Exact algorithms for barrier coverage with line-based deployed rotatable directional sensors

Zijing Ma[a], Shuangjuan Li[a*], Dong Huang[a]
[a]College of Mathematics and Informatics, South China Agricultural University, China
Email: mazijingscau@hotmail.com, lishj2013@hotmail.com, huangdong06@163.com
[*]*Corresponding author*

***Abstract*—Barrier coverage is an important coverage model for intrusion detection, which requires a chain of sensors across the deployment area with the adjacent sensors' sensing areas overlapping. Directional sensors are often dispersed from an airplane following a predetermined line. However, barrier coverage cannot be guaranteed after initial sensor deployment due to the sensors' random offsets and random orientations. Fortunately, directional sensors can rotate to mend the barrier gaps using this line-based sensor deployment model. Existing work proposed a greedy heuristic approach to mend the gaps by rotating sensors, but it cannot answer whether there exists a barrier. We fill in this gap by presenting an exact algorithm which can determine whether there exists a barrier. We first introduce the notion of feasible orientation range and then try to calculate each sensor's feasible orientation starting from the leftmost sensor. We also propose a fast algorithm of choosing the sensors' orientations from their feasible orientation ranges to form a barrier if there exists a barrier, or form a set of sub-barriers if there does not exist a barrier. Simulation results show that our algorithm outperforms the distributed algorithm in the existing work.**

***Index Terms*—wireless sensor networks, directional sensor, strong barrier coverage, rotatable sensor**

I. Introduction

Wireless Sensor Networks (WSNs) have been widely used in many fields such as intrusion detection, border surveillance, critical resource protection and battlefield surveillance. Barrier coverage is an important coverage model of intrusion detection, which aims to form a barrier consisting of sensors across the region of interest (ROI) such that any intruder passing through the ROI can be detected by at least one sensor [1]. Sensors are often dispersed from an airplane following a predetermined line in the ROI, where the offset of each sensor's actual landing location follows a normal distribution. It is important to study the barrier coverage problem of such line-based sensor deployment model([2], [4]–[6], [9]), which is more realistic than the poisson distribution model [12]. Due to the random offsets of the sensors' locations, it is difficult to guarantee barrier coverage after initial line-based sensor deployment and the barrier gaps are unavoidable.

Most of the research on barrier coverage are based on isotropic sensor whose sensing range is modeled as a circle [4], which is an ideal model. However, sensors with directional sensing range, often modeled as a sector [3], are widely used in many practical applications, such as cameras, audio sensors, infrared sensors and radar sensors. It is more practical to study how to achieve barrier coverage using directional sensors than the isotropic sensors. Due to the sensors' random offsets of locations and random orientations, the orientation of these sensors is also a key factor to form barrier coverage besides the positions of the directional sensors. There may exist some barrier gaps after initial line-based sensor deployment, which can be mended by changing the orientations of the sensors. However, it is challenging to determine the orientations of the directional sensors to mend these gaps, since the sensors can rotate 360 degrees.

Existing work in [5] proposed a distributed algorithm of rotating the sensors, called Distributed Gap Mending Algorithm, to mend the barrier gaps. However, this algorithm cannot give an exact answer to the problem of determining whether there exists a barrier. Distributed Gap Mending Algorithm used a greedy method, which determined the orientation of each sensor only using the information of its closest neighbor sensors. It always chose the orientation of each sensor such that its sensing area is the nearest to its right closest neighbor sensor. However, this chosen orientation may not be the optimal choice for deciding the orientations of other right neighbor sensors.

In this paper, we fill in this gap by proposing an exact algorithm to solve the problem of determining whether there exists a barrier. We first introduced a notion of feasible orientation range. That is, if an orientation of one sensor is not in its feasible orientation range, this sensor cannot form a barrier with other sensors with any possible orientations. Then we calculate each sensor's feasible orientation range starting from the leftmost sensor. If there exists one sensor whose feasible orientation range is null, it implies that there does not exist a barrier; otherwise, there exists a barrier and we propose an algorithm to find the orientations of all the sensors to form a barrier. Even if there does not exist a barrier, we can also find the orientations of all the sensors to form a set of sub-barriers such that the total number of gaps is minimized.

The rest of the paper is organized as follows. Section II reviews some related works. In section III we will establish the network model. Section IV proposes an algorithm of determining whether there exists a barrier. In section V we propose an algorithm of finding the sensors' orientations to form a barrier or a set of sub-barriers. In section VI we evaluate the performance of the algorithms by simulations. In section VII we conclude this paper.

978-1-7281-3106-1/20/$31.00 ©2020 IEEE

II. RELATED WORK

Barrier coverage is a hot topic in WSNs. It aims to construct a chain of sensors across the deployed area to detect any intruder crossing through it. The notion of barrier coverage was first proposed in the work [1], in which two types of barrier coverage were studied: weak barrier coverage and strong barrier coverage. Weak barrier coverage aims to detect any intruder crossing the ROI along the vertical paths, while strong barrier coverage aims to detect any intruder whose crossing paths are arbitrary. In this paper we study the strong barrier coverage, short for barrier coverage. The work in [6]–[10] studied how to achieve barrier coverage with sensors. However, most of these works assume that the sensing model of sensors are omnidirectional.

The work in [12] studied the barrier coverage problem in directional sensor network under the poisson distribution model. It defined the virtual node to reduce the solution space from continuous domain to discrete domain and then constructed a barrier graph using these virtual nodes as vertices. By finding a path in this graph, it could determine whether there exist sensors' orientations that can provide barrier coverage. However, the algorithm could not give us an exact answer but an approximate one, because the solution space is continuous other than discrete.

The work in [2] studied how to mend the barrier gaps by rotating the directional sensors for barrier coverage under the line-based deployment. It proposed algorithms for the weak barrier coverage and strong barrier coverage. The simple rotation algorithm was proposed to only rotate the sensor on either side of the gap to mend this gap. When this simple algorithm could not mend the gap, the chain reaction-based rotation algorithm was proposed to rotate the sensors one by one like a chain reaction to fix the gap. The work in [5] proposed distributed algorithms by rotating sensors to mend the gap based on line-based deployment. It determined the orientation of one sensor based on the closest left and right neighbor sensors. However, both of the algorithms in [2], [5] cannot answer whether there exists a barrier. We study the same scenario as the work in [2], [5], and try to fill in this gap by giving an exact answer to this problem.

III. NETWORK MODEL

The ROI is a rectangular belt region of length L and width H, where $L \gg H$. N directional sensors are deployed evenly on a predetermined horizontal line in ROI, where the X coordinate of sensor s_i's target location is calculated by $(2i-1)/2N$. However, due to environmental constraints, the actual locations of sensors have random offsets.

As shown in Fig. 1, a directional sensor s_i is modeled as a sector, denoted as $< (x_i, y_i), \theta, \varphi, \mathrm{R}_s >$. Here, (x_i, y_i) are the cartesian coordinates of sensor s_i, θ is the view angle, φ is the orientation angle and R_s is the sensing range. The sector of sensor s_i' view area is denoted as S_i. For any point p on the arc of S_i, we rotate $\overrightarrow{s_i p}$ around s_i in a clockwise direction until we meet the first endpoint of this arc denoted as m_i and the other endpoint denoted as n_i. The vector $\overrightarrow{s_i m_i}$ is called the starting edge of sector S_i while $\overrightarrow{s_i n_i}$ is called the end edge of S_i. Sensor s_i can rotate to change its orientation angle and can rotate from 0 to 360 degree. Assume that each directional sensor knows its coordinate (x_i, y_i) using GPS or other localization algorithms. Fig. 2 shows an initial line-based deployment of directional sensors, which has some barrier gaps, denoted by $G_1, G_2, ...G_9$.

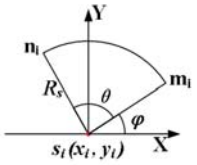


Fig. 1. Directional sensing model

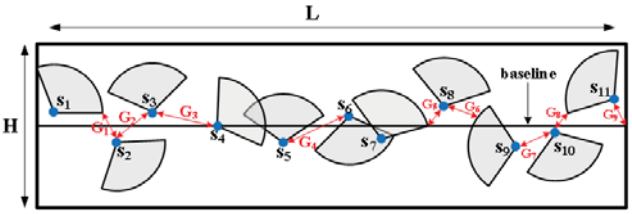


Fig. 2. Initial line-based sensor deployment with gaps $G_1, G_2, ...G_9$

We study how to rotate the directional sensors to achieve barrier coverage. We first define the problem of how to determine whether there exists a barrier as follows:

Definition 1. The decision problem of barrier coverage (DBC): Given n directional sensors deployed along a predetermined horizontal line with random offsets, the problem is to determine whether there exists an orientation angle of each sensor such that the sensing ranges of the adjacent sensors overlap with each other to form a barrier crossing from the left boundary of the region to the right boundary.

We also define the problem of how to choose the orientation angles of these sensors to form a barrier or a set of sub-barriers as follows:

Definition 2. The orientation problem of barrier coverage(OBC): Given n directional sensors deployed along a predetermined horizontal line with random offsets, the problem is to find the orientation angle of each sensor such that the sensing ranges of these sensors overlap to form a barrier crossing from the left boundary of the region to the right boundary or a set of sub-barriers such that the total numbers of gaps is minimized.

IV. MOTIVATION

In this section we analyze the Distributed Gap Mending Algorithm [5] and show that this algorithm may not form a barrier even if there exists a barrier. This algorithm chose the orientation of each sensor such that its sensing area is

the nearest to its closest right neighbor sensor, which is local optimal.

Given the orientation of sensor s_{i-1}, this algorithm always chose the orientation of sensor s_i such that the corresponding sector is the closest to the vector $\overrightarrow{s_i s_{i+1}}$. For example, as shown in Fig. 3(a), S_i rotates to overlap with S_{i-1}. If S_i rotates anti-clockwise to touch S_{i-1}, a_i is the point where S_i intersects with S_{i-1}. If S_i rotates clockwise to touch S_{i-1}, b_i is the point where S_i intersects with S_{i-1}. S_i finally rotates clockwise to touch $\overrightarrow{s_i b_i}$, since $\overrightarrow{s_i b_i}$ is closer to $\overrightarrow{s_i s_{i+1}}$ than $\overrightarrow{s_i a_i}$, as shown in Fig. 3 (b). Similarly, we rotate S_{i+1} to touch $\overrightarrow{s_{i+1} b_{i+1}}$, since $\overrightarrow{s_{i+1} b_{i+1}}$ is closer to $\overrightarrow{s_{i+1} s_{i+2}}$ than $\overrightarrow{s_{i+1} a_{i+1}}$, as shown in Fig. 3 (c). However, S_{i+2} cannot intersect with S_{i+1}, since S_{i+1} cannot intersect with the circle which S_{i+2} is located at.

In fact, as shown in Fig. 3 (d), S_{i+2} can rotate to touch S_{i+1}, if S_i rotates to touch $\overrightarrow{s_i a_i}$ other than $\overrightarrow{s_i b_i}$. Thus, Distributed Gap Mending Algorithm may not find the proper orientations of sensors to form a barrier if there exists one.

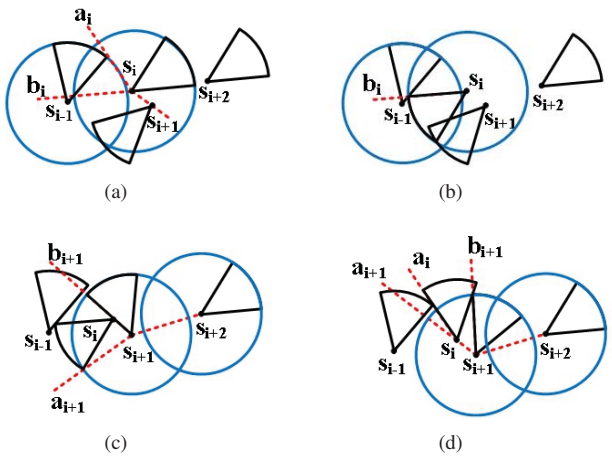


Fig. 3. A failure case of Distributed Gap Mending Algorithm [5]

V. DBC Algorithm

In this section, we will propose an algorithm for the decision problem of barrier coverage, short for DBC algorithm. To determine whether there exists a barrier, we try to calculate each sensor's feasible orientation range.

A. calculate the feasible orientation range

We first introduce some notions.

Definition 3. Virtual circle: Virtual circle, denoted as C_i, is the circle where the sector S_i is located at.

As shown in Fig. 4(a), sensor s_i's sensing area is represented by the sector bounded by black edges and its virtual circle is the circle in blue.

Definition 4. Sub-barrier: A sub-barrier is formed by a set of sensors if the sensing ranges of the adjacent sensors overlap with each other.

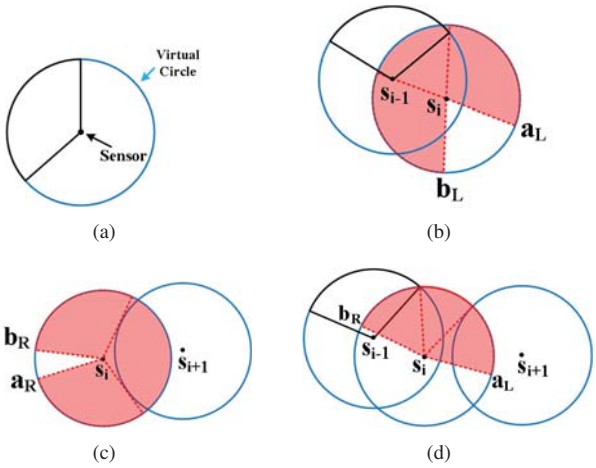


Fig. 4. **(a)** virtual circle; **(b)** feasible orientation range; **(c)** right orientation range; **(d)** updated feasible orientation range

If a sub-barrier also overlaps with the left and right boundary of the ROI, this sub-barrier is called a barrier.

Definition 5. Feasible orientation: For sensor s_i, one orientation angle is called the feasible orientation if the sensor can form a sub-barrier together with its left neighbor sensors.

It is easy to know that there may be many feasible orientations. Thus, we define the feasible orientation range as follows:

Definition 6. Feasible orientation range: For sensor s_i, its feasible orientation range, denoted as O_i, includes all possible feasible orientations.

We also define the feasible region.

Definition 7. Feasible region: For sensor s_i, its feasible region, denoted as T_i, is the union of the sector S_i whose orientation is within its feasible orientation range.

Let $\alpha\left(\overrightarrow{f}\right)$ denote the included angle of vector $\overrightarrow{f}$ and the x-axis. The feasible orientation range O_i and feasible region T_i are calculated shown in Fig. 4 (b). Suppose the feasible region T_{i-1} of sensor s_{i-1} is the sector with black edges. The sector S_i first rotates such that it does not intersect with T_{i-1}. Then rotate S_i around s_i anticlockwise until it first meets T_{i-1}, denoting its starting edge as $\overrightarrow{s_i a_L}$. Continue rotating S_i around s_i anticlockwise until it first leaves T_{i-1}, denoting its end edge as $\overrightarrow{s_i b_L}$. Thus, the feasible orientation range O_i is the range from $\alpha\left(\overrightarrow{s_i a_L}\right)$ to $\alpha\left(\overrightarrow{s_i b_L}\right) - \theta$ in the anticlockwise direction and the feasible region T_i is the region enclosed by $\overrightarrow{s_i a_L}$, $\overrightarrow{s_i b_L}$ and the arc $\overset{\frown}{a_L b_L}$(i.e. the shadowed sector in Fig. 4(b)).

Given the feasible orientation range of sensor s_{i-1}, we calculate the feasible orientation range of s_i as follows:

1) If s_i is the first sensor, the feasible orientation range of s_i is calculated according to the intersection of circle C_i and the left boundary of region. The two intersection points are

denoted as p_i and p_j respectively (if C_i and the left boundary are tangent, p_i and p_j are the same point). We rotate sector S_i to intersect with the left boundary only at point p_i or p_j, denoting its orientation as A_i or A_j respectively. Thus, the feasible orientation range of s_i is the range from A_i to A_j .

2) If s_i is not the first sensor, then four cases are considered according to the locations of s_i and s_{i-1}.

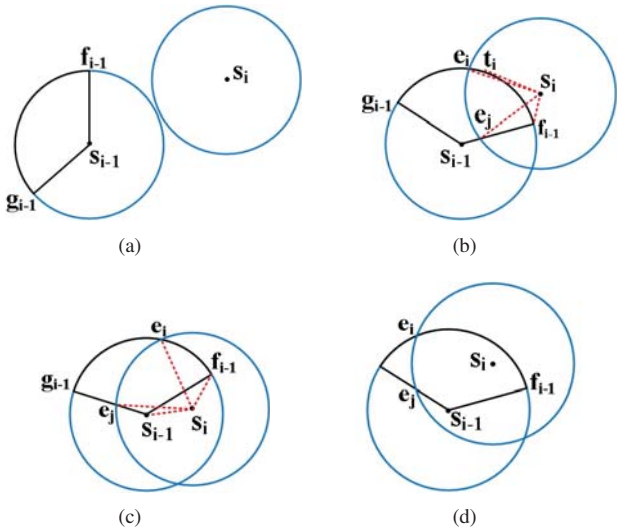


Fig. 5. Examples of calculating feasible orientation range

Case 1: C_i does not intersect with C_{i-1}. It implies that S_i cannot overlap with S_{i-1}, as shown in Fig. 5(a). Thus, there is a gap between these two sectors and the feasible orientation range of s_i is set to be [0,2π].

Case 2: C_i intersects with C_{i-1} and s_i is located outside C_{i-1}, as shown in Fig. 5(b). We define a vector set D. First draw two tangent lines from s_i to C_{i-1} with tangent points denoted as t_i and t_j. If t_i or t_j is in C_i and also in the sector T_{i-1}, then $\overrightarrow{s_it_i}$ or $\overrightarrow{s_it_j}$ will be added to D. The vectors $\overrightarrow{s_ie_i}$ and $\overrightarrow{s_ie_j}$ will be also added to D, where e_i and e_j are the intersection points of sector T_{i-1} and C_i. Suppose f_{i-1} and g_{i-1} are the end points of the arc of the sector T_{i-1}. If f_{i-1} or g_{i-1} is inside C_i, then $\overrightarrow{s_if_{i-1}}$ or $\overrightarrow{s_ig_{i-1}}$ will be added to D. We choose the vector in D with the smallest angle, denoted as v_i. We also choose one with the largest angle, denoted as v_i'. Thus, the feasible orientation range of s_i is the range from $\alpha\left(\overrightarrow{v_i}\right)-\theta$ to $\alpha\left(\overrightarrow{v_i'}\right)$. As shown in Fig. 5(b), the feasible orientation range of s_i is the range from $\alpha\left(\overrightarrow{s_it_i}-\theta\right)$ to $\alpha\left(\overrightarrow{s_if_{i-1}}\right)$.

Case 3: C_i intersects with C_{i-1} and s_i is inside C_{i-1} but outside T_{i-1}, as shown in Fig. 5(c). We also define a vector set D. If T_{i-1} and C_i do not intersect, S_i cannot overlap with S_{i-1}, which implies that there is a gap between them and the feasible orientation range of s_i is set to be [0,2π]; otherwise, $\overrightarrow{s_ie_i}$ or $\overrightarrow{s_ie_j}$ will be added to D, where e_i and e_j are the intersection points of sensor T_{i-1} and C_i. If f_{i-1} or g_{i-1} is inside C_i, then $\overrightarrow{s_if_{i-1}}$ or $\overrightarrow{s_ig_{j-1}}$ will be added to D. Similar as Case 2, we choose the vector in D with the smallest and largest angle, denoted as v_i and v_i'. Thus, the feasible orientation range of s_i is the range from $\alpha\left(\overrightarrow{v_i}\right)-\theta$ to $\alpha\left(\overrightarrow{v_i'}\right)$. As shown in Fig. 5(c), the feasible orientation range is the range from $\alpha\left(\overrightarrow{s_if_{i-1}}-\theta\right)$ to $\alpha\left(\overrightarrow{s_is_{i-1}}\right)$.

Case 4: C_i intersects with C_{i-1} and s_i is inside C_{i-1} as well as T_{i-1}, as shown in Fig. 5(d). Since s_i is located inside T_{i-1}, S_i intersects with T_{i-1} in any orientation, thus the feasible orientation range of s_i is [0,2π].

It is easy to know that the feasible orientation range of s_i is also related with the location of its closest right neighbor sensor. Thus, we introduce the notion of the right orientation range and then update the feasible orientation range using the right orientation range.

B. calculate the right orientation range

We first define the right orientation range.

Definition 8. Right orientation range**:** For sensor s_i**,** its right orientation range, denoted as R_i, includes all the orientations which satisfy that S_i can overlap with its closest right neighbor sensor or the right boundary of ROI.

The sector S_i first rotates such that it does not intersect with C_{i+1}. Then rotate S_i around s_i anticlockwise until it first meets C_{i+1}, denoting its starting edge as $\overrightarrow{s_ia_R}$. Continue rotating S_i around s_i anticlockwise until it first leaves C_{i+1}, denoting its end edge as $\overrightarrow{s_ib_R}$. Thus, the right orientation range O_i is the range from $\alpha\left(\overrightarrow{s_ia_R}\right)$ to $\alpha\left(\overrightarrow{s_ib_R}\right)-\theta$ in the anticlockwise direction, as shown in Fig. 4(c). Then, update the feasible orientation range O_i as the range from $\alpha\left(\overrightarrow{s_ia_L}\right)$ to $\alpha\left(\overrightarrow{s_ib_R}\right)-\theta$ and the feasible region T_i is the region enclosed by $\overrightarrow{s_ia_L}$, $\overrightarrow{s_ib_R}$ and the arc $\widehat{a_Lb_R}$(i.e. the shadowed sector in Fig. 4(d)).

In this subsection we will calculate the right orientation range of s_i.

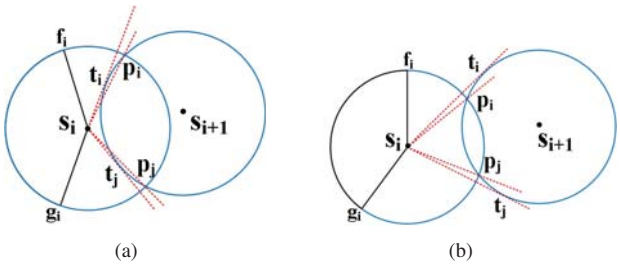


Fig. 6. Examples of calculating right orientation range

If the feasible orientation range of s_{i+1} has not been calculated, three cases will be considered.

Case 1: C_i does not intersects with C_{i+1}. It implies that S_i cannot overlap with S_{i+1}. Thus, there is a gap between them.

Case 2: C_i intersects with C_{i+1} but s_i is outside C_{i+1}, as shown in Fig. 6. We draw two tangent lines from s_i to C_{i+1} with the tangent points denoted as t_i and t_j. Let p_i and p_j denote the intersections of C_i and C_{i+1}. If two tangent points t_i and t_j are inside C_i, we rotate S_i to intersect with C_{i+1} only at point t_i or t_j, denoting the orientation of sensor s_i as

A_i or A_j respectively; otherwise, we calculate A_i and A_j by rotating S_i to intersect with C_{i+1} only at p_i or p_j. Then the right orientation range is the range from A_i to A_j. In Fig. 6(a), the right orientation range is $\alpha\left(\overrightarrow{s_it_i}-\theta\right)$ to $\alpha\left(\overrightarrow{s_it_j}\right)$. In Fig. 6(b), the right orientation range is $\alpha\left(\overrightarrow{s_ip_i}-\theta\right)$ to $\alpha\left(\overrightarrow{s_ip_j}\right)$.

Case 3: C_i intersects with C_{i+1} and s_i is inside C_{i+1}. It implies that whatever orientation S_i rotates, S_{i+1} can rotate to intersect with it, so the right orientation range of s_i is [0,2π].

After the right orientation range is calculated, update the feasible orientation range of sensor s_i by the intersection of it and right orientation range. Besides, we will update the feasible orientation ranges of all the sensors on its left side belonging to the same sub-barrier. The process of calculation is similar as the process of calculating the feasible orientation range in the above subsection.

C. Algorithm Description

In this subsection, we propose the algorithm of determining whether there exists a barrier.

The basic idea of this algorithm is to calculate the feasible and right orientation range of each sensor from the left to the right. If none of the intersection of these two ranges is null, it implies that there exists a global barrier; Otherwise, there does not exist one and we will calculate the feasible orientation ranges of the sensors to form a set of sub-barriers.

The procedures of this algorithm are described as follows:

1) For each sensor s_i, calculate its feasible and right orientation range using the methods in the above two subsection.

2) If the intersection of these two ranges is null, then it means that there does not exist a global barrier and the current sub-barrier ends with sensor s_i. A new sub-barrier starts with sensor s_{i+1}, go to 1); otherwise, go to 3).

3) For each sensor s_j $(j < i)$ back to forth, which belongs to the current sub-barrier, recalculating its right orientation range. If the intersection of these two ranges is null, then it means that there does not exist a global barrier and the current sub-barrier ends with sensor s_i. A new sub-barrier starts with sensor s_{i+1}, and recalculate its feasible and right orientation range, go to 2); If none of the intersection of all these sensors is null, update their feasible orientation ranges by the intersections and go to 1).

Note that if s_i is the first or the last sensor, its feasible or right orientation range should be calculated based on the locations of left or right boundary of ROI. If a new sub-barrier starts with sensor s_i, then its feasible orientation range is initially set to be [0, 2π].

The detail of this algorithm is shown in Algorithm 1. Let cur denote the first sensor's index of the current sub-barrier. Let isSuccess denote the variable indicating whether there exists a global barrier.

Algorithm 1 Algorithm of DBC

Input: Sensor Set S
Output: Feasible Orientation Range Set O=$\{O_i|1 \le i \le n\}$, boolean isSuccess

```
1: cur = 0, isSuccess = true
2: for each sensor s_i ∈ S
3:    calculate feasible and right orientation range of S_i as O_i and R_i
4:    if O_i ∩ R_i is null then
5:       isSuccess = false
6:       cur = i + 1
7:       continue
8:    else
9:       O'_i = O_i
10:      O_i = O_i ∩ R_i
11:   end if
12:   if i > 1 then
13:      position = i − 1
14:      while position ≥ cur do
15:         calculate right orientation range of S_position as R'_position
16:         O'_position = O_position ∩ R'_position
17:         if O'_position is null then
18:            O_i = O'_i
19:            cur = i + 1
20:            isSuccess = false
21:            break
22:         end if
23:         position = position − 1
24:      end while
25:   end if
26:   if position < cur then
27:      position = i − 1
28:      while position ≥ cur do
29:         O_position = O'_position
30:         position = position − 1
31:      end while
32:   end if
33: end for
34: return isSuccess and O
```

VI. Algorithm of OBC

In this section, we propose an algorithm of finding the orientations of all the sensors to form a barrier or form a set of sub-barriers, called algorithm of OBC.

The basic idea of this algorithm is to choose an orientation of each sensor from its feasible orientation range such that its sensing area overlaps with that of its closest left neighbor sensor. If there is no such orientation, it implies that there is a gap between this sensor and its left closest neighbor sensor and a new sub-barrier starts from this sensor. Rotate this sensor to the boundary of this range; otherwise, rotate this sensor to intersect with its closest left neighbor sensor by choosing one orientation from its feasible orientation range. Note that the feasible orientation range of one sensor may include more than one ranges.

Algorithm 2 shows the detail of OBC algorithm.

Fig.7 presents a part of final deployment of sensors using our proposed algorithm and the Distributed Gap Mending

Algorithm 2 OBC Algorithm

Input: Sensor Set S

Output: The orientations of Sensors S

```
Calculate feasible orientation range set O by Algorithm 1
for each sensor s_i ∈ S
    if i = 1
        rotate S_i to one boundary of s_i's feasible orientation range
    else
        isRotated = false
        for each range of feasible orientation range o ∈ O_i
            if there is an orientation in o such that S_i can intersect with S_{i+1} then
                rotate S_i to this orientation
                isRotated = true
            end if
        end for
        if isRotated = false
            rotate S_i to one boundary of s_i's feasible orientation range
        end if
    end if
end for
return S
```

Algorithm [5]. Sensors are deployed with view angle θ = 60. Fig. 7(a) shows a barrier formed using our algorithm. Fig. 7(b) shows the orientations of sensors computed in the work [5] under the same initial sensor deployment, which results in a barrier gap.

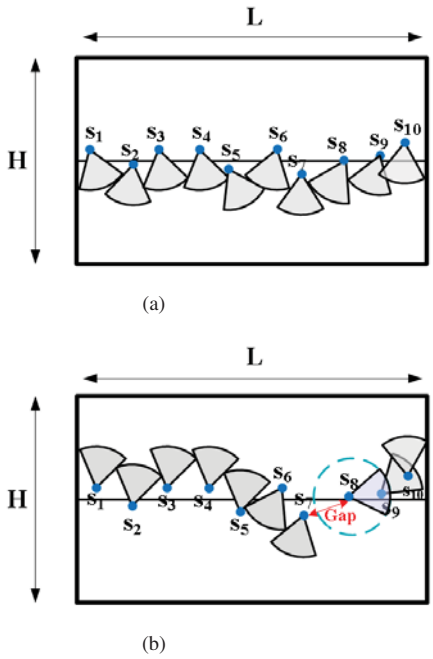


Fig. 7. **(a)** The final sensor rotation scheme by our algorithm; **(b)** The final sensor rotation scheme by the Distributed Gap Mending Algorithm [5]

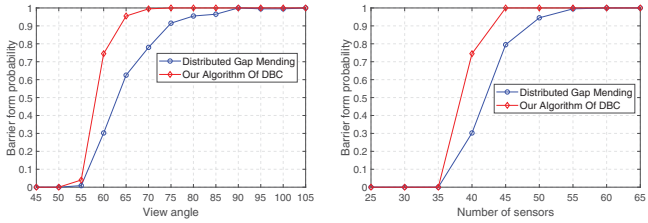


Fig. 8. **(a)**Probability of forming a strong barrier vs view angle of sensors **(b)**Probability of forming a strong barrier vs number of sensors

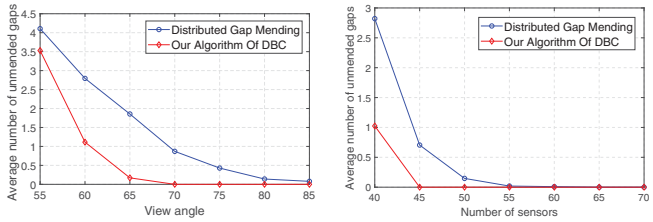
VII. Simulation Results

In this section, we evaluate our proposed algorithm using Java by compared to the Distributed Gap Mending Algorithm [5]. We assume the ROI is a rectangle belt region with length L = 500m and H = 100m. Sensors are deployed randomly along the middle-line of this region. The initial orientation angle of the sensors is randomly chosen from $[0, 2\pi]$. The sensing range of sensor is 15m. In each experiment, we evaluate the probability of forming a barrier and unmended gaps of forming a strong barrier. The result is an average result of 100 experiments.

First, we evaluate the probability of forming a barrier when the view angle changes. The number of sensors is 40. As shown in Fig. 8(a), the probability of forming barrier increases as the view angle of sensors increases. Our proposed algorithm has a higher successful ratio than the distributed algorithm. Moreover, our algorithm can form a barrier when the view angle is not smaller than 70 while the distributed algorithm can form a barrier when the view angle must be larger than 90. Since sensors with larger view angle consume more energy to sense, the barrier formed by our algorithm with smaller view angle can have a long lifetime than that by the distributed algorithm.

Then, we investigate the probability of forming barrier when the number of sensors changes. The view angle of sensors is 60. The number of sensors varies from 25 to 65 with a step 5. Fig. 8(b) shows that the probability of forming barrier increases as the number of sensors increases. Our proposed algorithm always has a higher successful ratio than the distributed algorithm. Moreover, our algorithm can form a barrier only using 45 sensors while the distributed algorithm must use 55 sensors. It implies that under the same conditions, our algorithm needs fewer directional sensors, which can decrease the cost of directional sensors.

Furthermore, we evaluate the unmended barrier gaps when the view angle and number of sensors changes respectively. In Fig. 9(a), the number of sensors is 40. It demonstrates that our algorithm of DBC always results in fewer unmended gaps than the distributed algorithm as the view angle of sensors increases. Moreover, the result obtained by our algorithm decreases more quickly than that by distributed algorithm. Fig.9(b), the view angle of sensors is 60. The result shows



(a) (b)

Fig. 9. **(a)**Unmended gaps of forming a strong barrier vs view angle of sensors **(b)**Unmended gaps of forming a strong barrier vs number of sensors

that our proposed algorithm has a better performance than the distributed algorithm as the number of sensors increases. Our algorithm results in only one gap while the distributed algorithm results in 2.8 gaps when the number of sensors is 40. Thus, our algorithm can leave fewer gaps than the distributed algorithm when the number of sensors is not sufficient.

VIII. Conclusion

We present an exact algorithm of determining whether there exists a strong barrier by rotating the directional sensors with the line-based directional sensors. Then we also propose an algorithm of determining the sensors' orientations for forming a strong barrier or a set of sub-barriers if the number of sensors is not sufficient. Simulation results indicate that our algorithm outperforms the distributed algorithm.

IX. Acknowledgment

This work is supported by National Natural Science Foundation of China (Grant No. 61702198 and 61976097).

References

[1] Santosh Kumar, Ten H Lai, and Anish Arora, "Barrier coverage with wireless sensors", in Proceedings of the 11th annual international conference on Mobile computing and networking (2005), pp. 284--298.

[2] Jiaoyan Chen, Bang Wang, Wenyu Liu, Laurence T Yang, and Xianjun Deng, "Rotating directional sensors to mend barrier gaps in a line-based deployed directional sensor network", IEEE Systems Journal 11, 2 (2014), pp. 1027--1038.

[3] Zhibo Wang, Jilong Liao, Qing Cao, Hairong Qi, and Zhi Wang, "Achieving k-barrier coverage in hybrid directional sensor networks", IEEE Transactions on Mobile Computing 13, 7 (2013), pp. 1443--1455.

[4] Anwar Saipulla, Cedric Westphal, Benyuan Liu, and Jie Wang, "Barrier coverage of line-based deployed wireless sensor networks", in IEEE INFOCOM 2009 (2009), pp. 127--135.

[5] Yueshi Wu and Mihaela Cardei, "Distributed algorithms for barrier coverage via sensor rotation in wireless sensor networks", Journal of Combinatorial Optimization 36, 1 (2018), pp. 230--251.

[6] Anwar Saipulla, Benyuan Liu, and Jie Wang, "Barrier coverage with airdropped wireless sensors", in MILCOM 2008-2008 IEEE Military Communications Conference (2008), pp. 1--7.

[7] Jie Shen, Zhibo Wang, and Zhi Wang, "Fault tolerant line-based barrier coverage formation in mobile wireless sensor networks", International Journal of Distributed Sensor Networks 11, 10 (2015), pp. 930585.

[8] Zhao Zhang, Weili Wu, Jing Yuan, and Ding-Zhu Du, "Breach-Free Sleep-Wakeup Scheduling for Barrier Coverage With Heterogeneous Wireless Sensors", IEEE/ACM Transactions on Networking 26, 5 (2018), pp. 2404--2413.

[9] Jiaoyan Chen, Bang Wang, Wenyu Liu, Xianjun Deng, and Laurence T Yang, "Mend barrier gaps via sensor rotation for a line-based deployed directional sensor network", in 2013 IEEE 10th International Conference on High Performance Computing and Communications & 2013 IEEE International Conference … (2013), pp. 2074--2079.

[10] Shuangjuan Li and Hong Shen, "Minimizing the maximum sensor movement for barrier coverage in the plane", in 2015 IEEE Conference on Computer Communications (INFOCOM) (2015), pp. 244--252.

[11] Lu Zhao, Guangwei Bai, Hang Shen, and Zhenmin Tang, "Strong barrier coverage of directional sensor networks with mobile sensors", International Journal of Distributed Sensor Networks 14, 2 (2018), pp. 1550147718761582.

[12] Dan Tao, Shaojie Tang, Haitao Zhang, Xufei Mao, and Huadong Ma, "Strong barrier coverage in directional sensor networks", Computer Communications 35, 8 (2012), pp. 895--905.