

Irregular-Shaped Event Boundary Estimation in Wireless Sensor Networks

Srabani Kundu, Nabanita Das, Sasanka Roy and Dibakar Saha

Abstract In a wireless sensor network (WSN), sensor nodes are deployed to monitor a region. When an event occurs, it is important to detect and estimate the boundary of the affected area and to gather the information to the sink node in real time. In case, all the affected nodes are allowed to send data, congestion may occur, increasing path delay, and also exhausting the energy of the nodes in forwarding a large number of packets. Hence, it is a challenging problem to select a subset of affected nodes, and allow them only to forward their data to define the event region boundary satisfying the precision requirement of the application. Given a random uniform node distribution over a 2-D region, in this paper, three simple localized methods, based on local convex hull, minimum enclosing rectangle, and the angle of arrival of signal, respectively, have been proposed to estimate the event boundary. Simulation studies show that the angular method performs significantly better in terms of area estimation accuracy and number of nodes reported, even for sparse networks.

Keywords WSN · Event area · Convex hull · Boundary detection · Minimum enclosing rectangle

S. Kundu (✉)
Guru Nanak Institute of Technology, Kolkata, India
e-mail: srabani6@gmail.com

S. Kundu · N. Das · S. Roy · D. Saha
Indian Statistical Institute, Kolkata, India
e-mail: ndas@isical.ac.in

S. Roy
e-mail: sasanka.ro@gmail.com

D. Saha
e-mail: dibakar.saha10@gmail.com

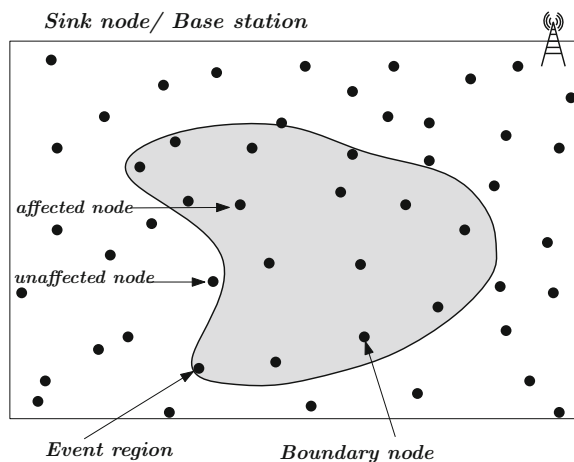
1 Introduction

In general, to monitor large inaccessible regions, wireless sensor networks are deployed with tiny, inexpensive sensor nodes distributed over an area to collect ground data [1, 2]. At regular intervals, the nodes sense data and forward it to the sink node via multihop paths.

A sensor node is basically a small device capable of sensing data, some processing, and communicating with its neighboring nodes. Here, the sensor nodes are assumed to be homogeneous and static. In most of the cases nodes are battery powered with limited or no recharging facility at all. Also, the computing capability of a node is elementary with small amount of storage. It is to be noted that typically, communication demands most of the energies of a node, whereas sensing and computing take only a small share. So, to enhance the network lifetime, it is extremely essential to limit the number of packets in the network. This necessitates in-node processing, i.e., instead of forwarding the incoming packets to the sink continuously, nodes may process data and forward the relevant information only toward the sink node. However, with limited computing power and limited memory, the in-node processing should be simple in terms of computation complexity and storage.

When an event occurs within the area to be monitored as shown in Fig. 1, it may spread over the region and it should be identified immediately. If all affected nodes start to route their information to the sink node, the network gets congested immediately resulting increase in packet delay. Also, due to huge number of packet forwarding, nodes will die out faster which can create a network failure. Hence, it is always better to choose a small subset of affected nodes which are critical to reconstruct the event region boundary, and to allow them to send their packets to the sink node only. The reduction in the number of reporting nodes at one hand limits the traffic in the network, saving energy significantly. On the other hand it also helps to reduce

Fig. 1 Event boundary and affected nodes



congestion, hence path delay for real-time reporting. Again, since it results some loss of information, it is challenging to optimize the number of reporting nodes to satisfy the precision requirement of the concerned application. Knowing all the affected nodes, the problem can be easily mapped to the classical problem of computational geometry, namely the convex hull computation. However, it is to be noted that in WSNs instead of optimal centralized algorithms, it is wise to adopt self-organized light-weight localized algorithms based on local neighborhood information only that converges with limited rounds of communication.

A lot of research activities have been reported so far on event boundary estimation problem in WSN, formulated in various ways to combat their inherent hardness. The important challenges are to limit the amount of computation and rounds of communication, and at the same time the computation should be based on minimum neighborhood information, since message communication is the only way to gather knowledge about the neighborhood of a node, and it is expensive in terms of energy. Authors in [3] presented a boundary estimation method based on two centrality measures of nodes, betweenness and closeness, respectively. In [4, 5], a graph-theory-based solution is developed to detect the event boundary, irrespective of any communication model. Based on the concept of image processing, Chintalapudi et al. in [6] proposed an algorithm to detect the network boundary. Another statistical approach to identify the boundary nodes and the topology of the region has been presented in [7].

Authors in [8, 9] proposed techniques based on computational geometry. A polynomial-based boundary estimation algorithm has been proposed in [10], where the *query tree* was constructed to route the event information in the form of a polynomial to the *sink* node. In [11–13], authors proposed some heuristic-based solutions to detect and identify the event boundary for a wireless sensor network. The gradient-based data distribution model is followed by the authors in [14, 15] to detect the event boundary for an irregular-shaped event area. In [16, 17], authors proposed a low latency event boundary detection heuristic where it generates a reduced boundary node set, without knowing the neighbors' locations and forward it to the sink node with minimum latency. Most of the above algorithms are either computation intensive, or are based on many unrealistic assumptions. In WSN, the sensed data are highly error prone and the assumption of graded data distribution is not always true. Again, detection of the boundary with the help of neighbor nodes location information requires large memory which is really very difficult to manage.

Considering a uniform random node distribution over a 2-D region, in this paper, we focus on three simple distributed methods to estimate the irregular-shaped event boundary region in WSN. First, we present two naive techniques—one based on localized convex hull, and the minimum enclosing rectangle, respectively. Finally, we propose a simple light-weight distributed algorithm based on angle of arrival of signal with $O(d \log d)$ computation and $O(d)$ space complexity in each affected node, where d is the maximum number of neighbors of a node. Each node is assumed to be equipped with directional antenna and is capable of measuring the angle of arrival of received signal. For in-node processing, each node requires the node ids of its adjacent neighbors, and only their locations are not required. Extensive

simulation studies show that the angular boundary detection method needs minimum computation and communication overhead and it also can detect the event boundary more accurately compared to the others even when the region is sparsely populated. The paper is organized as follows. Section 2 defines the problem. Section 3 proposes the distributed algorithms for the selection of the boundary nodes. Section 4 shows the simulation results and finally, Sect. 5 concludes with some open issues.

2 Network Model and Preliminaries

In our model of wireless sensor networks, the 2-D region under consideration is deployed with n homogeneous sensor nodes, randomly distributed over the area. Each node i can communicate directly with a node j if it lies within its transmission range T .

Definition 1 Two sensor nodes i and j are *neighbors* of each other, if and only if, sensor node i can communicate with node j directly, i.e., the Euclidean distance $D(i, j)$ between nodes i and j is less than the *transmission range* T , i.e., $D(i, j) \leq T$.

Definition 2 A WSN is represented by an undirected *topology graph* $G(V, E)$, where V is the set of nodes distributed over a 2D region and E is the set of edges such that an edge $(i, j) \in E$, if and only if j is a *neighbor* of i and vice versa, with $i, j \in V$.

Definition 3 In a *topology graph* $G(V, E)$, the *hop count* of a node i is represented as its distance in terms of number of hops from the sink node via shortest path.

Let us assume that each sensor node senses the environment at a regular interval of time and when required, routes the sensed data to the sink node. When an event occurs, in general, it spans over a region which may be of irregular shape. To estimate the event region boundary of irregular shape, by selecting a few boundary nodes only, it is an important and challenging problem in WSN. With the above network model, we consider the problem of selecting a reduced set of boundary nodes in a distributed fashion, which reports to the sink node which reconstructs the event boundary and estimates the affected area in terms of the convex hull enclosing all reported nodes.

Definition 4 Given a set of points S distributed over a 2-D region, the convex hull of S is defined as the smallest convex polygon enclosing all points of S .

To achieve the solution with acceptable accuracy level, we propose three distributed algorithms with simple in-node processing based on limited neighborhood information that converges with small number of communication rounds. In our proposed model, sensor nodes do not require the actual data value, and no assumption has been taken about the data distribution within the event area.

3 Algorithms for Estimating the Irregular-Shaped Event Boundary Region

To detect the change of environmental phenomenon at a regular interval of time, we assume that sensor nodes are deployed randomly over an area. When an event occurs, it spans an area R of arbitrary shape without any hole. If the sensed data crosses a threshold value, then a node executes the boundary detection algorithm as described below. We propose three simple distributed schemes for selecting the boundary nodes and compare their performance by simulation.

3.1 Boundary Detection by Localized Convex Hull

Here, we present a distributed algorithm based on localized convex hull computation to detect the boundary nodes of an event region R as shown in Fig. 2. Here, each affected node i detects all its affected neighbors and constructs a local convex hull by considering all its affected neighbors with their locations. If node i itself is one of the vertices of the convex hull, node i announces itself as a boundary node, and forwards its location to the sink. For routing, a spanning tree may be constructed in the WSN to forward data via minimum delay path as has been proposed in [17]. Initially, each node broadcasts a 'Hello' packet with its node id and location, and from the 'Hello' packets received from others it prepares neighbor list with their locations. Each node senses data at regular interval; in case it exceeds the predetermined threshold value, it broadcasts an 'Affected' message with its node id and location. From the

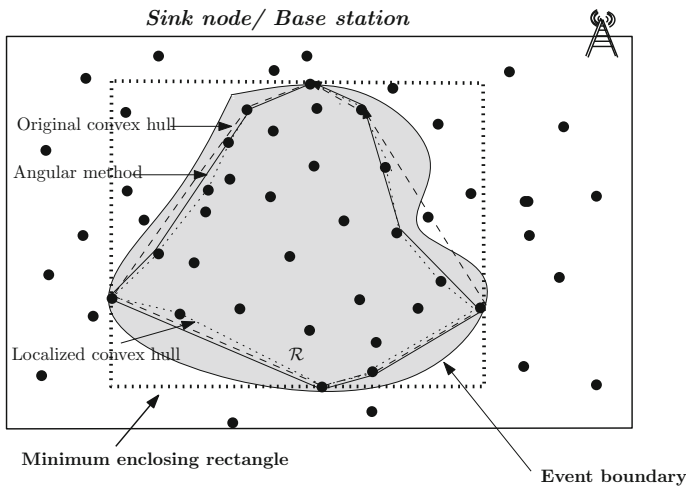


Fig. 2 Boundary detection by convex hull, minimum enclosing rectangle, and angular method

received ‘Affected’ messages from neighbors, it computes the convex hull enclosing itself and all its affected neighbors by the well-known *Jarvis March* algorithm [18]. The algorithm is the simplest one for constructing convex hull enclosing points on a two-dimensional plane with $O(h.n)$ time complexity, where h is the number of vertices of the convex hull and n is the number of points given. In real-life examples, the *Jarvis March* algorithm outperforms other convex hull algorithms when n is small or h is expected to be very small compared to n . In our case, n is limited by the maximum node degree d , and $h \leq d$, hence in the worst case, the complexity is $O(d^2)$. The space complexity of the procedure is $O(d)$ only. It is evident that with a collision-free message protocol, each node transmits only 3 messages, and the procedure terminates in 3 rounds only.

Therefore, the above procedure is simple, with $O(d^2)$ time complexity, and constant message complexity. Each node takes the decision of selection by itself based on the locations of its affected neighbors only. Also, the procedure converges in 3 rounds only. But the performance in terms of accuracy in boundary estimation is not guaranteed.

3.2 *Boundary Detection by Minimum Enclosing Rectangle*

By the most naive approach, the event area is estimated by finding the minimum rectangle enclosing all affected nodes. For this, the sink node should know the extreme co-ordinates of the affected nodes. Each affected node i knows its co-ordinates (x_i, y_i) , and sets $x_{min}(i) = x_{max}(i) = x_i$ and $y_{min}(i) = y_{max}(i) = y_i$, and broadcasts it. Next, it listens to its neighbors. Each time if it receives a packet from its neighbor j and if $x_{min}(j) < x_{min}(i)$, then $x_{min}(i) \leftarrow x_{min}(j)$ and it is broadcasted. If $y_{min}(j) < y_{min}(i)$, then $y_{min}(i) \leftarrow y_{min}(j)$, and then it is broadcasted. Similarly, if $x_{max}(j) > x_{max}(i)$, then $x_{max}(i) \leftarrow x_{max}(j)$. If $y_{max}(j) > y_{max}(i)$, then $y_{max}(i) \leftarrow y_{max}(j)$. If there is any update it is broadcasted. If any unaffected node receives any updated value of the four variables mentioned above, it broadcasts it. The procedure terminates after P rounds of communication, where P is the maximum hop count of a node in $G(V, E)$. Finally, the sink computes the minimum enclosing rectangle with x_{min} , y_{min} , x_{max} and y_{max} . Figure 2 shows an event area enclosed by the minimum enclosing rectangle. Though the computation involved is very simple, but gathering of the extreme co-ordinates of the affected nodes necessarily requires flooding in the network that in the worst case may take P rounds of communication. The computational complexity of each node is $O(P.d)$. The message complexity is $O(P)$. It needs only the information of its own location. It is clear that this approach always over estimates the area, and the convergence is rather slow. In the worst case it may take $O(n)$ rounds to complete.

3.3 Angular Boundary Detection

Finally, we propose another approach, based on the angular location of the neighbors. It is assumed that each node is equipped with directional antenna, such that when it receives a signal, it can estimate the angle of arrival. Hence, each affected node selects a subset of its affected neighbors as the reporting nodes. By *Angular boundary detection algorithm*, each affected sensor node i broadcasts a *Hello* ($i, flag = 1$) message with its id to its neighbors and listens from its neighbors ($flag = 0$ means unaffected node). If it receives a *Hello* ($j, flag = 1$) message from its neighbor j , it just includes it in its neighbor list NL with its $id, flag$ bit, and the angle of arrival θ_j .

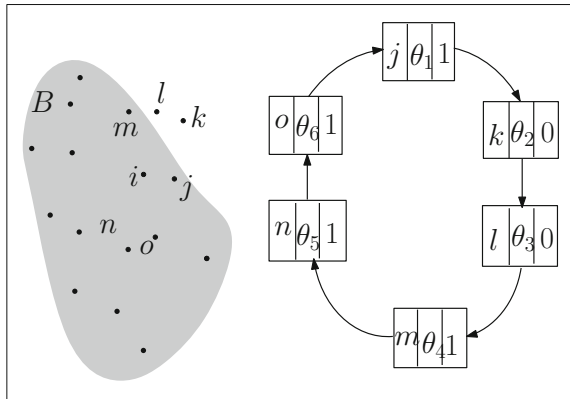
Next, each affected node i checks whether all of its neighbors are affected or not. If not, node i includes its neighbors in a circular list L in sorted order of θ_j (may be clockwise or anticlockwise) as shown in Fig. 3. Finally, node i starts to traverse L and checks for any transition from affected node to unaffected node or vice versa. If any transition is found then the affected node $j \in L$ is selected as boundary node, and it is added to a list and finally node- i broadcasts the list. Each node, if selected, forwards its location to the sink.

Algorithm 1 presents the steps formally.

Complexity Analysis:

- **Time complexity:** Each node computes the neighbor positions in terms of angles and sort them in a list L . After sorting, each node traverses the list only once. Assuming that in $G(V, E)$ the maximum node degree is d , each node, in the worst case, requires $O(d \log d)$ computation.
- **Space complexity:** Each node makes a list of neighbors NL . To make the list L , each entry consists three elements, i.e., node id, flag, and angle. If all the d neighbors get included in the list, we need $O(d)$ storage space. Hence, the space complexity is $O(d)$.

Fig. 3 Affected node i and its neighbors in L



Algorithm 1: Angular Boundary Detection

Input: Node i , $STATUS = 0$, $flag = 0$, list of neighbors NL
Output: $STATUS = 0/1$ of a node (boundary node or not)

```

for each node i do
  if an affected node then
    |  $flag \leftarrow 1$ ;
  end
  phase 1: node  $i$  broadcasts a  $hello(i, flag = 1)$  message ;
  wait and listen;
  if receives a  $hello(j, flag = 1/0)$  from its neighbor  $j$  then
    | include  $j$  in  $NL$  with its  $id, flag$  bit and angle of arrival;
  end
  Phase 2:
  for each node- $j \in NL$  do
    | if  $flag == 1$  then
    | |  $temp \leftarrow 1$ ;
    | else
    | |  $temp \leftarrow 0$ ;
    | | break;
    | end
  end
  if  $temp == 0$  then
    | for each node  $j \in NL$  do
    | | The angle of arrival  $\theta$  is included in  $L$  in sorted order;
    | end
  end
  Phase 3:  $temp \leftarrow flag$  // the flag value is the first node  $j \in L$ ;
  for each node- $j \in L$  do
    | if  $temp^! = flag$  //transition found then
    | | include affected node  $j$  in a temporary list  $L_t$  // for boundary node;
    | |  $temp \leftarrow flag$ ;
    | end
  end
  broadcasts  $selected(L_t)$  message;
  if receives  $selected(L_t)$  message then
    | if  $STATUS = 0$  then
    | | if  $i \in L_t$  and then
    | | |  $STATUS \leftarrow 1$ ;
    | | end
    | end
  end
  Terminate;
end

```

- **Message complexity:** Only two messages per node are required in the procedure, one *Hello* message and one *selected* message. Hence, per node message complexity is $O(1)$.

Example 1 Figure 3 shows an arbitrary event boundary B . In this example, node i collects angular location information from its 6 neighbor nodes and constructs a circular list by sorting the angles in anticlockwise direction as shown in Fig. 3. Now, node i starts to traverse through the list and finds transitions from node j to k and from l to m . As node j is affected, so node i declares node j as the boundary node. Similarly node i also declares node m as another boundary node.

4 Simulation Studies

For simulation study, given a $w \times w$ square area A with a random uniform distribution of n nodes, irregular-shaped event area is generated by diffusion process model following [15].

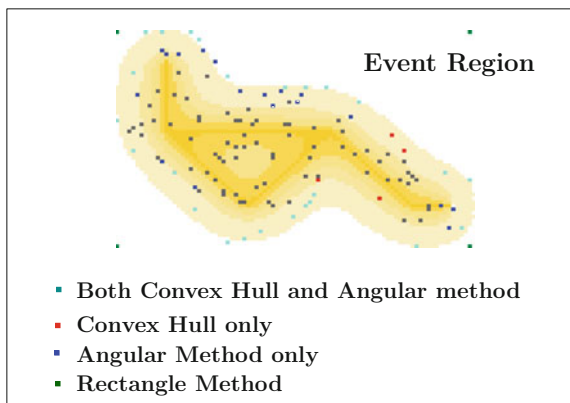
4.1 Arbitrary Event Area Generation

In [15], the event area is generated by two steps, *diffusion* and *softening*. Here, the entire area to be monitored is divided into $w \times w$ grid. Next, some grid cells are randomly chosen as *source cells* and initialized with a high data value. The cells other than the *source cells* are initialized to a fixed lower data value. In diffusion step, keeping the data of the *source cells* unaltered, the data values of all other cells are updated by the average of its four neighbor cells. After repeated application of this diffusion step, softening step is followed where the sources became non-source cells and some cells are again randomly chosen as sources except those previous cells. The process is repeated to generate the event area. In this work, we have customized this procedure to generate our event area within 200×200 grid. Here source cells are chosen randomly and they are adjacent with each other. Then the diffusion and softening steps are being carried out to generate the event area as shown in Fig. 4.

4.2 Results

For simulation, $1000 \leq n \leq 2500$ homogeneous nodes are distributed over the 200×200 region by considering a uniform random distribution. Here, different event

Fig. 4 Event area generated by diffusion model



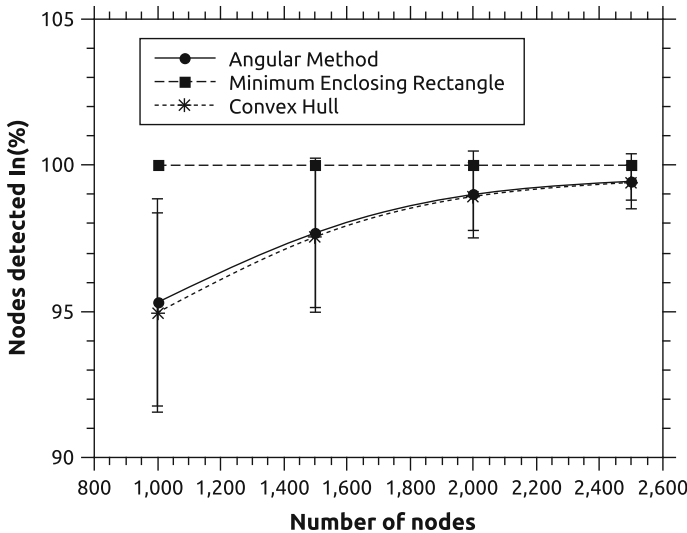


Fig. 5 Affected boundary nodes enclosed (%) versus n

regions are created by changing the source cells randomly and the experiments are repeated for different networks by varying the node set and the transmission radius. The transmission radius T varies between $6 \leq T \leq 12$. The simulation is implemented using Java 1.7.0_55 and Matlab.

Figure 5 shows how the percentage of affected nodes enclosed within the estimated area varies with n . It is evident that minimum enclosing rectangle method always encloses 100 % of affected nodes, whereas, for the other two methods, the percentage increases with n as is expected. On the other hand, Fig. 6 shows the variation of the number of unaffected nodes enclosed within the estimated boundary, termed here as *false positive*. For the rectangle method, the false detection rate is very high which will always over estimate the event area.

From the simulation studies, it is also clear that the angular method performs well even with low node density which is very suitable for real-life scenario.

Figure 7 shows that the angular boundary detection method reports small number of boundary nodes compared to the convex hull procedure, in case the node density is low. It is also evident from Fig. 8 that with small number of boundary nodes angular boundary detection method always gives better accuracy of area estimation compared to the other two methods.

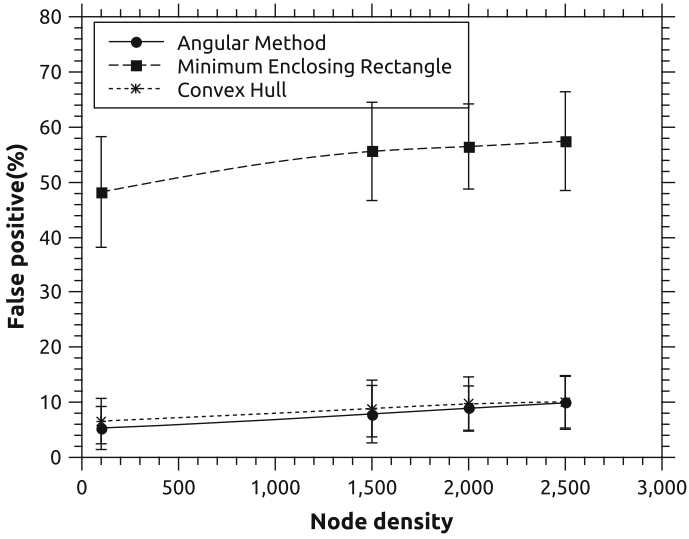


Fig. 6 Unaffected nodes enclosed (in %) versus n

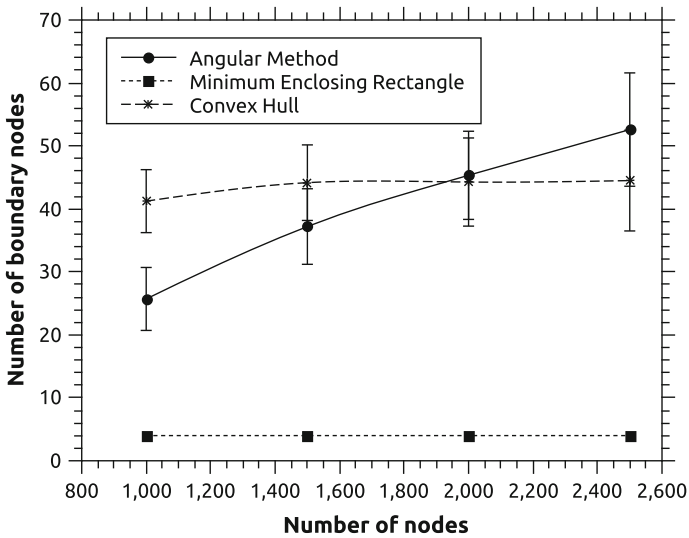


Fig. 7 Boundary nodes reported (in %) versus n

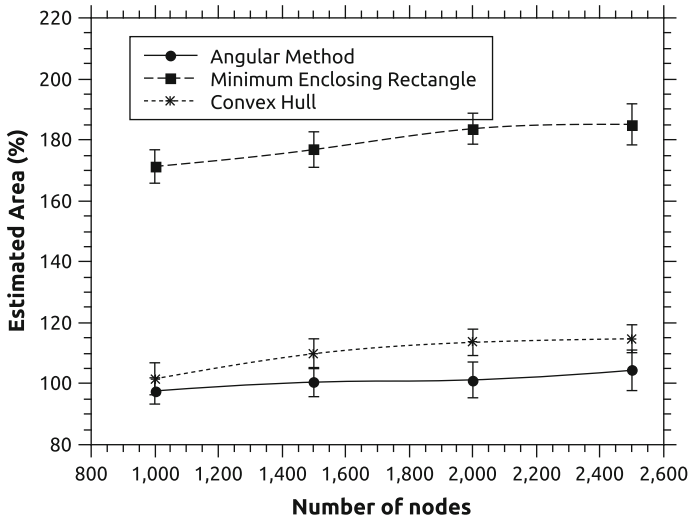


Fig. 8 Estimated area (in % of actual area) versus n

5 Conclusion and Future Work

Given a random node distribution over a bounded 2-D area, we focus on simple distributed approaches to estimate the irregular-shaped event boundary region in wireless sensor networks. We propose three algorithms, namely the (a) localized convex hull, (b) the minimum enclosing rectangle, and (c) the angular boundary detection. Complexity analysis (both time and message) and comparison studies by simulation show that the proposed angular boundary algorithm, without neighborhood location information, performs better in terms of accuracy in event boundary detection, number of reported boundary nodes, and rounds of communication.

References

1. A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, *Wireless sensor networks for habitat monitoring*, In Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications (Atlanta, Sept.). ACM Press, New York, 2002.
2. I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, *Wireless sensor networks: a survey*, Computer Networks Journal, Elsevier, March, 2002.
3. X Li, S He, J Chen, X Liang, R Lu and S Shen, *Coordinate-free Distributed Algorithm for Boundary Detection in Wireless Sensor Networks*, In Proceedings. of IEEE Globecom Houston, TX, USA Dec. 2011.
4. Y. Wang, J. Gao and J. S. B. Mitchell, *Boundary Recognition in Sensor Networks by Topological Methods*, In MobiCom'06 Los Angeles, California, USA Sep. 2006.
5. R. Ghrist and A. Muhammad, *Coverage and Hole Detection in Sensor Networks via Homology*, In 4th international Symposium on Information Processing in Sensor Networks, April 2005.

6. K. K. Chintalapudi and R. Govindan, *Localized edge detection in sensor fields*, In Proc. Ad Hoc Networks, Elsevier, 2003.
7. S. P. Fekete, A. Krolller, D. P. Fisterer, S. Fischer and C. Buschmann, *Neighborhood-Based Topology Recognition in Sensor Networks*, In Proceedings ALGOSENSORS, Springer LNCS 2004.
8. Q. Fang, J. Gao and L. Guibas, *Locating and Bypassing Routing Holes in Sensor Networks*, In Proceedings Mobile Networks and Application, Springer, 2006.
9. C. Zhang, Y. Zhang, Y. Fang, *Localized algorithms for coverage boundary detection in wireless sensor networks*, In Wireless Network, Springer, 2007.
10. T Banerjee, D Wang, B Xie and D. P. Agarwal, *PERD: Polynomial Based Event Region Detection in Wireless Sensor Network*, In Proceedings of IEEE International Conference on Communications 2007.
11. J. Bruck, et al., *Localization and routing in sensor networks by local angle information*, ACM Transaction on Sensor Networks, 2009.
12. W. C. Chu, K. F. Ssu, *Decentralized Boundary Detection without Location Information in Wireless Sensor Networks*, In Proceedings IEEE Wireless Communications and Networking Conference: Mobile and Wireless Networks, 2012.
13. B. Greenstein, E. Kohler, D. Culler, and D. Estrin, *Distributed Techniques for Area Computation in Sensor Networks*, In Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks (LCN'04).
14. S. Kundu and N. Das, *In-Network Estimation and Localization in Wireless Sensor Networks*, In Proceedings 7th IEEE International Conference, Globecom, Dec. 2012.
15. J. Lian, L. Chen, K. Naik, Y. Liu and G. B. Agnew, *Gradient Boundary Detection for Time Series Snapshot Construction in Sensor Networks*, IEEE Transaction on Parallel and Distributed Systems Oct. 2007.
16. S. Kundu and N. Das, *Event Boundary Detection and Gathering in Wireless Sensor Networks*, In Proceedings 2nd International Conference Applications and Innovations in Mobile Computing (AIMoC 2015), Feb. 2015.
17. S. Kundu, *Low Latency Event Boundary Detection in Wireless Sensor Networks*, In Proceedings International Conference on Advanced Networks and Telecommunications System (ANTS), 2015.
18. R. A. Jarvis, *On the identification of the convex hull of a finite set of points in the plane*, Information Processing Letters 2: 18–21, (1973). doi:[10.1016/0020-0190\(73\)90020-3](https://doi.org/10.1016/0020-0190(73)90020-3).