

# Distributed Algorithm for Full-View Barrier Coverage with Rotatable Camera Sensors

Rui Yang, Xiaofeng Gao\*, Fan Wu and Guihai Chen

Shanghai Key Laboratory of Scalable Computing and Systems,

Department of Computer Science and Engineering, Shanghai Jiao Tong University

Email: yang.rui.shirley@gmail.com, gao-xf, fwu, gcheng@cs.sjtu.edu.cn

**Abstract**—In wireless sensor networks, barrier coverage is a fundamental category of coverage problems and its surveillance capability can be reinforced when utilizing camera sensors. In this paper, we propose a distributed algorithm to solve the problem of Full-View Barrier Coverage with rotatable camera sensors (FBR). Our goal is to minimize the active sensor number when guaranteeing the surveillance capabilities. Correspondingly, we propose a Distributed Proliferation Algorithm (DPA) which is the first distributed algorithm to deal with such problem. In DPA, each sensor has five possible phases: Initial Phase, Update Phase, Expand Phase, Wait Phase, and Terminate Phase. The main idea of DPA is spreading the sensor direction along the barrier with the help of auxiliary conflicting graph. We design a greedy strategy based on DFS. Moreover, local Dijkstra algorithm is utilized to select the shortest path to reduce active sensor number from global view. The mass number of numerical experiments validate the efficiency of DPA which can construct a full-view barrier line with fewer sensors compared with previous work.

**Index Terms**—full-view barrier coverage, rotatable camera sensors, distributed algorithm

## I. INTRODUCTION

With the expeditious development of technology, camera sensor has been extensively used and achieved better effects than many other kinds of sensors in resource protection, disaster monitoring, military surveillance, and other critical fields. This aroused in-depth researches in this field which actuate the generation of the concept: Full-View Coverage (FVC). An object is full-view covered if there is always a camera sensor facing close to it no matter which direction it faces [1] (rigorous definition is given in Section III).

Further, researchers observe that FVC costs much more than traditional coverage models, and there are some cases that people merely want to know whether there is an intruder invading across the region. We named such coverage as barrier coverage and such deployment as barrier. A barrier is formed by a set of sensors whose sensing regions are contiguous and span across the monitored field [2], guaranteeing that every object attempting to go through the monitored region will be detected straightaway with high accuracy and little artificial intervention. Based on the above definitions, researchers raise

This work has been supported in part by China NSF grant (61202024, 61472252, 61133006, 61422208), China 973 project (2014CB340303), the Natural Science Foundation of Shanghai (Grant No.12ZR1445000), Shanghai Educational Development Foundation (Chenguang Grant No.12CG09), Shanghai Pujiang Program 13PJ1403900, and in part by Jiangsu Future Network Research Project No. BY2013095-1-10 and CCF-Tencent Open Fund.

X. Gao is the corresponding author.

the model: Full-View Barrier Coverage (FVBC) where each node in barrier is full-view covered.

FVBC problem bears similarity to the barrier problem with directional antenna, while it possesses such intricacy that traditional solutions like [3]–[5] cannot be applied. Thus some researchers study on this problem. Wang et al. first defined the concept FVC. Hereafter, the algorithms designed for solving FVBC came out, aiming to reducing sensor number [7] and solving Minimum Camera Barrier Coverage Problem (MCBCP) [8] respectively. All such solutions used static cameras which fairly cost much and wasteful. Hence, Gui considered a centralized algorithm [9] using rotatable camera sensors to diminish sensors.

However, all the algorithms mentioned above are centralized, but camera sensors are actually ad-hoc. Therefore, applying distributed algorithm can embody the characteristic of camera sensor network more practically. Correspondingly, we propose a Distributed Proliferation Algorithm (DPA) which is the first distributed algorithm to deal with such problem. In DPA, each sensor has five possible phases: Initial Phase, Update Phase, Expand Phase, Wait Phase, and Terminate Phase. The main idea of DPA is to spreading sensor direction across the barrier with the help of auxiliary conflicting graph. We use greedy strategy with the help of DFS. Moreover, local Dijkstra algorithm is utilized to select the shortest path which would reduce active sensor number from global view. Mass number of numerical experiments validate the efficiency of DPA, which can construct a full-view barrier line with fewer sensors. Our main contributions are:

- We propose a novel efficient distributed algorithm for full-view barrier coverage problem with rotatable camera sensors (FBR), including the procedure of sensor selection and camera rotation.
- We provide a complexity analysis for DPA and show that the time complexity is  $O(n^2)$  even when the monitored strip is pretty long. We also show that the message complexity of DPA is  $O(m)$ , where  $m$  is the number of a sensor's neighbours.
- We compare DPA with algorithms in [7]–[9] to validate the efficiency of DPA and prove the correctness of it.

The rest of the paper is organized as follows. Section II introduces related researches in past years. Section III introduces models, assumptions and problem definition. Section IV il-

illustrates how Distributed Proliferation Algorithm (DPA) work and validate. Section V evaluates the performance of DPA and Section VI summarizes the paper and states the future work.

## II. RELATED WORKS

Barrier coverage is a sustained hot topic and there are some comprehensive surveys conducted on directional sensor coverage [10]–[12]. Here we review the most relevant work.

Traditional barrier coverage was first introduced by K.-P. Shih [2] and has been studied in depth, while rotatable directional coverage was firstly utilized for object coverage [13] and demonstrated its availability in reducing sensor number. The barrier coverage with directional sensor was first introduced in [6]. It showed a heuristic centralized algorithm named CoBRA to find the barrier line with homogeneous rotatable camera sensors. In fact, the authors in [6] considered camera sensor the same with other directional sensors, omitting its unique visual characteristics.

Compared with traditional directional sensors, camera sensors can collect different views of the same object from different view directions. Considering this property, Wang et al. introduced a novel model called full-view coverage into area coverage and proposed an efficient solution for FVC detection problem [1]. With the same model, they further studied on the problem of constructing a camera barrier as a connected zone across the monitored field such that every point within the zone is full-view covered [7]. Fig. 1 shows a typical FVB line. Later, Huan Ma introduced in [8] Minimum Camera Barrier Coverage Problem (MCBCP) and proposed a feasible solution to minimize the sensor number. Based on previous researches, Wang and Cao extended existing barrier coverage models into camera sensor networks, introduced weak/strong barrier coverage verification problems and proposed new detection methods [14].

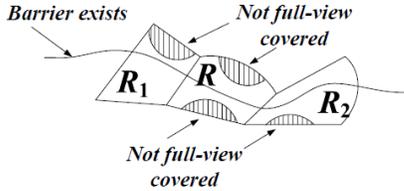


Fig. 1. A valid full-view barrier shown in [7].

The most up to date research is [9] in which Gui et al. proposed a greedy algorithm to partition the monitored area and choose feasible barrier line. However, all the above mentioned literature proposed centralized algorithms to solve camera sensor coverage problem, while the camera sensor network is indeed ad-hoc. Thus considering a distributed algorithm for FVBC problem is pretty practical and meaningful.

## III. MODELS AND PRELIMINARY

This section specifies the system model including sensing model of camera sensors and network model. Some primary notations, definitions and terminologies are also introduced.

### A. Sensing Model

**Definition 1** (Ordinary Camera Sensor (OCS)). *Ordinary camera sensors (OCS) are sensors actually deployed in field of interest (FoI). Each OCS is defined by sensing radius  $r_s$ , communication radius  $r_c$ , orientation vector  $\vec{f}_i$  and offset angle  $\varphi$  which shows the field-of-view (FoV) on both side of  $\vec{f}_i$ .  $s_i$  represents the  $i^{\text{th}}$  sensor. A point  $p$  is said to be covered by  $s_i$  if and only if  $\|s_i p\| \leq r$  and  $\vec{s}_i \vec{p} \cdot \vec{f}_i \geq \|s_i p\| \cos \varphi$  where  $\|s_i p\|$  denotes the distance from  $s_i$  to  $p$ .*

**Definition 2** (Edge Sensors and Boundary Sensors). *Edge Sensors  $s_e$  are the sensors located on the vertical boundaries. A Boundary Sensor  $s_b$  is the sensor located near a boundary  $b$  and the distance between  $s_b$  and  $b$  is less than  $r_s$ .*

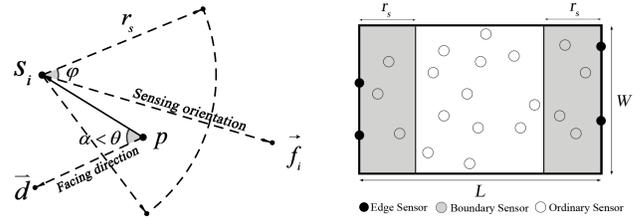


Fig. 2. Directional Sensing Model Fig. 3. Coverage & Network Model

The diagrammatic sketch of sensing and coverage model are picturesquely shown in Fig. 2 and Fig. 3.

### B. Coverage Model

**Definition 3** (Full-View Coverage). *A point  $p$  is said to be full-viewed covered if for any direction  $\vec{d}$  of  $p$ , there is a sensor  $s_i$  that  $p$  is covered by  $s_i$  and the angle between  $\vec{d}$  and  $\vec{ps}_i$  is less than  $\theta$ , where  $\theta \in [0, \frac{\pi}{2}]$  is a predefined parameter, named Maximum Difference Angle (MDA).*

**Definition 4** (Full-View Barrier). *In a rectangle FoI  $A$ , a full-view barrier (FVB) is a connected region inside  $A$  such that every point in FVB is full-view covered and each path from the lower border to upper border intersects with FVB.*

### C. Assumptions and Problem Statement

We assumed that the monitored area is a strip rectangle where all cameras are initially randomly deployed. Moreover, the sensor density is assumed to be high enough to construct a FVB and each sensor has a unique ID. We consider that all the cameras are homogeneous and know their own location coordinates. For each camera, its communication radius  $r_c$  is twice as much as its sensing radius  $r_s$ , which means cameras can know which sensor stands within a  $r_c$  disk, but can only detect enemies  $r_s$  away.

Now we can define our problem: in this paper, we aim to rotate the angle of the sensing direction of camera sensors to form a FVB using as less sensors as possible. By doing this, we can clearly monitor the region and clearly get the face of the enemy no matter how it invades FoI. We denote the problem as FBR.

#### IV. DISTRIBUTED PROLIFERATION ALGORITHM

In this section, we propose an efficient structure for each OCS to store information, detect whether it can construct FVB, and decide its rotation angle. In our algorithm, we transform the initial problem into field selection problem, and then transform it back into sensor selection problem with rotation information. Here is also the challenging part of this topic.

##### A. Algorithm Overview

In our distributed algorithm, sensors except for edge sensors mainly includes five phases, **Initial Phase**, **Update Phase**, **Expand Phase**, **Wait Phase** and **Terminate Phase**.

**Definition 5** (Backbone Sensor, Leaf Sensor, and Exterior Sensor). *The OCS in DPA can be classified into three categories: **Backbone Sensor (BS)** is the sensor who receives Expand Request for at least one time. It is obliged to choose local barrier line and can influence the rotation of its neighbors. The sensor which is influenced by Backbone Sensor and has to rotate involuntarily is **Leaf Sensor (LS)**. Remaining sensors which receive no operation order are **Exterior Sensors (ES)**.*

In all, DPA utilizes the core ideology of greedy algorithm and DFS (Depth-First Search) as the outer frame. DPA begins with an Edge Sensor which sends Expand Request to its nearest sensor  $s_{b1}$ .  $s_{b1}$  is the 1<sup>st</sup> BS and will analyze his sensing disk, find possible FVBs, chooses one FVB and  $s_{b2}$  as the next BS and his son, sends Turn Request to the sensors which is needed to form the local FVB, and sends Expand Request to  $s_{b2}$ .  $s_{b2}$  will repeat the process and this procedure will be iterated until Extend Request reaches an edge sensor  $s_f$  on the opposite edge. Eventually,  $s_f$  sends Finish Request back to its father, where finish information will be sent stepwise, back along the BS network, guaranteeing each sensor rotating to the ordered direction. The phase status of each sensor is always broadcast to its neighbors. Additionally, if a BS fails to construct a local FVB, it will send error information back to his father, asking his father to choose another FVB with different exit or another son as alternative.

##### B. Phase Introduction

We need to know some definitions first.

**Definition 6** (Neighbor). *If  $s_i$  and  $s_j$  satisfy  $\|s_i s_j\| < r_c = 2r_s$ , we call that  $s_i$  and  $s_j$  are neighbours.*

There still exists some parameters we ought to define: *triedpath* represents the number of paths that have already been tried to expand the barrier; *FC* records the union of possible full-view covered region in one sensing disk. Moreover, we treat OCS and boundary sensors equally except that we omit the region that exceeds the monitored region.

1) *Initial Phase*: Arbitrarily choose an Edge Sensor as start point. Each sensor broadcasts its ID and location, and catches its neighbors' information. Afterwards, all the sensors except for start point switch to Update Phase. The information exchange is done in a limited but abundant time. After this limited time, the sensors with no receiving data are forced to

close and do not participate in following steps. After information exchange, start point will send out Expand Request to the nearest sensor and jump to Wait Phase.

2) *Update Phase*: In this phase, sensors receive the neighbors' status and update the information stored in themselves. Sensors in this phase will only respond to Expand Request or Turn Request. During this phase,

- if  $s_i$  receives Expand Request, it will further receive information packet and turn into Expand Phase;
- if  $s_i$  receives Turn Request, it will rotate itself and switch to Wait Phase.

The content of information packet is enumerated in Table I. Obviously, the *ExPos* of father sensor is the entrance position

TABLE I  
CONTENT OF INFORMATION PACKET

Variable name	Meaning
<i>ExPos</i>	The location range of exit arc
<i>nsnum</i>	Current needed sensor number
<i>bbid</i>	ID of current sensor in BS network

of his son sensor. Similarly, if sensors in Update Phase receives no request from other sensors for a certain long time, it will also be closed.

3) *Expand Phase*: The sensor in Expand Phase is considered to be the decision sensor for analyzing local information and expanding the barrier line. It will block other request until the expand algorithm is accomplished. Here we define an index to evaluate the quality of different lines:

$$q = \frac{|L_s L_f|}{\# \text{ of sensors needed to implement FVB line}}$$

where  $L_s$  is the central location of entrance arc and  $L_f$  is that of exit arc within the sensing circle. All the parameters in the above formula is for one sensor. Algorithm1 is shown below.

After executing Algorithm1, the sensor will send Information Packet to the next BS, record the needed direction of himself and switch to Wait Phase.

4) *Wait Phase*: Stayed in this phase:

- if  $s_i$  receives Error Feedback, it broadcast Release Request to his sons and transits to Expand Phase;
- if  $s_i$  receives Finish Request, it turns to Terminate Phase;
- if  $s_i$  receives Turn Request from his father sensor, it does not rotate himself but broadcasts the ordered direction and stays in Wait Phase;
- if  $s_i$  receives Release Request from his father, it switches to Update Phase.

5) *Terminate Phase*: When  $s_i$  is in Terminate Phase,

- it rotates itself to the calculated direction;
- if it is a Backbone Sensor, it sends Finish Request to its corresponding Leaf Sensors;
- if it is an Edge Sensor on start edge, DPA runs to the end and we can finish the algorithm;
- it sends Finish Request to its father BS.

---

**Algorithm 1** Expand Phase (run at camera sensor)

---

- 1: Receiving Information Packet or Error Feedback
  - 2: **if** the sensor receives information packet **then**
  - 3:   Record  $bbid$  as father sensor;
  - 4:    $FC$  = the return value of Algorithm 2;
  - 5:   Further partition  $FC$  and use Dijkstra Algorithm to detect FVB in  $FC$  and store the information in  $\mathbb{PL}$  (detailed illustration is in SectionIV-C4);
  - 6:   **for** each line  $l$  in  $\mathbb{PL}$  **do**
  - 7:     **if**  $l_i$ 's exit arc is the same with previous ones **then**
  - 8:       Preserve the line with a smaller  $nsnum$ ;
  - 9:     **end if**
  - 10:    Calculate  $q_i$  for  $l_i$ ;
  - 11:    **end for**
  - 12:    Sort the  $q_i$  and store corresponding barrier ID into  $Q$ ;
  - 13:     $triedpath = 0$ ;
  - 14:    **end if**
  - 15:    Choose  $Q[triedpath]$  as alternative line  $l_j$ ;
  - 16:    Send Turn Request for the Leaf Sensors related to  $l_j$ ;
  - 17:    Choose the sensor within the  $r_c$  and nearest from exit arc as  $next$  Backcone Sensor;
  - 18:    Update *Information Packet* and send Request;
- 

When the Edge Sensor on Start Edge receives Terminate Phase, the deployment procedure can terminate successfully.

### C. Single Sensor Full-View Barrier Detection

This part introduces the algorithms that will be applied in Expand Phase, which is designed for single Backbone Sensor.

1) *Sub-region partition*: This part illustrates how we primitively partition the possible monitored area of each sensor.

**Lemma 1** (Partial Full-View Coverage Detection). *For ordinary camera sensors  $s_i$ , every point  $p$  that  $\|s_i p\| < r_s$  can be judged whether it can be full-view covered.*

*Proof*: For every point  $p$ , if it can be full-view covered, then the sensors that cover it are up to  $2 \times r_s = r_c$  apart one another and when the two sensors are  $r_c$  apart,  $p$  must be at the right middle. At the same time, for every OCS  $s_i$ , it can detect each sensor within  $r_c$  circle, which means  $s_i$  grasp the locations of all sensors that may cover the point within its sensing range. ■

From Lemma 1, we can ensure that every point in the monitored area can be judged whether it is full-view covered as long as the sensor density reached the minimum standard. Moreover, all the possible facing direction of a camera can form a disk area with radius  $r_s$ . We named such disk as sensing disk and denoted it as  $R_i$ . The neighbor sensor can partition  $R_i$  into sub-regions. Then we can begin to judge whether each sub-region has the probability to be full-view covered.

2) *Full-View Coverage Judgement*: The sensing disk of each sensor can be divided into sub-regions as introduced in Section IV-C1. Then, we define a coverage list  $\mathbb{CL}_p$  (example is shown in Fig. 4) for each point  $p$  in  $R_i$ . We construct the list as follow: we initialize the list as an empty set and add  $\overrightarrow{ps_i}$

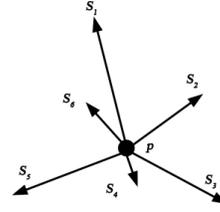


Fig. 4.  $\mathbb{CL}_p$  of Single Point

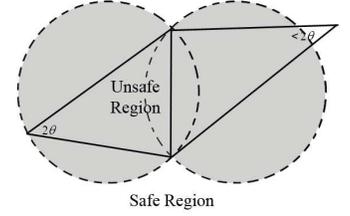


Fig. 5. Safe & Unsafe Region

into  $\mathbb{CL}_p$ . Then, we rotate the vector clockwise around  $p$ , until it encounter another sensor  $s_j$  so that  $\overrightarrow{ps_j} \parallel \overrightarrow{ps_i}$ . Add  $\overrightarrow{ps_j}$  into the list  $\mathbb{CL}_p$  and repeat this process until the rotating vector is parallel to  $\overrightarrow{ps_i}$  again. Then, the coverage list is finished, where

$$\mathbb{CL}_p = \{\overrightarrow{ps_{p_1}}, \overrightarrow{ps_{p_2}}, \dots, \overrightarrow{ps_{p_m}}\}$$

and  $|\mathbb{CL}_p|$  (the cardinality of  $\mathbb{CL}_p$ ) equals  $m$ , representing the sensor number that  $p$  needs to accomplish FVC. Based on the definition of  $\mathbb{CL}_p$ , we can state Lemma 2 as follows:

**Lemma 2** (Full-View Coverage Point [7]). *A point  $p$  is full-view covered if and only if the angle between  $\overrightarrow{ps_{p_i}}$  and  $\overrightarrow{ps_{p_{i+1}}}$  is less than or equals  $2\theta$  for any  $1 \leq i \leq m$ , where  $p_{m+1} = p_1$ . If represented by mathematical expression, Lemma 2 is:*

$$\forall \overrightarrow{ps_{p_i}}, \overrightarrow{ps_{p_{i+1}}} \in \mathbb{CL}_p (p_{m+1} = p_1), \angle(\overrightarrow{ps_{p_i}}, \overrightarrow{ps_{p_{i+1}}}) \leq 2\theta$$

According to Lemma 2, we can directly draw the conclusion that the sufficient condition of point FVC is that  $|\mathbb{CL}_p| \geq \lceil \pi/\theta \rceil$ . This can be a shortcut to exclude irrequired point. To further eliminate the irrequired points, we define following terms:

**Definition 7** (Safe and Unsafe Region). *For any two sensors  $s_i$  and  $s_j$ , we define safe region  $SR(i, j)$  that*

$$\forall p \in SR(i, j), \angle(\overrightarrow{ps_i}, \overrightarrow{ps_j}) \leq 2\theta$$

*and unsafe region  $UR(i, j)$  that*

$$\forall p \in UR(i, j), \angle(\overrightarrow{ps_i}, \overrightarrow{ps_j}) > 2\theta$$

Schematic diagram is shown in Fig. 5. The definitions are the corollary of Lemma 2. And additionally in terms of the shortcut above, we can propose the Algorithm. 2 for each sensor to detect which sub-region in his sensing disk has the probability to be full-view covered.

3) *Further Partition*: There may exist the situation that several FVC sub-regions are in one sensor's sensing disk while that sensor could not cover them simultaneously. Thus we give a definition called conflict sub-region to further partition the FVC sub-region.

**Definition 8** (Conflict Sub-region [9]). *For two FVC regions  $R_v$  and  $R_w$ , they conflict to each other if there is a camera  $s_i \in CL_{R_v} \cap CL_{R_w}$  cannot cover them simultaneously.*

Further, there may exist such situation that  $s_i$  can only cover total  $R_v$  and partial  $R_w$  simultaneously. We denote the largest covered part of  $R_w$  as  $R_{wc}$  and then divide  $R_w$  into  $R_{wc}$  and  $R_{wu} = R_w - R_{wc}$  to guarantee that any two regions in the sensing disk can only be conflict or both covered by  $s_i$ .

---

**Algorithm 2** Individual Sensor FVC Sub-Region Judgement

---

**Require:** All locations of sensor neighbors  $\mathbb{NL}$ .**Ensure:** Full-view coverage region  $FC$  within sensing disk.

```
1:  $FC = \phi$ ;  $m = \lceil \pi/\theta \rceil$ ;
2:  $nnum = |\mathbb{NL}|$ ;
3: for each neighbor location  $L_i$  in the list  $\mathbb{NL}$  do
4:   Draw  $L'_i$ 's sensing border for partition (fan-shaped for
   neighbors in Wait Phase and round for others);
5: end for
6: List partitioned region to  $\mathbb{FCL}$  (Full-view Coverage List);
7: for each sub-region  $R$  in  $\mathbb{FCL}$  do
8:   Construct  $\mathbb{CL}_p$  for sub-region.
9:   if  $|\mathbb{CL}_p| \geq m$  then
10:     $FC = FC \cup R$ ;
11:   end if
12:   for each  $s_i, s_{i+1}$  in  $\mathbb{CL}_p$  do
13:     $FC = FC - UR(i, j)$ ;
14:   end for
15: end for
16: return  $FC$ ;
```

---

4) *Local Full-View Barrier Line Detection:* We map FVC region into a conflict graph  $G = (V, E_a, E_c)$ , where node in  $V$  is abstracted from a possible FVC sub-region, edge in  $E_a$  represents a relation of two adjacent region and each edge in  $E_c$  shows the relation of two conflict sub-regions. Then, we use Dijkstra algorithm to detect the shortest paths from entrance arc to all the other partitioned arcs on sensing circle.

#### D. Model Analysis

1) *Correctness of DPA:* DPA is an efficient solution to the problem we discuss.

**Theorem 1.** *DPA can successfully construct a feasible solution for FVBC problem with rotatable camera sensor.*

*Proof:* We will prove the theorem from three aspects:

*Termination:* In DPA, due to the time limitation in Initial Phase and Update Phase, unused sensor will be definitely closed. Additionally, each sensor that once converts to Expand Phase and Wait Phase will be sure to have a son sensor and father sensor separately, which guarantees that they will receive Finish Request from its son or father and finally terminate.

*Validity:* In each iteration, BS will detect local FVB in its sensing disk and choose the next BS according to the selected exit sub-region. The next BS will consider this sub-region as entrance. This procedure guarantees the continuity of global FVB. On the other hand, when a sensor  $s$  is asked to offer help for a BS, it will broadcast it ordered direction. This action guarantees that following BS will not consider sensing disk of  $s$  as feasible area, but a sector as available region.

*Agreement:* In each phase, the sensors have a determinate algorithm and will choose a local optimal choice basing on a fixed strategy. Therefore, for a specific initial deployment,

DPA will give out a determinate feasible solution no matter how much time the algorithm is applied. ■

2) *Complexity of DPA:* It is clear that the complexity of DFS is  $O(|V|+|E|)$  if we adjacency list as basic data structure. In other words, we need  $O(|N|)$  rounds on average to finish the algorithm, where  $N$  represents the number of sensors in FoI and plays the same role with  $V$  in graph theory.

Considering the inner algorithms, we first discuss the time complexity of Algorithm 2. We define  $n_N$  as the scale of sensor neighbors, which is determined by sensor density in monitored area. The construction of sub-region costs  $O(n_N^2)$  and the detection and judgement for  $\mathbb{CL}_p$  needs  $O(n_N^3)$  separately. Thus, the complexity for Algorithm 2 is  $O(n_N^3)$ .

**Theorem 2.** *Algorithm 1's time complexity is  $O(n_N^5)$ .*

*Proof:* For each sensor that receives Expand Request for the first time, it needs  $O(n_N^2+r_s^4)$  to partition the sensing disk, where  $r_s$  is the sensing radius of camera sensors. Looking for local FVB (Algorithm 2) costs  $O(n_N^3)$  as proven before.

The following step (line 5) in Algorithm 1 costs  $O(n_N^5)$  because for each sensor, its neighbours will create  $O(n_N^2)$  sub-region no matter whether there are intersection parts. Moreover, Dijkstra algorithm costs  $O(n^2)$  with  $n$  nodes and each sensor will have at most  $2n_N - 1$  possible exits. For each exit, the algorithm will utilize Dijkstra to find a shortest path with  $O(n_N^2)$  sub-regions. So, the total time consumption is  $O(n_N \times (n_N^2)^2) = O(n_N^5)$ . Here, the further partition costs at most  $O(n_N^2)$  (Scan every sub-region for one time), so the time complexity of the second step is  $O(n_N^5)$ .

Next step is to calculate index, sort index and choose the optimal exit, which will take  $O(n_N \log n_N)$ .

The remaining steps in Algorithm 1 need only  $O(1)$  time.

Considering the sensing radius as a fixed parameter, the complexity of Algorithm 1 is  $O(n_N^2+r_s^4)+O(n_N^3)+O(n_N^5)+O(n_N \log n_N) + O(1) = O(n_N^5)$ . ■

Synthetically considering above analysis, when the monitored strip is long enough,  $O(n_N^5)$  can be omitted and the time complexity of DPA is  $O(N^2)$ . If the strip is short, the time consumption depends more on the sensor density.

As far as message complexity, the most "busy" sensor needs to send information packets, collect neighbour information and receives the request from father sensors. The total message complexity for a sensor is  $O(n_N)$ .

## V. SIMULATION AND EVALUATION

In this section, we show the simulation results on FBR problem. DPA is simulated via C++ language and the diagrams are made via Matlab. The detailed parameters are listed in Table. II. The same with the parameter values in [9].

TABLE II  
SIMULATION PARAMETERS

Parameter	Monitored Area ( $L \times W$ )	$r_s$	AoV( $\varphi$ )	MDA( $\theta$ )
Data	$20m \times 10m$	$3m$	$\pi/3$	$2\pi/3$

Since the initial deployment is random, we run the algorithm for 200 times with different initial deployment and average the results as the final results.

### A. Comparison with Other Algorithms

In previous researches [1], [7], [8], [14], the authors use static camera sensors as the basic model, which results in a waste of sensors. In Fig. 6, PS stands for the Path Selection algorithm proposed in [7] while MCSPS is short for Minimum Camera Sensors Path Selection algorithm in [8] and RCBC stands for Rotatable Camera Barrier Coverage algorithm in [9]. We can conclude from the curve that using rotatable sensors can diminish sensor number to great extent, so we only compare our algorithm DPA with RCBC in following part.

Fig. 7 shows that the barrier line successfully constructed rate is positive correlative with initial deployed sensor number, and further, DPA algorithm executes more efficiently than RCBC when the sensor number are relative small while when the sensor density is high enough, the sensor number will not be a vital factor to increase barrier line successfully constructed rate.

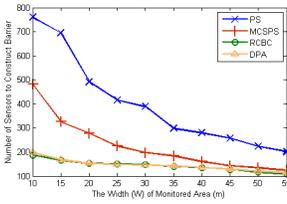


Fig. 6. Comparison with algorithm using static sensors

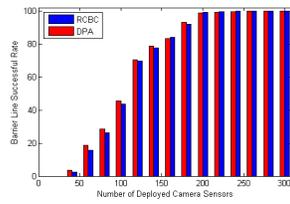


Fig. 7. Comparison of camera sensor number with centralized algorithm

### B. Impact of Simulation Parameters

Here we only discuss the factors that obviously influence the barrier construction. We assume the initial deployed sensor number is 0.5 per square meters.

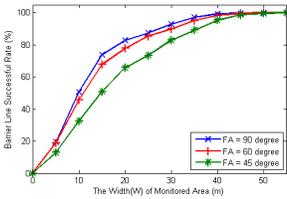


Fig. 8. Comparison with algorithm using static sensors

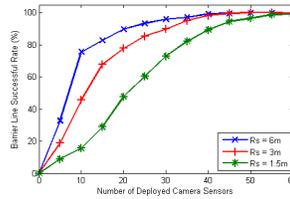


Fig. 9. Comparison of camera sensor number with centralized algorithm

In Fig. 8, we can see that a larger angle-of-field leads to a high successful range, but a double degree cannot bring a double success rate. Moreover, a doubled sensing radius can also bring about the increase on success rate, as shown in Fig. 9. Considering that a doubled sensing radius will extend the FoV more than 4 times while a doubled degree can only lead to a doubled FoV, the different rate of the increase can be illustrated to some extent.

## VI. CONCLUSION AND FUTURE WORK

This paper proposes a Distributed Proliferation Algorithm (DPA) to find a full-view barrier (FVB) in wireless networks with rotatable camera sensors. This is the first distributed algorithm tries to deal with such problem. In DPA, the backbone sensor will detects the local FVB line considering the neighbors and orders the next backbone sensor to expand FVB line. This procedure will be continuous operated until a Expand Request reached another edge. Once guaranteeing that a FVB can be constructed, Finish Request will come back along the backbone network and finally all the sensors rotate to the required direction. The simulation results show that DPA can find a FVB with relative fewer camera sensors and higher successful rate. In the future, DPA will be optimized to find the rotation protocol to rotate minimum angles to save more energy or extend to a cuboid space.

## REFERENCES

- [1] Yi Wang, Guohong Cao, *On Full-View Coverage in Camera Sensor Networks*, IEEE International Conference on Computer Communications (INFOCOM), pages 1781-1789, 2011.
- [2] Santosh Kumar, Ten H. Lai, and Anish Arora, *Barrier coverage with wireless sensors*, ACM International Conference on Mobile Computing and Networking (MOBICOM), pages 284-298, 2005.
- [3] Benyuan Liu, Oliver Dousse, Jie Wang, Anwar Saipulla, *Strong barrier coverage of wireless sensor networks*, ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), pages 411-420, 2008.
- [4] Shibo He, Jiming Chen, Xu Li, Xuemin Shen, *Cost-effective barrier coverage by mobile sensor networks*, IEEE International Conference on Computer Communications (INFOCOM), pages 819 - 827, 2012.
- [5] Shibo He, Xiaowen Guo, Junshan Zhang, Jiming Chen, *Barrier coverage in wireless sensor networks: From line-based to curve based deployment*, IEEE International Conference on Computer Communications (INFOCOM), pages 470 - 474, 2013.
- [6] Kuei-Ping Shih, Chien-Min Chou, I-Hsin Liu, Chun-Chih Li. *On Barrier Coverage in Wireless Camera Sensor Networks*, IEEE International Conference on Advanced Information Networking and Applications (AINA), pages 873 - 879, 2010.
- [7] Yi Wang, Guohong Cao. *Barrier Coverage in Camera Sensor Networks*, ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), pages 1-10, 2011.
- [8] Huan Ma, Meng Yang, Deying Li, Yi Hong, Wenping Chen. *Minimum Camera Barrier Coverage in Wireless Camera Sensor Networks*, IEEE International Conference on Computer Communications (INFOCOM), pages 217-225, 2012.
- [9] Yang Gui, Fan Wu, Xiaofeng Gao, Guihai Chen. *Full-View Barrier Coverage with Rotatable Camera Sensors*, IEEE/CIC International Conference on Communications in China (ICCC), pages 818-822, 2014.
- [10] M. Amac Guvensan, A. Gokhan Yavuz. *On Coverage Issues in Directional Sensor Networks: A Survey*, Ad Hoc Networks, pages 1238-1255, 2011.
- [11] Jian Shen, Wenyong Zheng, Jin Wang, Zhihua Xia, Zhangjie Fu. *Routing Protocols Using Directional Antennas in Ad Hoc Networks: A Comparative Review*, International Journal of Grid and Distributed Computing (IJGDC), 6(5): 39-50, Oct. 2013.
- [12] Junbin Liang, Ming Liu, Xiaoyan Kui. *A Survey of Coverage Problems in Wireless Sensor Networks*, Sensors and Transducers, 163(1): 240-246, 2014.
- [13] Tzung-Shi Chen, Hua-Wen Tsai, Chih-Ping Chen, Jiun-Jie Peng. *Object Coverage with Camera Rotation in Visual Sensor Networks*, International Wireless Communications and Mobile Computing Conference (IWCMC), pages 79-83, 2010.
- [14] Yi Wang, Guohong Cao. *Achieving Full-View Coverage in Camera Sensor Networks*, ACM Transactions on Sensor Networks (TOSN), 10(1): 1-31, 2013.