Network Localization in Partially Localizable Networks

David K. Goldenberg   Arvind Krishnamurthy   Wesley C. Maness
Yang Richard Yang   Anthony Young
Computer Science Department, Yale University, New Haven, CT 06511

A. Stephen Morse, Andreas Savvides
Electrical Engineering Department, Yale University, New Haven, CT 06511

Brian D.O. Anderson
Australian National University and National ICT Australia limited, Canberra

***Abstract*— Knowing the positions of the nodes in a network is essential to many next generation pervasive and sensor network functionalities. Although many network localization systems have recently been proposed and evaluated, there has been no systematic study of *partially localizable networks*, *i.e.*, networks in which there exist nodes whose positions cannot be uniquely determined. There is no existing study which correctly identifies precisely which nodes in a network are uniquely localizable and which are not. This absence of a sufficient uniqueness condition permits the computation of erroneous positions that may in turn lead applications to produce flawed results. In this paper, in addition to demonstrating the relevance of networks that may not be fully localizable, we design the first framework for two dimensional network localization with an efficient component to correctly determine which nodes are localizable and which are not. Implementing this system, we conduct comprehensive evaluations of network localizability, providing guidelines for both network design and deployment. Furthermore, we study an integration of traditional geographic routing with geographic routing over virtual coordinates in the partially localizable network setting. We show that this novel cross-layer integration yields good performance, and argue that such optimizations will be likely be necessary to ensure acceptable application performance in partially localizable networks.**

I. Introduction

Knowing the correct positions of network nodes is essential to many functionalities in next-generation pervasive and sensor networks. For example, realizing the vision of pervasive computing (*e.g.*, [12], [41]) requires that the locations of the devices and the users be known (*e.g.*, [17], [38]); to efficiently route traffic to a geographic location (*e.g.*, [21], [44]), the nodes need to know their locations; to cover a region, the sensors use their positions to determine the quality of coverage (*e.g.*, [24]); to guide a user across a field [23], the guiding sensors need to know their positions; in order to detect events and track targets, the tracking sensors need to know their positions to pinpoint the movement of the targets and to implement efficient state transfer (*e.g.*, [45]).

David Goldenberg was supported in part by NSF Graduate Research Fellowship DGE0202738. Arvind Krishnamurthy was supported in part by NSF grant ANI-0207399. Yang Richard Yang was supported in part by NSF grants ANI-0207399 and ANI-0238038. A. Stephen Morse was supported in part by the NSF. Brian D.O. Anderson was supported in part by an Australian Research Council Discovery Projects Grant and by National ICT Australia, which is funded by the Australian Government's Department of Communications, Information Technology and the Arts and the Australian Research Council through the Backing Australia's Ability initiative and the ICT Centre of Excellence Program.

Given the importance of knowing the positions of the network nodes, much research effort has been invested into network localization, which refers to the process of determining these positions. Two straightforward methods to achieve localization are manual configuration and the Global Positioning System (GPS) [18]. However, neither methods scales well and both suffer from inherent physical limitations. For example, GPS receivers are costly both in terms of hardware and power requirements. More importantly, since GPS requires line-of-sight between the receiver and the GPS satellites, it may not work well indoors, underground, or in the presence of obstructions such as dense vegetation, buildings, or mountains blocking the direct view to these satellites.

The limitations of manual configuration and GPS have motivated the search for alternative ad-hoc methods, with a large number of localization systems having recently been proposed and evaluated (*e.g.*, [2], [3], [5], [6], [7], [8], [9], [11], [13], [14], [17], [20], [26], [27], [28], [31], [33], [35], [39], [40], [42]). The predominant type of approach, called fine-grained localization, involves nodes measuring the distances between themselves and their neighbors, with only some nodes called "beacons" having to be informed of their position through GPS or manual configuration. While some of the previous schemes are cleverly engineered, it has remained an open challenge in the field to determine precisely which nodes are *uniquely localizable*. For example, one prominent scheme has proposed that each node with three node-disjoint paths to three distinct beacons is uniquely localizable. We will see later that this is not a sufficient condition. Furthermore, most of the previous localization schemes determine node positions using optimization techniques and simply assign coordinates to non-localizable nodes corresponding to a local minimum[1]. When there are multiple configurations satisfying a given instance

[1]Most previous approaches observe that increasing node density reduces errors. However, errors due to ambiguity are obscured by the customary network-aggregate error measures.

of the localization problem, the returned configuration may not be the one that corresponds to reality. If an erroneous configuration is used by an application, for instance event detection, then incorrect or misleading conclusions may be drawn. In [10], Eren et al. proposed the construction of a *grounded graph* whose properties can be used to check the unique localizability of a network. However, their condition decides whether the *entire network*, including all of its nodes, can be uniquely localized. As we will show later, for realistic networks in many environments, it is unlikely that all of the nodes can be uniquely localized. Thus, such a collective test is likely to fail, unless the network is highly dense and regular. Furthermore, many applications can function properly as long as a sufficient number of nodes are uniquely localized, so it is not imperative that every single node be uniquely localizable.

Motivated by above observations, in this paper we propose the concept of the *partially localizable network* (PLN). These are networks in which not all nodes can be uniquely localized. We believe that partially localizable networks are likely to be the most prevalent networks in practice.

The first major challenge in studying PLNs is to identify the uniquely localizable nodes. In this paper, we present a sufficient graph-theoretic condition for a node to be uniquely localizable. Applying the condition, we identify localizable nodes by efficiently partitioning the network into components which are *redundantly rigid and triconnected*. Coordinates for the nodes in these components can then be uniquely determined subject to error due to noise in the distance measurements. We note that in simulations under realistic conditions, nodes determined to be uniquely localizable by our tests are only very rarely rendered ambiguous by errors in edge length measurements as described in [25].

Since our algorithm identifies localizable nodes efficiently, we are also able to use it to thoroughly explore appropriate network parameters in order to achieve a desired localization-dependent application goal. For the first time, it is possible to observe exactly how many nodes one can expect to be localizable in medium density and sparsely connected random networks. Using our tool, we are also able to guide the deployment of networks by adding nodes systematically to the network so as to increase the proportion of localizable nodes. As applications progress and adapt to operation in PLNs, our tool will undoubtedly find uses in more sophisticated planning and analysis tasks.

The second major challenge in studying PLNs is to determine how to best make use of nodes that cannot be uniquely localized. One possibility is for applications to simply ignore such nodes as if they do not exist, but this is clearly a worst-case option. The application setting in which we study this issue is geographic routing over PLNs. We propose an integration of geographic routing without location information and standard geographic routing. We show that by using virtual localization techniques for the non-localizable nodes, one can achieve a higher routing success rate between localized nodes than by ignoring non-localizable nodes.

The contributions of this paper can be summarized as follows.

- We propose the novel PLN paradigm. We develop efficient algorithms to ascertain which nodes can be uniquely localized and which cannot.
- Implementing our system, we conduct comprehensive experimental evaluations of network localizability, and describe implications on both network design and on the use of novel network deployment algorithms.
- We argue that a cross-layer approach involving feedback between the localization layer and the application layer should be adopted in PLNs. As an example of this, we study an integration of geographic routing without location information and standard geographic routing. We show that this cross-layer integration improves network performance.

The rest of this paper is organized as follows. In Section II, we formulate the unique localizability problem, derive conditions for a node to be uniquely localizable, and present our algorithm to identify the nodes that can be localized. In Section III, we apply the algorithm to explore the parameter space of the localization problem. In Section IV, we present our algorithms for geographic routing in PLNs. We discuss related work in Section V. Our conclusions and future work are described in Section VI.

II. Identifying Localizable Nodes in Partially Localizable Networks

In this section, we describe how to identify uniquely localizable nodes in a partially localizable network. We call this problem the *node-localizability problem*. However, before we study this problem, we first review previous results on how to check whether a complete network is localizable, *i.e.*, all nodes in the network are uniquely localizable. We refer to this problem as the *network-localizability problem*. Readers who are familiar with [10] can skip to Section II-C. Note that unless otherwise stated, we work in two dimensions.

A. Problem Formulation of the Network-Localizability Problem

In the network-localizability-problem formulation, we have a network in real d-dimensional space $\{d = 2 \text{ or } 3\}$ consisting of a set of $m > 0$ nodes labeled 1 through m which represent "beacons" together with $n - m > 0$ additional nodes labeled $m+1$ through n which represent sensors. Each node is located at a fixed position in $\mathbb{R}^d$ and has associated with it a specific set of "neighboring" nodes.

Let $\mathbb{G} = \{V, E\}$ be the network with vertex set $V = \{1, 2, \ldots, n\}$ and edge set E defined so that (i, j) is one of the graph's edges precisely in the case that nodes i and j are neighbors. The *network localization problem with exact distance information* is to determine the locations p_i of all network nodes in $\mathbb{R}^d$ given the graph of the network $\mathbb{G}$, the positions of the beacons $p_j,\ j \in \{1, 2, \ldots, m\}$ in $\mathbb{R}^d$, and the distance $l_{i,j}$ between each neighbor pair $(i, j) \in E$. The *network-localizability problem* is to determine if there is exactly one set of vectors $\{p_{m+1}, \dots p_n\}$ in $\mathbb{R}^d$ which is consistent with the given data $\mathbb{G}$, $\{p_1, p_2, \dots, p_m\}$, and $l : E \to \mathbb{R}$. If there is exactly one set, we say that the network is localizable.

We can see that the network-localizability problem is closely related with the Euclidean graph realization problem in which coordinates are assigned to vertices of a weighted graph such that the distance between coordinates assigned to nodes joined by an edge is equal to the weight of the edge. Clearly, if the graph $\mathbb{G}$ of a network is uniquely realizable, the network is localizable up to global rotations, translations, and reflections.

However, this connection to Euclidean graph realization is still incomplete. The network-localizability problem is not equivalent to the unique realizability of $\mathbb{G}$, but of a graph with a slightly larger edge set that includes edges from every beacon to every other. We call this augmented graph the *grounded graph* of the network. In the graph abstraction, an edge represents a distance constraint between its endpoints. Since the distance between each pair of beacons is known, edges must exist between all pairs of beacons. This observation is crucial in order to capture all available constraints in the network. If the grounded graph of the network is uniquely realizable, then the network is uniquely localizable up to rotations, translations, and reflections. In two dimensions, three non-collinear beacons are necessary to resolve the global orientation of the network to a single possibility.

B. A Sufficient and Necessary Condition for Network Localizability

Now that we have defined the network-localizability problem and its grounded graph abstraction, we can proceed to describe the precise conditions for unique localizability.

We must first state that the following is a *generic* characterization of unique realizability, *i.e.*, one that holds for *almost all* configurations of network nodes. What this means is that for all network configurations other than a set of configurations containing certain degeneracies among node positions, unique localizability is a graph-theoretic property of the network connectivity and independent of the positions of the nodes. Given this fact, randomization aids in the classification of networks as uniquely realizable. For any reasonable probability distribution on node positions, degenerate configurations have zero probability of appearing, and one can be justified in assuming the network nodes to be in general position. It is worth noting however, that in the presence of errors in edge length measurements, configurations indistinguishable from degenerate may very well occur. We begin with descriptions of the three ways in which a graph can fail to have a unique realization.

1) Not rigid
A realization of a graph may be subject to deformations that allow the coordinates assigned to vertices to vary continuously while simultaneously satisfying all of the edge constraints. This non-uniqueness can either be continuous or discontinuous. A graph is *flexible* if it admits a continuous deformation other than global rotation, translation, and reflection; otherwise, it is called *rigid*. Flexible graphs have an infinite number of realizations. Fig. 1 shows a flexible network. The figure also provides a counter example to the claim that if a node has three node-disjoint paths to three distinct beacons, it is uniquely localizable.

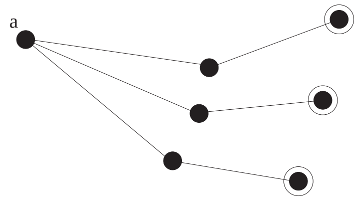


Fig. 1. An example flexible graph. It also shows that although node a has three node-disjoint paths to the three beacons on the right (the implicit edges among the beacons are not drawn), the position of node a is not unique. For example, imagine dragging node a to the left.

Flexibility arises due to unconstrained degrees of freedom in the graph structure. One can see that in two dimensions, a graph of n nodes has at most $2n$ degrees of freedom. Each edge constraint eliminates at most a single degree of freedom, but there are 3 degrees of freedom corresponding to the rotation and translations of a rigid body that cannot be eliminated by any number of edges. One would guess then that at least $2n-3$ edges are necessary for rigidity in a graph of n vertices. However, having the requisite total number of edges is not sufficient for rigidity, as the edges could all be crammed between only a few vertices, leaving the rest under-constrained. The intuition then, is that $2n-3$ *well-distributed edges* are needed. More precisely, in graphs with $2n-3$ edges, no subset of n' nodes may have more than its fair share of $2n'-3$ edges between its nodes. If a subgraph has more than $2n'-3$ edges, some of them are *redundant*. Non-redundant edges are called *independent*. Each independent edge eliminates a degree of freedom in the structure, so the presence of $2n-3$ independent edges is sufficient for rigidity. This intuition turns out to be correct in two-dimensions, resulting in the well-known Laman condition [22]. Unfortunately, this argument does not hold in three dimensions and no graph-theoretic characterization of rigidity for dimensions greater than 2 is yet known.

Theorem 1: (Laman) The edges of a graph $G = (V, E)$ are independent in two dimensions iff no subgraph $G' = (V', E')$ has more than $2n'-3$ edges, where $n' = |V'|$.

Corollary 1: A graph having n vertices and $2n-3$ edges is rigid in two dimensions iff no subgraph (V', E') has more than $2n'-3$ edges, where $n' = |V'|$.

2) Not $d+1$-connected
Rigid graphs are still susceptible to discontinuous non-uniqueness. Specifically, they may be subject to fold ambiguities in which a set of nodes have two possible configurations corresponding to a "reflection" across a set of mirror nodes as shown in Fig. 2. This type of ambiguity is not possible in $d+1$-vertex-connected graphs.

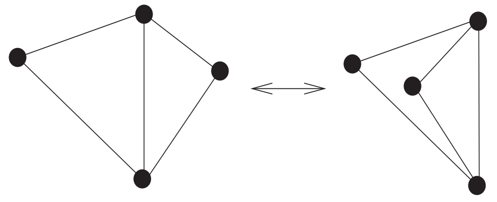


Fig. 2. An example showing two realizations due to reflection.

3) Not redundantly rigid
A $d+1$-vertex-connected rigid graph may still be subject to a flex ambiguity. Fig. 3 shows a triconnected rigid graph which becomes flexible upon removal of an edge. More specifically, after the removal of an edge, a subgraph can swing into a different configuration in which the removed edge constraint is satisfied and then reinserted. This type of ambiguity is eliminated by *redundant rigidity*, the property that a graph remains rigid upon removal of any single edge.

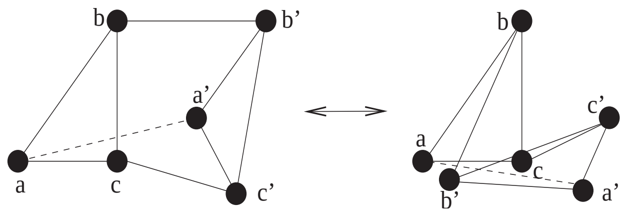


Fig. 3. An example from [15] showing a rigid triconnected graph with two realizations. If edge (a, a') is removed, triangle $a'b'c'$ swings along a path until the distance (a, a') is the same as it originally was.

Summarizing the conditions for eliminating ambiguities in graph realization, we have the following:

Theorem 2: A graph G with $n \geq 4$ vertices is uniquely realizable in two dimensions iff it is redundantly rigid and triconnected.

Hendrickson [15] proved the necessity of triconnectedness and redundant rigidity for unique graph realizability, and it was recently shown by Berg and Jordan [4] that it these conditions also sufficient.

Recalling that three non-collinear beacons are necessary for unique network localizability to absolute positions, we have the following result:

Theorem 3: ([10]) Let $\mathbf{N}$ be a network in $\mathbb{R}^2$ containing at least three non-collinear beacons. Let the grounded network graph $\underline{\mathbb{G}}$ be the graph whose vertices correspond to the network nodes and whose edges include all neighbor pairs and all beacon pairs in $\mathbf{N}$. The network is localizable in two dimensions if and only if $\underline{\mathbb{G}}$ is redundantly rigid and triconnected.

C. Identifying Uniquely Localizable Nodes

The results in the preceding subsection for network localizability can be extended to the node-localizability problem. To determine if a node is uniquely localizable, we first have the following necessary condition:

Lemma 1: If a node is uniquely localizable, it must have three node-disjoint paths to three distinct beacons.

Proof: Suppose the underlying (grounded) graph is $G = (V, E)$. Suppose that there are three beacon nodes and v is a localizable node.

Let a node w located at the centroid of the three beacon nodes be adjoined to G. Furthermore, suppose that w is a degree 3 vertex with edges joining w to each of the three beacon nodes. Call the new graph G'. Then clearly w is a localizable node of the graph G'.

Now consider the nodes v and w, and let r be the maximum number of nonintersecting paths that will join them. Since w has degree 3, $r \leq 3$. If $r = 3$, there is one nonintersecting path from v to w passing through each beacon node. If in each of these paths the edge joining w to the beacon node is deleted, there results three nonintersecting paths joining v to each of the three beacon nodes.

So suppose, in order to derive a contradiction to the result we are trying to prove, that $r < 3$. Then G' cannot be triconnected. So there exists a separating vertex pair x and y such that $G' = H_1 \cup H_2$ with the vertex set of the intersection of the two subgraphs H_1 and H_2 given by $V(H_1 \cap H_2) = \{x, y\}$ and with H_1 and H_2 possessing three or more vertices each (including x and y).

We can argue that the three beacon nodes and w can be taken to lie in one of the subgraphs, without loss of generality H_1. For suppose this is not the case. Then of the four nodes, at least one must lie in H_1 and not in H_2, and one must lie in H_2 but not in H_1. As a consequence, these last two nodes cannot be adjacent. But since the three beacon nodes and w form a complete graph, these last two nodes are necessarily adjacent, which is a contradiction.

The vertex v may lie in H_1, or in H_2 and not in H_1. We analyze these two possibilities.

Under the second possibility, a reflection argument (reflecting about the line joining x and y) can be applied to conclude that v is not localizable, which is a contradiction.

If $v \in H_1$ and there is a path in H_2 from x to y, then we replace G' by H_1 augmented with a new edge (x, y) and repeat the process that decomposed G' as $H_1 \cup H_2$. If $v \in H_1$ and there is no path in H_2 from x to y, we replace G' simply by H_1 and repeat the process. In either case, denote the replacement of G' as H_1'. Within H_1', the maximum number of nonintersecting paths that will join v to w must be less than 3, else there would be a contradicion of the property that in G', there are less than three nonintersecting paths between v and w. Hence H_1' is not triconnected. This argument can be repeated, until we arrive at a graph L which is not triconnected, a decomposition of L as $L = L_1$ and L_2 given by $V(L_1 \cap L_2) = \{x_L, y_L\}$ such that the beacon nodes and w lie in L_1, while v lies in L_2 but not in L_1. That such a graph must eventually be found follows from the fact that at each step in the process, the number of vertices in the subgraph containing the three beacon nodes and w decreases, while it is clearly bounded below.

At this point a reflection argument can be applied to conclude that v is not localizable, which is a contradiction. ■

Lemma 1 notwithstanding, Fig. 1 shows that the existence of three-node disjoint paths is not the only necessary condition for node localizability. Intuition tells us that the node should also belong to some "rigid" structure. In addition, Fig. 3 provides an example suggesting that a uniquely localizable node should belong to a subgraph that is even "stronger" than rigid. However, whether it is necessary that uniquely localizable nodes belong to some redundantly rigid subgraph is an open problem at this time.

A sufficient condition which follows from Theorem 3 requires that a uniquely localizable node belong to a redundantly rigid subgraph that is triconnected and contains three beacons. We call this condition the *RRT-3Beacon condition*, shortened here to RRT-3B. This condition allows us to identify uniquely localizable nodes one component at a time, instead of one node at a time. Before we discuss how to identify RRT components, we comment that since the RRT-3B condition is a sufficient condition, it may identify only a subset of all of the uniquely localizable nodes, as illustrated by Fig. 4. A necessary and sufficient condition for generic node localizability is not yet known.

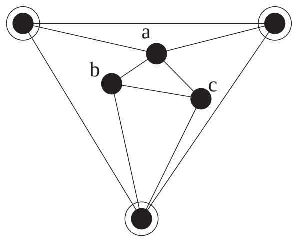


Fig. 4. An example showing that the RRT-3B condition fails to identify node a as uniquely localizable. The three nodes on the boundary are beacon nodes. Node a is uniquely localizable despite the fact that the graph is not triconnected.

To identify the RRT components, we first extract all triconnected subgraphs. There are multiple ways to test for the triconnectivity of a subgraph. We do this in the simplest possible way; for each vertex in a subgraph, we remove it, test the reduced component for biconnectivity, and replace the vertex. If the subgraph remains biconnected after removal of each of its vertices, then the subgraph is triconnected. We then use the Pebble Game [19] to discover and tag redundantly rigid subgraphs.

1) Identifying redundantly rigid subgraphs using the Pebble Game: Recalling Laman's condition for rigidity in the plane, we see that as stated it suggests a very poor algorithm for testing graph rigidity involving counting the number of edges of all exponentially many subgraphs. There are several alternative formulations of Laman's condition that yield polynomial time algorithms. The Pebble Game uses the following formulation that results in a particularly intuitive algorithm.

Theorem 4: For a graph $G = (V, E)$ with n vertices, the following are equivalent:

- The edges of G are independent in two dimensions.
- For each edge (a, b) in G, the graph formed by adding three additional edges (a, b) has no n' vertex subgraph with more than $2n'$ edges.

This formulation leads to an algorithm in which a set of independent edges is grown by adding one edge at a time. A new edge is added if it is determined to be independent of the existing set. If $2n-3$ independent edges are found, where n is the number of vertices in the graph, then the graph is rigid. We will see that the structure of the problem allows us to efficiently check new edges for independence.

To elaborate, assume we have a set of independent edges $\hat{E} \subset E$. Recall that the independence of edges in a graph can be conceptualized as a property of "well-distributedness". As such, to determine if another edge $e \in E$ is independent of $\hat{E}$, we see by the alternative Laman condition that e is independent of $\hat{E}$ if and only if there is no subgraph with too many edges (i.e., n' vertices and $> 2n'$ edges) after any edge in $\hat{E} \cup e$ is "quadrupled" (three additional copies added). It turns out that only the candidate edge e needs to be quadrupled (see [19] for proof), meaning that the complexity of testing a graph for rigidity amounts to $O(n)$ times the complexity of checking a graph for the well-distributedness of its edges.

In order to test a graph for the well-distributedness of its edges, we use the following "pebble game". Every node is given two pebbles which it must keep, each of which can be used to "cover" an edge incident upon it. We would like to assign pebbles to edges so that all the edges in the graph are covered. Such an assignment is called a *pebble covering*. The existence of a pebble covering is equivalent to there being no n' node subgraph with more than $2n'$ induced edges. Stepping back for a moment, we see that edge e is independent of $\hat{E}$ if and only if there exists a pebble covering once e is quadrupled.

We will now discuss how to find a pebble covering. Assume inductively that we have a set of edges already covered by pebbles and we want to add a new edge. We first look at the node endpoints of the new edge. If either of them has a free pebble, it can be used to cover the new edge, and we are done. If neither of them has a free pebble, then both of their pebbles are being used to cover existing edges. If the node at the other end of one of these existing edges has a free pebble, then we can use that pebble to cover the existing edge, freeing up a pebble to cover the new edge. The general procedure is to search along a directed version of the underlying graph with edges directed away from their pebble-covered end until a free pebble is found, use that free pebble to cover the last edge, and perform a series of swaps reversing the direction of all the edges along the successful search path until a pebble is freed up at an endpoint of the new edge.

In order to test a graph for rigidity, at most $n(n-1)/2$ edges will be tested for independence. Each independence test involves 4 pebble searches, each of which requires $O(n)$ time, for a total of $O(n \cdot |E|)$ time, where $|E|$ is the number of edges in the graph. For a network with edges only between nodes located close to each other, the number of edges will be $o(n)$. Therefore, the running time of the Pebble Game on sensor networks is $o(n^2)$.

The Pebble Game is attractive for its intuitive appeal as well as for its efficiency. Each pebble can be interpreted to represent a degree of freedom of the node it belongs to. We have seen that there cannot be more than $2n-3$ independent edges between n nodes. Therefore, at all times, there will be at least 3 free pebbles in the assignment, representing the three degrees of freedom of any rigid body in two dimensions. Because of this, three copies of an edge will be always be successfully covered. If the fourth copy is not pebble coverable, then the current set of independent edges already consists of $2n'-3$ edges connecting n' nodes, and we have identified a fully constrained portion of the network. Because of this, redundantly rigid regions of the graph are identified as the edges and nodes traversed by failed pebble searches. In this way, the Pebble Game is an efficient tool to identify the redundantly rigid regions of a graph.

2) An algorithm to identify RRT components: In order to discover all of the RRT components of the network, it is not enough to simply intersect the triconnected components with the redundantly rigid components, as this intersection will not necessarily possess both properties. We employ a recursive decomposition in which the algorithm for triconnected component discovery alternates with the Pebble Game for redundantly rigid component discovery. Our algorithm is outlined in Fig. 5.

```
1  FindRRTComponents(Graph G)
2       if not triconnected then
3         recurse on each triconnected component
4       else if not redundantly rigid then
5         recurse on each redundantly rigid component
6       else return "graph G is an RRT"
```

Fig. 5. A recursive algorithm to identify RRT components.

III. Partially Localizable Networks: Experimental Investigations

The algorithm developed in the preceding section allows us to investigate at least three questions that could not heretofore be addressed: 1) in what deployment scenarios will non-localizable nodes comprise a significant proportion of the network, 2) how does the presence of non-localizable nodes affect the performance of typical location-dependent network functionalities, and 3) how might one deploy networks so as to optimize for localizability. In this section, we motivate and address these three questions.

A. Percentage of Localizable Nodes

We first evaluate the incidence of non-localizable nodes in typical scenarios. We generate random placements of nodes in a region according to two distributions: uniform and Gaussian. Uniform node placement is commonly used in simulations, even though Gaussian is likely to more accurately model practical random node deployments[2] (*e.g.,* nodes scattered from aircraft). We assume that two nodes can measure their separation distance if they lie within a given radius of one another. Imperfections in this unit disk model will merely serve to reduce connectivity, and hence localizability, so this study explores the best case scenario for localizability in ad-hoc networks.

The percentage of nodes found to be localizable by the RRT-3B condition as we increase the number of nodes placed in a fixed region is shown in Fig. 6. Throughout these experiments, all results are obtained by creating 20 instances of 100-node networks with the desired parameters and calculating the mean and 95% confidence interval for the relevant quantity. Node density is expressed as the expected node degree given the number of nodes uniformly placed in the field and the sensing radius of each node, neglecting boundary effects.

As expected, the percentage of localizable nodes increases with density. Nevertheless, even at expected node degree as high as 15, the percentage of non-localizable nodes remains significant. As the Gaussian distribution tends to produce networks with high connectivity in a central region and a sparsely connected periphery, average degree is not a very good characterization of these networks. Nevertheless, due to these properties of the distribution, at low expected node degree, more nodes are localizable under the Gaussian than the uniform distribution, whereas for higher expected degree, more nodes are localizable under the uniform distribution.

To observe how the presence of beacons affects the localizability of network nodes, we produced Fig. 7. For three different densities, we vary the number of beacons and observe how the number of localizable nodes changes. We can observe that the addition of beacons exhibits diminishing rewards in terms of the number of localizable nodes. By adding beacons past the point at which 10% of the nodes are beacons, the number of nodes rendered localizable per beacon is less than one. Since beacons are likely to be relatively expensive, such an approach is unlikely to be viable.

We also generated regular networks using a fixed-density concentric ring deployment with beacons at the periphery as in [32]. This deployment is by design isotropic, yet we still find that the number of localizable nodes varies as in Fig. 6, so we do not show the results here.

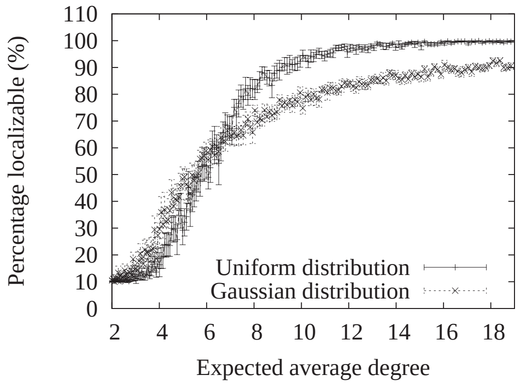


Fig. 6. Percent localizable nodes vs. density. Results shown are for 100-node randomly deployed networks including 10 beacons.

[2]With the caveat that the arbitrarily large deviations from the mean allowed by the Gaussian distribution are unrealistic.

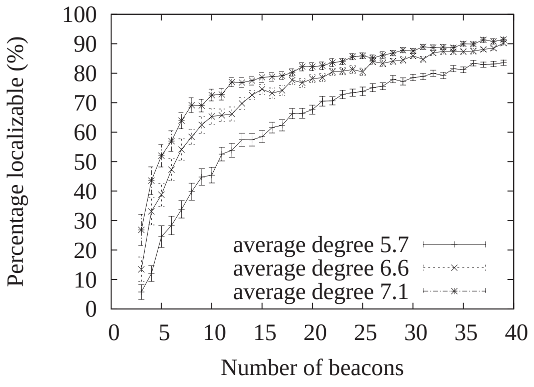


Fig. 7. Percent localizable nodes vs. number of beacons.

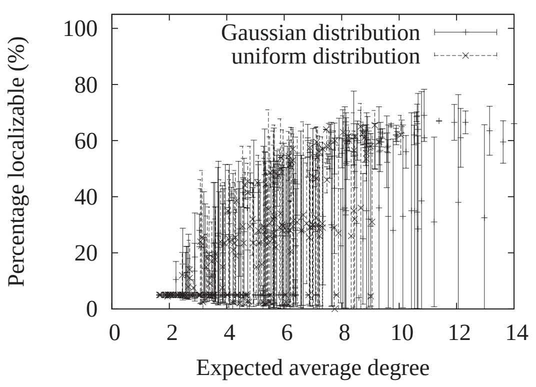


Fig. 9. Percent localizable nodes vs. density using obstacle map shown in Fig. 8. Results shown are for 5 beacon networks. We decided against using rank control in order to show extreme variance of confidence intervals in obstacle map domains.

In random deployments in realistic environments, networks are likely not to be isotropic. Distance measurements may be impaired by obstacles, degrading node localizability. We observe the effects of obstacles on the percentage of nodes that cannot be localized in the College campus environment shown in Fig. 8 and plot our results in Fig. 9. In this experiment, an edge exists between two nodes only if the straight line between them does not pass through any obstacles. We observe that the presence of obstacles renders a substantial percentage of nodes unlocalizable. In addition, the proportion of localizable nodes in such an urban environment is more unpredictable than in open ones, as evidenced by the much wider range of the 95% confidence intervals.

Overall, we observe that for virtually all practical node densities, randomly deployed networks are only partially localizable. Thus, in order to realize the potential of random ad-hoc networks, it is essential to more fully explore the PLN paradigm.

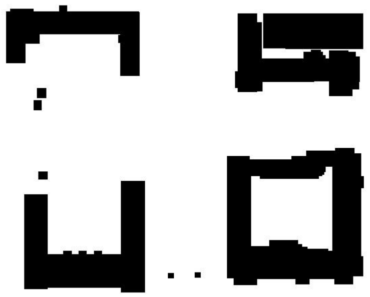


Fig. 8. Obstacle map used for all evaluations involving obstacles. Map is of Yale University's Cross Campus plaza.

B. Coverage of PLNs

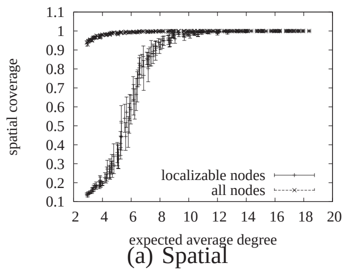
The percentage of localizable nodes, while an important quantity, is hard to associate directly with application performance. The impacts of partial localizability on networks can be better evaluated by considering specific network metrics relevant to some of the envisioned applications that require knowledge of node positions. Towards this end, we evaluate the behavior of network coverage metrics in partially localizable networks.

We define four metrics to evaluate coverage performance. Each of these metrics has its own merits depending on the specifics of the considered application.

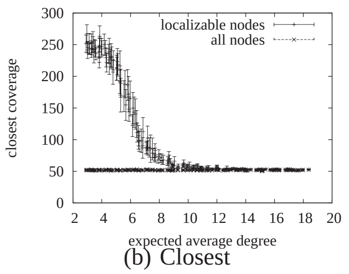
- *Spatial coverage*: the likelihood that a position chosen uniformly at random is within sensing range of a node. This metric expresses the chance an event in the region of interest will be observed by some node.
- *Closest coverage*: for a point, the distance between it and its nearest sensor node. For a network, this metric is defined as the mean value of closest coverage over points chosen uniformly at random in the area of interest. This metric reflects the most closely one can expect an event in the field to be observed.
- *Worst-case coverage*: the largest distance between a point in the network domain and a sensor node. This metric expresses how poorly an event can possibly be observed in the field.
- *Aggregate coverage*: a measure of the aggregate sensing "quality" of points in the field. Suppose a node at distance d from an event achieves a sensing quality of $1/d^2$ and that sensing quality from multiple nodes is additive. The aggregate coverage of that event is defined as the distance away from the event at which a single sensor would need to lie in order to achieve by itself the same sensing quality as the network. The aggregate coverage of the network is the expectation of aggregate coverage taken over points in the field chosen uniformly at random.

Note that these metrics could be generalized into multiple node measures that would be relevant to applications requiring events to be detected by multiple sensors. For instance, spatial coverage could be varied to read: the likelihood that a position chosen uniformly at random is within the sensing range of p sensors (where p is some fixed integer).

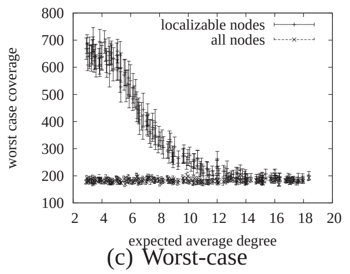
Fig. 10 shows the single sensor coverage achieved by all nodes, regardless of their localizability, and the coverage achieved by localizable nodes only. We make a distinction between localizable and non-localizable coverage because the value of sensed data may depend heavily on the ability



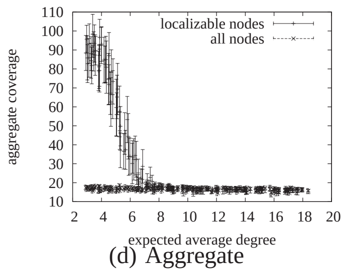
(a) Spatial



(b) Closest



(c) Worst-case



(d) Aggregate

Fig. 10. Coverage performance of 100 uniformly distributed nodes in a 1000 by 1000 field.

to associate it with a physical position. Detection by non-localizable nodes may still be of use, but it is important nevertheless to make this distinction. We observe that for a wide range of network density, coverage by all nodes and coverage by localizable nodes have very different values under all four coverage metrics. More specifically, all-node coverage does not exhibit much improvement over the evaluated density range, while localizable-node coverage requires a much higher density to achieve performance comparable to all-node metrics. By making a simple conversion from node degree to sensing range, it can be seen in part b) of Fig. 10 that for node degree all the way up to 6 neighbors on average, the expected closest coverage of an event in the field is *greater than the sensing range*. This means that even at moderate density, a randomly placed event will not be sensed by a localizable node in the average case. Looking at parts a) and c), we can see that even at the average node degree of 10 at which localizable spatial coverage is equal to all-node spatial coverage, sizeable gaps often occur in the localized coverage, as evidenced by the high values in part c) of localizable worst-case coverage as compared with all-node worst-case coverage. The coverage results for Gaussian distributions of nodes are similar to these shown for the uniform case.

These results are further evidence that the performance of many envisioned sensing applications is likely to be dramatically affected by the localizability properties of the network.

C. Beacon Placement through Smart Deployment

In this section, we will discuss practical and novel methods of smart deployment made possible by our localizability algorithm. We have seen that non-localizability of network nodes significantly impacts some of the network metrics relevant to typical location-dependent applications. The methods evaluated in this section seek to mitigate such effects by yielding networks with fewer non-localizable nodes than random deployment.

We first apply our localizability tool to guide the deployment of beacons. In this section, we use a network deployment model in which beacons can be placed approximately at a targeted location. As an example, this could be achieved in practice by firing a specially outfitted beacon node towards a target from a mortar launcher, or by using mobile beacons. We are motivated by the consideration that the relative expense of beacon nodes will likely make deterministic placement worthwhile.

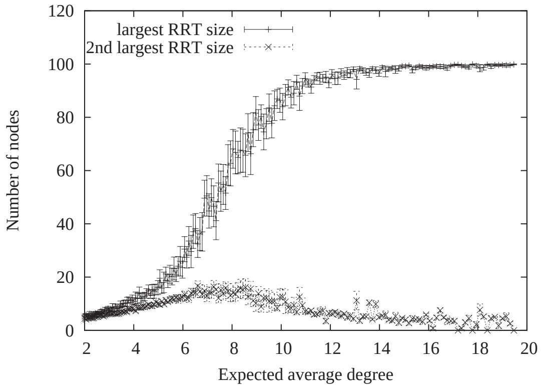


Fig. 11. Statistics regarding RRTs in 100-node networks with no beacons.

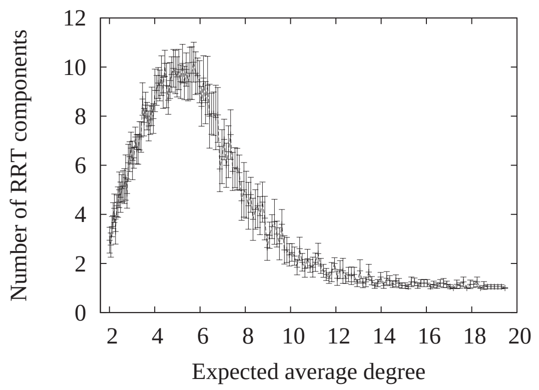


Fig. 12. The distribution of the number of RRTs in 100-node networks with no beacons. The addition of beacons shifts the curve slightly down and to the left.

At its core, *smart deployment* uses our tools to partition the graph into its RRT components. Each of these components would be localizable if it were to contain three beacons. Guided by this observation, we seek to place three beacons in RRT components so that the maximum number of non-localizable nodes will be rendered localizable. Of course, before there are beacons in the network, it is impossible to determine precise target points for beacon deployment. Therefore, we initially try to randomly place beacons into RRTs before then placing additional beacons deterministically. Note that this approach may be too conservative in that there may be non-RRT components that could be rendered RRT by the addition of three beacons at appropriate positions. As we have no way of identifying these situations, we adopt the conservative approach.

Shown in Fig. 11 is the distribution of the size of the largest and second largest RRT components versus average node degree. In Fig. 12 is shown the distribution of the number of RRTs. We can see from these figures that above average node degree 10, the network consists usually of one large RRT and only one or two other small RRTs. At these density levels, random beacon placement yields very good localizability performance. At lower densities, the potential exists for some improvement over random deployment. However, through simulation of a best-case oracle scheme, we found that this potential gain is rather small except at very low density (only greater than 20% for average node degree below 5). Because of this, we study smart beacon deployment on non-isotropic networks of Gaussian distributed clusters with uniformly distributed cluster centers. This type of network may be a good model for potentially important network deployment scenarios.

Our smart deployment algorithm randomly deploys m beacons, where m is the number of RRTs in the network. Next, it places additional beacons deterministically near placed beacons connected to an RRT until the entire connected RRT is made localizable. The results of smart deployment shown in Fig. 13 are compared against uniform random deployment which inserts all beacons uniformly at random. We see that even our simple scheme results in large performance gains for anisotropic networks by virtue of it being aware of the number and size of the RRT clusters.

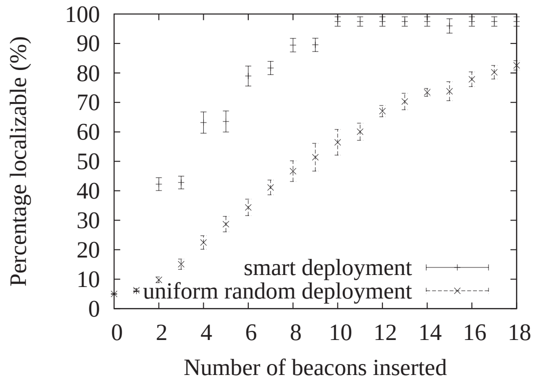


Fig. 13. Percent localizable nodes vs. number of beacons inserted. In this figure, m beacons were initially placed at random, where m is the number of RRTs in the network. Subsequently, up to $2m$ additional beacons are placed by the two algorithms.

D. Event-Based Network Training

The other smart deployment paradigm we introduce in this section we call *event-based training*. This is a novel approach consists of placing *events* in the network field to which network nodes can measure their distance. For instance, if the network nodes use time difference of arrival ranging, the events could be simultaneous ultrasound and RF bursts produced by inexpensive disposable devices designed specifically for this configuration purpose. When an event is detected by network nodes, it is treated as if it were a node for purposes of localization. Time-synchronization will be necessary for this scheme, and a deployment of training events could also follow a staggered pattern between potentially interfering sites. Using this technique, it is possible to greatly increase the effective network density available to the localization layer for a one-time computation of positions, and then reaping the cost benefits of a sparser network for extended operation. It is of course possible that the event-based distance mesasurements may be less accurate than inter-node measurements, but for simplicity we do not go into the details in this paper. However, we introduce a novel formulation of joint source and network localization that arises from this issue in the Appendix.

We evaluate the feasibility and potential benefit of event-based training by considering two simple algorithms. The first is a uniform dispersal of events over the network field. As expected, this dispersal is equivalent to an increase in density of the network and results in an increase in the number of localizable nodes consistent with Fig. 11. We investigate here whether it is possible to do better than this by deploying events with some control. We assume that positions for the localizable nodes in the network are computed before event deployment. We introduce a random jitter between the targeted position and actual deployment position of each event. This simulates realistic deployment uncertainty, as well as small errors in the computed positions of localizable nodes.

Through evaluation of various techniques, we found that a hybrid approach which switches between two methods depending on the density level performs best. In each method, each potential event deployment position is repeatedly chosen uniformly at random until it satisfies a certain condition. At low average node degree, (less than 5.8), the condition is that the event position must be within range of at least one beacon and at most 3 localizable nodes exclusive of beacons. At high densities, we ensure that the distance between the target and the centroid of the set of positions of localizable nodes within range of the target is at least half the sensing radius.

The rationale for these approaches can be understood by referring to Fig. 12. At densities below the peak in the number of RRTs, the network may contain a few small RRTs, but is not quite dense enough to allow widespread localization. By adding events close to beacons, density is locally increased in those regions of the network that have the potential to become immediately localizable by virtue of the presence of beacons.

This approach does not work in the denser networks, as the problem becomes less one of connecting beacons to clusters of unlocalizable nodes on the verge of becoming part of an RRT, but rather one of connecting outlying unlocalizable nodes in local voids to the few large RRTs in the network. In dense networks, requiring events to be within range of only 3 localizable nodes places them uselessly in obscure corners of the network away from the large RRTs. Because of this, we instead place events so that they are not too close to localizable nodes, but still likely to be within range of the large RRTs, so that they can bridge the gaps to isolated nodes.

We observe in Fig. 15 that the hybrid method compares favorably with uniform deployment for all densities other than those corresponding to the peak in the number of RRTs. At this point, networks consist of a few medium-sized RRTs and many small RRTs. It has proven difficult to exploit structure in order to outperform uniform deployment for such networks.

All the methods evaluated had very unpredictable performance on this regime. While these event-based training methods do not yield different asymptotic behavior from the random deployment of additional nodes, we can see in Fig. 16 that the performance gains are quite robust. This is especially true for sparser networks, which are likely to be important in any large-scale random network deployment, as locally sparse regions will surely arise as problem regions in such deployments. Performance along the orthogonal axis of number of training events is plotted in Fig. 14.

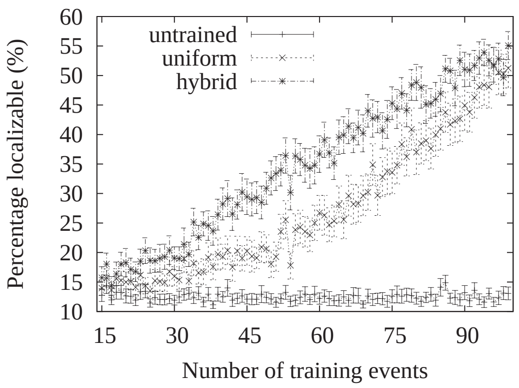


Fig. 14. Event-based training on a 100 node network with 10 beacons and an average node degree of 3.1. The trends are similar for higher densities.

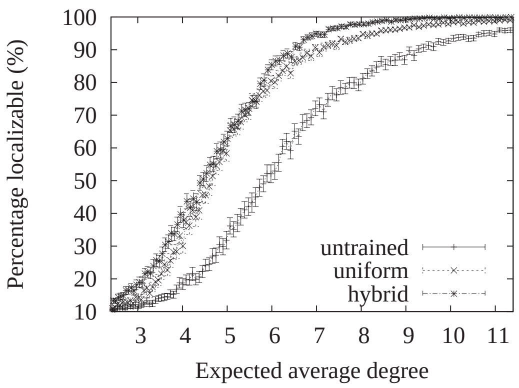


Fig. 15. Event-based training on a 100 node 10 beacons network deploying 30 training events. 100 samples were taken for each data point.

IV. Geographic Routing in Partially Localizable Networks

We have now established the likelihood that a significant proportion of nodes in realistic networks will be non-localizable, especially under deployments that do not explicitly seek to optimize for localizability. Furthermore, we have seen that this can have a significant impact on network performance as measured by various coverage metrics. Now we show how an important location-dependent application, namely geographic routing, is affected by the presence of non-localizable nodes.

In this section, we evaluate the performance of geographic routing on PLNs and propose methods to improve routing

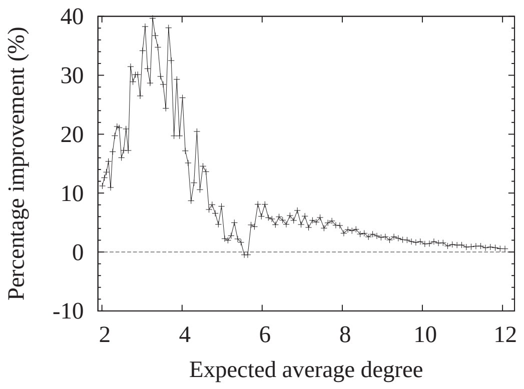


Fig. 16. Percent increase in the number of localizable nodes using hybrid training as compared with uniform training. We switch event deployment conditions at an average node degree of 5.8.

performance in such scenarios. Note that our main objective is to illustrate the relevance of the PLN paradigm and thus we use simplistic models.

A. Problem Formulation

We consider a geographic-routing application in which a network of sensor nodes is deployed in a field. The nodes perform localization, but as we have seen, some nodes can determine their positions while others cannot. Users may issue queries to the network specific to a destination position. For example, a user may want to query the temperature reading at the sensor which is at (or closest to) a given position, and will need nodes to route messages across the network to a particular physical location.

B. Solution Technique

Our solution technique for determining node positions is a three step process. First we determine the RRT components using the decomposition algorithm in Fig. 5. Second, upon those RRTs that contain at least three beacons, we use a standard localization method, MDS, to localize the nodes contained therein. We use the MDS-MAP implementation of MDS presented in [36]. Third, we compute estimates for the positions of the remaining nodes, using a position averaging technique presented in [29], wherein each of the non-localizable nodes computes a virtual position for itself by repeatedly taking the average of its neighbors' positions until convergence is reached. We refer to this scheme as *RRT + avg*.

We choose greedy routing to be our geographic routing algorithm, due to its simplicity and requirement for node positions. While the aim of this study is to clearly elucidate the issues surrounding geographic routing on PLNs, we would note that by using more sophisticated routing techniques, as in [34], one could further reduce routing error.

C. Localization in PLNs

Before we investivate geographic routing in PLNs, we discuss the actual localization process in these networks. As mentioned above, we use MDS-MAP to compute the positions of localizable nodes in the network and estimate the positions of non-localizable nodes using averaging. When running *RRT + avg*, the physical location error can be reduced at lower connectivities in comparison with running MDS over the entire graph, as shown in Fig. 17. The key observation to be made here is that by identifying RRTs and running *RRT + avg*, physical location error will decrease and the routing success rate will increase, supporting the value of our decomposition tools. By identifying localizable portions of the network, we avoid computing incorrect positions for non-localizable nodes.

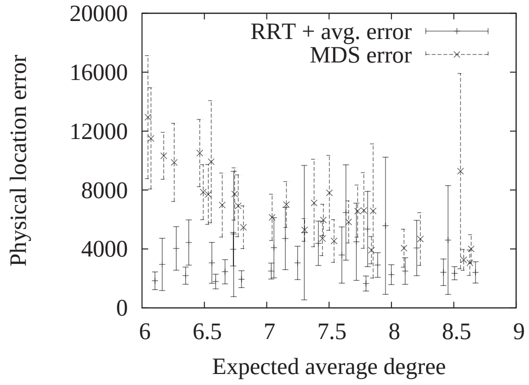


Fig. 17. Physical location error vs. connectivity. Results shown are for 100-node randomly deployed networks including 10 beacons.

A potential flaw in this approach, as noted in [25], is that in real-life scenarios, due to uncertainty in edge measurements, nodes determined to be uniquely localizable by generic conditions may not in fact have uniquely determined positions. Through simulation, we find that this does occur, although infrequently enough that the RRT decomposition remains a useful tool in realistic deployments.

We simulated uniform random deployments of 20 node networks containing 5 beacons with zero-mean Gaussian edge length measurement error with a standard deviation of 5% of the sensing range over a wide range of expected node degree. We ran a localization algorithm consisting of MDS followed by Kalman Filter based refinement on both the set of exact edge lengths and on the noisy edge lengths. Nodes which are localized with a small error when using the exact edge lengths but which exhibit a large localization error when using the noisy edge lengths are potentially those nodes upon which the unique localizability conditions break down in the presence of errors.

For expected degree greater than 6, due to the redundancy in the connectivity of RRT components, such nodes are almost never present. However, the minimally rigid structures prevalent at lower connectivity are indeed susceptible to becoming ambiguous due to edge measurement errors. An example of such a situation is shown in Fig. 18. In this example, in localizing the node at the top right using noisy edge length data, the network is flipped into a faulty configuration. The common situation however, which occurs for more than 90% of the tested random networks of average degree 6, is similar to that shown in Fig. 19. In these cases, the computed positions exhibit computation error but are not mislocalized into a faulty network configuration.

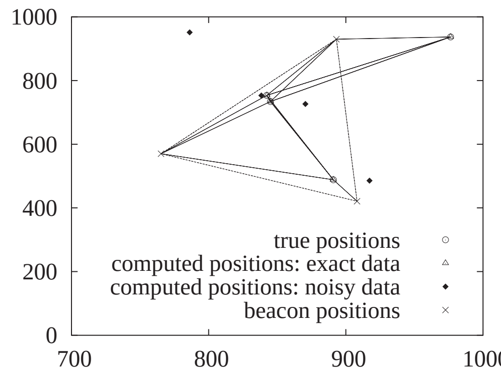


Fig. 18. Faulty outcome in localization with errors in which top right node is mislocalized. Solid lines represent distance measurements, dashed lines represent implicit inter-beacon distance measurements

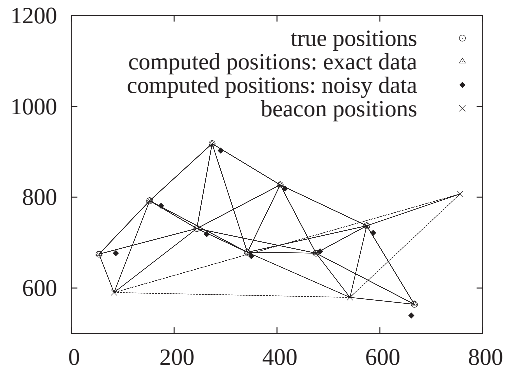


Fig. 19. Common outcome in localization with edge length errors. Solid lines represent distance measurements, dashed lines represent implicit inter-beacon distance measurements

D. Experimental Investigations: Geographic Routing

Returning to geographic routing in PLNs, we measure and compare the routing error obtained through various approaches to the problem. We define routing error as the distance between the target point of a query to the final point reached by routing, measured for all pairs of nodes between which we attempt to route a message.

We evaluate greedy routing in three different settings. We first establish a lower-bound on routing error by routing using the true positions. We then measure the routing error when routing is performed using the positions obtained by *RRT + avg*. We also measure the improvement in error as a result of performing a limited one-hop multicast of the query after the greedy routing terminates at a position. Fig. 20 presents the performance results. Again, as stated above, the routing error using positions obtained by *RRT + avg* is close to that of the lower-bound.

These error results are evidence that the performance of geographic routing is affected in PLNs. Further, once a portion of the network is identified as an RRT, applications such as greedy routing can exploit this knowledge and increase performance using hybrid schemes such as *RRT + avg*.

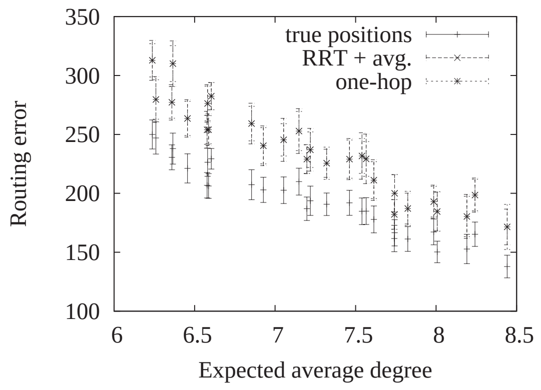


Fig. 20. Routing error vs. connectivity. Results shown are for 100-node randomly deployed networks with 10 beacons.

V. Related Work

Network localization is an active research field (see [17] for a survey). The previous approaches can be classified into two types: coarse-grained and fine-grained. The focus of this paper is fine-grained localization in which some nodes know their locations, and the distances between proximal pairs of nodes are measured. As we discussed in the Introduction, a major shortcoming of previous studies on fine-grained localization is that they cannot correctly identify the nodes that can be localized.

A problem related with network localization is the molecular conformation problem studied in the Chemistry community (*e.g.*, [16]). However, the focus of these studies is on three dimensions. Also, issues of network design and deployment are not studied, since the structure of molecules is fixed.

One major building block of the uniqueness condition is rigidity theory. Rigidity has long long studied in mathematics and structural engineering (see for example [15], [30], [43]) and has a surprising number of applications in many areas. This paper builds on the results from [10] by using grounded graphs for localization. However, the objective of [10] is to check the unique localizability of the entire network and thus does not identify subsets of nodes which can localize and consider applications of this identification.

We use the Pebble Game developed by Jacobs and Hendrickson [19], originally proposed in the field of computational physics, to identify redundantly rigid subgraphs of a graph. There are also other algorithms for identifying redundantly rigid graphs with better average-case complexity. However, we chose the Pebble Game for its simplicity and intuitive appeal. The original use of the Pebble Game was to find over-constrained regions in a two-dimesional lattices known as "network glasses". As such, its inventors did not need to worry about "flips" in the graph structure, and redundant rigidity was sufficient for structural stability. In network localization, we must test for the full conditions for unique localizability (including triconnectivity), and employ a recursive decomposition of the network that outputs redundantly rigid and triconnected components of the network grounded graph.

We study the effects of localization on network coverage and geographic routing. Although there is a large literature on both topics (see [1] for a survey), the assumption of the previous approaches is that the positions of all of the nodes are known. In [34], Seada et al. studied the effects of localization errors on geographic face routing; however, they consider only local random errors while our geographic routing evaluation considers global issues due to the existence of both localizable and non-localizable regions. Overall, there is no earlier study on partially localized networks.

VI. Conclusions and Future Work

In this paper, we designed a complete framework for localization with an efficient component which correctly determines which nodes are localizable and which are not. Implementing this system, we have conducted comprehensive evaluations of network localizability, as it affects both network design and deployment. We find that our method for identifying localizable nodes is robust enough in the presence of edge length errors to be practical. We further studied routing in partially localizable networks. We evaluated an integration of geographic routing with location and geographic routing without location information and showed that such novel cross-layer integrations can greatly improve network performance.

There are multiple avenues for further study. The focus of this paper is on unique localizability. Measurement errors also play an important role in localization and these effects should be investigated. One practical direction is in the identification of nodes that would be localizable in a scenario of perfect edge measurements but whose positions become ambiguous under the real life edge length error model. We envision that in order to increase the robustness of the uniqueness testing and decomposition, some measurement edges should not be used, if the lengths of the edges are close to singular positions.

Another line of research with good potential is partially localized networks for other applications beyond coverage, event detection, and geographic routing. Almost all previous research using node positions assumes that all of the nodes in the network know their positions. Our evaluations show that there can be a significant portion of the nodes that cannot determine their positions uniquely. Thus many of these protocols need to be revisited upon a likely setup for the future: a hybrid network in which some nodes know their positions, some know their positions subject to ambiguity (*e.g.*, two possible positions), and others do not know their positions at all. Designing protocols for such hybrid networks is likely to be both challenging and worthwhile.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cyirci. Wireless sensor networks: A survey. *Computer Networks*, 38(4):393–422, Mar. 2002.

[2] J. Albowicz, A. Chen, and L. Zhang. Recursive position estimation in sensor networks. In *Proceedings of the 9th International Conference on Network Protocols '01*, pages 35–41, Riverside, CA, Nov. 2001.

[3] P. Bahl and V. N. Padmanabhan. RADAR: An in-building RF-based user location and tracking system. In *Proceedings of IEEE INFOCOM '00*, pages 775–784, Tel Aviv, Israel, Mar. 2000.

[4] A. Berg and T. Jordan. A proof of Connelly's conjecture on 3-connected generic cycles. *J. Comb. Theory B.*, 2002.

[5] P. Biswas and Y. Ye. Semidefinite programming for ad hoc wireless sensor network localization. In F. Zhao and L. Guibas, editors, *Proceedings of Third International Workshop on Information Processing in Sensor Networks (IPSN'04)*, Berkeley, CA, Apr. 2004.

[6] N. Bulusu, J. Heidemann, and D. Estrin. GPS-less low-cost outdoor localization for very small devices. *IEEE Personal Communications Magazine*, 7(5):28–34, Oct. 2000.

[7] S. Capkun, M. Hamdi, and J.-P. Hubaux. GPS-free positioning in mobile ad-hoc networks. In *HICSS*, 2001.

[8] K. Chintalapudi, R. Govindan, G. Sukhatme, and A. Dhariwal. Ad-hoc localization using ranging and sectoring. In *Proceedings of IEEE INFOCOM '04*, Hong Kong, China, Apr. 2004.

[9] L. Doherty, K. S. J. Pister, and L. E. Ghaoui. Convex position estimation in wireless sensor networks. In *Proceedings of IEEE INFOCOM '01*, pages 1655–1633, Anchorage, AK, Apr. 2001.

[10] T. Eren, D. Goldenberg, W. Whiteley, Y. R. Yang, A. S. Morse, B. D. O. Anderson, and P. N. Belhumeur. Rigidity, computation, and randomization of network localization. In *Proceedings of IEEE INFOCOM '04*, Hong Kong, China, Apr. 2004.

[11] D. Estrin, R. Govindan, J. S. Heidemann, and S. Kumar. Next century challenges: Scalable coordination in sensor networks. In *Proceedings of the Fifth International Conference on Mobile Computing and Networking (Mobicom)*, pages 263–270, Seattle, WA, Nov. 1999.

[12] G. H. Forman and J. Zahorjan. The challenges of mobile computing. *IEEE Computer*, 27(4):38–47, Apr. 1994.

[13] L. Girod and D. Estrin. Robust range estimation using acoustic and multimodal sensing. In *IEEE/RSI International Conference on Intelligent Robots and Systems (IROS)*, 2001.

[14] T. He, C. Huang, B. Blum, J. Stankovic, and T. Abdelzaher. Range-free localization schemes in large scale sensor networks. In *Proceedings of the Ninth International Conference on Mobile Computing and Networking (Mobicom)*, pages 81–95, San Diego, CA, Sept. 2003.

[15] B. Hendrickson. Conditions for unique graph realizations. *SIAM Journal on Computing*, 21(1):65–84, 1992.

[16] B. Hendrickson. The molecule problem: Exploiting structure in global optimization. *SIAM Journal on Optimization*, 5(4):835–857, 1995.

[17] J. Hightower and G. Borriella. A survey and taxonomy of location systems for ubiquitous computing. *IEEE Computer*, 34(8):57–66, Aug. 2001.

[18] B. Hofmann-Wellenhof, H. Lichtenegger, and J. Collins. *Global Positioning System: Theory and Practice, Fourth Edition*. Springer-Verlag, 1997.

[19] D. Jacobs and B. Hendrickson. An algorithm for two dimensional rigidity percolation: The pebble game. *J. Computational Physics*, 137(2):346–365, 1997.

[20] X. Ji. Sensor positioning in wireless ad-hoc sensor networks with multidimensional scaling. In *Proceedings of IEEE INFOCOM '04*, Hong Kong, China, Apr. 2004.

[21] B. Karp and H. T. Kung. GPSR: Greedy perimeter stateless routing for wireless networks. In *Proceedings of the Sixth International Conference on Mobile Computing and Networking (Mobicom)*, Boston, MA, Aug. 2000.

[22] G. Laman. On graphs and rigidity of plane skeletal structures. *Journal of Engineering Mathematics*, 4:331–340, 2002.

[23] Q. Li, M. D. Rosa, and D. Rus. Distributed algorithms for guiding navigation across a sensor network. In *Proceedings of the Ninth International Conference on Mobile Computing and Networking (Mobicom)*, San Diego, CA, Sept. 2003.

[24] S. Meguerdichian, F. Koushanfar, G. Qu, and M. Potkonjak. Exposure in wireless ad-hoc sensor networks. In *Proceedings of the Seventh International Conference on Mobile Computing and Networking (Mobicom)*, Rome, Italy, July 2001.

[25] D. Moore, J. Leonard, D. Rus, and S. Teller. Robust distributed network localization with noisy range measurements. In *Proceedings of The Second ACM Conference on Embedded Networked Sensor Systems (SenSys)*, Baltimore, Maryland, Nov. 2004.

[26] D. Niculescu and B. Nath. Ad-hoc positioning system. In *Proceedings of IEEE Globecom 2001*, Nov. 2001.

[27] D. Niculescu and B. Nath. Ad hoc positioning system (APS) using AOA. In *Proceedings of IEEE INFOCOM '03*, San Francisco, CA, Apr. 2003.

[28] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The Cricket location-support system. In *Proceedings of the Sixth International Conference on Mobile Computing and Networking (Mobicom)*, pages 32–43, Boston, MA, Aug. 2000.

[29] A. Rao, C. Papadimitriou, S. Shenker, and I. Stoica. Geographic routing without location information. In *Proceedings of the Ninth International Conference on Mobile Computing and Networking (Mobicom)*, pages 96–108, San Diego, CA, Sept. 2003.

[30] B. Roth. Rigid and flexible frameworks. *American Mathematical Monthly*, 88:6–21, 1981.

[31] C. Savarese, J. Rabay, and K. Langendoen. Robust positioning algorithms for distributed ad-hoc wireless sensor networks. In *USENIX Technical Annual Conference*, Monterey, CA, June 2002.

[32] A. Savvides, W. Garber, S. Adlakha, R. Moses, and M. B. Srivastava. On the error characteristics of multihop node localization in ad-hoc sensor networks. In *Proceedings of the 2nd International Workshop on Information Processing in Sensor Networks (IPSN'03)*, Palo Alto, CA, Apr. 2003.

[33] A. Savvides, C.-C. Han, and M. B. Strivastava. Dynamic fine-grained localization in ad-hoc networks of sensors. In *Proceedings of the Seventh International Conference on Mobile Computing and Networking (Mobicom)*, pages 166–179, Rome, Italy, July 2001.

[34] K. Seada, A. Helmy, and R. Govindan. On the effect of localization errors on geographic face routing in sensor networks. In F. Zhao and L. Guibas, editors, *Proceedings of Third International Workshop on Information Processing in Sensor Networks (IPSN'04)*, Berkeley, CA, Apr. 2004.

[35] Y. Shang and W. Ruml. Improved MDS-based localization. In *Proceedings of IEEE INFOCOM '04*, Hong Kong, China, Apr. 2004.

[36] Y. Shang, W. Ruml, Y. Zhang, and M. Fromherz. Localization from mere connectivity. In *Proceedings of the Fourth ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, Annapolis, MD, June 2003.

[37] X. Sheng and Y.-H. Hu. Maximum likelihood wireless sensor network source localization using acoustic signal energy measurements. *IEEE Transactions on Signal Processing*, 2003.

[38] M. Srivastava, R. Muntz, and M. Potkonjak. Smart kindergarten: Sensor-based wireless networks for smart developmental problem-solving environments. In *Proceedings of the Seventh International Conference on Mobile Computing and Networking (Mobicom)*, Rome, Italy, July 2001.

[39] R. Want, A. Hopper, V. Falcão, and J. Gibbons. The active badge location system. Technical Report 92.1, Olivetti Research Ltd. (ORL), 24a Trumpington Street, Cambridge CB2 1QA, 1992.

[40] A. Ward, A. Jones, and A. Hopper. A new location technique for the active office. *IEEE Personal Communications*, 4(5):42–47, 1997.

[41] M. Weiser. Some computer science problems in ubiquitous computing. *Communications of ACM*, July 1993.

[42] J. Werb and C. Lanzl. Designing a positioning system for finding things and people indoors. *IEEE Spectrum*, 35(9):71–78, Oct. 1998.

[43] W. Whiteley. Some matroids from discrete applied geometry. In J. E. Bonin, J. G. Oxley, and B. Servatius, editors, *Contemporary Mathematics*, volume 197. American Mathematical Society, 1996.

[44] Y. Yu, R. Govindan, and D. Estrin. Geographical and energy aware routing: A recursive data dissemination protocol for wireless sensor networks. Technical Report UCLA/CSD-TR-01-0023, UCLA Computer Science Department, May 2001.

[45] F. Zhao, J. Liu, J. Liu, L. Guibas, and J. Reich. Collaborative signal and information processing: An information directed approach. *Proceedings of the IEEE*, 91(8):1199–1209, 2003.

Appendix

A. Joint Source and Network Localization

We consider a network of deployed sensors to detect a (known) target, which we call a source. The sensors can measure some relative distances among themselves and the signal strength of the target. We assume that the sensors are static but the source can move. The objective of the sensors is to determine the positions of the source.

We formulate a joint source and network localization problem as follows. For network localization, we assume that the measured distance $L_{i,j}$ between nodes i and j is: $L_{i,j} = l_{i,j}(1 + e_{i,j})$, where $l_{i,j}$ is the true distance between nodes i and j, and $e_{i,j}$ is the measurement noise. We assume that this noise is a zero mean Gaussian noise with variance $\sigma_{i,j}^2$ [32]. As for source localization, we assume that the sensors have synchronized clocks and collect measurements periodically. Let the detected signal strength at node i for the k-th measurement (called an event) be: $s_{k,i} = \frac{S_k}{l_{k,i}^2} + e_{k,i}$, where S_k is the signal strength of the event at its origin, $l_{k,i}$ the distance from event k to node i, and $e_{k,i}$ the noise. Note that the above model assumes an inverse-square model and is valid for many types of signals such as acoustic signals [37]. We assume that the noise is a zero mean Gaussian noise with variance $\omega_{k,i}^2$. We assume that all of the measurement noises are independent. Thus the maximum likelihood (ML) estimate, which maximizes the log likelihood function, minimizes the following:

$$\sum_{(i,j)\in E} \frac{1}{\sigma_{i,j}^2}\left(\frac{L_{i,j}}{l_{i,j}} - 1\right)^2 + \sum_{(k,i)\in K} \frac{1}{\omega_{k,i}^2}\left(s_{k,i} - \frac{S}{l_{i,j}^2}\right)^2$$

Remark The above formulation has two terms. If we consider network localization alone, we have the first term. We can easily change this term to handle other types of range measurement errors. If we assume that the network is completely localizable, we have the second term. By considering both network localization and source localization, we have both terms. The above optimization formulation can be solved using EM or projection algorithms.