

Angle Coverage Maximization in Wireless Sensor Networks

Kit-Yee Chow, King-Shan Lui and Edmund Y. Lam

Department of Electrical and Electronic Engineering, The University of Hong Kong

Abstract

In this letter, we study the angle coverage problem in wireless sensor networks. We consider the situation that the target is very large, such as a building or a lake, and each sensor can only monitor a portion of its perimeter. We study the Minimum Cover problem to identify a minimum set of sensors which ensure that the target's perimeter is completely covered by the sensors. A distributed solution with a small overhead is developed and evaluated through simulations.

I. INTRODUCTION

Wireless sensor networks have been an emerging technology in habitat monitoring, target tracking, military applications, etc [1]. An important issue in sensor networks is power scarcity, where mechanisms that optimize sensor energy utilization have a great influence on prolonging network lifetime. Coverage problem is another fundamental concern, which reflects how well an area or a target is monitored by the sensors [2] [3] [4]. Most of the current studies on target coverage assume that a target can be covered by a single sensor. However, there are situations where a sensor can only monitor a certain portion of the object. Examples include image capturing and coastline monitoring.

Our objective is to ensure that the target's perimeter is completely covered by the sensors. To save energy, we want to use as few sensors as possible to cover the whole perimeter. To the best of our knowledge, we are the first to study the problem of identifying a minimum set of sensors for this purpose. In this letter, we describe the problem definition and an optimal distributed solution to solve the problem. [5] is an earlier work of this problem.

II. NETWORK MODEL

We consider a tracking system where an object of interest is monitored by the sensors surrounding it. We assume that the sensors are randomly distributed and each sensor knows its physical location by means of GPS or some localization algorithms [6], [7].

A. Cover Range

Cover Range is defined as the portion of perimeter of the object of interest covered by a sensor node. In our work, we represent the cover range in terms of angle for ease of discussion. As long as sensors are able to identify the ranges of the perimeter of the object of interest it can cover, our algorithms work. We now briefly describe how a sensor obtains its cover range if the object of interest is cylindrical with a radius R_o as shown in Figure 1.

1) *General Sensor Networks*: In general sensor networks, the cover range of a sensor is determined by its sensing range and the distance between the node and the centre of the object of interest. In Figure 1, 2β is the covered angle, κ is the sensing range of the sensor and d is the node distance. Since every sensor knows its physical location, d is known. The covered angle can then be calculated by using the following equation:

$$\beta = \cos^{-1}\left\{\frac{R_o^2 + d^2 - \kappa^2}{2dR_o}\right\}. \quad (1)$$

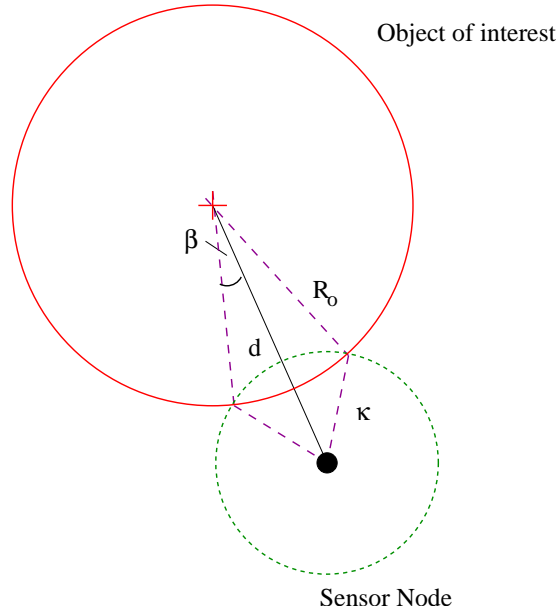


Fig. 1. Relationship between the object of interest and the sensor in a general sensor network.

It is easy to see from Figure 1 that only the sensors with a node distance less than or equal to $R_o + \kappa$ are able to cover the perimeter of the target. If κ is a constant, the closer the sensor node to the object of interest, the larger the cover range is.

2) *Visual Sensor Networks*: In visual sensor networks, where all the sensors are equipped with cameras, the cover range is determined by its camera's field-of-view (FOV) instead of sensing range. In Figure 2, 2β is the angle of view covered by the sensor node and 2α is the FOV of the node. Let d be the distance between the camera node and the center of the object of interest. Since every sensor node knows its physical location and orientation,

d and β can be evaluated by the following equations:

$$y = x \tan(\alpha) \quad (2)$$

$$\begin{aligned} d &= \sqrt{R_o^2 - y^2} + x \\ &= \sqrt{R_o^2 - x^2 \tan^2(\alpha)} + x \end{aligned} \quad (3)$$

$$\begin{aligned} \tan(\beta) &= \frac{y}{d - x} \\ &= \frac{y}{\sqrt{R_o^2 - y^2}} \end{aligned} \quad (4)$$

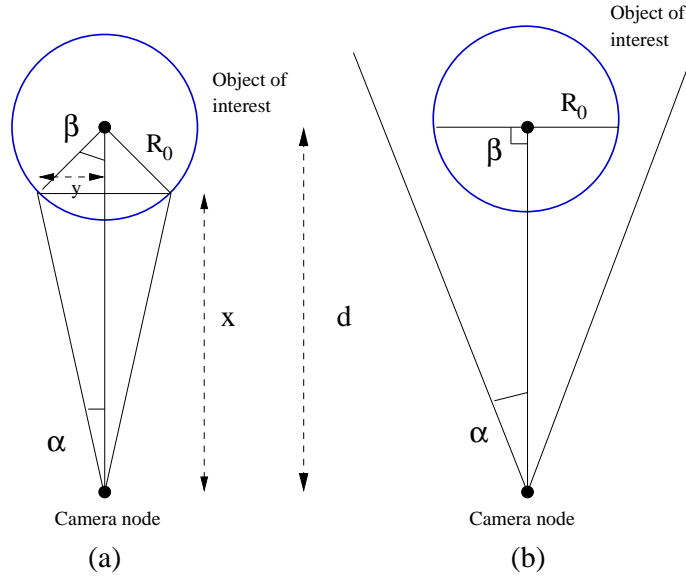


Fig. 2. Cover range of visual sensor with: (a) Small FOV; (b) large FOV

3) *Relationship Between Image Resolution And Node Distance*: Suppose the raw image size of the camera node is 512×512 . As the case in Figure 2(a), the full view of the captured image is occupied by the object of interest and $2y$ meters long visual data are projected onto 512 pixels horizontally. We may say that the image resolution is $\frac{512}{2y}$ pixels per meter. As the camera node is farther away from the target, x increases and thus y increases. As y increases, the image contains more visual data of the object of interest. In other words, for the same amount of visual data, they are represented by fewer number of pixels as the camera node is farther away and hence the resolution decreases. This shows that image resolution and node distance are inversely proportion to each other.

According to Equation 4, β increases as y increases. Since y is directly proportional to node distance and FOV, we can deduce that under the same FOV, the longer the node distance, the larger the captured range. Similarly, under the same node distance, the larger the FOV, the larger the captured range. As illustrated in Figure 2, the largest captured angle would be 180° . If this is the case, 2 images would be enough to give a round view of the

target. However, it is not desirable as the side view of the object of interest cannot be seen clearly.

Intuitively, larger β would facilitate the system to require fewer images to cover 360° and less energy would be needed. It implies that we should select images which are taken farther away from the object or by a camera with larger FOV. However, the resolution will be reduced if the node distance is increased. Consequently, there is a tradeoff between resolution and number of images needed.

B. Cover

Given a set of sensors S , let the cover range of sensor node $i \in S$ be $V(i) = [s_i, t_i]$. If $t_i < s_i$, sensor i covers 0° of the perimeter. A set $F \subseteq S$ is a *cover* if for each angle $\gamma \in [0^\circ, 360^\circ]$, there exists a sensor i in F such that $\gamma \in [s_i, t_i]$. In other words, $\bigcup_{i \in F} V(i) = [0^\circ, 360^\circ]$. Figure 3 illustrates a scenario of 9 sensors surrounding an object of interest. Each arrow represents the cover range of a node. $\{1, 3, 5, 7, 8\}$, $\{1, 2, 3, 5, 6, 9\}$, and $\{1, 3, 5, 7, 9\}$ are all valid covers. It is also possible that a cover does not exist. For example, in Figure 4, there is a range that is not covered by any sensor and there is no cover for this network.

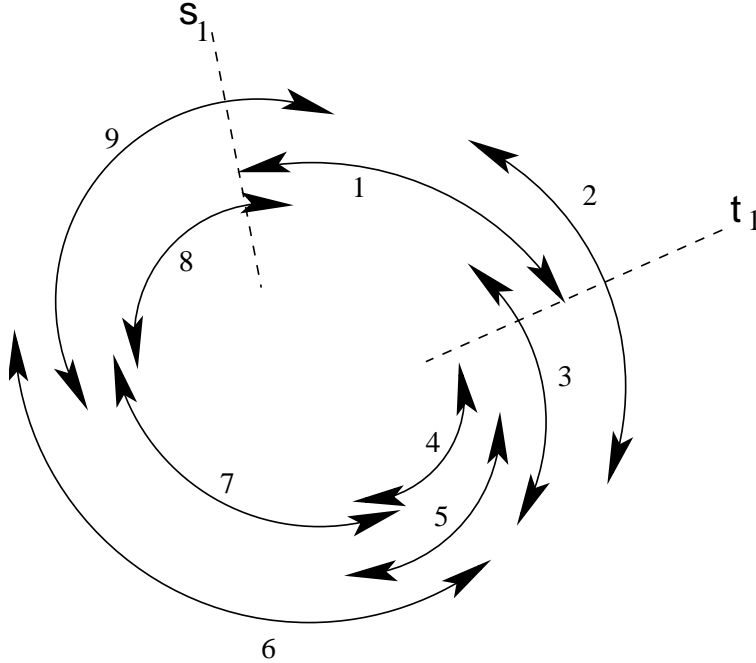


Fig. 3. Illustrations for various possibilities of sensor covers.

III. MINIMUM COVER ALGORITHM

We first explain how to solve the problem where there is no cover range spanning across 0° in a centralized manner. We then extend our algorithm to the case where at least one sensor covers 0° , and finally describe the distributed solution.

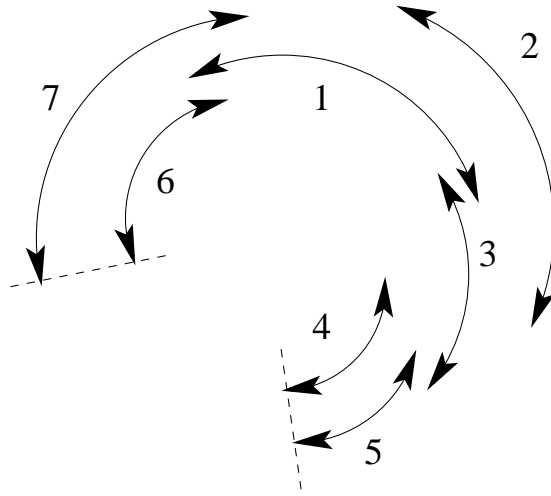


Fig. 4. Example of set of sensors which cannot cover 360°

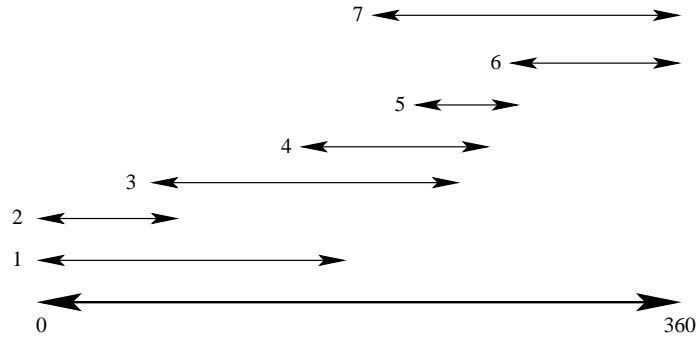


Fig. 5. Example of cover without sensor spanning across 0°

In the case where no sensor spanning across 0° , we can represent the cover ranges as shown in Figure 5. Intuitively, to reduce the number of sensors needed, we should try to select a sensor that covers as much of the target's perimeter that has yet been covered as possible. Suppose that the sensors selected are covering $[0^\circ, current_angle]$. Initially, no sensor is selected and $current_angle = 0^\circ$. The next sensor to be included should have a cover range that starts before $current_angle$ and ends as far as possible. Then, the portion that remains uncovered will be minimized. The selection process ends when the whole range $[0^\circ, 360^\circ]$ is fully covered. Referring to the example in Figure 5, there are two sensors whose ranges start from 0° . Among Sensor 1 and Sensor 2, Sensor 1 covers more of the remaining portion and so it is selected. Now, the range being covered is $[0^\circ, t_1]$. The next sensor should start before t_1 , and therefore only Sensors 3 and 4 are candidates. Among the two, Sensor 4 covers more and the algorithm selects it. Finally, 7 is selected and we are done. This strategy still works even if there is a gap between the sensors, as in Figure 4. Theoretically, the algorithm can identify a smallest sensor set to cover each sub-range. The smallest set for covering 360° or as wide as possible is the union of all the smallest sets for the sub-ranges. The pseudocode of the algorithm is as follows:

Theorem 1: Algorithm FIND_MIN_COVER is correct.

FIND_MIN_COVER(C, S)

```

1:  $C \leftarrow \emptyset$  /* initialize the minimum cover to be empty set */
2: find  $j$  where  $t_j \geq t_i \forall i \in S$ 
3:  $max\_angle \leftarrow t_j$ 
4:  $current\_angle \leftarrow 0^\circ$ 
5: while ( $current\_angle < max\_angle$ )
6: {
7:    $D = \{i \mid current\_angle \in [s_i, t_i]\}$ 
8:   if ( $D = \emptyset$ ) { /* there is a gap between the sensors */
9:     find  $k$  where
        $0^\circ < s_k - current\_angle \leq s_i - current\_angle$ 
        $\forall i \in S - C$ 
10:     $current\_angle \leftarrow s_k$ 
11:   }
12:   else {
13:     find  $j$  where  $t_j \geq t_i \forall i \in D$ 
14:      $C \leftarrow C \cup \{j\}$ 
15:      $current\_angle \leftarrow t_j$ 
16:   }
17: }
```

TABLE I
PSEUDOCODE OF FIND_MIN_COVER

Proof: To prove the theorem, we have to argue that Algorithm FIND_MIN_COVER finds a minimum cover if it exists, or finds a minimum set of sensors that preserves the widest angle coverage in case a cover does not exist, as in Figure 4, where some portion is not covered by any sensor.

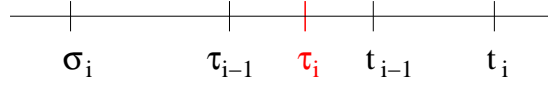
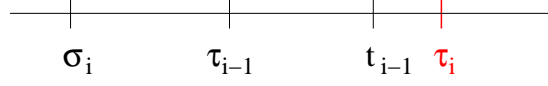
Suppose that for a certain S , there exists a minimum cover or minimum set C' of size k' . The cover ranges of those sensors are $[\sigma_1, \tau_1], [\sigma_2, \tau_2], \dots, [\sigma_{k'}, \tau_{k'}]$, where $\tau_i < \tau_{i+1}$ for $1 \leq i < k'$. We further assume that C returned by the algorithm is of size $k \geq k'$ and the cover ranges of those in C are $[s_1, t_1], [s_2, t_2], \dots, [s_k, t_k]$, where $t_i < t_{i+1}$ for $1 \leq i < k$. Note that $\sigma_1 = s_1 = 0^\circ$ and $\tau_{k'} = t_k = max_angle$. From Line 2 of the pseudocode in Table I, max_angle is the maximum ending angle among all the sensors. If C' is a minimum cover, $max_angle = 360^\circ$. If we can show that $\tau_i \leq t_i$ for $1 \leq i \leq k'$, we prove $k = k'$ and complete the proof.

According to the algorithm, among those cover ranges that start from 0° , t_1 should be the largest ending angle. That is, $\tau_1 \leq t_1$. By induction, we can show that $\tau_i \leq t_i$ for $1 \leq i \leq k'$ as follows:

Case I: C' is a minimum cover, $\sigma_i \leq \tau_{i-1} \forall i$. That is, we can assume $\sigma_i \leq \tau_{i-1} \leq t_{i-1}$. There are two cases:

- 1) $\tau_i \leq t_{i-1}$: It is obvious that $\tau_i \leq t_i$, as illustrated in Figure 6.
- 2) $\tau_i > t_{i-1}$: This scenario is shown in Figure 7, after setting $current_angle$ to t_{i-1} in Line 15 of the algorithm, sensor of range $[\sigma_i, \tau_i]$ is a candidate in D since the range covers t_{i-1} . Among all the candidates, the largest ending angle is selected to be included in C . Hence, $\tau_i \leq t_i$.

Case II: C' cannot cover 360° , there is a gap between the cover ranges.

Fig. 6. Case 1 $\tau_i \leq t_{i-1}$ Fig. 7. Case 2 $\tau_i > t_{i-1}$

In this case, we argue that our algorithm identifies the sensors that cover as much as possible. Line 8 detects the existence of a gap. In this situation, the search starts from starting angle of the next sub-range, which is identified in Line 9. Once the starting point of the next sub-range is identified, according to the proof in Case I, our algorithm always finds a minimum number of sensors to cover a continuous range. Therefore, FIND_MIN_COVER can identify a smallest set that achieves the widest angle coverage.

□

Algorithm FIND_MIN_COVER works when we can define a starting *current_angle* in Line 4. It may not be appropriate for us to set the *current_angle* to 0° if there is a cover range spanning across 0° . We should first determine whether there is any angle that is covered by one sensor, say i , only. We refer to such a node as **default member**. If this is the case, every minimum cover must contain i and the *current_angle* in Line 4 is set to be t_i . We only have to find a minimum cover starting from angle t_i and wrap around to angle s_i . For example, in Figure 3, Sensor 1 is the default member and the *current_angle* is set to be t_1 . The next sensor to be included should start before t_1 and only Sensors 2 and 3 can be candidates. The latter one is selected as it covers more of the remaining portion. After that, the *current_angle* is set to be t_3 . The selection process continues in the same fashion until the *current_angle* is greater than s_1 . If every angle is covered by more than one sensor, we identify the sensors that cover a certain angle, say 0° . For each sensor i that covers 0° , the minimum cover from $[t_i, s_i]$ is found. The smallest minimum cover is then selected. Note that the search can either go clockwise or anticlockwise, but cannot be both. We assume clockwise in the rest of the discussion.

In a large sensor network, centralized algorithms are not appropriate, and we develop a distributed solution for such cases. In the distributed algorithm, each sensor needs to obtain the cover ranges of their neighbors only. That is, if the cover range of sensor i is $[s_i, t_i]$, i knows the cover range of neighbor j if $s_i \leq s_j \leq t_i$ or $s_i \leq t_j \leq t_i$. Referring to the example in Figure 3, Sensor 1 knows the cover ranges of neighbors 2, 3, 8 and 9.

We first describe how a node knows whether it is in a minimum cover. After getting the cover ranges from neighbors, a node k first determines whether it is definitely in a minimum cover. k checks whether it fulfills either one of the two following conditions: (1) k is a default member, or (2) k does not have any anticlockwise (backward) neighbor and t_k is largest among all its neighbors. Sensor 7 in Figure 4 is an example of condition (2). It does

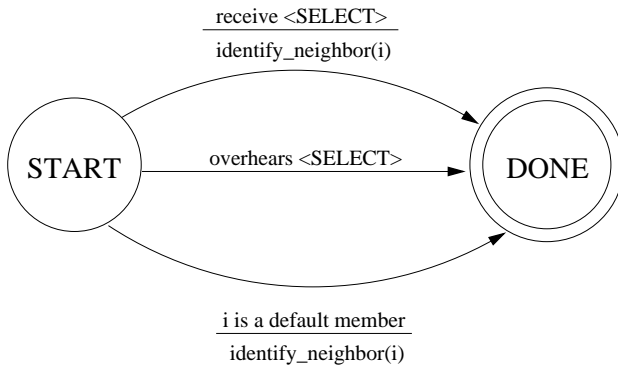
not have any backward neighbor and ends farther than its neighbor. After knowing that it is a minimum cover node, k announces to its neighbors that it is selected and then starts to identify the next sensor. According to FIND_MIN_COVER, if a sensor k is selected, $current_angle$ will be set to t_k . The next sensor to be included should be a neighbor j of k that has largest t_j . k can identify j with its local neighborhood information. k then informs Sensor j by sending a $\langle SELECT \rangle$ message. Sensor 6 in turn can find the next sensor accordingly. Other neighbors of k who can overhear the message to j would also aware that they are not selected. Referring to the example in Figure 3, Sensor 1 is a default member. Then, it informs its neighbors, Sensors 2, 3, 8, and 9, that it is selected, and then identifies the next sensor to be included in the minimum cover. Sensor 1 selects Sensor 3 and sends a message to the latter informing that it is selected. Sensor 3 then finds the next sensor, Sensor 5, which in turn selects Sensor 6. Similarly, 6 selects Sensor 9. When Sensor 9 knows that it is selected and realizes that it has a neighbor (Sensor 1) that is already selected, it stops the search. At this stage, a minimum cover $\{1, 3, 5, 6, 9\}$ is identified. Although no sensor knows the whole minimum cover, each of them knows whether it is in the cover or not and can send its data accordingly. It is also worth noting that the mechanism works if there are more than one default members. All of them will start searching for the next sensor. Once a selected node detects that it has a neighbor already selected, it stops the search. It is not difficult to see that the message overhead is very small. After acquiring neighborhood information, the only messages passing around would be for selected sensors to be included in the minimum cover.

In our algorithm, a node terminates the search when it receives or overhears a $\langle SELECT \rangle$ message. $\langle SELECT \rangle$ message is sent only if a node can identify that it is selected based on neighborhood information. If there is no default member, no $\langle SELECT \rangle$ would be triggered based on local information. Therefore, as in the centralized algorithm, when a node i that covers 0° does not hear anything for a certain period of time, it should invoke the process by assuming it is selected and send $\langle A_SELECT_i \rangle$ to its selected neighbor. To allow nodes covering 0° to identify the best cover, $\langle A_SELECT \rangle$ messages should also carry the membership of the cover. When $\langle A_SELECT_i \rangle$ is sent back to i , i knows the smallest cover that includes itself. By exchanging $\langle A_SELECT_i \rangle$ for all i that covers 0° , the minimum cover can be identified. The nodes in the cover can be informed in a distributed fashion then. To reduce overhead, we can restrict only one of them to start the search. The cover found, however, may not be optimal. We refer this situation as *sub-optimal*. The state diagram of the distributed algorithm can be found in Figure 8.

IV. SIMULATION

In this section, we present the simulation results of our protocol. Figures 9 – 11 study the performance of FIND_MIN_COVER under the scenario of visual sensor networks. Depending on applications, the user may request for images with different resolution. Only the sensors with images that fulfill the requested resolution will be

Node i that does not cover 0^0



Node i that covers 0^0

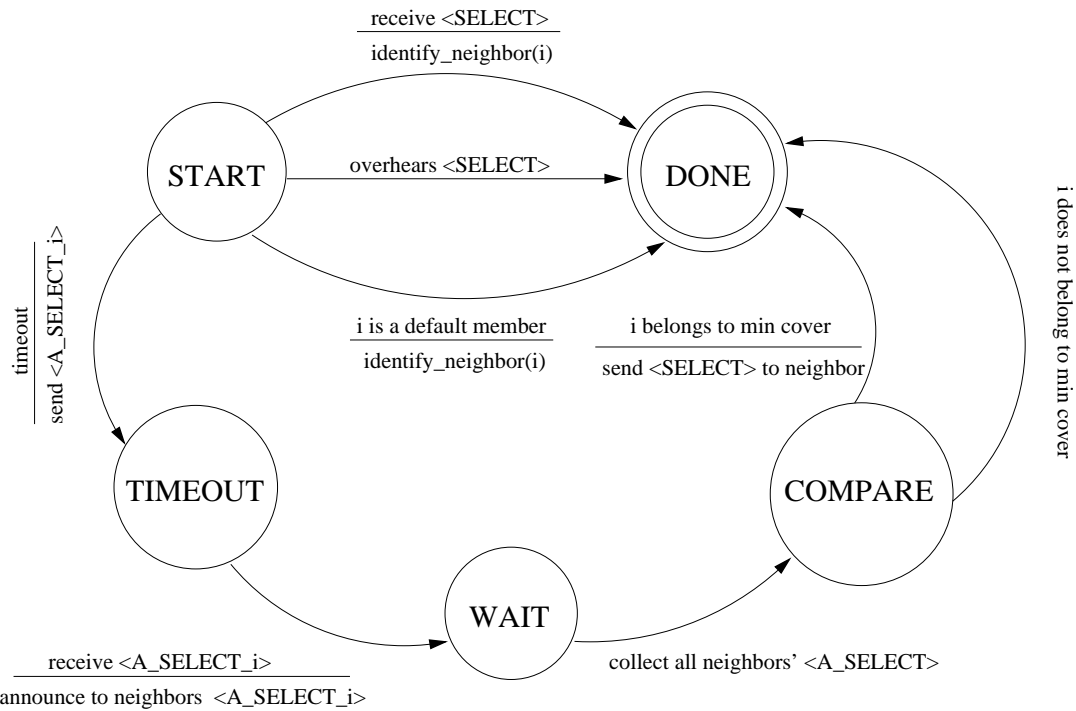


Fig. 8. State Diagram of distributed algorithm

FUNCTION identify_neighbor(i)

/ this function is for i to identify a neighbor to be included in the minimum cover */*

- 1: $N \leftarrow \{j \mid s_j < t_i < t_j\}$
 - 2: $selected \leftarrow j$ where $t_j \leq t_{j'}$, $\forall j' \in N$ and j is a not default member
 - 3: send $\langle SELECT \rangle$ to j
-

TABLE II
PSEUDOCODE OF FUNCTION IDENTIFY_NEIGHBOR(i)

considered. The simulation results are generated using *J-Sim* (formerly known as JavaSim) [8]. The whole network area is divided into $200M \times 200M$ with 400 grids where M denotes a unit length. There is at most one camera node in each grid and the probability that a grid has a sensor depends on the density and is generated randomly. In our simulation, the probability is set to be 0.8, and the communication range is set to be $25\sqrt{2} = 35.355$ units. The orientations of all camera nodes are assigned randomly and the size of the raw images captured by each camera is 512×512 . We assume that the object of interest is in cylindrical shape with radius $R_o = 50$ units.

Any nodes with node distance less than or equal to 85 units are the sensors that will be able to capture images of the object of interest. 30 topologies are generated and each topology is simulated with 7 different requested image resolutions and 3 different FOVs. Each point in the following simulation results is the average value of 30 topologies. In our simulations, the image resolution is defined as the number of pixels required to represent one unit distance in the image.

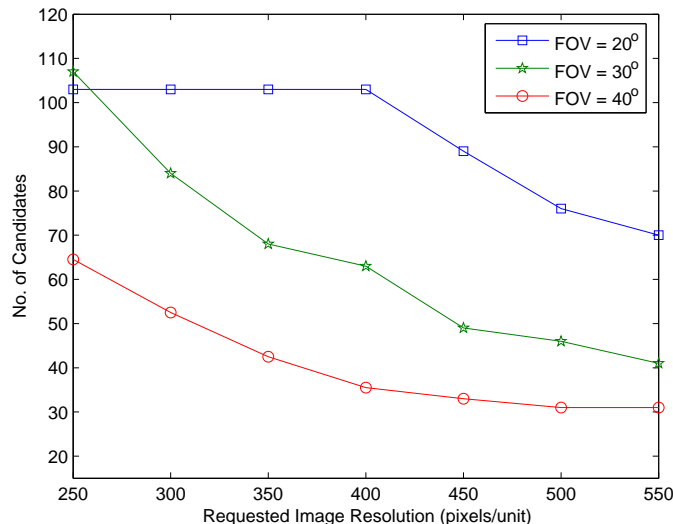


Fig. 9. Relationship between requested resolution and number of candidate

Figure 9 shows the relationship between the requested image resolution and the number of candidates involved. It can be observed that the number of candidates declines as the resolution requirement increases. Additionally, when the FOV is larger, there would be fewer candidates. It is because when the FOV is larger, sensor has to be closer to the object in order to obtain a high resolution image. In figure 10, it can be observed that the angle coverage is getting poorer when the FOV is larger or the image resolution requirement is higher. Under a particular resolution level, larger FOV would have fewer candidates. Similarly, if we require a higher resolution, there will be fewer candidates and thus more difficult to cover 360° .

Figure 11 shows the number of selected images under different resolution and different FOV. Since the candidate set of a higher resolution is a subset of the candidate set of a lower resolution, it is expected that the size of

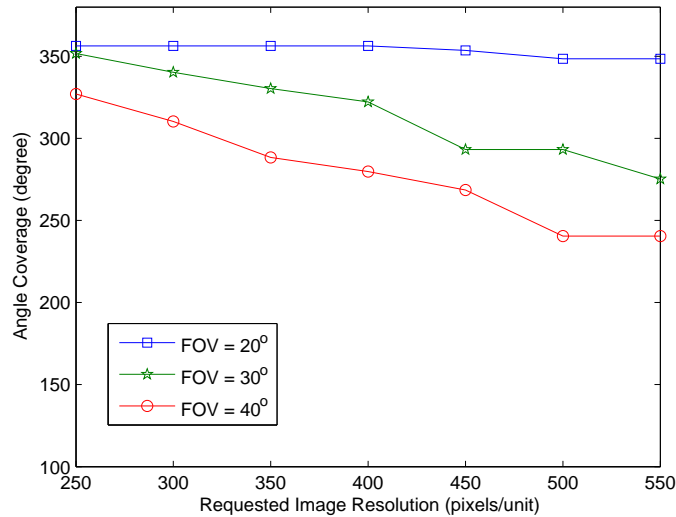


Fig. 10. Relationship between requested resolution and angle coverage

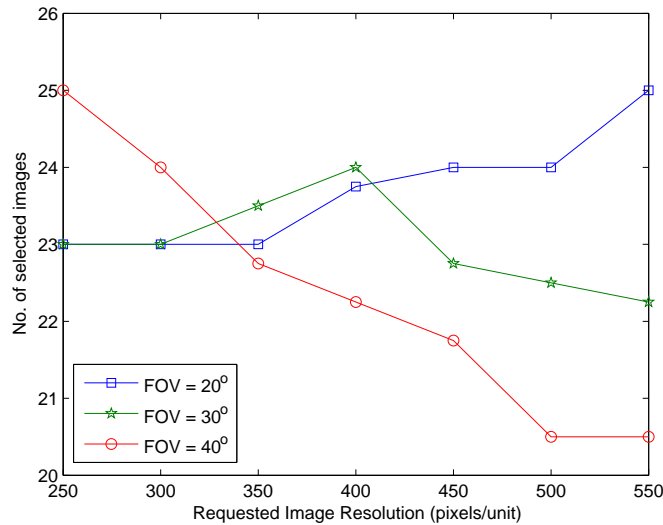


Fig. 11. Relationship between requested resolution and number of selected image

minimum cover should increase with resolution. We observe this trend when $FOV = 20^\circ$. The reason why this is not the case when $FOV = 30^\circ$ and $FOV = 40^\circ$ is because a cover cannot be formed when requested resolution is relatively large. It can be observed that there is a decreasing trend when the angle coverage is less than 340° . As the images are covering a narrower view, fewer images will be needed. The drop can be regarded as an indication of a threshold value of the requested image resolution with satisfactory angle coverage.

We also study the performance of our algorithm in general sensor networks. We assume that all the sensors have identical sensing ranges which is set to be 18 units and the object of interest is cylindrical with radius R_o . The effect of varying R_o on the size of selected cover is investigated in Figure 12. In the dummy case, all the nodes which are capable of sensing the target are included, it shows the total number of candidates of the selecton

process. It is obvious that the number of candidates increases when R_o increases. Both optimal and sub-optimal algorithms can achieve a significant reduction in cover size when compared with the dummy case.

The energy performance of the algorithm for one-off monitoring is evaluated. To facilitate our discussion, we adopt the following notations:

E_m : Energy needed in transmitting or receiving one control message

E_c : Energy needed in sensing

M : selected set of sensors

$|M|$: size of selected cover

κ : number of control messages in identifying the selected images

Let E_{total} be the total energy consumption,

$$\begin{aligned} E_{total} &= |M| \times E_c + \kappa \times E_m \text{ and therefore,} \\ \frac{E_{total}}{E_m} &= |M| \times \frac{E_c}{E_m} + \kappa. \end{aligned} \quad (5)$$

Depending on different applications or network settings, the ratios of $E_m : E_I : E_c$ can vary substantially. Figures 13 and 14 shows the normalizaed total energy consumption ($\frac{E_{total}}{E_m}$) in optimal and non-optimal cases respectively. It can be observed that the sub-optimal case consumes much less energy than the optimal case. The reason lies in the fact that the message overhead of sub-optimal case is few times less than that of the optimal case. In addition, as shown in Figure12, the size of their selected covers are comparable.

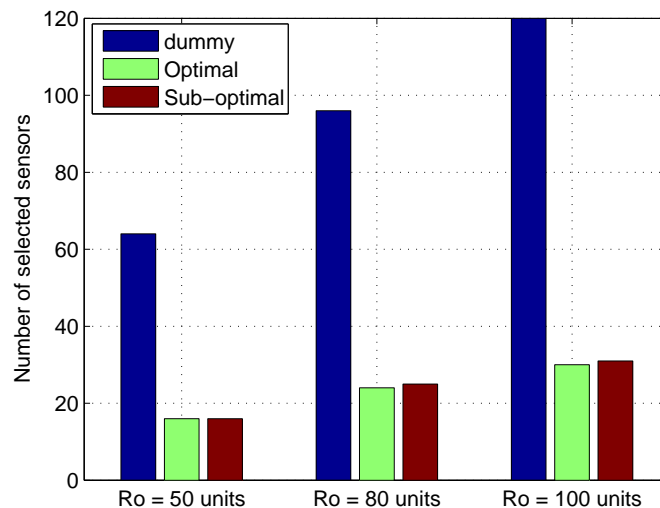


Fig. 12. Cover size comparison when R_o varies

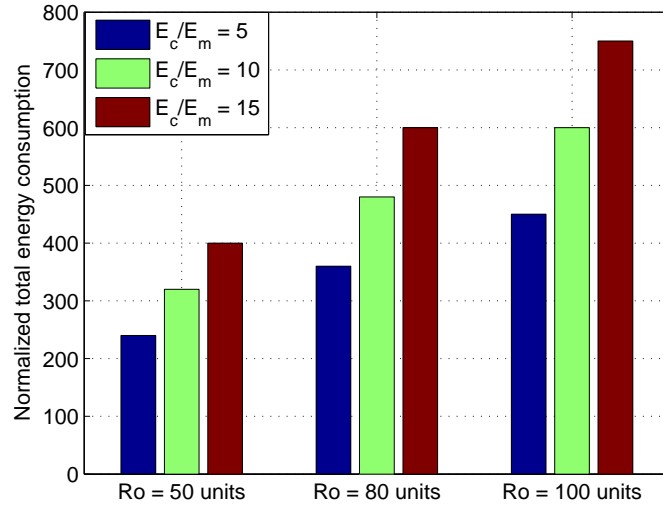


Fig. 13. Normalized total energy consumption comparison (Optimal)

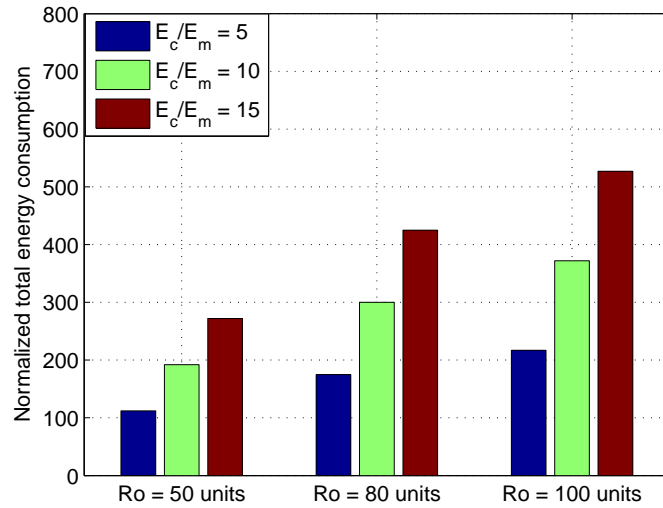


Fig. 14. Normalized total energy consumption comparison (Sub-Optimal)

V. CONCLUSION

In this letter, we present the angle coverage problem in wireless sensor networks. We develop a distributed algorithm with a small overhead to obtain the minimum cover. The algorithm is evaluated through extensive simulations and the results show that it can identify the minimum cover effectively.

REFERENCES

- [1] G. J. Pottie and W. J. Kaiser, "Wireless integrated network sensors," in *Communications of the ACM*, vol. 43, no. 5, 2000, pp. 51 – 58.
- [2] S. Gao, C. T. Vu, and Y. Li, "Sensor scheduling for k-coverage in wireless sensor networks," in *International Conference on Mobile Ad-hoc and Sensor Networks (MSN)*, 2006, pp. 268–280, INCS 4325.
- [3] J. Carle and D. Simplot-Ryl, "Energy-efficient area monitoring for sensor networks," in *IEEE Computer*, no. 2, 2004, pp. 40–46.
- [4] M. Cardei and D.-Z. Du, "Improving wireless sensor network lifetime through power aware organization," in *ACM Wireless Networks*, 2005, pp. 333 –340.
- [5] K.-Y. Chow, K.-S. Lui, and E. Y. Lam, "Maximizing angle coverage in visual sensor networks," in *IEEE International Conference on Communications, ICC*, June 2007.
- [6] J. Liang, J. Shao, Y. Xu, J. Tan, D. B.T., and B. P.L., "Sensor network localization in constrained 3-d spaces," in *IEEE International Conference on Mechatronics and Automation*, June 2006, pp. 49–54.
- [7] R. K. Patro, "Localization in wireless sensor network with mobile beacons," in *IEEE Convention of Electrical and Electronics Engineers in Israel*, Sept 2004, pp. 22–24.
- [8] A. Sobeih, W.-P. Chen, J. C. Hou, L.-C. Kung, N. Li, H. Lim, H.-Y. Tyan, and H. Zhang, "J-sim: A simulation and emulation environment for wireless sensor networks," in *IEEE Wireless Communications Magazine*, April 2005.