Robotics and Autonomous Systems 60 (2012) 266–277

Contents lists available at SciVerse ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

Perimeter detection in wireless sensor networks

Kyle Luthy [a], Edward Grant [a], Nikhil Deshpande [a], Thomas C. Henderson [b,*]

[a] North Carolina State University, United States
[b] University of Utah, United States

ARTICLE INFO

Article history:
Received 24 July 2010
Received in revised form
12 October 2011
Accepted 18 October 2011
Available online 18 November 2011

Keywords:
Wireless sensor networks
Boundary detection
Coverage hole
Convex hull
Robotic repair

ABSTRACT

For a mobile robotic agent to bridge the gaps between disconnected networks, it is beneficial for the robot to first determine the network coverage boundary. Several techniques have been introduced to determine the boundary nodes of a network, but the correctness of these techniques is often ill-defined. We present a technique for obtaining boundary node ground truth from region adjacency analysis of a model-based image created from a network graph. The resulting ground truth baseline is then used for quantitative comparison of several boundary detection methods including a local application of the image region adjacency analysis and the computation of the local convex hull with the addition of a perturbation value to overcome small boundary concavities in the node location point set. Given our proposed metrics of the techniques evaluated, the perturbed convex hull technique demonstrates a high success rate for boundary node identification, particularly when the convex hull is formed using two-hop neighborhoods. This technique was successfully implemented on a physical 25-node network, and the performance of this network implementation is evaluated.

1. Introduction

As demonstrated in [1], a mobile robotic agent can be used to fill in the gaps between networks that are spatially disconnected. This is accomplished by tracking nodes at a distance specified by the strength of its communication link. Doing so ultimately results in the robot following the communication boundary of a subnetwork and "plugging the gap" when it comes in contact with another subnetwork. Earlier research on this problem showed that this technique, while effective, could be improved if topological information was gathered from the network, e.g., determining which nodes were the network perimeter nodes. Perimeter information resulted in a shortened repair time in 66% of the simulated trials conducted, with an average distance improvement of 60.6%. Considering all trials produces an average distance reduction of 23%. Fig. 1 is an example of a network repair without its perimeter labels. In this figure, each node is presented as a letter surrounded by its communication disk, where connectivity is denoted by letter and color. The perimeter repair path is the bold line where the diamond symbol (◇) is the randomly chosen starting point and the star (⋆) is the repair point. As shown by the path line, the robot spends a considerable amount of time searching within the network before it reaches the perimeter.

There are two general perspectives taken on boundary node detection in wireless sensor networks (WSNs). The first determines the boundary based on the event being sensed, e.g., tracking a fire or tracking a chemical plume. Examples of this type of boundary determination can be found in [2–8]. The second perspective used for determining boundaries of WSNs uses the topology of the network itself. In this approach, the network topology is used to identify which nodes lie on the physical boundary of the network. It is this second view that is used in the research reported upon in this paper. Finding boundaries based on topology is more appropriate in this research because the goal is to extend the sensing range of the network to detect far away events, or events that are undetectable at the limits of the current boundary.

This paper presents and evaluates several techniques for boundary node detection. More importantly, however, it provides correctness measures for comparing boundary detection techniques. We introduce a model-based method of perimeter and coverage hole identification based on image region detection. While not feasible for network implementation due to memory constraints, it provides a ground truth baseline against which coverage boundary detection algorithms are tested.

The remainder of this paper is organized as follows. Section 2 provides an overview of the literature for network boundary detection in wireless sensor networks. Section 3 defines the problem in terms of sensory or communication models, and details our primary contribution, the model-based image technique for obtaining true boundary nodes. Several techniques are introduced and analyzed in Section 4. These methods are: (1) a localized

* Corresponding author.
E-mail address: tch@cs.utah.edu (T.C. Henderson).

0921-8890/$ – see front matter
doi:10.1016/j.robot.2011.10.004

Table 1
List of definitions.

Ground truth	Refers to the nodes that are true perimeter node; the standard against which techniques are tested
Image region adjacency	Differentiation of a region in an image with one label (color) from the other regions or the image background
Hop-contour/iso-contour	The nodes of a network that are the same number of hops from a beacon node
Degree	The number of connections, or one-hop neighbors of a node
Recall	When conducting a search, the percentage of correct or relevant results returned: $(actual \cap results)/actual$
Precision	The percentage of search results that are also correct or relevant: $(actual \cap results)/results$
Uniform random distribution/network	It is equally likely that a node will be placed at any point within the deployment region
Normal error	The error is random over a Gaussian distribution of mean, μ, and standard deviation, σ
Convex hull	The minimal number of points in a graph (node locations) within which all other nodes are contained

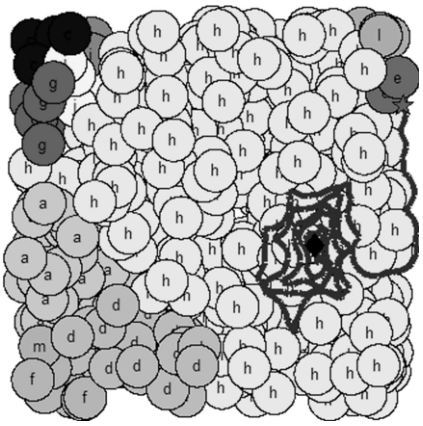

Fig. 1. Example of autonomous network repair by following communication perimeters.

version of the model-based image technique, (2) the iso-hop-contour method described in [9], (3) a centralized technique that defines the boundary nodes from the global degree histogram, and (4) a technique that locally computes the convex hull, which is implemented on a physical, 25 node network. Section 5 discusses the potential of localization algorithms for improved performance and topological detection to better aid in repair decisions. Lastly, the results are summarized in Section 6. The techniques presented in this paper draw from several different disciplines including graph theory, networking, information retrieval, and computer vision. Therefore it may be useful to consult Table 1 for a list of terms and their definitions.

2. Related work

The related literature on physical perimeter and hole boundary detection algorithms can generally be classified as either centralized or localized. Centralized implies that a single node determines whether or not each node in the network is an edge node after accumulating data from all nodes in the network. The primary disadvantage of centralized algorithms is, that as networks grow, so do the communication and memory overhead. The advantage of centralized algorithms, however, is that they allow for the detection of complex topological constructs. With localized algorithms, on the other hand, each node is responsible for determining its own state. Distributed boundary detection algorithms therefore do not require the communication and memory resources of their centralized counterparts. Whether the underlying technique relies on the detection of chordless cycles, analysis of node degrees, connectivity contours, or geometric constructs, they all have the same end goal. Each method has its own merit, but they are difficult to compare as the ground truth of the boundary nodes is often vague or not defined at all. Others are well defined and very specific, but do not address coverage boundaries and therefore are not well suited to the particular problem that we are addressing.

Kroeller et al. [10] have devised a technique that searches the network for chordless cycles. They show that chordless cycles are those that follow the perimeter of the network as well as the boundary of any internal holes. These cycles are found by searching for constructs called "flowers", which allow the algorithm to determine whether or not a node is inside, or outside, of a cycle. This technique guarantees correct results but at the cost of complex computation across a dense eight-hop neighborhood.

Saukh et al. [11] take a similar approach of finding patterns within a network that can positively identify inner nodes. Using several patterns of varying complexity, they have a more generally applicable method that is not limited to networks of high density. This method can be applied to neighborhoods ranging from 2 to 6 hops depending on the density of the region of interest. Varying parameters can aid in adaptability across non-uniform networks.

Likewise, the work presented by Ghrist and Muhammad [12] and de Silva and Ghrist [13] is targeted on chordless cycles. These authors focus on the construction of the Rips complex [13], a geometric construction forming triangles and polygons based on the edge information within the network, to detect holes in network coverage, but assume that the network's outer perimeter nodes are known.

Chordless cycles are also the focus of Shirsat and Bhargava [14]. They propose a localized boundary detection algorithm which identifies the geographical perimeter of holes in a sensor network assuming that the location-aware two-hop neighbor information is available to every node. The proposed algorithm requires synchronization of the two-hop neighborhood of sensor nodes in the network as well as the knowledge of the relative clock-wise orientation of all the immediate neighbors of a node for reliable detection of chordless cycles.

A different approach taken by Funke [9] proposes a perimeter and hole detection algorithm that uses hop counts to create iso-contours across the region of interest. The end-points of each contour line segment correspond to either the region perimeter or the edge of a hole. This algorithm is shown to work quite well for networks of degree as low as 20 and was shown to be viable for both unit disk graphs and quasi-unit disk graphs [9]. Funke and Klein [15] have also introduced a distributed version of this technique. This algorithm considers tunable parameters for hole size and neighborhood radius and addresses shortcomings of their original approach that resulted in false positives. This approach is shown to perform well for distributions of degree 25 or perturbed grid networks of degree 10.

Dinh [16] also focuses on contour breaks. He proposes a distributed topological connectivity approach where the nodes use their two-hop neighborhood to classify perimeter nodes. This is done by determining whether or not the nodes within the two-hop neighborhood can be used to form a ring about the node of interest. If not, it is classified as a boundary hole. This distributed algorithm is shown to work well for network densities with an average degree of 7.

Similar to the contour concept, Wang et al. [17] use shortest path trees to discover tight communication cycles around network holes. An interesting benefit that is derived from the algorithm is that it can also be used to generate the medial axis of a network, which can aid in routing. Continued work in this area has applied similar techniques to perform connection-based localization [18].

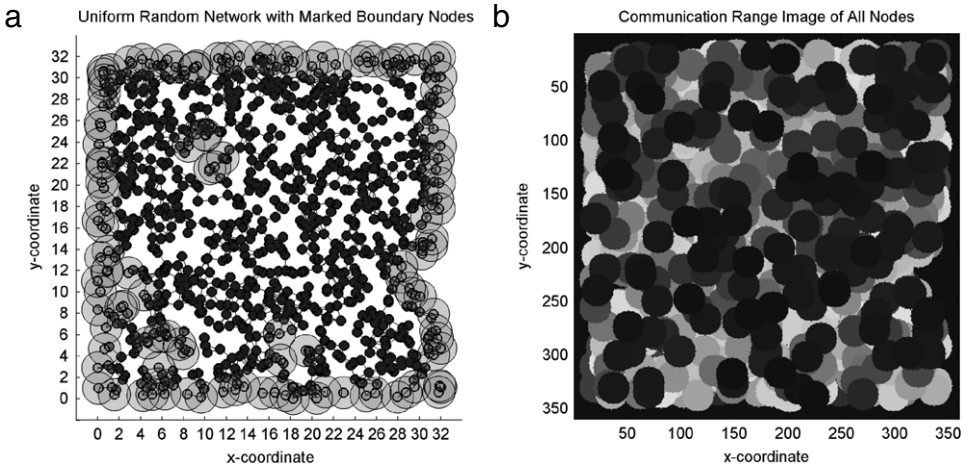

Fig. 2. Demonstration of the proposed imaging technique for boundary detection. (a) The boundary nodes (circles with large halos) as differentiated from the remaining nodes (small circles) by analyzing the network communication range image of (b) where individual nodes are labeled with different colors on a common background color. Nodes whose coverage regions are adjacent to the background are defined as boundary nodes.

Other techniques focus on the differences in node neighbor degree at the boundaries and within the network. Fekete et al. [19] present a technique for discerning boundary nodes, near boundary nodes, node distances to the nearest boundary, and Voronoi nodes within a network through evaluation of the node neighbor degree distribution of the network. Nodes near the boundaries are expected to have fewer neighbors than inner nodes. In addition, they provide a means of differentiating the outer boundary of the network from hole boundaries by evaluation of boundary length for simple hole structures or more complex evaluation of curvature from edge overlapping for more elaborate shapes. This approach is shown to work best for densely populated nodes (degree > 100).

Similarly, Bi et al. [20] present a distributed, cooperative scheme which uses information exchanged in the two-hop neighborhood of a sensor node to identify holes and network perimeters. The algorithm is based on the premise that a boundary node would have a lower nodal degree than an in-network node, in a dense, uniformly distributed, sensor network. The average degree of two-hop and one-hop neighbors is compared with the degree of the node of interest to determine whether or not it lies on the boundary. Additionally, a low complexity heuristic is employed to avoid non-boundary sensor nodes from implementing the complete algorithm, thus reducing communication and computational overhead.

The procedure introduced by Zhang et al. [21] requires relative location and direction information within a one-hop neighborhood to compute local Voronoi regions for each node. If a node is not enclosed in a Voronoi region, then it can be considered as an edge node. Zhang et al. [21] also examined the local computation of the convex hull.

Deogun et al. [22] also consider the immediate neighbors of the node as being of interest. If it is found that this node is not enclosed by a triangle formed by any of its neighbors, then it can be considered as an edge node. This algorithm relies on inter-node distances and relative location to compute the bounding triangles.

The technique introduced by Fang et al. [23] focuses on communication routing holes within the network. Routing holes are detected using the *Tent* algorithm which identifies stuck nodes along a route, those nodes which are closer to the target node than any of their neighbors. In geographic routing, when a message reaches a stuck node, it cannot find a closer point to be routed through and becomes stuck at a local minimum. The *BoundHole* algorithm [23] can then be employed to incrementally determine the other nodes forming the hole boundary. This algorithm results in well defined routing holes. However, these holes can be quite small and do not necessarily relate to holes from a coverage perspective.

Schieferdecker et al. [24] classify network nodes into three categories: (1) mandatory boundary nodes, (2) optional boundary nodes consisting of those nodes within the maximum communication range of mandatory boundary nodes, and (3) interior nodes. They also present two distributed algorithms based on multi-dimensional scaling (MDS) and enclosed circles (ECs) which require only connectivity information of at most three-hop neighborhoods for classification. These techniques are quantitatively and qualitatively compared to other connectivity based approaches and are shown to perform quite well for networks of degree greater than 10.

3. Definition of boundary nodes

While the work reviewed in the previous section presents several viable in situ boundary detection techniques, it is difficult to compare their correctness. Some performance metrics are introduced, but little has been done in the way of defining boundaries or establishing correctness measures, which would allow for a comparative assessment of techniques. This section focuses on defining true boundary nodes with respect to coverage for a given network and the appropriate correctness measures for evaluating boundary detection methods.

Boundary nodes, whether they be on the perimeter of a network or on the perimeter of a hole, can be defined in terms of coverage. Simply put, a boundary node is one that is adjacent to an area not covered by any other nodes of the network. This implies that the boundary definition is model dependent, relying on the detectable range of the device concerned, i.e., sensory or communication range. Modeling the network in this manner allows the network to be viewed as an image rather than a set of vertices and edges in a graph. This is best explained via an example.

Consider the uniform random network depicted in Fig. 2(a) where each node is marked by a small blue circle. Connections between the nodes specify nodes that are in communication range of one another. The true boundary nodes are signified by a small circle surrounded by a halo denoting the communication range of these nodes. More will be said about this shortly. From this graph, the image of Fig. 2(b) is constructed. The communication disk for each node is labeled using the node's unique identification number on the top of a global background label. From the previously introduced definition of a boundary, a node in the image is considered as a boundary node if its label is adjacent to the

```
// create default image background
image_size_x ⇐ xmin−model_max_range−1 to xmax+model_max_range+1
image_size_y ⇐ ymin−model_max_range−1 to ymax+model_max_range+1
image[image_size_x,image_size_y] ⇐ background label
// create a uniquely labeled image for each node
FOR i ⇐ 1 to num_nodes
    fill model from centerpoint (node(i)_x,node(i)_y) with unique identifier, i
ENDFOR
// determine which nodes are adjacent to the image background
FOR x ⇐ image_xmin to image_xmax
    FOR y ⇐ image_ymin to image_ymax
        IF image(x,y) is not a background pixel and node(i) corresponding
        to image(x,y) is not identified as an edge node
            IF any neighbors of image(x,y) are background pixels
                label node(i) as an edge node
            ENDIF
        ENDIF
    ENDFOR
ENDFOR
```

Fig. 3. Algorithm for boundary node detection using model-based image analysis.

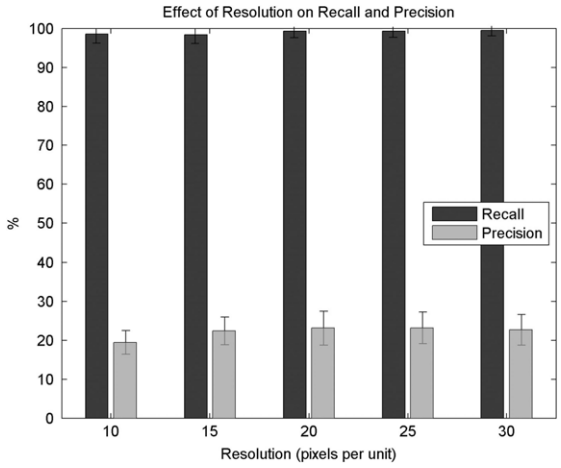

Fig. 4. The average recall and precision over 50 uniform random networks of 1024 nodes using localized image analysis for different resolutions.

background label. Parsing the image in this manner results in the boundary nodes, which are labeled in Fig. 2(a). The halos of the defined boundary nodes are plotted in Fig. 2(a) to demonstrate the continuity of both perimeter and hole boundaries. Note that, by this definition, it is not necessary for the nodes of a common boundary to be members of a cycle in the graph. Although the disk model was used for this example, this technique is not limited to this model. Pseudocode for this algorithm is presented in Fig. 3.

Knowing the true boundary nodes of the network, it is possible to assess the correctness of a given technique for defining coverage boundary nodes. Borrowing from the field of information retrieval, the correctness can be measured in terms of recall and precision. Recall reports the percentage of boundary nodes that have been identified. Precision on the other hand is the percentage of nodes identified as boundary nodes that are actually boundary nodes. This lends insight into the number of false positives reported. It is common for the ratio of the two, the F-measure, to be reported, however, different applications may weigh the importance of recall and precision differently so both will be reported herein. The following section applies the definition of true boundary nodes and the correctness measures of recall and precision to several boundary detection techniques including one based on the local application of the model image.

4. Analysis of boundary detection techniques

This section analyzes four boundary detection techniques in terms of placement error, and where applicable, localization error. The placement error occurs when a node's actual location is different from its intended location of deployment. The placement error applies to all boundary detection techniques. The localization error is a product of the network's localization algorithm and therefore only affects those algorithms which require the location information of the node. While the focus of this analysis is on the correctness of the proposed algorithms, a comparison of the performance metrics of the techniques is provided in Table 2. The first algorithm is a localized version of the model image adjacency algorithm. The second is a centralized hop contour technique. The third algorithm is based on an analysis of the node degrees throughout the network. Fourth, and lastly, a local convex hull technique is presented, with perturbation added to offset small concavities.

4.1. Local image modeling

As demonstrated in the previous section, using an image created from the sensory or communication model is an effective method to define the boundary nodes of a network. Unfortunately, applying this technique on a deployed network is not feasible for large node populations. This is a centralized technique, so it would scale poorly on resource constrained sensor nodes with memory requirements governed by Eq. (1). In Eq. (1), P is the resolution or number of pixels per point, R is the radius of the model (assuming a disk model) centered at (x, y), and S is the number of bytes required for storage of the largest node id in the network.

$$\begin{aligned} memory = {} & P * ((\max(x) + R) - (\min(x) - R)) \\ & * ((\max(y) + R) - (\min(y) - R)) * S. \end{aligned} \tag{1}$$

Although the imaging technique does not lend itself to a centralized network implementation, it can be localized by considering each node and its neighbors individually. Eq. (1) still applies to the local case. However, the maximum width and height of an image considering only immediate neighbors is four times the radius of the model for a worst case memory requirement of $(4 * R * S)^2$. To minimize the consumption of memory, it is necessary to reduce the image resolution. To assess the impact of resolution on this technique, it was run on 50 uniform random networks with resolutions ranging from 10 pixels per unit to 30 pixels per unit in 5 pixel increments. Fig. 3 shows the resulting average recall and precision of these trials with 95% confidence intervals. Recall is the primary measure of interest when considering the effects of resolution. The local application of this technique should identify at least all of the nodes identified by global application so the recall should be 100%. Although the data demonstrates a very high recall, the results are not perfect. This is due to pixel misalignment between the original image and its local counterpart. A shift by as much as a single pixel can cause a node to be incorrectly identified. Although the precision improves slightly with increased resolution, the overall precision is quite poor. The poor precision measure relates primarily to the identification of holes within the network. When the image analysis is applied globally, all nodes are considered in the image and contribute to network coverage. This is not so with the local application, hence smaller holes are reported and the precision decreases. All further analyses apply a resolution of 20 pixels per unit (see Fig. 4).

While robust performance in all cases is desirable, a uniform random deployment is not generally considered ideal. Controlled deployments can provide better coverage characteristics, along with potential benefits for routing options and redundancy. Consider the sample grid deployment of Fig. 5, where 1024 nodes are deployed in a square grid with 1 unit spacing. The communication range of each node is 1.5 units and adheres to the disk model. In this deployment, each node has an average number

Table 2
Table comparing the performance metrics of the evaluated boundary detection algorithms.

Algorithm	Type	Run-time complexity	Space complexity	Communication
Local image model	Distributed	$O(rc)$ r = pixels/row c = pixels/col	$O(rc)$ r = pixels/row c = pixels/col	N-Hop neighbor discovery
Iso-hop contours	Centralized	$O(\|V\| + \|E\|)$ per iso-segment V = # vertices E = # edges	$O(\|V\| + \|E\|)$ per iso-segment V = # vertices E = # edges	Network flood for S sources
Nodal degree analysis	Centralized	$O(n)$ n = #nodes	$O(n)$ n = #nodes	Single flood and neighbor discovery
Perturbed Graham scan	Distributed	$O(n \log n)$ n = node degree	$O(n)$ n = node degree	N-Hop neighbor discovery

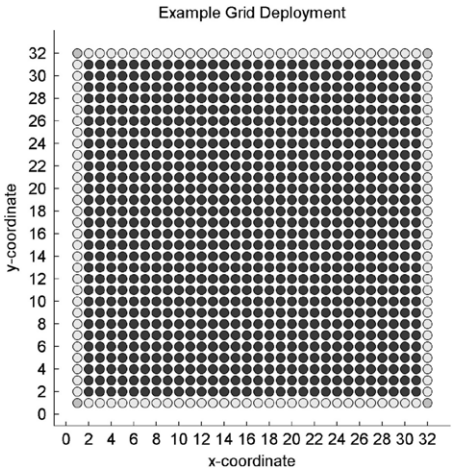

Fig. 5. An example square grid of 1024 nodes with unit placement and communication range of radius 1.5 units. The node color signifies nodes with the same degree.

of connections, or degree, of 7.63 which is regional as denoted by the color of each node specifying its degree. In this ideal case, the local image technique results in nearly 100% recall and precision. However, it is unrealistic to assume an ideal network distribution, both in terms of node placement during deployment and node localization during network formation.

First, consider the placement error that is introduced during the network deployment. The normal error is introduced to the grid placements of Fig. 5, with incremental standard deviation from 0.1 to 0.5 in steps of 0.1 for both x and y coordinates. Each standard deviation is tested on 30 networks, the results are compiled in Fig. 6(a) along with the results obtained from the uniform random distribution. It is clear from Fig. 6(a) that recall is relatively immune to the placement error while precision decreases rapidly as the standard deviation increases beyond 0.1. This same experiment was run using a two-hop neighborhood, the results are displayed in Fig. 6(b). Developing a two-hop image rather than a one-hop image eliminates several of the smaller coverage discontinuities within the network resulting in an improvement in precision. The average precision using one-hop neighbors drops to 60% with a placement error of 0.2 whereas the two-hop scenario maintains over 90% precision through a placement error of 0.2 and drops to approximately 60% at 0.3 error.

Since this technique relies on known node coordinates it is also important to assess the effect of localization error on the local image technique. Consider again the ideal deployment of Fig. 5. The true edges are determined using this ideal network while the local image technique is applied after the localization error is introduced by altering the coordinates, by introducing a normal error as before. Again, this was done for 30 networks with the standard deviation ranging from 0.1 to 0.5 in steps of 0.1. Fig. 7(a) (left) shows an improved performance in precision when compared to the placement error, but with an associated loss of recall. Again, localization error effects are reduced by using two-hop neighborhoods; see Fig. 7(b) (right).

These tests demonstrate that the local application of image boundary detection technique is robust when used for detecting the boundary nodes for both placement and localization errors. However, if high precision is the desired metric, it is best to use a larger neighborhood. In addition, care must be taken to ensure that the hardware has adequate memory to create a network image of an acceptable resolution.

4.2. The iso-contour technique

As described in Section 2, Funke [9] has introduced a perimeter and hole finding method that detects the breaks in contours formed by hop counts from one or more source nodes within the network. An example of this technique using 1 source node is shown in Fig. 8. One advantage of this technique is that it requires no position information, only hop counts, hence it is immune to the localization error. This method requires centralized computation and as such will not scale particularly well, however, Funke and Klein [15] have developed a distributed algorithm based on this technique. The results presented in [9] demonstrate that the centralized method works well for node densities where each node has an average degree of 20. This is reasonable for the uniform random distributions analyzed by Funke. However, average degree does not always indicate how well this algorithm works. For instance, the iso-contour method works extremely well on the network of Fig. 5 which has an average degree of only 7.63. This is demonstrated in Fig. 8 where each color band denotes nodes on the same iso-contour, those nodes that are the same distance in terms of hops from a given source node. The degree of the nodes is not an indication of the continuity of their connection across the whole network, or in this case, a hop contour. Furthermore, nodes form regions of degrees, with nodes on boundaries often having lower degrees than other nodes, Fig. 5 being an extreme example. The corner nodes in Fig. 5 have the lowest degree, 3, while the non-corner edges have a degree of 5, and the central nodes each has 8 connections. This observation inspired the technique presented in the next section. Given the regional nature of degrees, it is important to realize that the standard deviation of degree in a given deployment can be quite large. It is therefore suggested that average degree is not an appropriate measure to characterize a network deployment with regard to iso-contours.

This technique takes the number of source nodes as a parameter. Funke uses 4 source nodes and accumulates the discovered boundary nodes from each run [9]. This parameter is re-assessed here on 30 uniform random networks of 1024 nodes with 1–6 sources. As in [9], the first source node of each run is chosen randomly and the remaining are chosen by keeping track of the hop count to the closest source node. The node that has the highest hop count to the closest source node will be the next source node. This ensures that the source nodes are not all co-located. The results are presented in Fig. 10. The recall of this method, while not as high as the local image technique, is, on average, above 80% for all source node populations. Likewise, the precision is quite poor, with values consistently less than 20% as demonstrated in the uniform random network example of Fig. 9. This is not surprising given the discontinuities in the hop contours of the uniform random network at

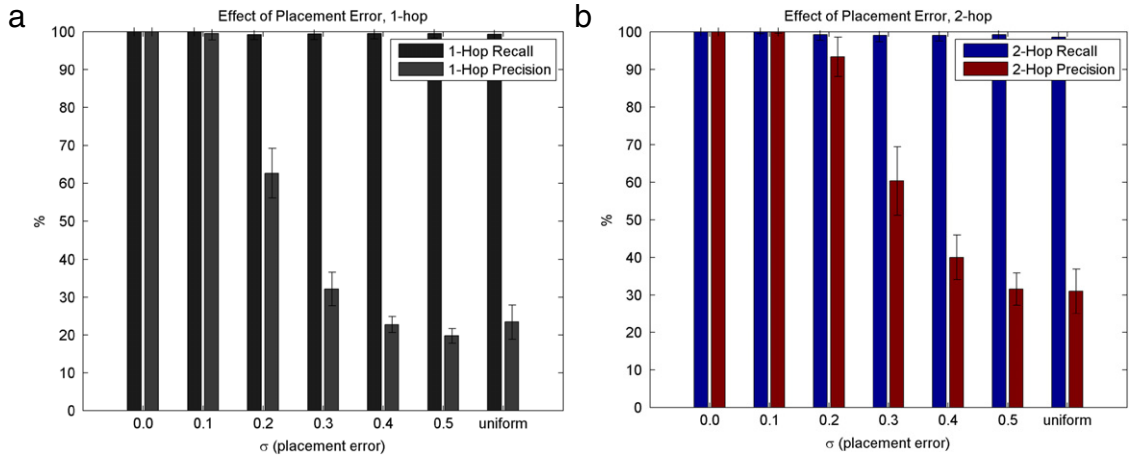

Fig. 6. The effect of normal placement error on recall and precision for (a) one-hop and (b) two-hop local images.

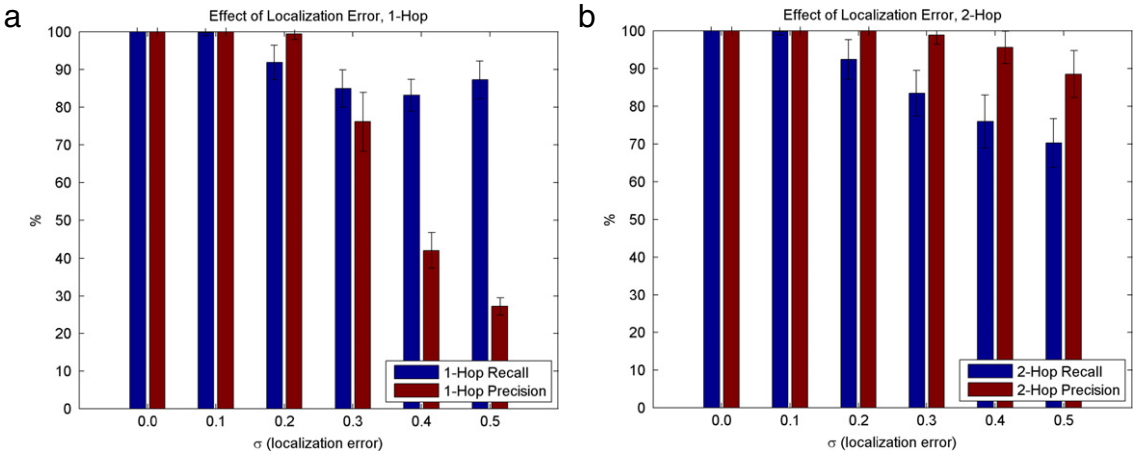

Fig. 7. The effect of normal localization error on recall and precision for (a) one-hop and (b) two-hop local images.

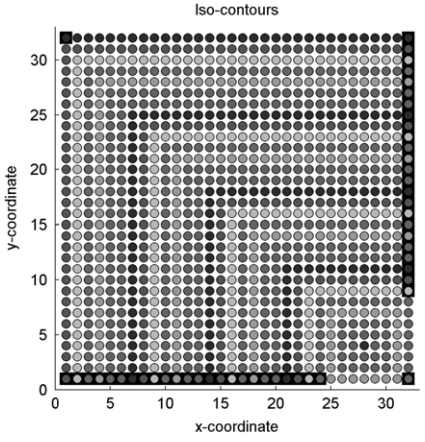

Fig. 8. An example of hop iso-contours using Funke's technique. Strips of consistent color denote nodes that are the same hop count from the beacon source. The ends of these contour segments are identified as boundary nodes.

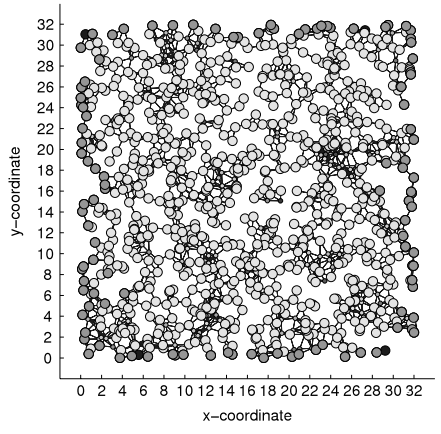

Fig. 9. An example of using iso contours on a uniform random network with average degree of 6.71. The recall shown by the large medium gray nodes is 96%. The large dark nodes are the true boundary nodes that were not correctly identified. The large light nodes are false positives, hence the precision is only 13%.

this density. The precision will be higher when run on grid networks with fewer contour breaks. The data presented in this figure also validates Funke's assessment that little is gained by using more than 4 source nodes and therefore 4 sources will be utilized for the remainder of the analysis. While more source nodes do improve recall, precision begins to decrease even further past 4 sources.

As with the local image technique, the effect of placement error can be assessed. Again, 30 square grid networks containing 1024 nodes were deployed with normal error with standard deviation from 0 to 0.5 in increments of 0.1. Fig. 11 shows the recall and precision that this test produced. The average recall is over 85% for all placement errors but as seen with the local image technique, the precision is greatly reduced, to as little as 10%, as error is introduced into the deployment. For the iso-contour method, this implies that there are numerous discontinuities in the iso-contours

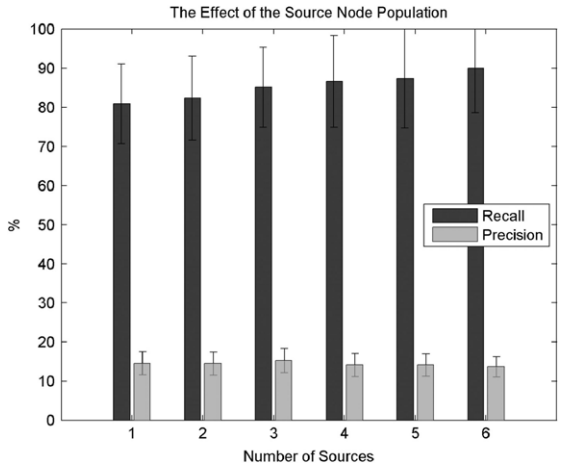

Fig. 10. Performance of hop-contours with 1–6 source nodes.

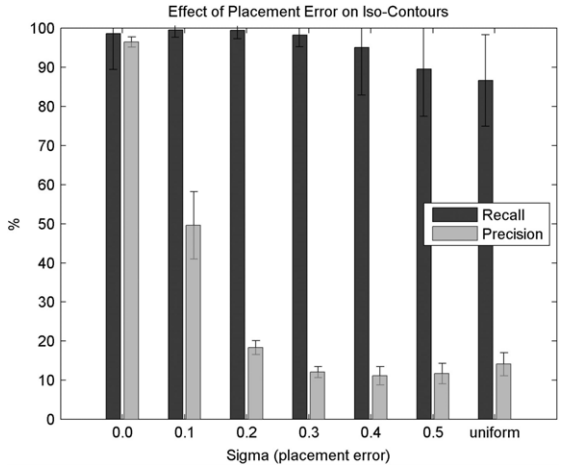

Fig. 11. The effect of placement error on the performance of the iso-contour technique with 4 sources applying variances from 0 to 0.5 in steps of 0.1.

being reported as holes, however, these are often not classifiable as coverage holes, and therefore, not of particular interest in the context of this paper. A pruning method was introduced in [9] that removes connected components consisting of a number of nodes less than a defined threshold. This is intended to eliminate the smaller boundary features and would likely serve to increase the precision of the technique.

4.3. Degree analysis for boundary detection

As mentioned in the discussion of iso-contours and as presented in several related works [19,20], the degree, or the number of immediate neighbors, of a node is often dependent upon its location in the network. Nodes with many neighbors are typically on the interior of a network, while nodes on the periphery have fewer neighbors. The degree of edge nodes however is relative, depending largely upon the density of the deployed network. A sparsely populated network may have edge nodes of degree 4, while a denser deployment may be on the order of degree 10. To determine an adequate threshold, it is therefore important to analyze the degree distribution of the entire network. The degree threshold to define neighbors can then be set to some multiple of the standard deviation of the degree of the network subtracted from the mean. This is demonstrated in the degree histogram of Fig. 12(a) for a grid network deployment with a standard deviation in placement of 0.3. For instance, if the threshold is set to 1.5 standard deviation below the mean, then the boundary nodes can be marked as in Fig. 12(b) where the node color denotes the node's degree (blue is lowest) and boundary nodes are circled in black.

Much like the iso-contour technique, this method is centralized and is not location dependent, making it immune to the localization error. The placement error is still a concern however. The placement error test is once again performed, however, this time the standard deviation threshold is taken into account by varying it from 0.5 to 1.5 times the standard deviation in increments of 0.25. Here, the recall and precision are reported separately, Fig. 13(a) and (b) respectively. This data indicates that recall is best when the threshold is close to the mean and precision is best when the threshold is far from the mean. A threshold of 1.25 times the standard deviation balances the recall and precision the best although both decrease to under 60% with the placement error. The performance of this technique on a uniform random distribution yields comparable results to those achieved with a placement error of 0.5.

4.4. Convex hull boundary detection

The convex hull is a natural construct for perimeter detection. By definition, the convex hull is the minimum number of points in a set forming a perimeter in which all other points are enclosed. Determining the perimeter is the goal. However, the convex hull does not readily identify all the perimeter nodes that span adjacent vertices in the graph of the network. Therefore, the global application of the convex hull will by default suffer from poor recall. Although the global computation of the convex hull yields a less than desirable identification of perimeter nodes, it is possible to distribute the algorithm to include additional boundary nodes. Rather than run the algorithm on the entire network, it can be run at each node, considering only that node's immediate neighbors. If the source node is returned as an edge node, it is labeled as such. Not only does this result in a higher recall, but it requires only a small amount of resources and computation per node, allowing it to scale well with network size. This technique can be considered a refinement of the encapsulating triangle method introduced by Deogun et al. [22] in that it does not require their node selection scheme. This technique is also referred to by Zhang et al. [21] who apply a similar construct in forming neighbor embracing polygons. Further improvement to this technique can be gained by introducing a perturbation variable which introduces small shifts in the location of the target node. These shifts allow nodes that are close to the hull to be counted as well by negating small concavities.

There are several convex hull algorithms, notable among these are the Jarvis March Algorithm [25] and Chan's Algorithm [26]. Here however, focus is placed on the Graham Scan algorithm [27], which was introduced by Graham in 1972. The Graham Scan Algorithm has proved to be an efficient planar convex hull algorithm and runs in $O(n \log n)$ for n input points, or nodes. The Jarvis March is an output sensitive algorithm that has a runtime of $O(nh)$, where h is the number of hull vertices. It can be seen that in instances where there are several nodes but only a few hull points, the Jarvis March will out perform the Graham Scan. Chan's Algorithm is also output sensitive and has a runtime of only $O(n \log h)$. It partitions the point space into smaller point clusters based on an estimate of h on which another convex hull algorithm, often the Graham Scan, is run. An overall scan is then run on the points resulting from the partitions. While both the Jarvis March and Chan's Algorithm have potentially faster execution times than the Graham Scan, this runtime advantage has not been seen in practice for point sets under 1000 [26]. Currently, most deployed networks are well under 1000 nodes. Furthermore, upon distributing the algorithm in the network, point populations will

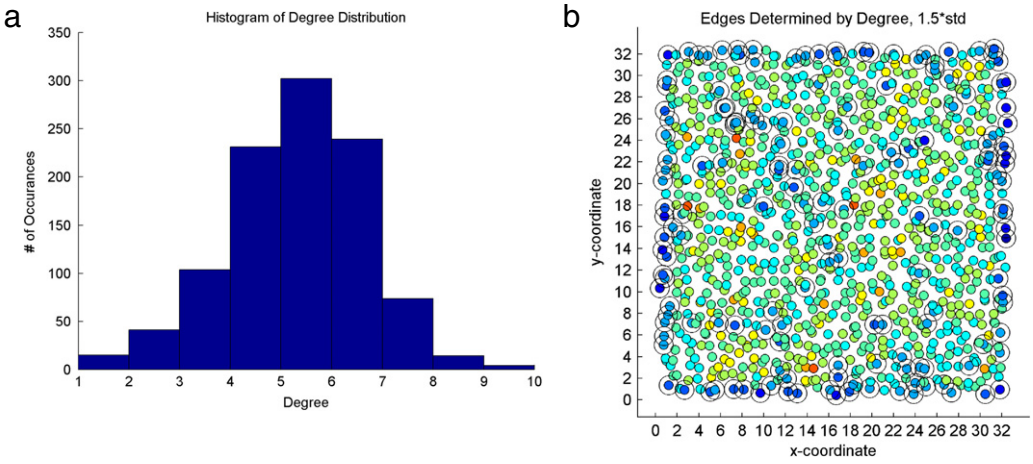

Fig. 12. The degree histogram, (a), and the resulting network using a degree threshold set 1.5 standard deviation from the mean, (b), The network is a grid placement with a variance of 0.3. The colors represent the degree of the node (lower degrees are darker) and dark circles surround nodes that are identified as boundary nodes.

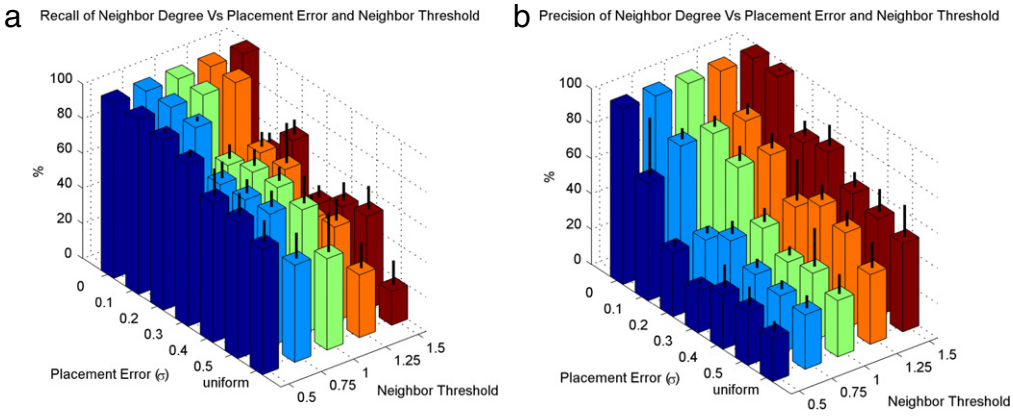

Fig. 13. The recall, (a), and the precision, (b), of the degree threshold boundary detection technique considering placement error and the degree threshold.

be on the order of 10's of nodes. Finally, Chan's Algorithm can be extended to three dimensions and it commonly utilizes the Graham Scan.

The Graham Scan Algorithm functions by first choosing a node with the lowest y-coordinate, defaulting to the lowest x-coordinate in case of a tie. The angles between this point and all other points in the search space are then determined. Next the nodes are sorted on the angle from lowest to highest. The algorithm moves counter-clockwise from the start point considering three points at a time where the middle node is the point of interest. If, in moving from point 1 to point 3 through point 2, a left hand turn is made, then node 2 is an edge node. The next iteration commences as node 2 becomes the starting point and the next two nodes after node 2 in the sorted angle list are considered. In the case of a right hand turn, node 2 is no longer considered and the next two nodes from the sorted angle list are considered with respect to node 1.

The convex hull boundary detection method is location dependent and as such it will be affected by both placement and localization errors. As with the threshold variable in the degree threshold technique, variation in the perturbation variable must also be taken into consideration. The recall of this experiment is depicted in Fig. 14(a) and the precision in Fig. 14(b). The recall is above 70% for all placement errors with a perturbation of 0.1 times the communication range, an improvement of as much as 30% when no perturbation is utilized. For larger perturbation values, recall rates of 85% and above are realized. The precision with a 0.1 perturbation is the highest when compared to the other perturbation values but is generally poor with significant decreases, to near 30%, with increased placement error. Performing this same experiment using two-hop networks as shown in Fig. 14(c) and (d) demonstrates drastic improvement in precision, particularly with lower position error, with little sacrifice in recall when a perturbation value of 0.2 is utilized. The higher ranges of the placement error still result in low (<40%) precision for higher perturbation values.

Although the perturbed Graham Scan boundary detection method demonstrates good performance in the presence of placement error, it is a location dependent algorithm, it is therefore subject to the localization error. As with the local image detection technique, this test assumes the ideal grid placement of nodes with normal error applied to their self-localized positions. Again, this is subject to different behavior under different perturbation values. The results of the recall and precision are plotted in Fig. 15(a) and (b), respectively. Recall remains quite high for all localization errors when the perturbation value is increased. Precision follows the same trend displayed in the placement error, it decreases as the perturbation value increases. The precision is approximately 70% for localization errors up to 0.2 for perturbation values up to 0.3.

The results obtained using the convex hull method for boundary node determination are promising. Because of this, and because of the simplicity of the algorithm, this technique was implemented on a network of 25 T-Mote Sky [28] nodes. The radio range was software limited to approximate a larger deployment area. Each node queries its neighbors and retrieves their locations. After processing the Graham Scan, the nodes signify that they are

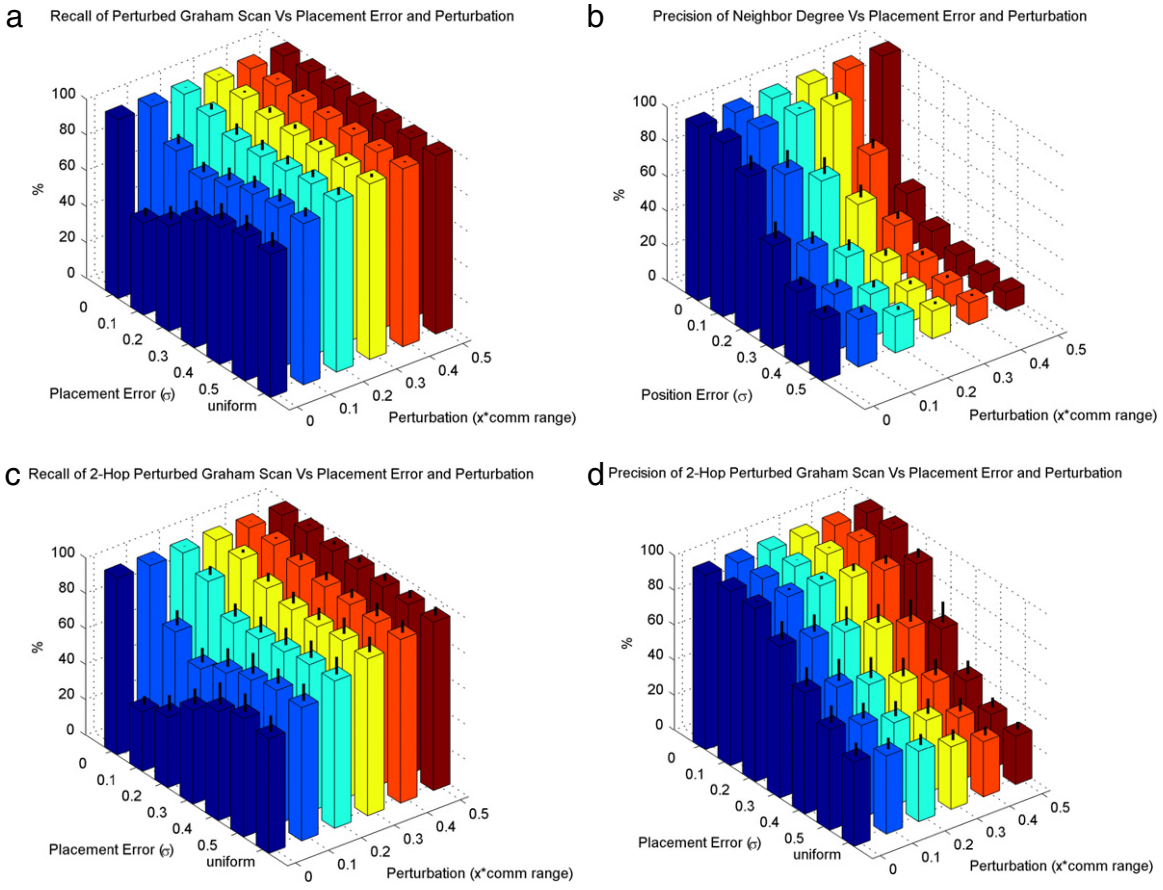

Fig. 14. The recall, (a), and the precision, (b), of the convex hull boundary detection technique considering placement error and increasing perturbation. This is repeated considering two-hop neighborhoods resulting in the recall profile of (c), and precision profile of (d).

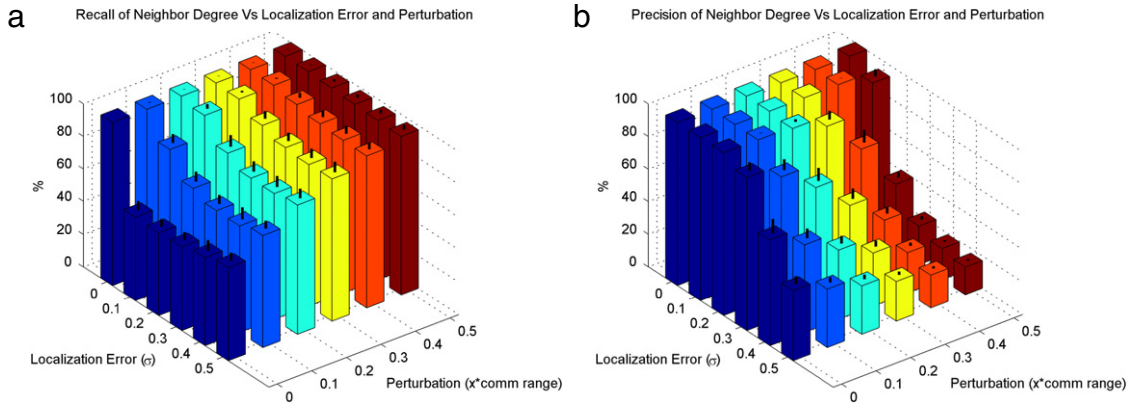

Fig. 15. The recall, (a), and the precision, (b), of the convex hull boundary detection technique considering localization error and increasing perturbation values.

perimeter nodes by lighting the green LED or not, by lighting the red LED. A test formation is shown in Fig. 16(a) where identified perimeter nodes are framed with a green square, while non-perimeter nodes are signified with a red circle frame. The network is also shown in Fig. 16(b). Here, the connections between the nodes are included. The nodes that were identified as perimeter nodes are depicted as large red circles while those that are identified in simulation given the real connections are depicted by a blue square.

The real-world deployment yielded only 2 discrepancies when compared to the simulated run. Node 4 was not identified as a perimeter node in the real deployment while node 19 was identified on the real deployment but not in the simulation. This inconsistency is due to the random perturbation resulting in a small movement of node 4 and a large displacement of node 19 in the real deployment.

It is also interesting to note that the connections within the real-world deployment are irregular, not adhering to a defined range. Node 1 for instance is a neighbor of node 25 although they are not physically near one another. There are also some instances of asymmetric links within the network. This suggests that future simulations should be performed with a more realistic model;

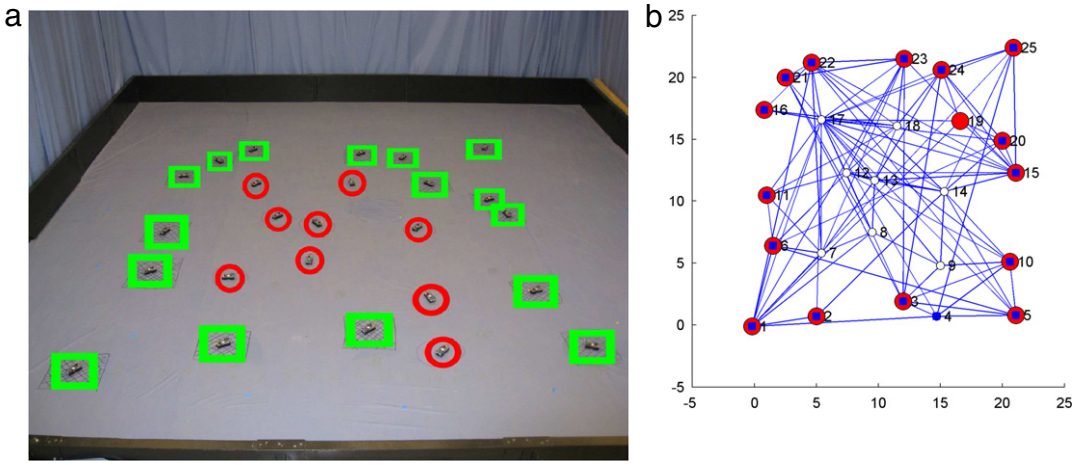

Fig. 16. The real-world implementation (a) layout and (b) connectivity graph.

one that includes both node orientation and the antenna radiation pattern.

5. Future work

As mentioned in the previous section, there is still room for improvement in perimeter and hole detection. Specifically, the current techniques should be evaluated in more realistic simulations, particularly those that are model-based. Also, asymmetric links are problematic for all techniques. Of particular importance for model-based techniques is the accuracy of the localization algorithm employed by the network.

Additional avenues of investigation open up in the topological analysis of the network, e.g., discriminating between perimeter nodes and hole boundary nodes. Ideally, if this could be accomplished locally, then the technique would be scalable to larger networks. This is particularly important in the context of network repair. Furthermore, being able to estimate the shape of the network based on its boundary nodes could provide insight for intelligent searching and network expansion.

5.1. The localization problem

It has been demonstrated how localization error can adversely affect boundary detection algorithms that are location dependent, such as the local image and convex hull techniques. It is therefore important to consider the effectiveness of the localization algorithm employed by the network. Unfortunately, localization is not a trivial task. In an attempt to accomplish this task economically, many techniques have been presented in the literature [29–31]. Recent techniques have focused on using the hop counts between nodes within the network to solve this problem. One such technique that will be considered for future work was presented by Gianni Giorgetti [32] which employs a self-organizing map (SOM) to create virtual coordinates from inter-node hop counts. Giorgetti suggests that localization performance can be improved if orientation or distances are known. We plan to obtain rough measures of these variables with directional antennas and distance approximations from received signal strength readings.

A potential advantage of model-based boundary detection algorithms is that the model may provide corrective information to the localization algorithm. Conversely, if the localization of the network is correct, with a high degree of confidence, it can be used to adapt the model to the real-world deployment. For instance, if two nodes are neighbors but they are not in each others coverage region by the model then either the model needs to be adjusted or the localization is in error.

5.2. Topological analysis

While it is beneficial to detect both perimeter and hole nodes, in some applications it is important to be able to differentiate between the two. For instance, network coverage can be improved by filling in network holes. In this situation, it is useful to know which nodes are hole nodes. Conversely, for repair situations where multiple disjoint networks are present, the perimeter nodes are the feature of interest. Current investigation focuses on collecting neighbor metrics to aid in discriminating between hole and perimeter nodes. One drawback of this approach is its sensitivity to network density. Another possible method for differentiating hole nodes from perimeter nodes is to determine the length of boundary node cycles in the network. The longest cycle should correspond to the perimeter while the others should denote holes. However, local techniques would be preferable for scalability reasons.

Once the perimeter of a network is identified, it could be useful to identify the shape of the network perimeter. If node locations are known with reference to a fixed frame, determining the shape is trivial. However, localization techniques such as [32] create only local virtual coordinates which are not registered globally with the entire network. This shape may be useful for repair by determining where best to search for disconnected networks. It may also provide insight into environmental features such as ponds which enveloped dropped nodes during deployment.

6. Conclusions

This paper has introduced a technique to identify perimeter and coverage hole boundary nodes in wireless sensor networks by creating a model-based image and detecting regions adjacent to the background. Although this is not a viable solution for implementation within a real network deployment, it does provide ground truth which is useful for evaluating other viable in situ techniques.

Of the four techniques analyzed, it was found that the localized convex hull technique computed by the Graham Scan performed well in the presence of placement error and localization error when two-hop neighborhoods are considered. This technique was also successfully implemented on a physical 25 node network with only two identification errors.

The local image model technique also performed well when two-hop neighborhoods were considered, providing high ranges of recall and precision under both placement and localization errors. This method, however, does not scale well in terms of

memory with increased neighborhood size. While the two-hop neighborhood improves the precision of the results, it is important to consider memory required for image creation.

The local degree analysis and iso-contour methods are quite effective at detecting the perimeter nodes but are not very precise, resulting in many false positives. The advantage of these two methods is that they are immune to the localization error as they do not rely on the positions of the nodes within the network.

While the perturbed Graham Scan algorithm has been successfully demonstrated, more work still needs to be done. Specifically, for this and other techniques to be useful, it is vital to remove the dependence on globally referenced node coordinates. As discussed, node localization algorithms exist that can provide virtual coordinates on which perimeter and hole detection algorithms can be run. It is also of great interest to be able to differentiate between hole and perimeter nodes in order to better direct network repairs.

References

[1] K. Luthy, E. Grant, T. Henderson, Leveraging RSSI for robotic repair of disconnected wireless sensor networks, in: IEEE International Conference on Robotics and Automation, Rome, Italy 2007, pp. 3659–3664.

[2] K. Chintalapudi, R. Govindan, Localized edge detection in sensor fields, in: Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications, 2003, pp. 59–70. doi:10.1109/SNPA.2003.1203357.

[3] M. Ding, D. Chen, K. Xing, X. Cheng, Localized Fault-tolerant event boundary detection in sensor networks, in: INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies, 2005, pp. 902–913.

[4] M. Ding, X. Cheng, Robust event boundary detection in sensor networks—a mixture model based approach, in: IEEE INFOCOM 2009 (ISSN: 0743-166X), 2009, pp. 2991–2995. doi:10.1109/INFCOM.2009.5062273.

[5] G. Jin, S. Nittel, NED: an efficient noise-tolerant event and event boundary detection algorithm in wireless sensor networks, in: 7th International Conference on Mobile Data Management, (MDM 2006) (ISSN 1551-6245), 153, 2006. doi:10.1109/MDM.2006.110.

[6] C. Mallery, M. Medidi, Robust edge detection in wireless sensor networks, in: IEEE GLOBECOM 2008, (ISSN: 1930-529X), 2008, pp. 1–5. doi:10.1109/GLOCOM.2008.ECP.61.

[7] R. Nowak, U. Mitra, Boundary estimation in sensor networks: theory and methods, in: IPSN'03: Proceedings of the 2nd International Conference on Information Processing in Sensor Networks, Springer-Verlag, Palo Alto, CA, USA, 2003, pp. 80–95.

[8] K. Ren, K. Zeng, W. Lou, Secure and fault-tolerant event boundary detection in wireless sensor networks, IEEE Transactions on Wireless Communications 7 (1) (2008) 354–363.

[9] S. Funke, Topological hole detection in wireless sensor networks and its applications, in: DIALM-POMC 05: Proceedings of the 2005 Joint Workshop on Foundations of Mobile Computing, New York, NY, USA, 2005, pp. 44–53.

[10] A. Kroeller, S. Fekete, D. Pfisterer, S. Fischer, Deterministic boundary recognition and topology extraction for large sensor networks, in: SODA 06: Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithms, New York, NY, USA, 2006, pp. 1000–1009.

[11] O. Saukh, R. Sauter, M. Gauger, P. Marron, On boundary recognition without location information in wireless sensor networks, ACM Transactions on Sensor Networks 6 (3) (2010) 20:1–20:35.

[12] R. Ghrist, A. Muhammad, Coverage and hole-detection in sensor networks via homology, in: Fourth International Symposium on Information Processing in Sensor Networks, IPSN 2005, doi:10.1109/IPSN.2005.1440933, 2005, pp. 254–260.

[13] V. de Silva, R. Ghrist, Coordinate-free coverage in sensor networks with controlled boundaries via homology, International Journal of Robotics Research 25 (12) (2006) 1205–1222.

[14] A. Shirsat, B. Bhargava, Local geometric algorithm for hole boundary detection in sensor networks, Security and Communication Networks (2011) doi:10.1002/sec.271. n/a–n/a ISSN 1939-0122.

[15] S. Funke, C. Klein, Hole detection or: how much geometry hides in connectivity? in: Proceedings of the 2006 Symposium on Computational Geometry, Sedona, AZ, USA, 2006, pp. 377–385.

[16] T.L. Dinh, Topological boundary detection in wireless sensor networks, Journal of Information Processing Systems 5 (3) (2009) 145–150. doi:10.3745/JIPS.2009.5.3.145.

[17] Y. Wang, J. Gao, J. Mitchell, Boundary recognition in sensor networks by topological methods, in: Proceedings of the 12th Annual International Conference on Mobile Computing and Networking, ACM, New York, NY, USA, 2006.

[18] Y. Wang, S. Lederer, J. Gao, Connectivity-based sensor network localization with incremental delaunay refinement method, in: Proceedings of the 28th Annual IEEE Conference on Computer Communications, ACM, New York, NY, USA, 2009, pp. 2401–2409.

[19] S. Fekete, A. Kroller, D. Pfisterer, S. Fischer, C. Buschmann, Neighborhood-based topology recognition in sensor networks, CoRR.

[20] K. Bi, K. Tu, N. Gu, W.L. Dong, X. Liu, Topological hole detection in sensor networks with cooperative neighbors, in: International Conference on Systems and Networks Communications, ICSNC'06, 31, doi:10.1109/ICSNC.2006.71, 2006.

[21] C. Zhang, Y. Zhang, Y. Fang, Localized algorithms for coverage boundary detection in wireless sensor networks, Wireless Networks 15 (1) (2009) 3–20.

[22] J. Deogun, S. Das, H. Hamza, S. Goddard, An algorithm for boundary discovery in wireless sensor networks, in: Proceedings of the 12th International Conference on High Performance Computing, vol. 3769, Goa, India, 2005, pp. 343–352.

[23] Q. Fang, J. Gao, L.J. Guibas, Locating and bypassing holes in sensor networks, Mobile Networks and Applications (ISSN: 1383-469X) 11 (2) (2006) 187–200. doi:10.1007/s11036-006-4471-y.

[24] D. Schieferdecker, M. Volker, D. Wagner, Efficient algorithms for distributed detection of holes and boundaries in wireless networks, Communication (2011) 1–16.

[25] R. Jarvis, On the identification of the convex hull of a finite set of points in the plane, Information Processing Letters 2 (1) (1973) 18–21.

[26] T. Chan, Optimal output-sensitive convex hull algorithms in two and three dimensions, Discrete and Computational Geometry 16 (1996) 361–368.

[27] R. Graham, An efficient algorithm for determining the convex hull of a finite planar set, Information Processing Letters 1 (1972) 132–133.

[28] Moteiv, Tmote-Sky Datasheet, moteiv, 2006.

[29] C. Alippi, G. Vanini, A RSSI-based and calibrated centralized localization technique for wireless sensor networks, in: Proceedings IEEE Conference on Pervasive Computing and Communications, Washington, DC, USA, ISBN 0-7695-2520-2, 2006, p. 301.

[30] R. Peng, M. Sichitiu, Probabilistic localization for outdoor wireless sensor networks, SIGMOBILE Mobile Computer and Communications Review 11 (1) (2007) 53–64.

[31] M. Terwilliger, A. Gupta, V. Bhuse, Z. Kamal, M. Salahuddin, A localization system using wireless network sensors: a comparison of two techniques, in: Proceedings of the First Workshop on Positioning, Navagation, and Communication, Hanover, Germany, 2004.

[32] G. Giorgetti, Resource-constrained localization in sensor networks, Ph.D. Thesis, Universita' Degli Studi di Firenze, Italy, Firenze, Italy, 2007.

Kyle Luthy received his B.S. in Electrical Engineering and Computer Science from Louisiana State University in 2001 and his M.S. and Ph.D. degrees in Computer Engineering from North Carolina State University (NCSU) in 2003 and 2009, respectively. At NCSU he was a member of the Center for Robotics and Intelligent Machines (CRIM) where his research focus was on autonomous robotic interactions with wireless sensor networks. As part of the CRIM team he was also involved in center efforts concerning sensing and control systems for various industry projects. He currently resides in San Diego where he works with autonomous systems as an Engineer with SPAWAR Systems Center Pacific.

Edward Grant is a Full Professor of Electrical and Computer Engineering and Biomedical Engineering at North Carolina State University. In 2010 Dr. Grant was appointed as a Senior Researcher at the Italian Institute of Technology. He received his B.Sc. (Hons) degree in Mechanical Engineering from the University of Abertay Dundee, his M.Eng. degree in fluid power control from the University of Sheffield, and his Ph.D. in Computer Science from the University of Strathclyde. Dr. Grant is a Chartered Engineer (C.Eng.) and a Fellow of the Institution of Mechanical Engineers (F.I.Mech.E.). His research interests include: evolutionary robotics, medical robotics, and wearable sensing and control systems for healthcare. Dr. Grant is Director of the Center for Robotics and Intelligent Machines at North Carolina State University.

Nikhil Deshpande is currently a Ph.D. student in the Department of Electrical and Computer Engineering at North Carolina State University. He received his Bachelors degree in Electrical Engineering from the Government College of Engineering, under the aegis of the University of Pune, India in 2003, and his Masters degree in Integrated Manufacturing Systems Engineering from North Carolina State University in 2007. His general interests are in the area of Wireless Sensor Networks and Robotics, and more specifically, in studying the effectiveness of robot navigation and coordination through the assistance of WSNs and Received Signal Strength. He is a student member of IEEE and was inducted into the Phi Kappa Phi Honor society in 2006.

Thomas C. Henderson received his B.S. in Math with Honors from Louisiana State University in 1973 and his Ph.D. in Computer Science from the University of Texas at Austin in 1979. He is currently a Full Professor in the School of Computing at the University of Utah. He has been at Utah since 1982, and was a visiting professor at DLR in Germany in 1980, and at INRIA in France in 1981 and 1987, and at the University of Karlsruhe, Germany in 2003.

Prof. Henderson was chairman of the Department of Computer Science at Utah from 1991 to 1997, and was the founding Director of the School of Computing from 2000 to 2003. Prof. Henderson is the author of Discrete Relaxation Techniques (University of Oxford Press), Computational Sensor Networks (Springer-Verlag), and editor of Traditional and Non-Traditional Robotic Sensors (Springer-Verlag); he served for 15 years as Co-Editor-in-Chief of the Journal of Robotics and Autonomous Systems and was an Associate Editor for the IEEE Transactions on Pattern Analysis and Machine Intelligence and IEEE Transactions on Robotics and Automation. His research interests include autonomous agents, robotics and computer vision, and his ultimate goal is to help realize functional androids. He has produced over 200 scholarly publications, and has been principal investigator on over $8M in research funding. Prof. Henderson is a Fellow of the IEEE, and received the Governor's Medal for Science and Technology in 2000. He enjoys good dinners with friends, reading, playing basketball and hiking.