Computing and Informatics, Vol. 29, 2010, 801–821

FINDING PERIMETER OF QUERY REGIONS IN HETEROGENOUS WIRELESS SENSOR NETWORKS

Ahmed M. Khedr, Walid Osamy

Mathematics Department
Zagazig University
Zagazig, Egypt
e-mail: {amkhedr, w.osamy}@zu.edu.eg

Manuscript received 22 October 2009; revised 5 December 2009
Communicated by Ilsun You

Abstract. Some applications in wireless sensor networks (WSNs) only need to record the information of a target entering or leaving some specific regions of WSNs perimeter. One important issue in this context is to detect the perimeter of the deployed network to ensure that the sensor nodes cover the target area. In this paper we propose two distributed algorithms to elect the perimeter nodes of query regions in a WSN. We consider the most general case, where every sensor has a different sensing radius. We provide performance metrics to analyze the performance of our approach and show by simulation that the proposed algorithms give good performance.

Keywords: Boundary estimation, distributed algorithm, heterogenous sensing radius, wireless sensor networks

1 INTRODUCTION

WSNs are composed of many low cost, low power devices with sensing, local processing and wireless communication capabilities [3, 8]. Typical applications of WSNs may require a random deployment of sensor nodes over a large target area. In addition, military surveillance systems also require detection of any activities around the boundaries of the target surveillance area. Therefore, the system should be capable of detecting and identifying any object that enters or leaves the monitored area.

As energy consumption is a limiting factor for the lifetime of a node, communication has to be minimized. Upon startup, the swarm of sensors form a decentralized and self-organizing network that monitors the region. From an algorithmic point of view, the basic characteristic of a WSN requires working under a paradigm that is different from classical models: absence of a central control unit, limited capabilities of nodes, and limited communication between nodes. These require developing new algorithmic ideas that combine methods of distributed computing and network protocols with traditional centralized network algorithm. In other words, the real question is how can we use a limited amount of strictly local information in order to achieve distributed knowledge of global network properties?

A perimeter is an area enclosing given, specified characteristics. In this paper, we focus on the perimeter detection of query regions in a WSN with heterogeneous sensing ranges, where perimeter detection has a wide range of uses in several areas, including

1. military, (e.g., locating minefield or surrounding a target);
2. nuclear/chemical industries, (e.g., tracking radiation/chemical spills);
3. oceans, (e.g., tracking oil spills);
4. tracking forest fires; and
5. space, (e.g., planetary exploration) [4].

The nodes that represent the perimeter of the target area under surveillance are called perimeter nodes. Hence, the development of mechanisms by which perimeter nodes of network can be identified is an important and a challenging problem.

In this paper, we propose two distributed algorithms to elect the perimeter nodes of a query region in a heterogenous WSN. Therefore, to consider specific region inside the whole monitoring area, the end user will transmit the coordinate information of the four vertices of the rectangle to the WSN. The sensor nodes within the query region will become an independent WSN and execute our proposed approach to determine the perimeter nodes of the region. The perimeter nodes cooperate with each others to monitor the query region.

The rest of the paper is organized as follows: Section 2 describes related research on perimeter detection problem. The description of relevant terms is listed in Section 3. In Section 4, we introduce our approach to carry the proposed problem. Simulation of our approach is presented in Section 5. We conclude our work in Section 6.

2 RELATED RESEARCH

With emerging need for environmental monitoring, military surveillance and security protection, research on WSNs has recently received an increased interest [3, 8, 15, 16]. Many challenging research problems [17] are being looked upon that include scalability of the algorithms so as to incorporate a large number of nodes, design of simple and efficient schemes for different network operations, devising power-conserving protocols, defining security mechanisms, and development of exciting new applications that exploit the potential of WSN to the fullest extent. In all these applications, characteristics of the monitored area ought to be collected and analyzed.

Boundary/Edge estimation is an important data processing problem in WSNs. It is of considerable interest in detecting contours and boundaries in a scalar field, like the temperature distribution [5] or when the phenomenon evolves over time, like in scenarios with diffusion phenomena such as chemical leaks whose perimeter needs to be tracked. In the context of WSNs, edge detection can be considered as a powerful primitive which can be used as a building block for a variety of other applications. Numerous perimeter detection approaches have been proposed in the literature. The work described in [10] develops a non-localized distributed protocol that allows nodes to identify themselves as being located near the boundary of the polygonal region. The authors show that a restricted central stress is a useful index for extracting topological boundary information from a geometrically distributed WSN, provided distribution of nodes follows a random distribution. They compare several centralized measures commonly used in the analysis of social networks and show that a restricted central stress is particularly suited for geometric networks, and finally they provide mathematical as well as experimental evidence for the accuracy of this measure. In [21], Nowak and Mitra proposed an edge approximation method for a WSN, using recursive dyadic partition (RDP). In [2] the authors proposed approaches based on a localized edge detection technique for edge sensor detection; namely, the statistical approach, the filter-based approach and the classifier-based approach. The main difference between [21] and [2] is that a hierarchical network architecture is assumed and utilized in [21] whereas no hierarchy is present among sensors in [2]. Another major difference is that the real boundary has been approximated in [21] while only edge sensors were detected in [2]. Also, [2] outperforms [21] in terms of the communication cost, which is critical in WSNs.

In [7], the authors proposed a technique for boundary estimation in WSNs where observations from sensors are aggregated and confidence intervals around the true boundary are obtained for a set of points. They have also provided a distributed technique for realizing a non-parametric regression technique. They have also tackled the problem of satisfying accuracy constraints in terms of confidence interval, using the optimal number of sensors to be ON.

In [26], the authors proposed and analyzed two novel algorithms for outlying sensor identification and event boundary detection. These algorithms are purely localized and thus scale well for large WSNs. Also, they show by simulation results that these algorithms can clearly detect the event boundary and can identify outlying sensors with a high accuracy. Our approach differs in that the perimeter detection is independent of any other observations made by the sensor nodes, i.e., no recorded sensor information is used in perimeter detection. The work reported in [30] proposes a deterministic method for the boundary node detection based on localized Voronoi polygons, the technique originated from the computational geometry. The authors in [29] propose two deterministic, localized algorithms for coverage boundary detection in WSNs. Their algorithms based on two novel computational geometric techniques called localized Voronoi and neighbor embracing polygons. As compared to their previous work in [30], their algorithms outperform in terms computation and communication cost. A distributed localized algorithm for perimeter detection is introduced in [18]. Their algorithm works correctly in dense WSNs that satisfy a minimal degree of connectivity. The work in [26, 30, 29, 18] is close to ours, while our approach is based on local neighborhood information and the sensor nodes have heterogeneous sensing ranges.

Assuming a communication network follows a $\sqrt{2}/2$-quasi UDG ($\sqrt{2}/2$-QUDG) model, A. Kroller et al. [13] introduced combinatorial structures called flowers and augmented cycles in the network. This algorithm works in two stages: boundary recognition and topology extraction. The success of Kroller's design greatly depends on the identification of the flower structure, which may not always be the case, especially in a sparse network. Saukh et al. [23] extended the concept of patterns in UDG and $\sqrt{2}/2$-QUDG model to make it simple and tunable in sparse networks. These solutions come at a high cost as they need a complex combinatorial structure in a distributed manner. S. Funke [11] proposed an algorithm that finds out holes and their boundaries inside the WSN. It uses iso-contours to detect holes boundaries and the whole network boundaries. This algorithm requires more computational power than what is typically assumed of sensor nodes, and therefore could not be implemented in a distributed setting. Fang et al. [9] proposed simple greedy and distributed algorithms, the Tent rule and BoundHole for determining boundary cycles in the event that a transmission gets stuck at a node. A network node is found to follow a local minima if its neighborhood reflects geometry as defined by a tent rule. Using the tent rule, the detection of a boundary node as defined by greedy routing it local; however, a probe is required to identify all remaining nodes along the network edges. The main drawback of this algorithm is the lack of any simulation results. Moreover, to find the boundary, the use of suppressed start is required, which leads to a problem and use of enforce bit to solve this problem is not clear, i.e., when to set the enforce bit. A prerequisite in the algorithm of Zeinalipour-Yazti et al. [27] is that each node knows its own coordinate position along with the neighboring nodes and then the node that has minimum y coordinates in the sensor field is determined and marked as the starting perimeter node which then selects the neighboring perimeter node by measuring the polar angles of all the neighboring nodes on its x-axis. The line obtained by connecting the identified edge nodes is the boundary of the sensor field.

The main difference between our work and [13, 23, 11, 9, 27] is that our work provides remarkable energy savings. In addition, there are no restrictions on the communication disk model and sensing range of the sensor nodes; we also assume that the neighboring nodes can hear from each other.

In [31], the authors proposed three algorithms that use mobile sensor nodes to detect a boundary for certain phenomenon in the area. They assumed that no concentration or gradient information is known to the sensors. The algorithms enforce the sensors to cooperate and detect the boundary efficiently based on their relative locations, where mobile sensors provide their location information. An algorithm presented in [6] detects and traces the contour of a scalar field. The algorithm makes a mobile sensor node to approach a given contour. The algorithm uses local communication between the mobile node and its immediate neighbors. In [24], the authors proposed a simple distributed algorithm that correctly detects nodes on the boundaries and connects them into meaningful boundary cycles. They do not assume any knowledge of the node locations or inter-distances, and the communication graph does not follow the unit disk graph model. In our paper, we assume that the sensor nodes are static nodes and the perimeter detection is independent of any other observations. We propose a non-hierarchical decentralized localized perimeter detection algorithm that focuses on the identifying perimeter nodes with heterogenous sensing range.

In [14], Khedr et al. presented a decentralized localized algorithm where sensor nodes determine whether they are located along the perimeter of a WSN. The proposed algorithm uses the location neighborhood information in conjunction with the Barycentric technique to determine whether the sensor node is enclosed by neighboring nodes, and consequently, whether it is located within the interior of the WSN. The main difference between our work and [14] is that our work removes the restrictions on the sensing range of the sensor nodes.

3 PROBLEM OVERVIEW

In this section, we give the main assumptions, definitions and notations, and then develop the formal definition our proposed problem.

3.1 Algorithm Assumptions

Our approach relies on the following key assumptions regarding the sensor field and sensor nodes:

1. All the sensor nodes are randomly deployed in the monitoring region, while in our paper we assume that the sensor nodes are static nodes and the perimeter detection is independent of any other observations. The sensor nodes are deployed densely enough in which there is no existing isolated node and the monitoring region is fully sensing covered.
2. The communication range of wireless sensors is fixed. However, we assume that the sensing range may be larger or smaller than the communication range (i.e., heterogeneous sensing ranges).
3. *Underlying Communication Protocol:* We assume that there is an underlying protocol that takes care of all the necessary communication of information within the network.

4. *Localization:* The position of each and every node is known in any arbitrary global coordinate system, possibly by using a localization system from [20, 22, 25, 1, 19]. For simplicity, we assume that every node knows its location in space in terms of an (x, y) coordinate. The neighbors of a particular node are determined based on its radio range.
5. *Stationary Nodes:* The sensor nodes are assumed to be static. The phenomenon being sensed can be dynamic.

3.2 Definitions and Notations

Definition 1. For a sensor node S, there is a region, called *sensing region* $R(S)$, which signifies the area in which sensor S can sense a given physical phenomenon.

The *sensing range* of a sensor S indicates the maximum distance between sensor S and any point p in the sensing region of sensor S. A point p is covered or monitored by a sensor node S if the Euclidean distance between p and S is less than the sensing range of sensor S. The sensing region of node S_i can intersect with the sensing region of node S_j if the Euclidean distance between them is less than the sum of the sensing range of S_i and the sensing range of S_j. We say node S_i is a *coverage neighbor* of node S_j if and only if they have non-empty overlapping sensing region, i.e., if $R(S_i) \bigcap R(S_j) \neq \phi$; then S_i is called coverage neighbor of S_j.

Definition 2. The *communication region* of sensor s defines the area in which the sensor S can communicate with other sensor nodes directly.

The maximum distance between S_i and any other node S_j, where S_j is in *communication range* of S_i, is called the communication range of sensor S_i. Node S_i can communicate with node S_j if the Euclidean distance between them is less than the communication range of S_i. Then S_i is called a *neighbor* of S_j. The set of neighbors of S is represented by $N(S)$. Two nodes S_i and S_j can communicate directly with each other only if $S_i \in N(S_j) \bigwedge S_j \in N(S_i)$, i.e., they are neighbor to each other.

Definition 3. We define the quarter sets as the ones generated by dividing the communication space of a sensor node into four quarters, each quarter containing a set of neighboring sensor nodes.

3.3 Problem Definition

Some applications may only need to record target's entering or leaving information of specific regions rather than the precise location or the moving trajectory. Therefore, for such applications each sensor node should be able to recognize itself as a perimeter node or not, and the perimeter nodes should be elected to enclose the region. Furthermore, the elected perimeter nodes should cooperate with each other to monitor the perimeter of the region and take the responsibility to detect when the target enters or leaves. Thus, the global objective is to find out the Perimeter

Nodes (PNs) and to elect as few PNs as possible, taking into consideration that the sensing region of the PNs should overlap one by one and link with each other to enclose the region and keeping the communication cost as minimum as possible.

4 PERIMETER FINDING IN HETEROGENOUS WSNS

To find out the perimeter nodes in the border area of the randomly deployed heterogenous WSN, we propose two algorithms, namely non-cautious and cautious. Non-cautious algorithm has three phases: initial phase, election phase, and pruning phase. The second one is the cautious algorithm.

4.1 Non-Cautious Algorithm

In this section, we outline the description of each of the three phases.

4.1.1 Initial Phase

Each sensor S discovers its neighbor nodes by broadcasting its location information and sensing range along with its ID to its neighbor nodes. Then, according to the Y-coordinate, S divides its neighbors list into two sets:

1. top nodes ($TNodes$) includes the nodes that have Y-coordinate less than its Y-coordinate, and
2. down nodes ($DNodes$) includes the remaining nodes.

Then, S divides its neighbors list into two sets according to the X-coordinate:

1. left nodes ($LNodes$) includes the nodes that have X-coordinate less than its X-coordinate, and
2. right nodes ($RNodes$) includes the remaining nodes.

From the generated sets, S finds the intersection between each pair of sets (Q_1 being the intersection between $TNodes$ and $RNodes$, Q_2 being the intersection between $TNodes$ and $LNodes$, Q_3 being the intersection between $DNodes$ and $LNodes$, and Q_4 being the intersection between $DNodes$ and $RNodes$).

For more clarification, consider the following simple WSN in Figure 1. Sensor nodes are uniquely identifiable and each node location specified as ordered pair of (x, y) coordinates.

	$DNodes$	$TNodes$	$RNodes$	$LNodes$
Node 6	4, 5	2, 7	2, 7, 4	5
Node 7	6, 4, 5	2, 3	3, 4	2, 6, 5

Table 1. $DNodes$, $TNodes$, $RNodes$ and $LNodes$ sets for node 6 and 7

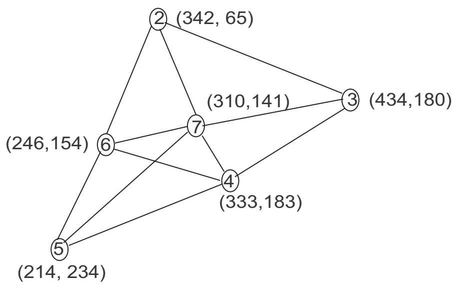


Fig. 1. Wireless sensor network

The generated sets $TNodes$, $DNodes$, $LNodes$, and $RNodes$, are shown in Table 1 for nodes 6 and 7. Table 2 shows the quarter Q_1, Q_2, Q_3, and Q_4 for nodes 6 and 7.

	Q_1	Q_2	Q_3	Q_4
Node 6	2,7	Ø	5	4
Node 7	3	2	6, 5	4

Table 2. Quarter sets for node 6 and 7

S directly declares itself as perimeter node (PN), if it has one or more empty quarter sets, and it considers itself as interior node if it has nonempty quarter sets. In the election phase, each PN will elect the necessary neighbors to be the new PNs and to complete its perimeter information.

4.1.2 Election Phase

If the perimeter information of PN is not complete, PN collaborates with its neighbors to elect the necessary neighbors to be the new PNs by executing the following steps:

1. Count the declaration messages from your 1-hop neighbors. If your count is less than or to equal one, or if your neighboring list belongs to the same quarter, decide that the network perimeter information is not completely discovered.

   In this case, PN elects a new PN node from its neighbors (PNN); the election process goes at every PN_j as follows:

   (a) Broadcast the *RequiredToCompletePerimeter* message to your neighbor nodes containing your ID.
   (b) Wait for the *RequiredSupport* messages from PNNs.
   (c) Among the *RequiredSupport* messages from PNNs, elect the the list of PN_is that belong to your quarter and have minimum calculated distance.
   (d) Send support message to the elected PNN to be the new PN.

The PNN_k that receives the *RequiredToCompletePerimeter* from PN_j executes the following steps:

1. Check whether the following conditions are satisfied for each PN_i in your list:
   (a) Both PN_i and PN_j do not belong to same quarter.
   (b) $R(PN_i) \bigcap R(PNN_k) \neq \phi$ and $R(PNN_k) \bigcap R(PN_j) \neq \phi$
   (c) $PN_i \in N(PNN_k)$ and $PN_j \in N(PNN_k)$.
2. If the above conditions are satisfied, PNN_k calculates the vertical distance (vd) to the line between PN_j and PN_i:
   (a) Send *RequiredSupport* message contains its ID, vd, and PN_i ID to PN_j,
   (b) Wait for support message.

In this way, PN_j will complete its perimeter information with PN_i.

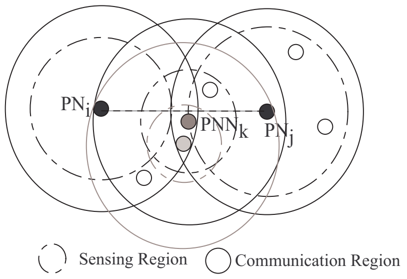


Fig. 2. New PN election process, PNN_k has minimum distance from the virtual line between PN_i and PN_j

For more clarification, in Figure 2, both light, and dark gray nodes have sensing overlapping with PN_i and PN_j. They response to *RequiredToCompletePerimeter* message from PN_j. PN_j supports the dark gray (PNN_k) to be the new PN, because it has the minimum distance from the joining line.

After the election phase, some redundant PNs may exist and require to be retired and return to be PNNs to save energy (pruning phase).

4.1.3 Pruning Phase

The main task of this phase is to retire the redundant PNs and return them to be PNNs for saving energy. Each PN's sensing region must be intersected with two PNs at different directions. This phase is divided into two main steps. In the first step, each PN checks the information of its neighboring list to make sure that there is no need to elect more new PNs and, in the second step, each PN decides to retire or not.

Step One: if you have at least two PNs at different directions, execute step two, otherwise return to the election phase.

Step Two: In each one of your quarters, elect the PN that has the largest sensing range. Then check whether the elected PNs are coverage neighbors; retire and return to be a PNN and declare yourself as a forward node between the elected PNs; otherwise remain a PN.

4.2 Cautious Algorithm

In the cautious algorithm, the sensor nodes that have nonempty quarter sets are called interior nodes. Each sensor S that has one or more empty quarter sets, broadcasts its ID, sensing range, and its total number of empty sets to its neighbors. Then, S declares itself as PN, if one of the following cases is satisfied:

- Case one: If S has the largest number of empty quarter sets among its neighbor nodes
- Case two: If S has the largest sensing range among its neighbor nodes; and its empty quarter sets equal to the maximum empty quarter set of its neighbors
- Case three: If S has the largest ID among its neighbor nodes and its sensing range equals to the maximum sensing range of its neighbors.

Once the PNs declare themselves, they will work as initiators and run the perimeter connectivity/coverage discovery procedure (*Next_PN_Procedure*) to complete the discovery steps. In order to reduce the number of messages in the discovery process, we restrict the number of involved sensors by using the following two techniques:

- First, based on predefined cases, we elect a sensor node that will initiate the discovery process.
- Second, we restrict the discovery process to be in one direction (left direction) by considering the set of coverage neighbors that are in the left direction, and then from that set we elect the left-most node to be the new PN.

4.2.1 Next_PN_Procedure

Here, we explain in detail the perimeter connectivity/coverage discovery procedure. Each sensor node is assumed to have the information of its coverage neighbors. Each PN (PN_j) executes the following steps:

1. PN_j assigns virtual sliding window centered at PN_j, with window size equal to the PN_j sensing range
2. PN_j dividing the window into left and right parts and PN_j considers only the left window part
3. ***exit*** if there is a PN node in the left window part

Algorithm 1 Non-Cautious Algorithm

```
1: Initially: If you have one or more empty quarter sets, declare yourself as
   a perimeter node ($PN$), otherwise declare yourself as an inner node.

   $PN$ Side: {The following code will be executed by $PN$s}
2: Count your $PN$ neighbors.
3: if (the count of $PN$ neighbors $\leq 1$) OR (all $PN$ neighbors at the same quarter)
   then
4:   Send RequiredToCompletePerimeter message to neighbors containing your ID.
5: else
6:   Elect the $PN$ neighbor which has the largest sensing range among your $PN$
     neighbors.
7:   if The elected $PN$s are sensing neighbors to each others then
8:     Retire and return to be $PNN$,
9:      Declare yourself as a forward node between the elected $PN$s
10:  end if
11: end if
12: if RequireSupport messages received then
13:   For each $PN_i$ that received in require to support messages.
14:   Elect the $PNN$ node that has minimum distance $vd$,
15:   Send support message to the elected $PNN$ to be new $PN$.
16: end if

   $PNN$ Side: {The following code will be executed by $PNN$s}
17: if RequiredToCompletePerimeter messages received from $PN_j$ then
18:   for each $PN_i$ in your neighbor list do
19:     if ($PN_i$ and $PN_j$ are not at the same quarter) AND ($R(PN_i) \bigcap R(PNN_k) \neq$
        $\phi$ and $R(PNN_k) \bigcap R(PN_j) \neq \phi$) And ($PN_i \in N(PNN_k)$ and $PN_j \in$
        $N(PNN_k)$) then
20:       Calculate the minimum distance $vd$ to the line between $PN_j$, and $PN_i$
          {this help $PN_j$ to elect the nearest $PNN$ to the line joining $PN_j$ and
          $PN_i$ }.
21:       Send RequireSupport message containing your ID, the calculated distance
          $vd$, and $PN_i$ ID to $PN_j$.
22:     end if
23:   end for
24: end if
25: if Support message received then
26:   Declare yourself as $PN$ to complete the perimeter information between $PN_j$
      and $PN_i$.
27: end if
```

Algorithm 2 Cautious Algorithm

```
1: Initially: Each sensor node $S$ in query region determines its quarter sets.
2: Compute $S$.NoEmptySet (the number of empty quarter sets)
3: Broadcast $S$.NoEmptySet, $S$.ID, and $S$.Rs
4: Determine the maximum of $NoEmptySet$ ($MaxNoEmptySet$) of your neighbors.
5: if $S.NoEmptySet > MaxNoEmptySet$ then
6:   Declare yourself as $PN$ node
7:   Call Next_PN_Procedure ($S$)
8: else
9:   if $S.NoEmptySet = MaxNoEmptySet$ then
10:     Determine the maximum sensing range of your neighbors $MaxRs$
11:     if $S.Rs > MaxRs$ then
12:       Declare yourself as $PN$ node
13:       Call Next_PN_Procedure ($S$)
14:     else
15:       if $S.Rs = MaxRs$ then
16:         Determine the maximum ID ($MaxID$) of your neighbors;
17:         if $S.ID = MaxID$ then
18:           Declare yourself as $PN$ node
19:           Call Next_PN_Procedure ($S$)
20:         end if
21:       end if
22:     end if
23:   end if
24: end if
```

4. otherwise, PN_j elects the left-most node as new center of the window then send 'Notify' message to it
5. each sensor node receives 'Notify' message from PN_j and declares itself as PN node and repeats what PN_j did before.

5 SIMULATION RESULTS

A simulator has been designed to evaluate the performance of our proposed algorithms. In our simulation, the sensor nodes are randomly deployed and uniformly distributed in a two dimensional plane. In order to carry out our experiments, we have taken the following performance metrics:

Energy consumption is measured by the total number of the network energy dissipation that was used to discover perimeter sensor nodes.

Number of perimeter nodes are the selected nodes to be close to the monitoring region.

Algorithm 3 Next_PN_Procedure

```
1: Input: sensor node $S$
2: $CN(S)$= Coverage Neighbors of $S$
3: $LCN(S)$ = Left Coverage Neighbors of $(S)$ {the set of Coverage Neighbors that
   are successors of $S$ on the perimeter}
4: if Notify message received then
5:     $S$ declare itself as PN node;
6: end if
7: if $\exists$ PN node $\in LCN(S)$ then
8:     End Procedure;
9: end if
10: $S_j$ = Left-most node($LCN(S)$)
11: $S$ send Notify message to $S_j$ to declare itself as PN node and run
    Next_PN_Procedure $(S_j)$
```

Communication overhead is measured by the total number of messages used in discovering perimeter sensor nodes.

In our simulation, we consider the following tunable parameters:

- The number of deployed sensor nodes varies from 300 to 700 nodes with an increment of 100 nodes.
- The communication range (R_c) of every sensor node is fixed at 60 m,
- Each sensor has P sensing ranges (R_s) where $P = 1, 2, \ldots, 5$ with values varying from 30 m to 70 m; the case with P sensing ranges allows each sensor to choose sensing ranges $R_{S_1} = 30, \ldots, R_{S_P} = 70$. The case when $P = 1$ is the case when all sensor have a fixed sensing range (R_s) with value = 30 m; in this case, communication radius is larger than or equal to double the sensing radius (i.e., $R_c \geq 2R_s$), the specification of $R_c \geq 2R_s$ as pointed out in [28] it is enough to ensure the network connectivity as long as the network region is completely covered; this specification holds for most commercially available sensors [32].

In order to measure the energy dissipation of nodes, we use the same energy parameters and radio model as discussed in [12], wherein energy consumption is mainly divided into two parts: receiving and transmitting messages. The transmission energy consumption needs additional energy to amplify the signal, depending on the distance to the destination. Thus, to transmit a k-bit message to a distance d, the radio costs will be

$$E_{Tx}(k,d) = \begin{cases} kE_{\text{elec}} + k\epsilon_{fs}d^2 & d < (\epsilon_{fs}/\epsilon_{mp}) \\ kE_{elec} + k\epsilon_{mp}d^4 & d \geq (\epsilon_{fs}/\epsilon_{mp}). \end{cases} \tag{1}$$

and to receive this message, the radio costs will be

$$E_{Rx}(k) = k * E_{\text{elec}}. \tag{2}$$

The simulated model parameters are set as $E_{elec} = 50\,\mathrm{nJ/bit}$, $\epsilon_{fs} = 10\,\mathrm{pJ/bit/m^2}$, $\epsilon_{mp} = \frac{13}{10\,000}\,\mathrm{pJ/bit/m^4}$ and the initial energy per node $= 2\,\mathrm{J}$. Moreover, we assume the whole WSN to be the query region and evaluate the performance of the query over the WSN. To evaluate the performance of our algorithms in terms of predefined metrics, we compare our algorithms with the Perimeter Algorithm (PA) proposed by Demetrios Zeinalipour [27], considering the case where the network perimeter is required.

In the first experiment, we compare the network energy dissipation of cautious, non-cautious and PA algorithms, where we consider the sensing range at $P = 1$, and vary the number of sensors between 300 and 700 with increment of 100. Figure 3 shows that energy consumption in our algorithms is less than the consumption energy of PA. This is due to the fact that in our algorithms, each node can determine whether it is a perimeter node or not. We can also observe that the difference between the three algorithms increases with the size of the network area. In general, it is obvious that our approach scales very well, fits large scale networks and can achieve remarkable energy saving.

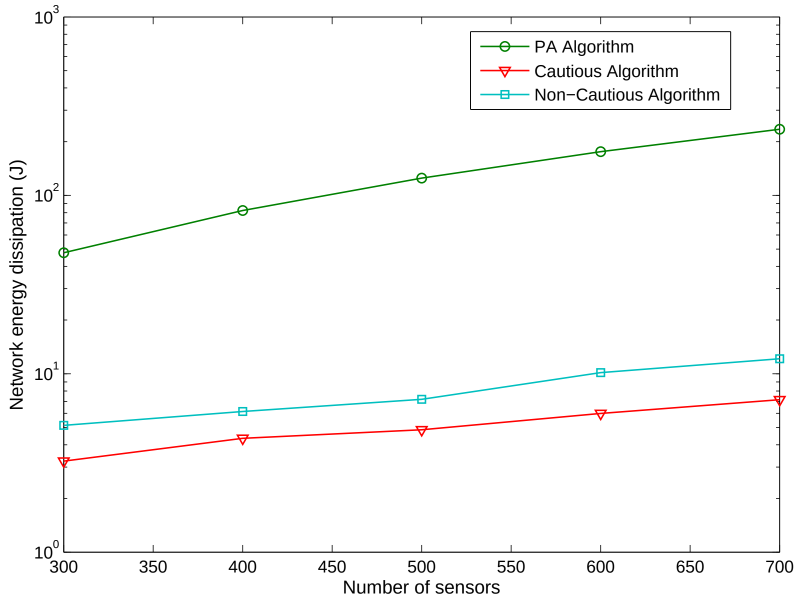


Fig. 3. Network energy dissipation of cautious, non-cautious and PA algorithms

Figure 4 shows the number of elected perimeter nodes by our approach and by the PA algorithm. We consider the sensing range at $P = 1$, and vary the number of sensors between 300 and 700 with increment of 100. The monitoring region is assumed to be 300 m × 300 m, and the communication range is fixed at 60 m. The simulation result demonstrates that when we increase the number of nodes in the network, the number of elected perimeter nodes is decreased in case of the cautious algorithm. This happens because the cautious algorithm can always select the sensor with the greatest contribution.

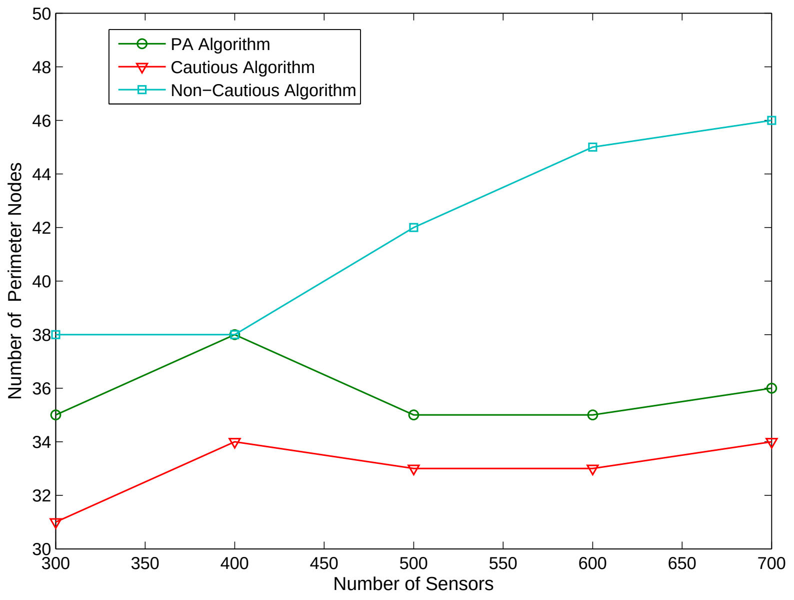


Fig. 4. Number of perimeter nodes by non-cautious, cautious and PA algorithms

We also observe that the number of elected perimeter nodes by non-cautious algorithm is higher than the number of elected perimeter nodes by PA or by the cautious algorithms. Since the optimal way to deploy the nodes that have the same sensing radius to achieve the border coverage of the region of interest is to deploy the nodes across the perimeter of the entire region such that any two adjacent nodes are tangent to each other and their sensing region centers lie on the perimeter of the entire region. Therefore, if we assumed that the region to be monitored is large in comparison to the sensing region of an individual node, then the number of nodes (n) required to cover the perimeter of the entire region equals to the perimeter length of the entire region divided by the diameter of the node sensing region, i.e.,

$$n = \frac{L}{2 * R_s}, \tag{3}$$

where L is the perimeter length of the entire region and R_s is the sensing radius of the node. From Equation (3), the lower bound to cover the perimeter of 300×300 network with sensor node with $R_s = 30$ will be 20 nodes. According to our simulation results, Figure 4 shows the final returned perimeter nodes using our algorithms and PA algorithm above the lower bound, we can also observe that our cautious algorithm has better performance than non-cautious and PA algorithms.

We compare the communication overhead in non-cautious, cautious and PA algorithms and find that the communication overhead of our approaches are slightly increased with increasing number of nodes. As shown in Figure 5, the percentage of average communication overhead of non-cautious algorithm is more than with the cautious algorithm. We can also observe that the communication overhead of our algorithms is less than that of the PA algorithm.

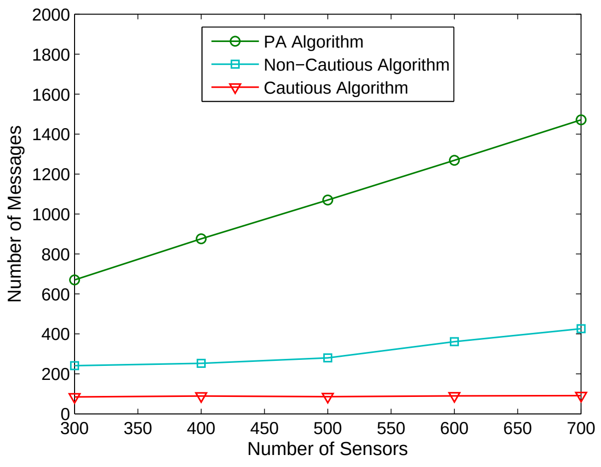
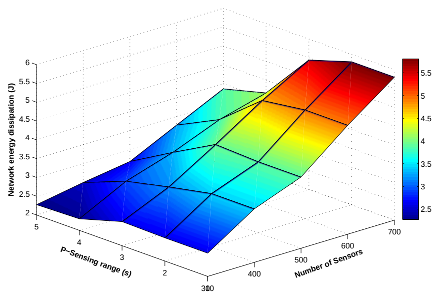


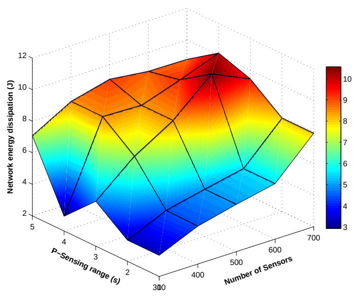
Fig. 5. Number of messages overhead in non-cautious, cautious and PA algorithms

In the second experiment, we study the impact of heterogenous sensing ranges with the number of sensors on our predefined metrics. We consider the sensing range vary from $P = 1$ to $P = 5$; then each node will elect its R_s ($R_s = 30$ when $P = 1$, $R_s = 20$, or 30 when $P = 2$, $R_s = 20$, 30, or 40 when $P = 3$, $R_s = 20$, 30, 40, or 50 when $P = 4$, and $R_s = 20$, 30, 40, 50, 60, or 70 when $P = 5$), and we vary the number of sensors between 300 and 700 with increment of 100. The monitoring region is assumed to be 300 m $\times$ 300 m, and the communication range is fixed at 60 m.

Figures 6 and 7 indicate that increasing P implies decreasing of the network energy dissipation and of the number of messages in cautious algorithm, and increasing network energy dissipation and the number of messages in non-cautious algorithm. Also, increasing the number of deployed nodes implies increasing of the network energy dissipation and the number of messages. This is due to the fact that number

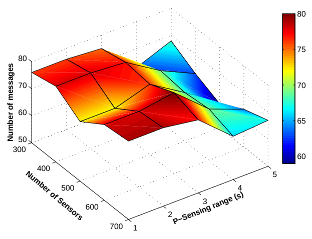


a)

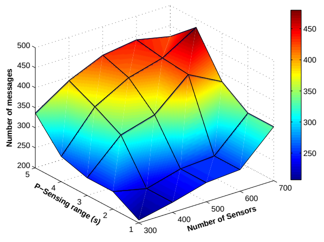


b)

Fig. 6. Network energy dissipation related to the number of sensors and P-sensing range(s) a) for cautious algorithm, and b) for non-cautious algorithm

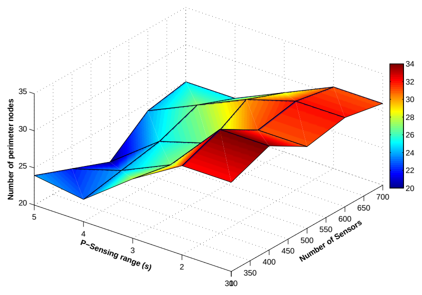


a)

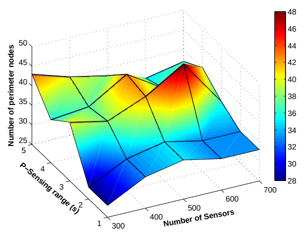


b)

Fig. 7. Number of messages overhead related to the number of sensors and P-sensing range(s) a) for cautious algorithm, and b) non-cautious algorithm



a)



b)

Fig. 8. Number of perimeter nodes related to the number of sensors and P-sensing range(s) a) for cautious algorithm, and b) for non-cautious algorithm

of neighbors for each node will be increased. This is attributed to the localized and distributed characteristics of our algorithms. The results indicate a highly scalable nature of our algorithms.

Comparing the cautious and non-cautious algorithms, the cautious algorithm has a better performance in terms of network energy dissipation and the number of message overhead than the non-cautious algorithm. This is because the cautious algorithm each time elects a suitable perimeter node towards the network perimeter.

Figure 8 demonstrates that increasing number of nodes in the network implies decreasing the number of elected perimeter nodes in cautious algorithm. This is due to increasing the probability of electing nodes with higher sensing range, where the monitoring region can be enclosed by fewer perimeter nodes with larger sensing ranges.

Varying the number of available sensing ranges P, the simulation results indicate that increasing P implies decreasing number of perimeter nodes produced by the cautious algorithm and increasing the number of perimeter nodes in case of non-cautious algorithm; this happened because in its start the non-cautious algorithm elects the perimeter nodes according to empty quarter sets ignoring node's sensing range while the cautious algorithm elects each time the sensor node with the greatest contribution to be the perimeter nodes.

These simulation results also show that heterogeneous sensing range can greatly contribute to decreasing the network energy dissipation.

6 CONCLUSION

In this paper, we have proposed two distributed algorithms (cautious and non-cautious) to elect the perimeter nodes of the query region in heterogenous WSNs. Each algorithm divides the communication space of the sensor node into four quarters, each quarter containing a set of neighbor nodes. Comparing our two algorithms, the cautious algorithm has lower communication overhead, elects fewer perimeter nodes, and has much better performance in terms of network energy dissipation than the non-cautious algorithm. This is due to the fact that the cautious algorithm each time elects the suitable perimeter node toward the network perimeter. The simulation results also indicate a highly scalable nature of our algorithms showing that heterogeneous sensing range can greatly contribute to decreasing the network energy dissipation.

REFERENCES

[1] Albowitz, J.—Chen, A.—Zhang, L.: Recursive Postion Estimation in Sensor Networks. ICNP '01, 2001.

[2] Chintalapudi, K.—Govindan, R.: Localized Edge Detection in Sensor Fields. IEEE International Workshop on Sensor Network Protocols and Applications, pp. 59–70, May 2003.

[3] Chong, C.-Y.—Kumar, S. P.—Hamilton, B. A.: Sensor Networks: Evolution, Opportunities, and Challenges. Proceedings of the IEEE, Vol. 91, 2003, No. 8, pp. 1247–1256.

[4] Clark, J.—Fierro, R.: Mobile Robotic Sensors for Perimeter Detection and Tracking. ISA Transactions, Vol. 46, 2007, No. 1, pp. 3–13.

[5] Dantu, K.—Sukhatme, G. S.: Contour Detection in Actuated Sensor Networks. Poster, First ACM Conference in Embedded Networked Sensor Systems (SenSys), 2003.

[6] Dantu, K.—Sukhatme, G. S.: Boundary Detection Using Actuated Sensor Networks. Center for Embedded Network Sensing, Posters, Paper 11, `http://repositories.cdlib.org/cens/Posters/11`, January, 2003.

[7] Duttagupta, S.—Ramamritham, K.—Ramanathan, P.: Distributed Boundary Estimation using Sensor Networks. Mobile Adhoc and Sensor Systems (MASS), 2006 IEEE International Conference, Vancouver, Oct. 2006, pp. 316–325.

[8] Estrin, D.—Govindan, R.—Heidmann, J.—Kumar, S.: Next Century Challenges: Scalable Coordination in Sensor Networks. In Proceedings of the ACM/IEEE International Conference on Mobile Computing and Networking, Seattle, Washington (USA), August 1999, pp. 263–270.

[9] Fang, Q.—Gao, J.—Guibas, L. J.: Locating and Bypassing Routing Holes in Sensor Networks. INFOCOM '04, 2004.

[10] Fekete, S. P.—Kaufmann, M.—Kroller, A.—Lehmann, N.: A New Approach for Boundary Recognition in Geometric Sensor Networks. In Proceedings of 17th Canadian Conference on Computational Geometry, 2005, pp. 82–85.

[11] Funke, S.: Topological Hole Detection in Wireless Sensor Networks and its Applications. The 3rd ACM/SIGMOBILE International Workshop on foundations of Mobile Computing, DIAL-M-POMC, 2005.

[12] Heinzelman, W.—Chandrakasan, A.—Balakrishnan, H.: An Application-Specific Protocol Architecture for Wireless Microsensor Networks. IEEE Trans. Wireless Comm., Vol. 1, 2002, No. 4, pp. 660–670.

[13] Kroller, A.—Fekete, S. P.—Pfisterer, D.—Fischer, S.: Deterministic Boundary Recognition and Topology Extraction for Large Sensor Networks. Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2006.

[14] Khedr, A. M.—Osamy, W.—Agrawal, D. P.: Perimeter Discovery in Wireless Sensor Networks. J. Parallel Distrib. Comput., Vol. 69, 2009, pp. 922–929.

[15] Khedr, A. M.—Osamy, W.: Tracking Mobile Targets Using Random Sensor Networks. The Arabian Journal for Science and Engineering, Vol. 32, 2007, pp. 301–315.

[16] Khedr, A. M.: New Mechanism for Tracking a Mobile Target Using Grid Sensor Networks. Computing and Informatics, Vol. 27, 2008, No. 6, pp. 853–874.

[17] Krishnamachari, B.—Iyengar, S.: Efficient and Fault Tolerant Feature Extraction in Sensor Networks. 2nd International Workshop on Information Processing in Sensor Networks, April 2003.

[18] Martincic, F.—Schwiebert, L.: Distributed Perimeter Detection in Wireless Sensor Networks. In: Wayne State University Technical Report, WSU-CSC-NEWS/03-TR03, `http://newslab.cs.wayne.edu/perimeter.pdf`, July 2004.

[19] Moses, D. K.—Patterson, R.: A Self-Localization Method for Wireless Sensor Networks. Eurasip Journal on Applied Signal Processing, Special Issue on Sensor Networks, October 2002.

[20] Niculescu, D.: Ad Hoc Positioning System. Proceedings of CLOBECOM, San Antoino, November 2001.

[21] Nowak, R.—Mitra, U.: Boundary Estimation in Sensor Networks: Theory and Methods. 2nd International Workshop on Information Processing in Sensor Networks, April 2003, pp. 22–36.

[22] Sawides, A. C. C.—Srivastava, M.: Dynamic Fine-Grained Localization in Ad-Hoc Networks of Sensors. Proceedings of the ACM/IEEE International Conference on Mobile computing and networking, Rome, July 2001.

[23] Saukh, O.—Sauter, R.—Gauger, M.—Marrón, P. J.: On Boundary Recognition without Location Information in Wireless Sensor Networks. In Proceedings of the 7th international Conference on information Processing in Sensor Networks, Information Processing In Sensor Networks. IEEE Computer Society, Washington, DC, pp. 207–218, `http://dx.doi.org/10.1109/`, IPSN.2008.11, April 22–24, 2008.

[24] Wang, Y.—Gao, J.—Mitchell, J. S. B.: Boundary Recognition in Sensor Networks by Topological Method. In Proc. of the ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom), September 2006.

[25] Wong, K. F. S.—Tsang, I. W.—Cheung, S.—Chan, H. G.—Kwok, J. T.: Position Estimation for Wireless Sensor Networks. Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM 2005), St. Louis, MO, USA, postscript: `http://www.cs.ust.hk/~jamesk/papers/globecom05.pdf`, November 2005.

[26] Wu, W.—Cheng, X.—Ding, M.—Xing, K.—Liu, F.—Deng, P.: Localized Outlying and Boundary Data Detection in Sensor Networks. IEEE Transactions on Knowledge and Data Engineering, Vol. 19, 2007, No. 8, pp. 1145–1157.

[27] Zeinalipour-Yazti, D.—Andreou, P.—Chrysanthis, P.—Samaras, G.: SenseSwarm: A Perimeter-based Data Acquisition Framework for Mobile Sensor Networks. 4th Intl. Workshop on Data Management for Sensor Networks DMSN DMSN '07 (with VLDB '07), Vienna, Austria, July 2007.

[28] Zhang, H.—Hou, J. C.: Maintaining Sensing Coverage and Connectivity in Large Sensor Networks. International Journal of Wireless Ad Hoc and Sensor Networks, Vol. 1, 2005, No. 1–2, pp. 89–124.

[29] Zhang, C.—Zhang, Y.—Fang, Y.: Localized Algorithms for Coverage Boundary Detection in Wireless Sensor Networks. Wireless Networks, `http://www.springerlink.com/content/k140t222737804r3`, 23 February 2007.

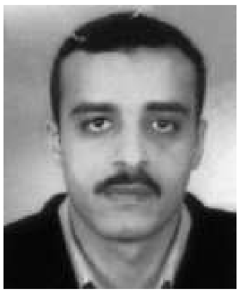
[30] Zhang, C.—Zhang, Y.—Fang, Y.: Detecting Coverage Boundary Nodes in Wireless Sensor Networks. Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611, USA., Networking, Sensing and Control, 2006,

ICNSC '06. Proceedings of the 2006 IEEE International Conference, April 23–25, 2006, pp. 868–873.

[31] Zhang, R.—Wang, J.: Contour Estimation using Collaborating Mobile Sensors. Electrical Engineering Department, University of California, Los Angeles, http://nesl.ee.ucla.edu/courses/ee206a/2004s/submissions/project/BoundaryTracking/FinalReport.pdf, 2004.

[32] http://www.xbow.com/Home/wHomePage.aspx.

Ahmed M. Khedr received his B. Sc. degree in mathematics in June 1989 and the M. Sc. degree in optimal control in July 1995, both from Zagazig University, Egypt. In July 1999 he received his M. Sc. and in March 2003 he received his Ph. D. degree, both in computer science and engineering, from University of Cincinnati, Ohio, USA. From March 2003 to January 2004, he was a Research Assistant Professor at ECECS Department, University of Cincinnati, USA. From January 2004 to May 2009, he worked as Assistant Professor at Zagazig University, Egypt, and from June 2009 till now he is working as Associate Professor at the Department of Computer Sciences, College of Computers and Information Systems, Taif University, KSA. In June 2009, he was awarded the State Prize of Distinction in Advanced Technology. He has co-authored 32 works in journals and conferences related to optimal control, wireless sensor networks, decomposable algorithms, and bioinformatics.



Walid Osamy received his B. Sc. degree in 2000 and his M. Sc. degree in 2006, both from the Department of Mathematics, Faculty of Science, Zagazig University, Egypt. He has been involved in the projects on Network Infrastructure, telecom, and Managment information systems at Communication Information Technology Center (CITC), Zagazig University. He is currently a Ph. D. student at the Faculty of science, Zagazig University. His research interests include computer graphics, computing, and wireless sensor networks.