Belt-Barrier Construction Algorithm for WVSNs

Chien-Fu Cheng
Department of Computer Science and Information Engineering, Tamkang University
New Taipei City, Taiwan (R.O.C.)

Kuo-Tang Tsai
Graduate Institute of Networking and Communication, Tamkang University
New Taipei City, Taiwan (R.O.C.)

***Abstract*—Previous research of barrier coverage did not consider breadth of coverage in Wireless Visual Sensor Networks (WVSNs). In this paper, we consider breadth to increase the Quality of Monitor (QoM) of WVSNs. The proposed algorithm is called Distributed β-Breadth Belt-Barrier construction algorithm (*D-TriB*). *D-TriB* constructs a belt-barrier with β breadth to offer β level of QoM, we call β-QoM. *D-TriB* can not only reduce the number of camera sensors required to construct a barrier but also ensure that any barrier with β-QoM in the network can be identified. Finally, the successful rate of the proposed algorithm is evaluated through simulations.**

Keywords- Barrier Coverage; Breadth of Image; Quality of Monitor; Wireless Visual Sensors Network.

I. INTRODUCTION

The barrier coverage problem is very important and challenging in Wireless Visual Sensor Networks (WVSNs). However, previous research of WVSNs did not consider breadth of coverage (Definition 1) and thus could not ensure high Quality of Monitor (QoM). For example, in Figure 1(a), two segments of an image are captured at time point t=0 and t=1 respectively. Because both of these two image segments have a small width, it is not easy to identify the intruder. If there is a requirement of breadth for each barrier (Figure 1(b)), we may be able to identify the intruder more easily and quickly. Therefore, we propose the following definition: a belt-barrier with a minimum breadth of β is called a β-breadth belt-barrier, and the QoM it can provide is called β-QoM (Definition 2).

Definition 1: Breadth

In WVSNs, a barrier is formed by the Field of Views (FoVs) of camera sensors. Breadth refers to the width of image that a camera sensor can capture at each time point. Because the sensing range of a camera sensor is in the form of a sector, the breadth of a barrier that a camera sensor can provide depends on where it is located in the sector.

Definition 2: β-Quality of Monitor (β-QoM)

The breadth of a barrier depends on the minimum breadth of the barrier. A belt-barrier whose minimum breadth is β is called a β-breadth belt-barrier, and the QOM it provides is defined as β-QoM.

However, some difficulties and challenges are involved in constructing a barrier of β-QoM. In WVSNs, a camera sensor's sensing range is in the form of a sector (with the radius of r and the central angle of θ ($0 \leq \theta \leq \pi/2$)) [8][11]. It is impossible to determine whether two camera sensors have any overlap in sensing range simply by the distance between them (smaller than $2r$ or not) [10]. In addition, our requirement of breadth for a constructed barrier has added more difficulty to this problem. Moreover, each sensor has information of only its neighbor sensors. It will be very difficult to find the optimum solution (minimum number of camera sensors) using a distributed algorithm. Thus, how to design an efficient algorithm that uses a smaller number of camera sensors and ensures that any barrier of β-QoM in the network can be identified is a key challenge for this research.

II. RELATED WORKS

In this section, we provide a review of literature related to wireless sensor networks (WSNs) and WVSNs.

2.1 The barrier coverage problem in WSNs

In the following paragraphs, we will introduce some related works of barrier coverage in WSNs. Studies [2][5][6] assumed the 0/1 sensing model and used omnidirectional sensors to solve the barrier coverage problem. Among them, Kumar et al. [5] proposed a centralized k-barrier algorithm for the barrier coverage problem. This algorithm detects intruders by constructing k barriers and provides two types of barrier coverage, one weak and one strong. Chen et al. [2] argued that using a centralized global algorithm to construct barriers is not very feasible in practical applications. Thus, the authors proposed a distributed algorithm called Localized Barrier Coverage Protocol (LBCP) to construct barriers and attempted to increase the network lifetime of barriers using a sleep-wake-up schedule. Liu et al. [6] proposed to divide the monitored area into multiple segments. Their method was to construct a horizontal barrier in each segment based on a distributed approach and then deploy a large number of sensors between segments to form vertical barriers. By doing so, the horizontal barriers in each segment can be connected by a neighboring vertical barrier (strong barrier). Their algorithm was called Divide-and-Conquer Algorithm to Construct Barrier Coverage (DCACBC). Although the 0/1 sensing model offers an effective solution to the above-mentioned problems, it can only detect whether an object is within the sensing range but not tell what the object is. Therefore, many later researchers have focused on WVSNs to discuss the various coverage problems.

2.2 The coverage problems in WVSNs

Below is a brief review of some works of the coverage problems in WVSNs. These works mainly address the target coverage problem [3][4][7][11] and the barrier coverage problem [10] in WVSNs. To achieve target coverage in WVSNs, Chow et al. [3] reduced the target coverage problem to an angle coverage problem and proposed an algorithm, called FIND_MIN_COVER algorithm, to solve this problem. This algorithm is based on the idea of viewing the target as the center to achieve 360° coverage of the target. In the selection of barrier members, this algorithm chooses the ones that can contribute more to the angle coverage of the target. In another paper [4], Chow et al. discussed how to reduce the number of camera sensors required and the transmission distance (hop counts) between camera sensors and the sink. Two versions of the algorithm, one centralized and one distributed, were proposed. Liu et al. [7] also used camera sensors to solve the target coverage problem. Moreover, they also considered the angle and focus of camera sensors to find a better combination of camera sensors that have a smallest overlap in coverage angles. Yang et al. [11] assumed that each target has differentiated coverage quality requirement, and the quality of coverage depends on the distance between the target and the sensor. Based on this concept, they proposed an algorithm called Maximal Network Lifetime Scheduling (MNLS). In the selection of barrier members, MNLS uses a heuristic approach to pick camera sensors that can jointly provide more coverage of the target and better quality of coverage. MNLS also considers power consumption of camera sensors to build a sensor network with a longer network lifetime.

To achieve barrier coverage in WVSNs, Shih et al. [10] proposed the Cone-based Barrier coveRage Algorithm (CoBRA). CoBRA is the first algorithm that solves the barrier coverage problem in WVSNs. CoBRA assumes that there is one sink on both the left and right barriers of an area, and all the sensors have rotation capability. In the initial phase, the sink on the left barrier broadcasts Barrier Request (BREQ) messages to all camera sensors in the area. These camera sensors will forward the BREQ messages to the sink on the right barrier. After collecting all the BREQ messages, the sink on the right barrier will select the barrier line that passes through a minimum number of hops. The main advantage of this method is that any barrier existent in the network can be identified. Its main weakness is that broadcasting of BREQ messages may result in a large overhead of data transmission and considerable packet collisions.

As mentioned earlier, early studies of coverage problem in WVSNs did not take into account of breadth [3][4][7][10][11], resulting in a poor quality of images collected. Besides, forwarding control messages via broadcasting will cause a large overhead of data transmission and possible packet collisions. In light of these problems, we will revisit the barrier coverage problem in WVSNs and propose more efficient distributed algorithms that consider breadth.

III. Problem Formulation and System Model

In this paper, we assume that the area to be monitored is in a rectangular shape. Four barrier lines are east barrier line L_e, west barrier line L_w, south barrier line L_s and north barrier line L_n. Camera sensors are scattered from helicopters or dropped by cannons. All camera sensors are aware of their own location information and directional information. Location information can be obtained from a built-in GPS or using a location algorithm [9]. The arguments of camera sensors fall in the range of [0°, 360°). Each camera sensor has a communication range of $2r$ and a FoV in the shape of a sector with a radius of r and a central angle of θ ($0 \le \theta \le \pi/2$). θ is represented as shown in Figure 2 (θ_1 denotes the small angle, θ_2 the large one, and θ the difference). Assume that a camera sensor is located at (x, y) and an intruder at (x', y'). If both Equation (1) and Equation (2) hold, the intruder can be detected by the camera sensor.

t=0 t=1 t=0 t=1

(a) image width - too small (b) image width - relatively larger

Figure 1. Monitoring with and without consideration of breadth

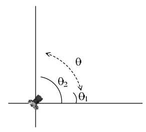


Figure 2. A sector-like sensing range

$$\sqrt{(x-x')^2+(y-y')^2} \le r \quad (1)$$

$$\theta_1 \le \sqrt{(x-x')^2/((x-x')^2+(y-y')^2)} \le \theta_2 \quad (2)$$

In the following section, we model this problem as an integer linear program. The parameters are defined as follows:

- N: all the camera sensors in the network, n=|N|;
- M: camera sensors selected to form the barrier, m=|M|, $M \subseteq N$;
- $active_i \in \{0, 1\}$: if camera sensor v_i is a member of the barrier, $active_i$=1; otherwise, $active_i$=0, $\forall v_i \in N$;
- $Lw_i\_\beta \in \{0, 1\}$: if the sensing range of camera sensor v_i intersects with the west barrier line L_w, and the breadth is greater than or equal to β, then Lw_i_β=1; otherwise, Lw_i_β=0,$\forall v_i \in N$;
- $Le_i\_\beta \in \{0, 1\}$: if the sensing range of camera sensor v_i intersects with the east barrier line L_e, and the breadth is greater than or equal to β, then Le_i_β=1; otherwise, Le_i_β=0,$\forall v_i \in N$;
- $sensor_{(i,j)}\_\beta \in \{0, 1\}$: If the sensing ranges of camera sensor v_i intersects with that of camera sensor v_j, and the breadth is greater than or equal to β, then $sensor_{(i,j)}\_\beta$=1; otherwise, $sensor_{(i,j)}\_\beta$=0, $\forall v_i, v_j \in N, i \neq j$.

As aforementioned, it is better if members of the barrier can be fewer. Hence, we set our objective function as expressed in Equation (3). Equations (4)~(7) are all intended to make sure that the constructed β-breadth belt-barrier is continuous, from L_w to L_e. Equation (4) ensures that there is only initiator of the barrier. Equation (5) ensures that there is only one terminator of the barrier. Equation (6) ensures that both the barrier initiator and terminator are respectively connected to only one other sensor. Equation (7) ensures that each barrier member v_i, excluding the initiator and the terminator, can be connected to only two camera sensors (one is the ancestor of v_i, and the other is the successor to v_i). This constraint helps avoid joining unnecessary camera sensors in the barrier.

$$minimize \ \ m=\sum_{i=1}^{n} active_i \quad (3)$$

subject to:

$$\sum_{i \in M} Lw_i\_\beta = 1 \tag{4}$$

$$\sum_{i \in M} Le_i\_\beta = 1 \tag{5}$$

$$\sum_{j \in M} sensor_{(i,j)}\_\beta = 1 \text{, for sensor } v_i \text{ where } Lw_i\_\beta=1 \text{ or } Le_i\_\beta=1,\ i \neq j. \tag{6}$$

$$\sum_{j \in M} sensor_{(i,j)}\_\beta = 2 \text{, for sensor } v_i \text{ where } Lw_i\_\beta \neq 1 \text{ and } Le_i\_\beta \neq 1,\ i \neq j. \tag{7}$$

IV. BARRIER CONSTRUCTION ALGORITHM

The proposed algorithm is called "Distributed β-Breadth Belt-Barrier construction algorithm" (*D-TriB*). *D-TriB* constructs a belt-barrier with a breadth of β based on a distributed approach. First of all, we find the initiator of β-breadth belt-barrier on the west barrier line L_w and select suitable relay sensors from left to right until the east barrier line L_e is reached. Since our goal is to use a smaller number of camera sensors to construct the barrier. Here, we use a *VLine* function to convert each camera sensor's FoV into a horizontal line. After the conversion, we can determine which candidate sensor's FoV is closer to the east barrier line and use this as a basis to select relay sensors. The pseudo code of *VLine* function is shown in Figure 3. Take a sensor's FoV that matches Case 3 as an example in Figure 4. In Case 3, the sensor's virtual line (contribution) should be calculated in two parts. The left part is determined by the length of FoV reflected upon the x-axis in the second quadrant, so it should be represented by the x coordinate of the left endpoint plus $r \cdot \cos\theta_2$. The right part is determined by the length of FoV reflected upon the x-axis in the first quadrant, so it should be represented by the x coordinate of the right endpoint plus $r \cdot \cos\theta_1$.

```
Function: VLine(v_i)
Input: the location information of camera sensor v_i
Output: the virtual line (x^{v_i}, x̄^{v_i})
if (θ1=0°) or (0°<θ1<90° and 270°<θ2<360°) do    //case 1
    return (x_i, x_i+r)
  else if (0°<θ1, θ2<90°) do    //case 2
    return (x_i, x_i+r·cosθ1)
  else if (0°<θ1<90°) and (90°<θ2<180°) do    //case 3
    return (x_i+r·cosθ2, x_i+r·cosθ1)
  else if (90°<θ1, θ2<180°) do    //case 4
    return (x_i, x_i+r·cosθ2)
  else if (90°<θ1≤180°) and (180°≤θ2<270°) do    //case 5
    return (x_i-r, x_i)
  else if (180°<θ1, θ2<270°) do    //case 6
    return (x_i+r·cosθ1, x_i)
  else if (180°<θ1<270°) and (270°<θ2<360°) do    //case 7
    return (x_i+r·cosθ1, x_i+r·cosθ2)
  else if (270°<θ1, θ2<360°) do    //case 8
   return (x_i, x_i+r·cosθ2)
```

Figure 3. Virtual Line Function

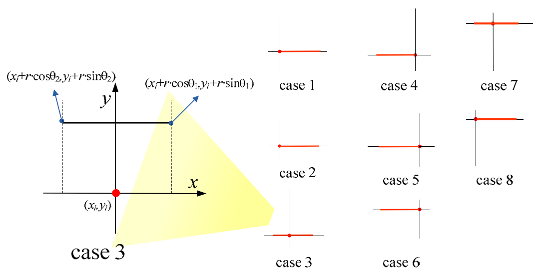


Figure 4. Conversion of the virtual ine

In this section, we will elaborate the execution process of the *D-TriB* algorithm. In the beginning of Task Initialization, *D-TriB* performs two tasks in parallel, namely BarrierConstruct and InitMsgCollection. The former task is about barrier construction, and the latter is responsible for collecting messages (line 4 in *D-TriB*). The initiator will be selected by using the *VLine* function and based on the constraint $Đ_*(L_w, v_i) \geq \beta$ (Definition 3) (line5-6 in *D-TriB*). If camera sensor v_i satisfies this constraint, v_i will forward the INIT message via geographic routing [1] to all the areas where there exists a possible initiator as well (line7 in *D-TriB*). After receiving the INIT messages from all the initiator candidates (line8-11 in *D-TriB*), the algorithm will find the camera sensor that can provide the largest contribution among them and inform it to be the barrier initiator (line12-14 in *D-TriB*).

After the initiator receives H_INT message, it will proceed to the relay sensor selection operation of barrier construction (line15 in *D-TriB*). In relay sensors selection, the selector v_i will not choose a neighbor sensor v_j that does not satisfy $Đ_{\#}(v_i, v_j) \geq \beta$ (Theorem 1) (line18-21 in *D-TriB*), any sensor in $backtrackSet_{Ind}$ (which leads to a dead end) or any sensor that has already become a member of the barrier B (line22-24 in *D-TriB*). If sensor v_i cannot find any succeeding relay sensor, it will be put in $backtrackSet_{Ind}$ to avoid being chosen by other camera sensors (line25-27 in *D-TriB*). At the same time, its ancestor will be informed to reselect a successor (line28-39 in *D-TriB*). If there is no relay sensor before v_i, the initiator selection operation will be re-executed (line28-36 in *D-TriB*). The above backtrack mechanism ensures that if there exists any β-breadth belt-barrier in the area, this barrier can be identified by the *D-TriB* algorithm. If sensor v_i can find a qualified relay sensor, it will pass the barrier construction task to the relay sensor. The relay sensor will follow the same procedure to find the next relay sensor. If multiple candidates meet the condition, the one that provides relatively more contribution will be selected to be the relay sensor (line40-48 in *D-TriB*). After the barrier is constructed, $Đ_*(L_e, v_i) \geq \beta$ can be satisfied, and the members of the barrier will be returned to the system (line16-17 in *D-TriB*).

Definition 3: The breadth coverage of border, $Đ_*(L, v_i)$.

If a straight line L has two points of intersection $((x_1,y_1), (x_2,y_2))$ with FoV, the distance between the two points $Đ_*(L, v_i)$ can be obtained using Equation (8). If there are not two points of intersection, then $Đ_*(L, v_i)=0$.

$$\begin{cases} \textbf{\textit{if}}\ (x+r\cdot\cos\theta_1<0)\ \textbf{or}\ (x+r\cdot\cos\theta_2<0)\ \textbf{then} \\ \quad Đ_*(L, v_i)=\sqrt{(x_1-x_2)^2+(y_1-y_2)^2} \\ \textbf{\textit{else}} \\ \quad Đ_*(L, v_i)=0 \end{cases} \tag{8}$$

Definition 4: Position function of space curves, $\phi(x(t), y(t))$.

The position function $\phi(x, y)$ is used to point out the location (coordinates) of a point. Because $\phi(x(t), y(t))$ varies with parameter t, the locus of its points is a curve. In other words, $\phi(x(t), y(t))$ is used to represent the position function of space curves.

Lemma 1: If curve ϕ_1 and curve ϕ_2 do not intersect, any point t_1 on ϕ_1 can find another point t_2 on ϕ_2 which makes the distance between t_1 and ϕ_2 the shortest. By trying all the points on curve ϕ_1, the shortest distance between the two curve $d(\phi_1, \phi_2)$ can be obtained.

Proof: According to Definition 4, ϕ_1 and ϕ_2 can be determined. Bring the point $(x(t_1), y(t_1))$ on ϕ_1 to get $\phi_2(x(t_1), y(t_1))$. Take the first derivative of $\phi_2(x(t_1), y(t_1))$ and set the derivative equal to zero, and the extrema (minima) can be obtained. If $t_1=s_1$, Equation (9) holds. We set the first derivative of the distance between the two points equal to zero (Equation (10)) to find the point $(x(s_1), y(s_1))$ on ϕ_1 that is closest to curve $\phi_2(x(t_2), y(t_2))$. If $t_2=s_2$, Equation (10) holds. We can bring the two points into Equation (11) to find the shortest distance between the two curves $d(\phi_1, \phi_2)$.

$$\frac{d\phi_2(x(t_1), y(t_1))}{dt_1}=0 \qquad (9)$$

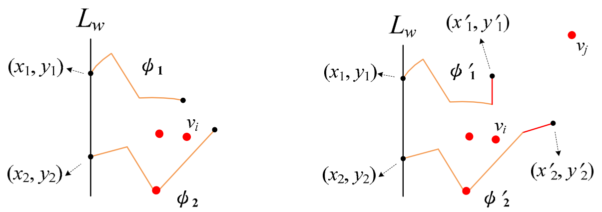
$$d\phi_2(x(t_1), y(t_1))=0$$

$$\frac{d\sqrt{(x(s_1)-x(t_2))^2+(y(s_1)-y(t_2))^2}}{dt_2}=0 \qquad (10)$$

$$d(\phi_1, \phi_2)=\sqrt{(x(s_1)-x(s_2))^2+(y(s_1)-y(s_2))^2} \qquad (11)$$

Theorem 1: In a β-breadth belt-barrier, the minimum breadth $Ɖ_{\#}(v_i, v_j)$ must be greater than or equal to β.

Proof: According to Lemma 1, the shortest distance between two curves can be determined. Due to the breadth consideration, the constructed barrier will be in a belt form. We can use two curves to represent the barrier. Let the two barriers denoted by ϕ_1 and ϕ_2 respectively, v_i be the last camera sensor to join the barrier, and the minimum breadth of the barrier is greater than or equal to β (Figure 5(a)). If a new camera sensor v_j joins the barrier, and ϕ_1 will ϕ_2 be respectively extended to become ϕ'_1 and ϕ'_2 (Figure 5(b)). Based on Lemma 1, we can compute the minimum breadth of coverage formed by the extended curves. If the minimum breadth is greater than or equal to β, this camera sensor will be accepted as a member of the barrier. By doing so, we can ensure that the minimum breadth of coverage $Ɖ_{\#}(v_i, v_j)$ of a β-breadth belt-barrier is always greater than or equal to β.



(a) before curve extension (b) after curve extension

Figure 5. Illustration of how the minimum breadth is determined

The pseudo code of *D-TriB* is shown in Figure 6. The *D-TriB* algorithm can be presented with the following primitives:

- Vector B is used to record the members of the barrier, where B has entries $B[1], B[2], \ldots, B[cn]$.
- The set of $backtrackSet_{Ind}$ is used to record the dead-end camera sensors.
- A region REG is represented as a 4-tuple, REG=(upper-left vertex, upper-right vertex ,lower-left vertex, lower-left vertex).
- activate task(T1, T2): Start the task T1 and T2 concurrently.
- *VLine*(v_i): Convert the contribution of v_i into a virtual line and obtain the positions of its left endpoint ($\underline{x}^{v_i}$) and right endpoint ($\bar{x}^{v_i}$) on the x-axis.
- $£_{v\sim x}(L)$: Obtain the x-coordinate of vertical line L.
- $£_{h\sim y}(L)$: Obtain the y-coordinate of horizontal line L.
- $Ɖ_*(L, v_i)$: Compute the breadth between border L and sensor v_i (Definition 3).
- $Ɖ_{\#}(v_i, v_j)$: Compute the minimum breadth between sensor v_i and sensor v_j (Theorem 1).
- *send*(〈*msg*〉, *rcvr*): Send a message 〈*msg*〉 to receiver *rcvr*.
  - *geosend*(〈INIT, v_i, $\bar{x}^{v_i}$ 〉, REG): Send a INIT message with the information of sensor v_i and $\bar{x}^{v_i}$ to sensors in the region REG, where REG=(($£_{v\sim x}(L_w)-r$, $£_{h\sim y}(L_n)$), ($£_{v\sim x}(L_w)+r$, $£_{h\sim y}(L_n)$), ($£_{v\sim x}(L_w)-r$, $£_{h\sim y}(L_s)$), ($£_{v\sim x}(L_w)+r$, $£_{h\sim y}(L_s)$)).
  - *send*(〈H_INIT, $backtrackSet_{Ind}$, B〉, v_l): Send a H_INIT message with set $backtrackSet_{Ind}$ and vector B to sensor v_l, 〈H_INIT, $backtrackSet_{Ind}$, B〉 is used to indicate that the task of barrier initiation has been handed over to v_l.
  - *send*(〈H_RLY, $backtrackSet_{Ind}$, B〉, v_l): Send a H_RLY message with set $backtrackSet_{Ind}$ and vector B to sensor v_l, 〈H_RLY, $backtrackSet_{Ind}$, B〉 is used to indicate that the task of barrier construction has been handed over to v_l.
- *neighbor*(v_i): Find out the set of neighbors of sensor v_i.

V. Simulation Result

In this section, the proposed protocols are evaluated via the C++ language. In our simulation, we use three different deployment methods (Random, Poisson and Gaussian distribution) to observe the successful rate of barrier construction. The detailed parameters are listed in Table 1. Besides, we also analyze the relationship between number of sensors required and the successful rate of barrier construction under different β requirements.

TABLE I. Simulation Parameters

Simulation parameters	
Area(W x L):	200 x 200 unit^2
Communication range:	60 unit
Sensing range(r):	30 unit
Field of View(θ):	$\pi/3$
β:	$r/3$, $r/6$ unit

First of all, a varying number of camera sensors are randomly scattered to observe the successful rate of barrier construction. In Figure 7, we can see *D-TriB* needs to use 159 camera sensors to construct a barrier of $\beta=r/6$ at a successful rate of 100%. Later, we scatter camera sensors based on Poisson distribution and observe the variation of the successful rate while changing the value of λ (λ equal to the expected number of occurrences during the given interval). As shown in Figure 8, results show that with the increase of λ, the successful rate of *D-TriB* can increase. *D-TriB* can achieve 100% successful rate in building a barrier of $\beta=r/6$ if λ is greater than or equal to 5 (the number of camera sensors required is 125). We can find that a smaller number of sensors are needed if sensors are deployed based on Poisson distribution. This is because Random deployment may easily cause uneven distribution of sensors in the area, which in turn lowers the successful rate of barrier construction. Besides, if we increase β from $r/6$ to $r/3$, the constraint to barrier construction will become stricter, and the successful rate will also decline (Figure 7 and Figure 8).

In reality, sensors are usually scattered from helicopters to construct the barrier. The distribution of sensors scattered from helicopters has the following characteristic. The number of sensors closer to the center of airdrop point is greater that the number of sensors farther from the center of airdrop point. The distribution is very similar to Gaussian distribution of the probability theory [12]. As shown in Figure 9, given 100 camera sensors to scatter, we can obtain a more even distribution of sensors in each unit area (the curve is relatively more horizontal) if the interval of airdrop point is 20. In the next step, we investigate the successful rate of barrier construction at various intervals of airdrop point. We set the parameter of Gaussian distribution σ =20 (σ equal to standard deviation) and scatter a fixed amount of 100 camera sensors at different intervals of airdrop point, from 100 to 10. For an area that is 200 units in length, there must be 11 ($\lfloor 200/20 \rfloor$+1) times of airdrops if the interval is set 20, and about 9 (100/11) cameras are scattered in each airdrop. As shown in Figure 10, the successful rate of *D-TriB* increases with the reduction of interval of airdrop point. This is due to the fact that with the shortening of the interval, the probably of sensor connection will increase. For *D-TriB*, if the interval is smaller than 60, the successful rate of constructing a barrier of $\beta=r/6$ can be 100%.

Finally, we investigate the successful rate of barrier construction under different requirements of β in each deployment. In the deployment of 200 camera sensors using Random distribution (Figure 11), *D-TriB* can reach 100% successful rate if β is less than or equal to 11; the successful rate drops to 0% if β is equal to or greater than 23. In the deployment of 200 camera sensors (λ =8) using Poisson distribution (Figure 11), *D-TriB* can reach 100% successful rate if β is less than or equal to 13; the successful rate drops to 0% if β is equal to or greater than 23. In the deployment of 200 camera sensors (interval of airdrop point is 20) using Gaussian distribution (Figure 11), *D-TriB* can reach 100% successful rate if β is less than or equal to 19; the successful rate drops to 44% if β is equal to or greater than 25. From the above simulation results, it can be concluded that given a fixed number of camera sensors to scatter, the successful rate of the proposed algorithms is significantly affected by β. Besides, we also observed that given the same amount of camera sensors, the β-QoM can be higher when camera sensors are deployed using Gaussian distribution than when deployed using the other two distribution methods.

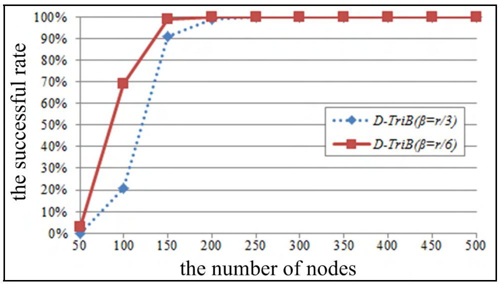


Figure 7. Random distribution

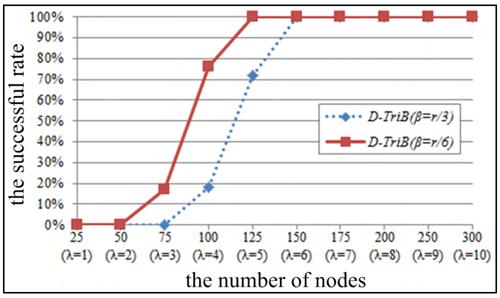


Figure 8. Poisson distribution with different value of λ

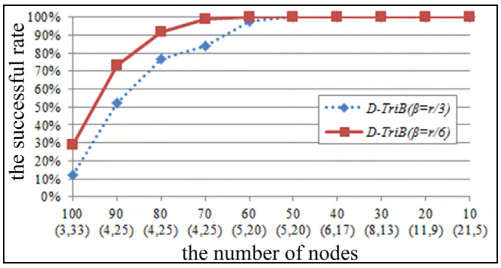


Figure 10. Gaussian distribution with different interval of airdrop point

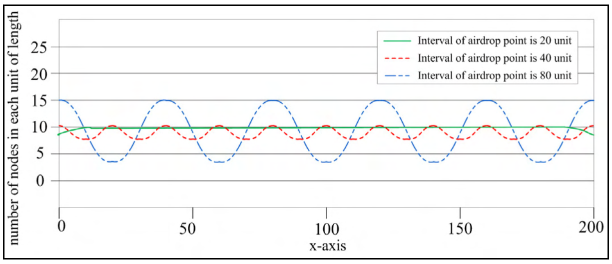


Figure 9. Density of sensors at different interval of airdrop point (100 sensors in total)

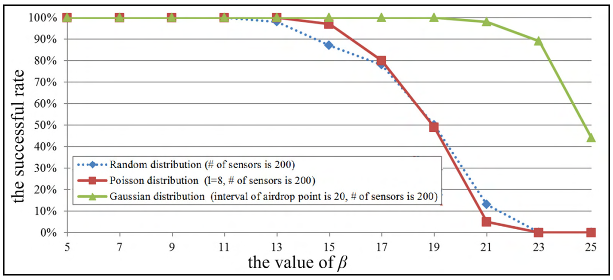


Figure 11. Random distribution, Poisson distribution and Gaussian distribution with different value of β

VI. CONCLUSION

For WVSNs, breadth of image coverage is also an important factor. However, this factor was not considered in previous research of WVSNs. Without this consideration, image identification in WVSNs may be difficult. In this paper, we revisit the barrier coverage problem in WVSNs. With consideration of breadth, we propose the *D-TriB* algorithm to construct a barrier of β-QoM. *D-TriB* uses a smaller number of sensors to construct a barrier of β breadth. We also analyze the successful rate of barrier construction under different requirements of β. Results show that the lower the β requirement, the higher successful rate.

REFERENCES

[1] C. Y. Chang, C. T. Chang, and S. C. Tu, "Obstacle-Free Geocasting Routing Protocol for Ad-Hoc Wireless Networks," *ACM/Baltzer Journal of Wireless Networks,* vol. 9, no. 2, pp.143-155, 2003.

[2] A. Chen, S. Kumar, and T. H. Lai, "Local Barrier Coverage in Wireless Sensor Networks," *IEEE Transactions on Mobile Computing*, vol. 9, no. 4, pp. 491 - 504, 2010.

[3] K.Y. Chow, K.S. Lui, and E.Y. Lam, "Maximizing Angle Coverage in Visual Sensor Networks," *Proceeding of the IEEE International Conference on Communications (ICC), pp. 3516–3521*, 2007.

[4] K. Y. Chow, K. S. Lui, and E. Y. Lam, "Achieving 360 Angle Coverage with Minimum Transmission Cost in Visual Sensor Networks," *Proceeding of the IEEE Wireless Communications and Networking Conference(WCNC)*, pp. 4112-4116, 2007.

[5] S. Kumar, T.H. Lai, and A. Arora, "Barrier Coverage with Wireless Sensors," *Proceeding of the Annual ACM International Conference on Mobile Computing and Networking (MobiCom)*, pp. 284-298, 2005.

[6] B. Liu, O. Dousse, J. Wang, and A. Saipulla, "Strong Barrier Coverage of Wireless Sensor Networks," *Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, pp. 411-419, 2008.

[7] L. Liu, H. Ma, and X. Zhang, "Analysis for Localization-Oriented Coverage in Camera Sensor Networks," *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 2579-2584, 2008.

[8] H. Ma and Y. Liu, "Some Problems of Directional Sensor Networks," *International Journal of Sensor Networks*, vol. 2, no. 1/2, pp. 44–52, 2007.

[9] S. F. A. Shah, S. Srirangarajan and A. H. Tewfik, "Implementation of a Directional Beacon-based Position Location Algorithm in a Signal Processing Framework," *IEEE Transactions on Wireless Communications*, vol. 9, no 3, pp. 1044-1053, 2010.

[10] K.P. Shih, C.M. Chou, I H. Liu and C.C. Li, "On Barrier Coverage in Wireless Camera Sensor Networks," *Proceeding of the IEEE Advanced Information Networking and Applications (AINA)*, pp. 873-879, 2010.

[11] H. Yang, D. Li and H. Chen, "Coverage Quality Based Target-Oriented Scheduling in Directional Sensor Networks," *Proceeding of the IEEE International Conference on Communications (ICC)*, pp. 1-5, 2010.

[12] G. Yang and D. Qiao, "Multi-Round Sensor Deployment for Guaranteed Barrier Coverage," *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, pp. 1-9, 2010.

[13] Z. Yun, X. Bai, D. Xuan, T. H. Lai and W. Jia, "Optimal Deployment Patterns for Full Coverage and *k*-Connectivity (*k*≤6) wireless sensor networks," *IEEE/ACM Transactions on Networking*, vol. 18, no. 3, pp. 934-947, 2010.

Algorithm: *D-TriB* //for each sensor v_i

Input: the location information of the camera sensor v_i
Output: a vector of camera sensor B

```
Initialization:
Set InitMsgSet ← ∅
Vector B ← ∅
Set backtrackSet_Ind ← ∅
activate task(BarrierConstruct, InitMsgCollection)
( x^{v_i} , x̄^{v_i} ) ← VLine(v_i)
if Đ*(L_w, v_i) ≥ β do
  geosend ( 〈INIT, v_i, x̄^{v_i}〉 , REG)
  wait until (all INIT messages from initial candidates have been received)
    for 〈INIT, v_j, x̄^{v_j}〉 ∈ InitMsgSet do
      X̄ ← X̄ ∪ { x̄^{v_j} }
    end
  if x̄^{v_i} =max( X̄ ) do
    B.push_back(v_i)
    send( 〈H_INIT, backtrackSet_Ind, B〉 , v_i)

Task BarrierConstruct:
when 〈H_INIT, backtrackSet_Ind, B〉 or 〈H_RLY, backtrackSet_Ind, B〉 is received do
  if Đ*(L_e, v_i) ≥β do
    return B
  for v_j ∈ neighbor(v_i) do
    if Đ#(v_i, v_j) < β do
      neighbor(v_i)←neighbor(v_i)－{v_j}
  end
  for n=0 to B.size( )-1 do
    neighbor_Ind←neighbor(v_i)－backtrackSet_Ind－{B[n]}
  end
  if neighbor_Ind = ∅ do
    B.pop_back(v_i)
    backtrackSet_Ind ← backtrackSet_Ind ∪ {v_i}
    if B.size( )=0 do
      X̄ ← X̄ －{ x̄^{v_i} }
      for v_b ∈ backtrackSet_Ind do
        ( x^{v_b} , x̄^{v_b} )← VLine(v_b)
        X̄ ← X̄ －{ x̄^{v_b} }
      end
      if x̄^{v_l} = max( X̄ ) do
        B.push_back(v_l)
        send( 〈H_INIT, backtrackSet_Ind, B〉 , v_l)
    else
      v_j←B.back( )
      send( 〈H_RLY, backtrackSet_Ind, B〉 , v_j)
  else
    relay sensor_cand ← ∅
    for v_j ∈ neighbor_Ind do
      relay sensor_cand ← relay sensor_cand ∪ x̄^{v_j}
    end
    for x̄^{v_k} ∈ relay sensor_cand do
      if x̄^{v_k} = max(relay sensor_cand) do
        B.push_back(v_k)
        send( 〈H_RLY, backtrackSet_Ind, B〉 , v_k)
    end

Task InitMsgCollection:
when initmsg_i = 〈INIT, v_j, x̄^{v_j}〉 is received do
  InitMsgSet ← InitMsgSet ∪ {initmsg_i}
```