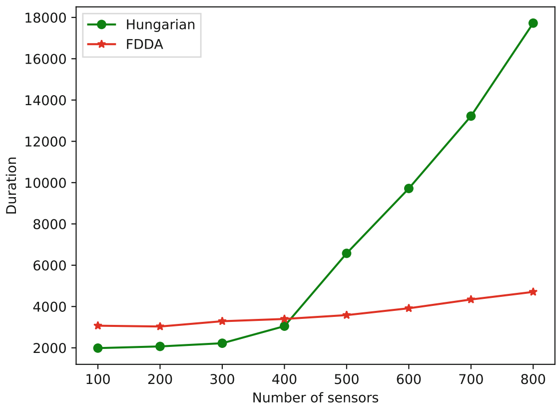

Fig. 8. Duration vs number of sensors.

6 Conclusion

In this paper, we devised a distributed algorithm for constructing maximum level underwater strong k-barrier coverage with available mobile sensors in 3D underwater environment. Considering that an underwater strong k-barrier coverage is constituted with k underwater strong 1-barrier coverage which is referred to as layer in this work, we first derive the optimal positions of the sensors in each layer, then we proposed a distributed algorithm for constructing maximum level underwater strong k-barrier coverage with available mobile sensors layer by layer from left to right in 3D underwater environment. By extensive simulations, we showed that the proposed algorithm outperforms the optimal centralized approach (i.e., Hungarian algorithm) in terms of duration, and achieves performance close to Hungarian algorithm with respect to two performance metrics, namely maximum movement distance of any sensor and total movement distance of all sensors.

References

1. Balister, P., Bollobás, B., Sarkar, A.: Barrier coverage. Random Struct. Algorithms **49**(3), 429–478 (2016). https://doi.org/10.1002/rsa.20656
2. Ban, D., Yang, W., Jiang, J., Wen, J., Dou, W.: Energy-efficient algorithms for k-barrier coverage in mobile sensor networks. Int. J. Comput. Commun. Control **5**(5), 616–624 (2010)
3. Barr, S.J., Wang, J., Liu, B.: An efficient method for constructing underwater sensor barriers. J. Commun. **6**(5), 370–383 (2011)
4. Chen, A., Lai, T., Xuan, D.: Measuring and guaranteeing quality of barrier-coverage in wireless sensor networks. In: Proceedings of the 9th ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc 2008, 26–30 May 2008, Hong Kong, China, pp. 421–430 (2008). https://doi.org/10.1145/1374618.1374674
5. Dobrev, S., et al.: Weak coverage of a rectangular barrier. CoRR abs/1701.07294 (2017). http://arxiv.org/abs/1701.07294
6. Fan, H., Li, M., Sun, X., Wan, P.J., Zhao, Y.: Barrier coverage by sensors with adjustable ranges. ACM Trans. Sens. Netw. (TOSN) **11**(1), 14 (2014)
7. Gage, D.W.: Command control for many-robot systems. Technical report, DTIC Document (1992)
8. Ganesan, V., Chitre, M., Brekke, E.: Robust underwater obstacle detection and collision avoidance. Auton. Robots **40**(7), 1165–1185 (2016)
9. He, S., Gong, X., Zhang, J., Chen, J., Sun, Y.: Barrier coverage in wireless sensor networks: from lined-based to curve-based deployment. In: Proceedings of the IEEE INFOCOM 2013, 14–19 April 2013, Turin, Italy, pp. 470–474 (2013). https://doi.org/10.1109/INFCOM.2013.6566817
10. Kuhn, H.W.: The Hungarian method for the assignment problem. Naval Res. Logistics Q. **2**(1–2), 83–97 (1955)
11. Kumar, S., Lai, T., Arora, A.: Barrier coverage with wireless sensors. In: Proceedings of the 11th Annual International Conference on Mobile Computing and Networking, MOBICOM 2005, Cologne, Germany, 28 August–2 September 2005, pp. 284–298 (2005). https://doi.org/10.1145/1080829.1080859

12. Li, S., Shen, H.: Minimizing the maximum sensor movement for barrier coverage in the plane. In: 2015 IEEE Conference on Computer Communications, INFOCOM 2015, 26 April–1 May 2015, Kowloon, Hong Kong, pp. 244–252 (2015). https://doi.org/10.1109/INFOCOM.2015.7218388
13. Li, S., Shen, H.: Minimizing the maximum sensor movement for barrier coverage in the plane. In: 2015 IEEE Conference on Computer Communications (INFOCOM), pp. 244–252. IEEE (2015)
14. Liu, B., Dousse, O., Wang, J., Saipulla, A.: Strong barrier coverage of wireless sensor networks. In: Proceedings of the 9th ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc 2008, 26–30 May 2008, Hong Kong, China, pp. 411–420 (2008). https://doi.org/10.1145/1374618.1374673
15. Rafferty, J.P., et al.: Oceans and Oceanography. Britannica Educational Publishing, Chicago (2010)
16. Saipulla, A., Liu, B., Xing, G., Fu, X., Wang, J.: Barrier coverage with sensors of limited mobility. In: Proceedings of the 11th ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc 2010, 20–24 September 2010, Chicago, IL, USA, pp. 201–210 (2010). https://doi.org/10.1145/1860093.1860121
17. Silvestri, S., Goss, K.: Mobibar: an autonomous deployment algorithm for barrier coverage with mobile sensors. Ad Hoc Netw. **54**, 111–129 (2017)
18. Wang, Z., Chen, H., Cao, Q., Qi, H., Wang, Z., Wang, Q.: Achieving location error tolerant barrier coverage for wireless sensor networks. Comput. Netw. **112**, 314–328 (2017)
19. Wu, F., Gui, Y., Wang, Z., Gao, X., Chen, G.: A survey on barrier coverage with sensors. Frontiers Comput. Sci. **10**(6), 968–984 (2016)
20. Yang, G., Qiao, D.: Multi-round sensor deployment for guaranteed barrier coverage. In: 29th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies, INFOCOM 2010, 15–19 March 2010, San Diego, CA, USA, pp. 2462–2470 (2010). https://doi.org/10.1109/INFCOM.2010.5462074