A Theory of Network Localization

J. Aspnes, T. Eren *Member, IEEE*
D.K. Goldenberg *Student Member, IEEE,* A. S. Morse *Fellow, IEEE*
W. Whiteley, Y. R. Yang *Member, IEEE,*
B. D. O. Anderson *Fellow, IEEE,* P. N. Belhumeur *Fellow, IEEE*

Manuscript received June 2004; revised December 2005.

J. Aspnes is supported by NSF grants CCR-9820888 and CCR-0098078. T. Eren and P.N. Belhumeur are supported by NSF grants ITR IIS-00-85864, IIS-03-25864, EIA-02-24431 and IIS-03-08185. D.K. Goldenberg is supported by NSF Graduate Research Fellowship DGE0202738. W. Whiteley is supported by grants from NSERC (Canada) and NIH (USA). A.S. Morse is supported by grants from NSF and the US Army Research Office. Y. R. Yang is supported by NSF grant ANI-0207399. B. D. O. Anderson is supported 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.

Abstract

In this paper we provide a theoretical foundation for the problem of network localization in which some nodes know their locations and other nodes determine their locations by measuring the distances to their neighbors. We construct grounded graphs to model network localization and apply graph rigidity theory to test the conditions for unique localizability and to construct uniquely localizable networks. We further study the computational complexity of network localization and investigate a subclass of grounded graphs where localization can be computed efficiently. We conclude with a discussion of localization in sensor networks where the sensors are placed randomly.

I. Introduction

Location service is a fundamental building block of many emerging computing/networking paradigms. For example, in pervasive computing [23], [59], knowing the locations of the computers and the printers in a building will allow a computer to send a printing job to the nearest printer. In sensor networks, the sensor nodes need to know their locations in order to detect and record events, and to route packets using geometric routing (*e.g.,* [38]).

Manual configuration is one method to determine the location of a node. However, this is unlikely to be feasible for large-scale deployments and scenarios in which nodes move often. GPS [32] is another possibility, however it is costly in terms of both hardware and power requirements. Furthermore, since GPS requires line-of-sight between the receiver and satellites, it may not work well in buildings or in the presence of obstructions such as dense vegetation, buildings, or mountains blocking the direct view to the GPS satellites.

Recently, novel schemes have been proposed to determine the locations of the nodes in a network where only some special nodes (called beacons) know their locations (*e.g.,* [28], [44], [53]). In these schemes, network nodes measure the distances to their neighbors and then try to determine their locations. The process of computing the locations of the nodes is called *network localization*. For example, in [53], Savvides *et al*. propose an iterative multilateration scheme to determine the locations of nodes that do not know their locations initially.

Although the designs of the previous schemes have demonstrated great engineering ingenuity and their effectiveness in certain settings verified through extensive simulations, some fundamental questions have not been addressed. As a result, the previous schemes are mainly heuristic-based and a full theoretical foundation of network localization is still lacking.

Specifically, we identify the following three fundamental questions:

1) *What are the conditions for unique network localizability?* Although the network localization problem has already been studied extensively, the precise conditions under which the network localization problem is solvable (*i.e.*, has a unique solution) are not known.
2) *What is the computational complexity of network localization?* Even though the computational complexity of graph embeddability has been investigated before (*e.g.*, general graphs by Saxe [54] and unit disk graphs by Breu and Kirkpatrick [9]), the computational complexity of determining the locations of the nodes in a uniquely localizable network has not been studied.
3) *What is the complexity of network localization in typical network deployment scenarios?* Furthermore, for a large-scale sensor network, it may not be possible to control the placement of the sensor nodes precisely. Rather, they may be placed uniformly or randomly in a region. The unique localizability and computational complexity of such scenarios have not been investigated.

The objective of this paper is to provide systematic answers to these three questions. Many but not all of the ideas of this paper were presented in preliminary form by Eren *et al.* in [20]. In this paper, we extend these ideas and provide formal proofs. In particular, we address the first question using graph rigidity theory, the second for arbitrary uniquely localizable networks and uniquely localizable unit disk networks, and the third for unit disk networks of randomly placed nodes.

More specifically, in order to answer the first question, we propose the notion of *grounded graphs*. In these graphs, each vertex represents a network node, and two vertices in the graph are connected if the distance between the two is known; that is, when the distance between the two nodes is measured or when the two nodes are beacon nodes and their distance is *implicitly* known. Given our construction of grounded graphs, we show that a network has a unique localization if and only if its corresponding grounded graph is *generically globally rigid*. By observing this connection, we are able to apply results from the graph-rigidity literature to network localization and thus provide a systematic and pleasantly intuitive answer to the first question. For example, to check if a network in the plane is unique localizable, we just need to check if the corresponding grounded graph is 3-connected and *redundantly rigid*, both of which can be efficiently checked.

In addition, we demonstrate conditions and inductive sequences for constructing uniquely localizable networks, both in the plane and in 3-space. For instance, we show that a network with a biconnected grounded graph is uniquely localizable if two-hop neighbors are connected, *e.g.*, by

doubling the range of distance measurements in a sensor network. By using our results, a designer of a network can be assured that the constructed network is uniquely localizable, thus avoiding expensive trial-and-error procedures.

To address the second question, we analyze the computational complexity of network localization when the grounded graph is a generically globally rigid graph and show NP-hardness with a reduction from set-partition. To strengthen this insight, we show that even in the idealized case that distance measurements are present between all nodes within less than a certain known distance of each other, localization is still NP-hard.

To address the third question, we explore the density-dependent average-case complexity of network localization in realistic settings like sensor networks, and study a class of graphs in the plane called *trilateration graphs*. We show that trilateration graphs are uniquely localizable and the locations of the nodes can be computed efficiently. We show that random geometric graphs are trilateration graphs with high probability if a certain node density or communication radius is reached. We provide asymptotic results on the densities of the beacons sufficient for trilateration to be carried out in $O(1)$ step, $O(\sqrt{\log(n)})$ steps, or $O(\sqrt{n})$ steps, respectively, where n is the number of nodes in the network.

The rest of this paper is organized as follows. The specific network localization problem to be addressed is formulated in Section II. The concepts of rigidity and global rigidity are discussed in Section III. In Section IV, sufficient conditions for localization and construction of localizable networks are presented. In Section V, we study the computational complexity of solving the localization problem. In Section VI, we study localization of random geometric graphs in the plane. In Section VII, we present simulation results for localization in 3-space geometric graphs. In Section VIII, we discuss related work. Our conclusion and future work are presented in Section IX.

II. Formulation

A. The Network Localization Problem

In this paper we shall be concerned with the "network localization problem with distance information" which can be formulated as follows. One begins with a network $\mathbf{N}$ in real d-dimensional space (where $d = 2$ or 3) consisting of a set of $m > 0$ nodes labelled 1 through m that represent special "beacon" nodes together with $n - m > 0$ additional nodes labelled $m + 1$ through n that represent ordinary nodes. Each node is located at a fixed position in $\mathbb{R}^d$ and has associated with it

a specific set of "neighboring" nodes. Although a node's neighbors are typically defined to be all other nodes within some specified range, other definitions could also be used (*e.g.*, those considering the effects of obstacles). The essential property we will require in this paper is that the definition of a neighbor be a symmetric relation on $\{1, 2, \ldots, n\}$ in the sense that node j is a neighbor of node i if and only if node i is also a neighbor of node j. Under these conditions $\mathbf{N}$'s neighbor relationships can be conveniently described by an undirected graph $\mathbb{G}_{\mathbf{N}} = (V, E_{\mathbf{N}})$ with vertex set $V = \{1, 2, \ldots, n\}$ and edge set $E_{\mathbf{N}}$ defined so that (i, j) is one of the graph's edges precisely when nodes i and j are neighbors. We assume throughout that $\mathbb{G}_{\mathbf{N}}$ is a connected graph. The *network localization problem with distance information* is to determine the locations p_i of all nodes in $\mathbb{R}^d$ given the graph of the network $\mathbb{G}_{\mathbf{N}}$, the positions of the beacons $p_j,\ j \in \{1, 2, \ldots, m\}$ in $\mathbb{R}^d$, and the distance $\delta_{\mathbf{N}}(i, j)$ between each neighbor pair $(i, j) \in E_{\mathbf{N}}$.

The network localization problem just formulated is said to be *solvable* if there is exactly one set of vectors $\{p_{m+1}, \ldots p_n\}$ in $\mathbb{R}^d$ consistent with the given data $\mathbb{G}_{\mathbf{N}}$, $\{p_1, p_2, \ldots, p_m\}$, and $\delta_{\mathbf{N}} : E_{\mathbf{N}} \to \mathbb{R}$. In this paper we will be concerned with "generic" solvability of the problem which means, roughly speaking, that the problem should be solvable not only for the given data but also for slightly perturbed but consistent versions of the given data. It is possible to make precise what generic solvability means as follows. Fix $\mathbb{G}_{\mathbf{N}}$ and let $e_1, e_2, \ldots, e_q$ denote the edges in $E_{\mathbf{N}}$. Note that for any set of n points $y_1, y_2, \ldots, y_n$ in $\mathbb{R}^d$ there is a unique distance vector z whose $k-th$ component (element) is the distance between y_i and y_j where $(i, j) = e_k$. This means that there is a well-defined function $f : \mathbb{R}^{nd} \to \mathbb{R}^{(md+q)}$ mapping $\{y_1, y_2, \ldots, y_n\} \longmapsto \{y_1, y_2, \ldots, y_m, z\}$. Solvability of the network localization problem is equivalent to f being injective at $\{p_1, p_2, \ldots, p_n\}$ in the sense that the only set of points $\{y_1, y_2, \ldots, y_n\} \in \mathbb{R}^{nd}$ for which $f(y_1, y_2, \ldots, y_n) = f(p_1, p_2, \ldots, p_n)$ is $\{y_1, y_2, \ldots, y_n\} = \{p_1, p_2, \ldots, p_n\}$. In this context it is natural to say that the network localization problem is *generically solvable* at $\{p_1, p_2, \ldots, p_n\}$ if it is solvable at each point in an open neighborhood of $\{p_1, p_2, \ldots, p_n\}$. In other words, the localization problem is solvable at $\{p_1, p_2, \ldots, p_n\}$ if there is an open neighborhood of $\{p_1, p_2, \ldots, p_n\}$ on which f is an injective function.

B. Point Formations

To study the solvability of the network localization problem, we reformulate the problem in terms of a "point formation". As we shall see, the point formation relevant to the network localization problem has associated with it the *grounded graph* of the network, $\hat{\mathbb{G}}_{\mathbf{N}}$, with the same vertices as

$\mathbb{G}_{\mathbf{N}}$ but with a slightly larger edge set which adds "links" or edges from every beacon to every other. It is a property of $\hat{\mathbb{G}}_{\mathbf{N}}$ rather than $\mathbb{G}_{\mathbf{N}}$ which proves to be central to the solvability of the localization problem under consideration.

We begin by reviewing the point formation concept. By a d-dimensional *point formation* [19] at $p \triangleq \text{column } \{p_1, p_2, \ldots, p_n\}$, written $\mathbb{F}_p$, is meant a set of n points $\{p_1, p_2, \ldots, p_n\}$ in $\mathbb{R}^d$ together with a set $\mathcal{L}$ of k *links*, labelled (i, j), where i and j are distinct integers in $\{1, 2, \ldots, n\}$; the *length* of link (i, j) is the Euclidean distance between point p_i and p_j. The idea of a point formation is essentially the same as the concept of a "framework" studied in mathematics [51], [60], [61] as well as within the theory of structures in mechanical and civil engineering. For our purposes, a point formation $\mathbb{F}_p = (\{p_1, p_2, \ldots, p_n\}, \mathcal{L})$ provides a natural high-level model for an n-node network in real 2 or 3 dimensional space. In this context, the points p_i represent the positions of nodes (*i.e.*, both beacons and ordinary nodes), in $\mathbb{R}^d$ and the links in $\mathcal{L}$ label those specific node pairs whose inter-node distances are given. Thus for the network $\mathbf{N}$, $\mathcal{L}$ would consist of all edges in $\hat{\mathbb{G}}_{\mathbf{N}}$, since the distance between every pair of beacons is determined by their specified positions.

Each point formation $\mathbb{F}_p$ uniquely determines a graph $\mathbb{G}_{\mathbb{F}p} \triangleq \{V, \mathcal{L}\}$ with vertex set $V \triangleq \{1, 2, \ldots, n\}$ and edge set $\mathcal{L}$, as well as a distance function $\delta : \mathcal{L} \to \mathbb{R}$ whose value at $(i, j) \in \mathcal{L}$ is the distance between p_i and p_j. Let us note that the distance function of $\mathbb{F}_p$ is the same as the distance function of any point formation $\mathbb{F}_q$ with the same graph as $\mathbb{F}_p$ provided q is *congruent* to p in the sense that there is a distance preserving map $T : \mathbb{R}^d \to \mathbb{R}^d$ such that $T(q_i) = p_i, i \in \{1, 2, \ldots, n\}$. In the next section, we will say that two point formations $\mathbb{F}_p$ and $\mathbb{F}_q$ are *congruent* if they have the same graph and if q and p are congruent. It is clear that $\mathbb{F}_p$ is uniquely determined by its graph and distance function *at most* up to a congruence transformation. A formation that is *exactly* determined up to congruence by its graph and distance function is called "globally rigid." More precisely, a d-dimensional point formation $\mathbb{F}_p$ is said to be *globally rigid* if each d-dimensional point formation $\mathbb{F}_q$ with the same graph and distance function as $\mathbb{F}_p$ is congruent to $\mathbb{F}_p$. It is clear that any formation whose graph is complete is globally rigid. The following simple generalizations of this fact in Lemma 1 provide sufficient conditions for global rigidity that are especially relevant to the network localization problem. In d dimensions, we say a set of points $p_1, \ldots, p_{d+1}$ is *in general position* if it does not lie in a proper subspace (*i.e.*, three points in the plane do not lie on a line, and four points in space do not lie in a plane).

Lemma 1:

Let $\mathbb{F}_p = (\{p_1, p_2, \ldots, p_n\}, \mathcal{L})$ be an n-point formation in $\mathbb{R}^2$ that contains three points p_a, p_b, and p_c in general position. Suppose that the graph of the formation $\mathbb{G}_{\mathbb{F}p}$ contains the complete graph on $\{a, b, c\}$. If the only n-point formation in $\mathbb{R}^2$ that contains these three points and has the same link set as $\mathbb{F}_p$ is $\mathbb{F}_p$ itself, then $\mathbb{F}_p$ is globally rigid.

This property is a direct consequence of the fact that the identity on $\mathbb{R}^2$ is the only distance preserving map $T : \mathbb{R}^2 \to \mathbb{R}^2$ that leaves p_a, p_b, and p_c unchanged. A directly analogous property holds in three dimensions. A proof of the lemma will not be given.

C. Solvability of the Network Localization Problem

With the previous definition of point formations, we can now restate the network localization problem in terms of its associated point formation $\mathbb{F}_p$. In the present context, the problem is to determine $\mathbb{F}_p$, given the graph and distance function of $\mathbb{F}_p$ as well as the beacon position vectors $p_1, p_2, \ldots, p_m$. Solvability of the problem demands that $\mathbb{F}_p$ be globally rigid; for if $\mathbb{F}_p$ were not globally rigid it would be impossible to determine $\mathbb{F}_p$ up to congruence, let alone to determine it uniquely. Assuming $\mathbb{F}_p$ is globally rigid, solvability of the network localization problem reduces to making sure that the group of transformations T that leaves the set $\{p_1, p_2, \ldots, p_m\}$ unchanged – namely distance preserving transformations $T : \mathbb{R}^d \to \mathbb{R}^d$ for which $T(p_i) = p_i,\ i \in \{1, 2, \ldots, m\}$ – also leaves unchanged the set $\{p_{m+1}, \ldots, p_n\}$. The easiest way to guarantee this in $\mathbb{R}^2$ is to require $\{p_1, p_2, \ldots, p_m\}$ to contain three points $p_{i_1}, p_{i_2}, p_{i_3}$ in general position; for if this is so, then the only distance preserving transformation that leaves $\{p_1, p_2, \ldots, p_m\}$ unchanged is the identity map on $\mathbb{R}^2$. Similarly, if in $\mathbb{R}^3$, $\{p_1, p_2, \ldots, p_m\}$ contains at least four points in general position, then T will again be an identity map, in this case on $\mathbb{R}^3$. We summarize the main result for the solvability of network localization as follows.

Theorem 1: Let $\mathbf{N}$ be a network in $\mathbb{R}^d$, $d = 2$ or 3, consisting of $m > 0$ beacons located at positions $p_1, p_2, \ldots, p_m$ and $n - m > 0$ ordinary nodes located at positions $p_{m+1}, \ldots, p_n$. Suppose that for the case $d = 2$ there are at least three beacons in general position. Similarly, for the case $d = 3$ suppose there are at least four beacons positioned at points in general position. Let $\mathbb{F}_p$ denote the point formation whose points are at $p_1, p_2, \ldots, p_n$ and whose links are those labelled by all neighbor pairs and all beacon pairs in $\mathbf{N}$. Then for both $d = 2$ and $d = 3$ the network localization problem is solvable if and only if $\mathbb{F}_p$ is globally rigid.

III. Rigidity and Global Rigidity

In the previous section, we have established that under certain mild conditions, the solvability of the network localization problem is equivalent to the "global rigidity" of point formation. In this section we review results from rigidity theory which allow us to check for "global rigidity" efficiently. Readers familiar with rigidity theory or not interested in the technical details can just read Theorem 4 (which gives an efficiently checkable condition for rigidity in R^2), the definition of redundant rigidity (rigidity after removal of any one edge), Theorem 6 and then proceed to next section. We refer the interested reader to [27] for an in-depth reference on this topic.

As we have already stated, a d-dimensional point formation $\mathbb{F}_p$ is globally rigid if each d-dimensional point formation $\mathbb{F}_q$ with the same graph and distance function as $\mathbb{F}_p$ is congruent to $\mathbb{F}_p$. In order to clearly present properties of global rigidity, we need several other mathematical concepts whose roots can be found in the rich classical theory of rigid structures.

A. Rigidity

Let $\mathbb{F}_p$ be a d-dimensional point formation, with the distance function measuring all edges in $\mathcal{L}$, $\delta : \mathbb{R}^{nd} \to \mathbb{R}^k$. We are interested in all possible formations with the same distances, that is, in $\delta^{-1}(\delta(p))$. This is a smooth manifold in $\mathbb{R}^{nd}$ [51] and we want to know whether it contains only points congruent to p. Our best tool for studying this manifold its tangent space and the matrix equation defining this tangent space with a linearized version of the distance constraints.

For each edge $(i,j) \in \mathcal{L}$, the distance equation $(p_i - p_j)^T(p_i - p_j) = \delta(i,j)^2$ generates the corresponding linear equation

$$(p_i - p_j)^T(\dot{p}_i - \dot{p}_j) = 0$$

in the unknown vector $(\dot{p}_1, \dot{p}_2, \ldots, \dot{p}_n)$. If a vector satisfies all these equations, then it lies in the tangent space. This entire system is written as a matrix equation:

$$R(\mathbb{F}_p)\dot{p} = 0, \tag{1}$$

where $\dot{p} = \text{column}\,(\dot{p}_1, \dot{p}_2, \ldots, \dot{p}_n)$, and $R(\mathbb{F}_p)$ is the specially structured $k \times dn$ array called the *rigidity matrix* of the formation. In structural engineering and mathematics, the solutions $\dot{p}$ are called *first-order flexes* (infinitesimal flexes, or virtual velocities) [51], [60], [61].

The tangent vectors to the congruences of the space $\mathbb{R}^d$ generate a subspace of trivial solutions, called the *trivial flexes*. In the plane, provided that we have at least two distinct points, this space

has dimension 3, generated by two translations and the tangent vector to a rotation about the origin. In 3-space, if we have three non-collinear points, this space has dimension 6, generated by three translations along the axes and the derivatives of three rotations about the three axes though the origin.

Definition 1: If the trivial flexes are the entire space of first-order flexes, the formation is *first-order rigid*.

In short, provided we have at least three vertices [51], [61]:

Theorem 2: Assume $\mathbb{F}_p$ is a formation with at least d nodes in d-space,

$$\text{rank } R(\mathbb{F}_p) \leq \begin{cases} 2n-3 & \text{if } d=2 \\ 3n-6 & \text{if } d=3. \end{cases}$$

The formation $\mathbb{F}_p$ in the plane is first-order rigid if and only if $\text{rank } R(\mathbb{F}_p) = 2n-3$. The formation $\mathbb{F}_p$ in 3-space is first-order rigid if and only if $\text{rank } R(\mathbb{F}_p) = 3n-6$.

It is easy to see from the form of the rigidity matrix that the entries in $R(\mathbb{F}_p)$ are polynomial (actually linear) functions of p. Because of this, the values of p for which the rank of $R(\mathbb{F}_p)$ is below its maximum value form a proper algebraic set in $\mathbb{R}^{dn}$. This observation lies at the roots of the following equivalences [60], [61]:

Theorem 3: Given a formation graph $\mathbb{G}$ with $n \geq 2$ vertices in the plane (resp. $n \geq 3$ vertices in 3-space) the following are equivalent:

1) for some formation $\mathbb{F}_p$ with this graph, $\text{rank } R(\mathbb{F}_p) = 2n-3$ (resp. $\text{rank } R(\mathbb{F}_p) = 3n-6$ in 3-space);
2) for all $q \in \mathbb{R}^{2n}$ in an open neighborhood of p, the formation $\mathbb{F}_q$ on the graph $\mathbb{G}$ is first-order rigid in the plane (resp. $q \in \mathbb{R}^{3n}$, $\mathbb{F}_q$ is first-order rigid in 3-space);
3) for all q in an open dense subset of $\mathbb{R}^{2n}$, the formation $\mathbb{F}_q$ on the same graph $\mathbb{G}$ is first-order rigid in the plane (resp. open dense subset of $\mathbb{R}^{3n}$, $\mathbb{F}_q$ is first-order rigid in 3-space).

When property 3) holds, we say that the graph $\mathbb{G}$ of $\mathbb{F}_p$ is *generically rigid* in the space. It is well known that first-order rigidity implies all of the other standard forms of rigidity for a formation, but the converse can fail [21], [51], [60]. For readers thinking of other concepts of rigidity, we point out that if one of these alternative forms of rigidity holds for a non-empty open set, then all of the properties in Theorem 3 hold [51], [60].

For the plane we have a strong combinatorial characterization of the generically rigid graphs. We note that this leads to a fast $O(|V|^2)$ algorithm for generic rigidity testing [29].

Theorem 4 (Laman [40]): A graph $\mathbb{G} = (V, \mathcal{L})$ with n vertices is generically rigid in $\mathbb{R}^2$ if and only if $\mathcal{L}$ contains a subset E consisting of $2n-3$ edges with the property that for any nonempty subset $E' \subset E$, the number of edges in E' cannot exceed $2n'-3$ where n' is the number of vertices of $\mathbb{G}$ which are endpoints of edges in E'.

There is no comparable complete result for 3-space, and no known polynomial time algorithm, though there are useful partial results [60], [61].

B. Conditions for Global Rigidity

We are interested in the stronger concept of generic global rigidity. This concept is intimately related with first-order rigidity. If the formation $\mathbb{F}_p$ is not first-order rigid, there is a non-trivial first-order flex $\dot{p}$ that does not come from a congruence. This is enough to guarantee that a small perturbation will create a formation that is not globally rigid.

Theorem 5 (Averaging Theorem [60], [61]): Given a non-degenerate formation $\mathbb{F}_p$ with a non-trivial flex $\dot{q}$, the formations $\mathbb{F}_{p+t\dot{q}}$ and $\mathbb{F}_{p-t\dot{q}}$ on the same graph, for all $t > 0$, have the same edge lengths for all links but are not congruent.

We say that a formation $\mathbb{F}_q$ is *generically globally rigid* if every sufficiently small perturbation q of p creates a globally rigid formation $\mathbb{F}_q$. The result above shows that any non-degenerate generically globally rigid formation $\mathbb{F}_p$ must be first-order rigid. However, as Fig. 1 illustrates, the converse is not true.

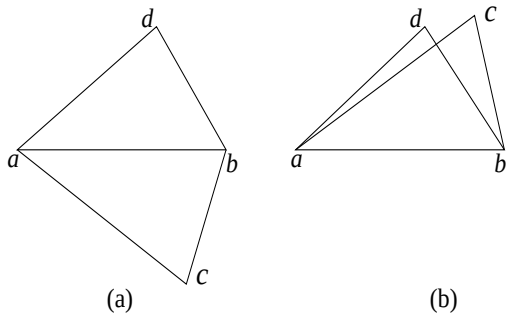

Fig. 1. Two first-order rigid formations with the same graph and the same distance values.

A graph $\mathbb{G} = \{\mathcal{V}, \mathcal{L}\}$ with n vertices is *generically globally rigid* in $\mathbb{R}^d$ if there is an open dense set of points $p \in \mathbb{R}^{dn}$ at which $\mathbb{F}_p$ is a globally rigid formation with link set $\mathcal{L}$. In the plane, a recent result gives a complete characterization of generically globally rigid graphs. To introduce the result, we first review the definitions of k-connectivity and redundant rigidity.

A graph $\mathbb{G}$ is *k-connected* if it remains connected upon removal of any set of $< k$ vertices. The k-connectivity of a complete graph with n vertices is defined to be $n-1$. A simple mental check also confirms that for more than $d+1$ vertices in dimension d, we need at least $d+1$ vertex connectivity, to avoid a reflection of one component through a mirror placed on a disconnecting set of size d.

A graph $\mathbb{G}$ is *redundantly rigid* in $\mathbb{R}^d$ if the removal of any single edge results in a graph that is also generically rigid in $\mathbb{R}^d$. Fig. 2 shows a graph that is not redundantly rigid. As Fig. 3 suggests, we need the graph to be generically redundantly rigid to ensure generic global rigidity.

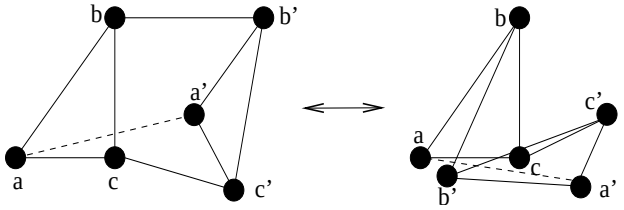

Fig. 2. An example from [29] showing a rigid 3-connected graph with two realizations in the plane. If edge (a, a') is removed, triangle $a'b'c'$ swings along a path until the distance (a, a') is the same as it originally was.

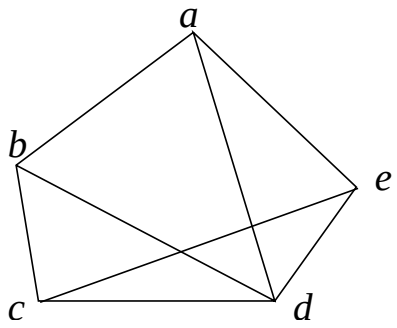

Fig. 3. A globally rigid formation in the plane.

Theorem 6 ([34]): A graph $\mathbb{G}$ with $n \geq 4$ vertices is generically globally rigid in $\mathbb{R}^2$ if and only if it is 3-connected and redundantly rigid in $\mathbb{R}^2$.

Notice that to actually carry out a test to decide whether or not a given graph $\mathbb{G}$ is generically globally rigid in $\mathbb{R}^2$, one must establish that it is both 3-connected and redundantly rigid in $\mathbb{R}^2$. Various tests for 3-connectivity are known, and we refer the reader to [33], [43] for details including measures of the complexity of the tests involved. Tests for redundant rigidity in $\mathbb{R}^2$ have been derived [29] based on variants of Laman's theorem [40].

Since these properties are also required for even a non-empty open set of globally rigid formations in the plane, we can see that the existence of one generically globally rigid formation $\mathbb{F}_p$ implies the graph is generically globally rigid. In 3-space, whether having one generically globally rigid formation is enough to show that the graph is generically globally rigid is an open question [13].

As with generic rigidity, we do not have a generalization of Theorem 6 to higher dimensions. However, it extends as a necessary but not sufficient condition.

Theorem 7 ([14], [29]): If a graph $\mathbb{G}$ with more than $d+1$ vertices is generically globally rigid in d-space, then $\mathbb{G}$ is redundantly rigid and at least $d+1$ connected. In all dimensions $d \geq 3$, there are redundantly rigid and at least $d+1$ connected graphs that are not generically globally rigid.

IV. Inductive Construction of Generically Globally Rigid Graphs

It is possible to derive useful sufficient conditions and inductive constructions for generically globally rigid graphs (i.e., solvable) in spaces of all dimensions [14], [21]. Such constructions can be useful in identifying and constructing uniquely localizable networks.

One simple construction inserts new nodes of degree $d+1$ into existing generically globally rigid formations to create larger generically globally rigid formations. Since we will use this construction later, we give some formal definitions using the term 'trilateration' from the plane as a general term.

Lemma 2: Given a generically globally rigid point formation $\mathbb{F}_p$, and a new point p_0 linked to $d+1$ nodes $p_1, ...p_{d+1}$ of $\mathbb{F}_p$, in general position, then the extended point formation $\bar{\mathbb{F}}_{p+p_0}$ is generically globally rigid.

Proof: Consider any location for the distances in $\bar{\mathbb{F}}_{p+p_0}$. We show that the location of p_0 is unique, given these prior locations.

We first give the proof in $\mathbb{R}^2$, where $\mathbb{F}_p$ has three non-collinear points p_a, p_b, p_c. We have the distances from p_0 to these three points. The distances from the first two points, p_a, p_b, define two intersections of corresponding circles centered at p_a and p_b. The distances from any third point p_c

to these two solutions are different, since p_c is not on the line through p_a, p_b. Therefore there is a unique position for p_0 for the given distance to p_c.

The same argument works in all dimensions, starting with the two points of intersection for d spheres with centers in general position.

Now, consider a second formation $\bar{\mathbb{F}}_{p+q_0}$ with the same link lengths as $\bar{\mathbb{F}}_{p+p_0}$. Since the generically globally rigid formation $\mathbb{F}_p$ is contained in this second extended formation, the location of its nodes is unique, up to congruence. The unique congruence T defined by the $d+1$ general position points of attachment induces a position $T(p_0)$ that satisfies our construction. Since the constructed point was unique, we conclude that $T(p_0) = q_0$ and the two extended formations are congruent. We conclude that the extended formation is globally rigid.

The general position property used is stable under small perturbations of p. Therefore the global rigidity holds for all small perturbations and the extended formation is generically globally rigid. ■

For the network setting in 2 dimensions, we can start with the globally rigid formation on $m \geq 3$ beacons as $\mathbb{F}_{p_m}$. We can then sequentially add new nodes as points $p_{m+1}, \ldots, p_n$, each along with 3 edges to distinct nodes in the preceding formation, to extend the preceding formation. Provided that all sets of points which will be used in extensions are in general position, we create a generically globally rigid formation $\bar{\mathbb{F}}_p$ with n points. This process can be worded in terms of generically globally rigid graphs.

Definition 2: A *trilateration extension* in dimension d of a graph $\mathbb{G} = (V, E)$, where $|V| \geq d+1$ produces a new graph $\mathbb{G}' = (V \cup \{v\}, E \cup \{(v, w_1), \ldots, (v, w_{d+1})\})$, where $v \notin V$, and $w_i \in V$.

Definition 3: A *trilaterative ordering in dimension* d for a graph $\mathbb{G}$ is an ordering of the vertices $1, \ldots, d+1, d+2, \ldots n$ such that $\mathbb{K}_{d+1}$, the complete graph on the initial vertices, is in $\mathbb{G}$, and from every vertex $j > d+1$, there are at least $d+1$ edges to vertices earlier in the sequence. Graphs for which a trilaterative ordering exists in dimension d are called *trilateration graphs in dimension* d.

Theorem 8: Trilateration graphs in dimension d are generically globally rigid in dimension d.

Proof: Any formation on the complete graph on $d+1$ vertices is generically globally rigid if the points are in general position. We take such a formation. We can then apply Lemma 2 to add each point along the trilaterative ordering, with its guaranteed $d+1$ edges, to create a larger generically globally rigid formation with all points in general position. We can then add any additional edges beyond the $d+1$ needed, without changing the generic global rigidity of the extended formation.

Repeated application of this leads to a generically globally rigid formation on the whole graph. Since the conditions of being in general position apply to an open dense subset of the space, we conclude that the graph is generically globally rigid. ■

A trilateration graph $\mathbb{G}$ may have more than one trilaterative ordering and even more than one *seed* — the initial complete graph $\mathbb{K}_{d+1}$. We will look at algorithmic aspects of trilateration graphs in the next section.

V. Computational Complexity of Localization

We have seen in preceding sections that global rigidity is a necessary condition for the solvability of network localization. We will now move from the decision problem of solvability to an associated search problem, graph realization.

Specifically, we define the graph realization problem as the problem of assigning coordinates to vertices of a weighted graph G so that the edge weight of every edge (i, j) equals the distance between the points assigned to vertices i and j. Note that a given graph may not be realizable under a particular set of edge weights. In the context of network localization, the graphs under study are the grounded graphs associated with network point formations.

A. Realizing Globally Rigid Graphs

Although global rigidity testing in the plane is computable in polynomial time, Saxe has shown that testing the realizability of weighted graphs is NP-hard [54]. Below, we will argue that realizing a graph is still hard, even if it is known that the graph is globally rigid and that it has a realization. The objective of this subsection is to build intuitive results. In the next subsection we will conduct a formal reduction and discuss the implications. Note that we will restrict ourselves to the plane in this section.

Recall that the SET-PARTITION-SEARCH problem is the following: Given a set of numbers S, find a partition of S as $A \cup S - A$ so that the sums of the numbers in the two sets are equal. We first prove a useful NP-hardness result for the SET-PARTITION-SEARCH problem.

Claim 1: Given a set S for which the existence of a set partition is guaranteed, the problem of finding a set partition is still NP-hard.

Proof: Assume that algorithm $\mathcal{A}$ solves set-partition-search. Let S be a set of numbers for which it is unknown whether there is a set-partition. Run $\mathcal{A}$ on input S for time t equal to the running time of $\mathcal{A}$ on a valid input of size $|S|$.

If $\mathcal{A}$ has not terminated, then S has no set-partition. If $\mathcal{A}$ has terminated, then S has a set-partition if and only the output of $\mathcal{A}$ is a set-partition of S. Since set-partition is NP-complete, set-partition-search is NP-hard. ■

We now show another result which will prove to be useful. Fig. 4 shows a particular realization of the wheel graph W_6.

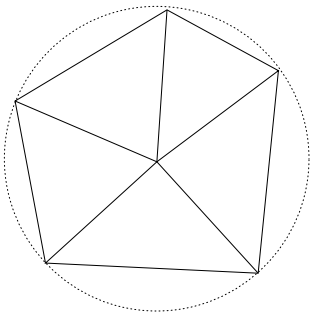

Fig. 4. Wheel graph W_6.

Claim 2: The wheel graph $\mathbb{W}_n$ is globally rigid.

Proof: We will refer to nodes in the cycle, $\mathbb{C}_{n-1}$, as *rim nodes*, the central node as the *hub*, an edge between the hub and a rim node as a *spoke*, and an edge between two rim nodes as a *rim edge*.

If we remove two rim vertices, the graph remains connected through the hub. If we remove the hub and one rim vertex, the graph remains a connected path on the remaining vertices. Therefore removing two vertices does not disconnect the graph, and it is 3-connected.

As Lemma 2.1 of [6] observes, a wheel is a minimally redundantly rigid graph for the plane. By Theorem 6, it is generically globally rigid. ■

We now analyze the complexity of realization of globally rigid graphs. A realistic formulation of the realization problem requires that the edge lengths be noisy measurements of underlying edge lengths subject to bounded errors. Note that with probability 1, these error-corrupted edge lengths will not correspond to realizable weights. In this case, the realization problem becomes an approximation problem; namely, finding an assignment of coordinates for the graph vertices so that the resulting discrepancies with the noisy weights are below a tolerance parameter. Below, we use a reduction from set partition to show that realization of globally rigid weighted graphs with

realizable *i.e.*, exact, edge weights is still hard. To construct the reduction, we use real numbers, which could potentially be irrational. The formal proof in the next subsection does not need to use real numbers.

Assume we have an algorithm $\mathcal{A}$ that takes as input a realizable globally rigid weighted graph and outputs the unique realization. Consider a set of n positive rational numbers $S = \{s_1, s_2, \ldots, s_n\}$, for which a set-partition exists, scaled without loss of generality such that $\sum_{i=1}^{n} s_i = \pi/2$. Let us now label the nodes of $\mathbb{W}_{n+1}$ as follows: we label the hub 0, and the rim nodes 1 through n, where there is an edge from i to $i+1$ for $i \in \{1, 2, \ldots, n-1\}$ and from n to 1. We will refer to the spoke from 0 to i as *spoke*$_i$.

Let us now construct a weighted version of $\mathbb{W}_{n+1}$. Let the weight of each spoke be r, where r is a positive rational number. Let the weight of the rim edge between node i and node $i+1$ for $i \in \{1, 2, \ldots, n-1\}$ be $2r\sin(s_i/2)$, and let the weight of the rim edge between node n and node 1 be $2r\sin(s_n/2)$. We now argue that this weighted version of $\mathbb{W}_{n+1}$, call it $\mathbb{W}'_{n+1}$, has a realization in the plane.

If we imagine s_i as the modulus of the angle between spoke$_i$ and spoke$_{i+1}$ for $i \in \{1, 2, \ldots, n-1\}$ and s_n as the modulus of the angle between spoke$_n$ and spoke$_1$ in a realization of $\mathbb{W}_{n+1}$, we can determine a set of edge weights. Fix the weight of each spoke to be r, where r is a positive real number. Then the weight of the rim edge between node i and node $i+1$ for $i \in \{1, 2, \ldots, n-1\}$ must be $2r\sin(s_i/2)$, and the weight of the rim edge between node n and node 1 must be $2r\sin(s_n/2)$. Since S has a set partition, we can form a cycle of these chords in the plane. Therefore the wheel graph with these edge weights, $\mathbb{W}'_{n+1}$ has a realization.

Note that despite the fact that the spokes might be inserted sequentially, it is not true that the ends of the spokes on the circumference necessarily occur sequentially as one moves continuously around the rim. The graph will in general fold up like a fan. In addition, note that the upper bound on the sum of the s_i ensures that in progressing through the cycle, there can be no net rotation around the hub, *i.e.*, the angles corresponding to clockwise rotation and those to counter clockwise rotation do not differ by some nonzero multiple of 2π.

Suppose we have an efficient algorithm $\mathcal{A}$ for graph realization. We run the algorithm on the realizable globally rigid weighted graph $\mathbb{W}'_{n+1}$ to obtain a realization. From this realization, determine whether it is clockwise or counter-clockwise to rotate spoke$_i$ to spoke$_{i+1}$ for $i \in \{1, \ldots, n-1\}$ and from spoke$_n$ to spoke$_1$. By construction, the set of angles corresponding to

clockwise rotation and that of counter-clockwise rotation form a set-partition of S.

This procedure solves set-partition-search with one call to a graph realization algorithm and polynomial time additional computation. Since set-partition-search is NP-hard, realizable globally rigid weighted graph realization in the plane is NP-hard.

B. Localization complexity for unit disk graphs

The preceding subsection considers arbitrary globally rigid graphs. However, the construction relies on a "folding fan" construction in which pairs of nodes close to each other in the unique realization may possibly not have an edge between them. We consider a special class of graphs called unit disk graphs, where a distance measurement is present between any pair of sensors if they are within some disk radius parameter r of each other. We will show that even when limited to this idealized class of graph, localization is still NP-hard. To avoid precision issues involving irrational distances, below we assume that the input to the problem is presented with the distances squared. If we make the further assumption that all sensors have integer coordinates, all distances will be integers as well.

We consider a decision version of the localization problem, which we call *UNIT DISK GRAPH RECONSTRUCTION*. This problem essentially asks if a particular graph with given edge lengths can be physically realized as a unit disk graph with a given disk radius in two dimensions. A similar result is obtained by Breu and Kirkpatrick in [9]. Our objective in this paper is to further connect to network localization.

The input is a graph G where each edge uv of G is labeled with an integer ℓ_{uv}^2, the square of its length, together with an integer r^2 that is the square of the radius of a unit disk. The output is "yes" or "no" depending on whether there exists a set of points in R^2 such that the distance between u and v is ℓ_{uv} whenever uv is an edge in G and exceeds r whenever uv is not an edge in G.

Our main result is that UNIT DISK GRAPH RECONSTRUCTION is NP-hard, based on a reduction from the NP-hard problem CIRCUIT SATISFIABILITY [24]. The constructed graph for a circuit with m wires has $O(m^2)$ vertices and $O(m^2)$ edges, and the number of solutions to the resulting localization problem is equal to the number of satisfying assignments for the circuit. In each solution to the localization problem, the points can be placed at integer coordinates, and the entire graph fits in an $O(m)$-by-$O(m)$ rectangle, where the constants hidden by the asymptotic

notation are small. The construction also permits a constant fraction of the nodes to be placed at known locations.

Formally, we show:

Theorem 9: There is a polynomial-time reduction from CIRCUIT SATISFIABILITY to UNIT DISK GRAPH RECONSTRUCTION, in which there is a one-to-one correspondence between satisfying assignments to the circuit and solutions to the resulting localization problem.

The proof of Theorem 9 depends on a sequence of constructions of logical gates and is given by Aspnes *et al.* in [5]. An application of the theorem to *sparse networks* shows that localization is hard. By sparse networks, we mean networks where the number of known distance pairs grows only linearly in the number of nodes. Sparse networks are of great importance, because in the limit as a network with bounded communication range and fixed sensor density grows, the number of known distance pairs grows only linearly in the number of nodes.

Corollary 1: There is no efficient algorithm that solves the localization problem for sparse sensor networks in the worst case unless P=NP.

Proof: Suppose that we have a polynomial-time algorithm that takes as input the distances between sensors from an actual placement in R^2, and recovers the original position of the sensors (relative to each other, or to an appropriate set of beacons). Such an algorithm can be used to solve UNIT DISK GRAPH RECONSTRUCTION by applying it to an instance of the problem (that may or may not have a solution). After reaching its polynomial time bound, the algorithm will either have returned a solution or not. In the first case, we can check if the solution returned is consistent with the distance constraints in the UNIT DISK GRAPH RECONSTRUCTION instance in polynomial time, and accept if and only if the check succeeds. In the second case, we can reject the instance. In both cases we have returned the correct answer for UNIT DISK GRAPH RECONSTRUCTION. ■

It might appear that this result depends on the possibility of ambiguous reconstructions, where the position of some points is not fully determined by the known distances. However, if we allow randomized reconstruction algorithms, a similar result holds even for graphs that have unique reconstructions. Below RP denotes the class of randomized polynomial-time algorithms [25].

Corollary 2: There is no efficient randomized algorithm that solves the localization problem for sparse sensor networks that have unique reconstructions unless RP=NP.

Proof: The proof of this claim is by use of the well-known construction of Valiant and Vazirani,

which gives a randomized Turing reduction from 3SAT to UNIQUE SATISFIABILITY [58]. The essential idea of this reduction is that randomly fixing some of the inputs to the 3SAT problem reduces the number of potential solutions, and repeating the process eventually produces a 3SAT instance with a unique solution with high probability. ■

Finally, because the graph constructed in the proof of Theorem 9 uses only points with integer coordinates, even an approximate solution that positions each point to within a distance $\epsilon < 1/2$ of its correct location can be used to find the exact locations of all points by rounding each coordinate to the nearest integer. Since the construction uses a fixed value for the unit disk radius r (the natural scale factor for the problem), we have

Corollary 3: The results of Corollary 1 and Corollary 2 continue to hold even for algorithms that return an approximate location for each point, provided the approximate location is within $\epsilon \cdot r$ of the correct location, where ϵ is a fixed constant.

What we do *not* know at present is whether these results continue to hold for solutions that have large positional errors but that give edge lengths close to those in the input. Our suspicion is that edge-length errors accumulate at most polynomially across the graph, but we have not yet carried out the error analysis necessary to prove this. If our suspicion is correct, we would have:

Conjecture 1: The results of Corollary 1 and Corollary 2 continue to hold even for algorithms that return an approximate location for each point, provided the relative error in edge length for each edge is bounded by ϵ/n^c for some fixed constant c.

C. Global/Distributed Optimization for Localization

The preceding subsections have shown that the computational complexity of network localization is likely to be high. In practice, one way to solve the general localization problem is to formulate it as an optimization problem. Specifically, realization of a graph $\mathbb{G} = (V, E)$ with edge weight function $\delta(i, j)$ can be formulated as a global optimization over vectors of points $\{x_1, x_2, \ldots, x_{|V|}\}$ of the following form,

$$minimize \sum_{(i,j)\in E} \left(\delta(i,j) - \parallel x_i - x_j \parallel\right)^2.$$

This formulation of the problem has been used by biologists studying molecular conformation [15]. Because such optimization is computationally expensive, strategies such as divide-and-conquer [30] and objective function smoothing [45] have been proposed. Recently, in [7], Biswas

and Ye show that network localization in unit disk graphs can be formulated as a semidefinite programming problem and thus can be efficiently solved. A condition of their algorithm, however, is that the graphs are densely connected. More specifically, their algorithm requires that $\Omega(n^2)$ pairs of nodes know their relative distances, where n is the number of sensor nodes in the network. However, as we see from the preceding section, for a general network, it is enough for the localization process to have a unique solution when certain $O(n)$ pairs of nodes know their distances.

In the context of network localization, distributed optimization algorithms may be desirable. In this case, algorithms such as [30] may be applied by dividing the global network into small globally rigid sub-components [36] (clusters) to reduce overall complexity. Each cluster computes its relative localization using some optimization technique. Then the global localization can be achieved by merging the localizations of individual components. With these algorithms, a tradeoff will likely emerge between the advantage of small cluster size and the disadvantage of having to reconcile a large number of localized clusters.

D. Realizing Trilateration Graphs

Although realization of general globally rigid graphs is hard, we have already seen a class of globally rigid graphs that are computationally efficient to realize. In what follows, we define *trilateration* to be the operation whereby a node with known distances to three other nodes in general position determines its own position in terms of the positions of those three neighbors. We assume that this operation is efficiently computable.

Theorem 10: A trilateration graph $\mathbb{G} = (V, E)$ with realizable edge weights is realizable in a polynomial number of trilaterations.

Proof: There is a sequence of trilateration extensions that result in $\mathbb{G}$ when applied to $\mathbb{K}_3$. If we know a seed of $\mathbb{G}$, then we can do the following: Localize one of the nodes of the seed at the origin, another on the positive x-axis, and the remaining node at a position with a positive y coordinate. At each step, we can calculate positions for all unlocalized nodes with edges to three localized nodes. Because $\mathbb{G}$ is a trilateration graph, we are guaranteed to be able to calculate positions for all nodes with at most $|V| - 3$ trilaterations.

If we do not know any seed of $\mathbb{G}$, we can guess it in at most $\binom{n}{3}$ tries, which is polynomial. A guess is correct if and only if the above procedure succeeds in localizing all nodes in a linear number of steps. Hence, we can realize a trilateration graph in a polynomial number of steps. ■

As we shall see, there are scenarios in which it is reasonable to assume that we know a seed of the trilateration graph, and in these cases, the linear algorithm will be applicable.

VI. Localization in Random Geometric Graphs in the Plane

In previous sections, we presented theory for localization of general networks. In this section, we specialize to the setting of sensor networks with a large number of randomly distributed sensors and explore the average case behavior of a specific localization algorithm. An abstraction that corresponds well to this setting is the random geometric graph.

A. Definition and Properties of Random Geometric Graphs

We define random geometric graphs in terms of point formations.

Definition 4: Given $n \in \mathbb{N}$ and $r \in [0, 1]$, the *random geometric graphs* $\mathbb{G}_n(r)$ are the graphs associated with two dimensional point formations $\mathbb{F}$, with all links of length less than r, where $p = \{p_1, p_2, \ldots, p_n\}$ is a set of points in $[0, 1]^2$ generated by a two dimensional Poisson point process of intensity n.

The parameters of the model, n and r, correspond respectively to the physical parameters of sensor density and sensing radius.

We next review some useful properties of the connectivity of $\mathbb{G}_n(r)$. Note that the results we present in this section are asymptotic and that because of this, we neglect collinearity as a low probability phenomenon.

As in the case of the Erdös-Rényi random graph model [8], there is a phase transition in the random geometric graph model at which the graph becomes connected with high probability [4]. Penrose [48] generalizes this to k-connectivity with the result that if $\mathbb{G}_n(r)$ has a minimum vertex degree of k then with high probability $\mathbb{G}_n(r)$ is k-connected.

Since it is was proved in [41] that for some $r \in O(\sqrt{\frac{\log n}{n}})$, $\mathbb{G}_n(r)$ asymptotically has a minimum vertex degree of k for $k \in O(1)$ with high probability, $r \in O(\sqrt{\frac{\log n}{n}})$ can also ensure k-connectivity.

B. Global Rigidity of Random Geometric Graphs

Recalling that 3-connectivity is a necessary condition for global rigidity, and using a recent result that 6-connectivity is sufficient for global rigidity in the plane [34], we conclude that $\mathbb{G}_n(r)$ is globally rigid with high probability for some $r \in O(\sqrt{\frac{\log n}{n}})$.

Next we have the following interesting result:

Theorem 11: If $\mathbb{G} = (V, E)$ is 2-connected, then the graph $\mathbb{G}^2 = (V, E \cup E^2)$, where E^2 is the set of edges between endpoints of paths consisting of two edges in $\mathbb{G}$, is globally rigid.

Proof:

Let $\mathbb{G} = (V, E)$ be 2-connected. Take any two nodes u and v in V. Since there are at least two node-disjoint paths from u to v, they lie on a cycle. Let us denote the cycle of n nodes by C_n. We will show that C_n^2 is globally rigid, and from this, it follows that the distance between every pair of nodes in V is fixed in $\mathbb{G}^2$, *i.e.*, $\mathbb{G}^2$ is globally rigid.

By a result from [6], every globally rigid graph has a globally rigid subgraph that can be obtained from K_4 by a sequence of node addition operations, termed edge splitting. Edge splitting preserves global rigidity, and in it, a new node v is added by replacing an existing edge (u, w) by edges (u, v) and (v, w), and adding an edge (v, z) for some $z \neq u, v$. We show that C_n^2 is globally rigid by constructing a class of globally rigid graphs C_n' which are spanning subgraphs of C_n^2, as illustrated in Fig. 5.

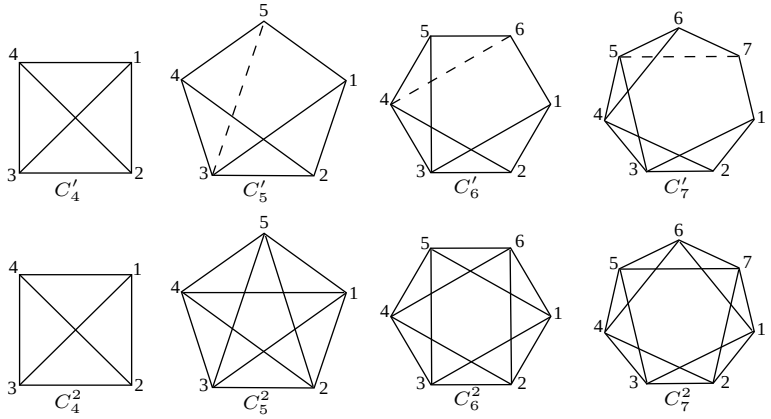

Fig. 5. In the top row are the globally rigid C_n' graphs, $n = 4, 5, 6, 7$. The dotted edge connects a newly added node n to node $n-2$. Note that C_n' is a spanning subgraph of C_n^2.

Starting from K_4, we label the nodes $1 \ldots 4$ and add nodes sequentially. In the $n-4$th step, we insert a node n by adding an edge $(n, n-2)$, and subdividing the edge $(n-1, 1)$ by replacing it with $(n-1, n)$ and $(n, 1)$. The resulting graph C_n' is globally rigid. It is easy to see that C_n' is a spanning subgraph of C_n^2 for all $n \geq 4$. Since adding edges to a globally rigid graph cannot result in a non-globally rigid graph, C_n^2 is globally rigid. Hence, if G is biconnected, the distance

between all pairs of nodes is fixed in G^2, and G^2 is globally rigid.

■

For random geometric graphs, the preceding theorem means that $\mathbb{G}_n(2r)$ is globally rigid with at least the probability that $\mathbb{G}_n(r)$ is 2-connected. This result is extended and related results for 3-space and trilateration graphs proven in recent work by Anderson *et al.* in [3]

For some large n and $\delta \in (0,1)$, let r_i denote the smallest radius at which $\mathbb{G}_n(r)$ becomes i-connected with probability $1-\delta$ and let r_g denote the radius at which it becomes globally rigid with probability $1-\delta$. Note that $r_2 \leq r_3 \leq r_g \leq r_6$ and that $r_g \leq 2r_2$. This behavior is illustrated in Fig. 6.

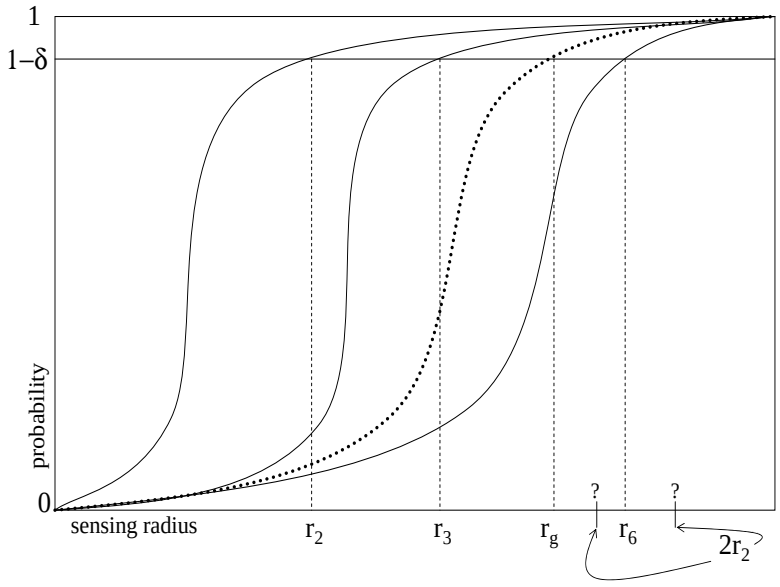

Fig. 6. Probability that $\mathbb{G}_n(r)$ is k-connected. Dotted line represents the probability that $\mathbb{G}_n(r)$ is globally rigid.

C. Realization of Random Geometric Graphs

We now explore conditions for $\mathbb{G}_n(r)$ to yield an efficient realization computation [1].

Theorem 12: If $\lim_{n\to\infty} \frac{nr^2}{\log n} > 8$, with high probability, $\mathbb{G}_n(r)$ is a trilateration graph.

Proof: Partition $[0,1]^2$ into $\frac{cn}{\log n}$ square cells of equal size where $8\log n/nr^2 < c < 1$. That such a c exists is assured by the theorem hypothesis. Note that with high probability, every cell

[1]with respect to a particular algorithm

contains at least three nodes. This is because if A is the area of a square, the probability it contains no nodes, one node, or two nodes is e^{-nA}, nAe^{-nA} and $(1/2)(nA)^2e^{-nA}$. When $A = logn/cn$, the sum of these three probabilities, call it $q(n)$, goes to zero as n goes to infinity. In fact, it is easily seen that $\frac{cn}{\log n}q(n)$ goes to zero as n goes to infinity, from which one can argue that every cell contains at least three nodes with probability approaching 1 as n goes to infinity. Additionally, since $r > 2\sqrt{2}\sqrt{\frac{\log n}{n}}$, every node has edges between itself and all nodes in its own cell and those adjacent cells sharing a corner or edge with its cell.

Starting from some cell we label as 0, we iteratively label every cell in $[0,1]^2$. In step $i \in \{1, \ldots, \sqrt{\frac{cn}{\log n}}\}$, we label with i every unlabelled cell that adjoins a cell labelled $i-1$ horizontally, vertically, or diagonally. We will refer to the union of all cells with the same label i as a *layer*, L_i.

We now iteratively label all n nodes in the grid such that each node has a unique label. In step -1, we choose three nodes in L_0 and label them 1, 2, and 3. In step 0, we label the rest of the nodes in L_0 sequentially with numbers greater than 3. In step i, we label sequentially all nodes in L_i with numbers larger than every label in L_{i-1}.

Every node in L_0 with a label greater than three has edges to 1, 2, and 3. By construction, a node labelled m in L_i, $i > 0$ has edges to at least three nodes in L_{i-1} with labels less than m. Thus we have a trilaterative ordering from Definition 3, and $\mathbb{G}_n(r)$ is a trilateration graph.

■

An intuitive argument that perhaps yields more insight into the previous result is the following. In the limit of large n, assume that nodes 1, 2, and 3 can be considered to occur at a single point p_0. If every node in $\mathbb{G}_n(r)$ is connected to three other nodes closer than itself to p_0, then $\mathbb{G}_n(r)$ has a trilaterative ordering. Since p_0 can be in any direction from an arbitrary point, this is assured in the event that every node has three neighbors in any 120° sector of the circle with radius r about it, or at least nine neighbors. Denoting by r_t the radius at which $\mathbb{G}_n(r)$ has probability $1-\delta$ of being a trilateration graph, we suspect that r_t approaches r_9 from above in the limit of large n.

These results immediately yield insight into the complexity of realizing $\mathbb{G}_n(r)$.

Theorem 13: For some $r \in O(\sqrt{\frac{\log n}{n}})$, if the positions of three nodes with edges to each other are known, then with high probability, a realization of $\mathbb{G}_n(r)$ is computable in linear time.

Proof: By the proof of Theorem 12, the three nodes with known positions form the seed of a spanning trilateration graph $\mathbb{G}$ with high probability. By Theorem 10, the positions of all nodes in $\mathbb{G}$ can be computed in linear time. Since $\mathbb{G}_n(r)$ is spanned by $\mathbb{G}$, it can be realized in linear time.

■

D. Localization in Random Sensor Networks

We now study a simple localization protocol for random sensor networks we call ITP in Fig. 7. Theorem 13 allows us to analyze the effectiveness of our procedure.

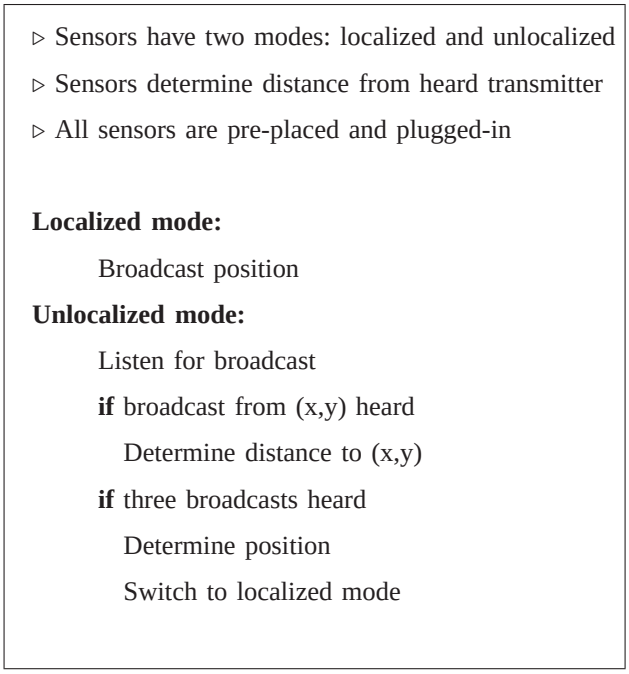

Fig. 7. The iterative trilateration protocol (ITP).

Definition 5: A *random sensornet* $\mathbb{S}_n(r)$ is a sensornet of n sensors with sensing radius r placed at random on $[0,1]^2$ by a two-dimensional Poisson point process. A *beacon* is a sensor that knows its position.

One could define a random sensornet in terms of a uniform distribution over $[0,1]^2$, but we do not consider this case.

The following results are summarized in Table I.

Claim 3: For some $r \in O(\sqrt{\frac{\log n}{n}})$, with high probability, all sensors in $\mathbb{S}_n(r)$ will have determined their positions with ITP by $O(\sqrt{\frac{n}{\log n}})$ time if three beacons are placed anywhere in $[0,1]^2$ so that they are in sensing range of each other.

Proof: We set r and partition $[0,1]^2$ into square cells as in the proof of Theorem 12. We will now show that we can have an entire grid cell within range of the three beacons. Let the beacons

lie at points P_1, P_2, and P_3. We know that $d(P_i, P_j) \leq r$, for $i \neq j$. Consider the smallest circle C enclosing the three beacons. Assume the center of C is P_c.

Consider the case that P_1, P_2, and P_3 are all are on C. We can bound the radius R of C as follows. Consider the angles of the three sectors $P_1P_cP_2$, $P_2P_cP_3$, and $P_3P_cP_1$, all less than $180°$.

For each $P_iP_cP_j$, to guarantee that $d(P_i, P_j) \leq r$, we have the constraint that $R \leq \frac{r}{2sin(P_iP_cP_j/2)}$. The most restrictive of these contraints on R is the one corresponding to the largest $P_iP_cP_j$, which is at most $180°$. Thus, we have that $R \leq r/2$. Now we draw a circle C' centered at P_c with radius $(1 - 1/2)r$. Using the triangle inequality we have that the distance from P_i to any point inside C' is less than or equal to $(1/2)r + (1 - 1/2)r = r$.

The case that only two of P_1, P_2, and P_3 are on C is similar since we also have that $R \leq r/2$.

Thus we have a circular area of $\Theta(r^2)$ wholly within range of the three beacons. We offset the grid partition such that an entire cell is within this area and thus localized in the first time-step. We label this cell 0 and proceed with labelling the remaining cells as in the proof of Theorem 12. We say that a layer is localized when all sensors in that layer have determined their positions. Assuming ITP broadcast, distance calculation, and trilateration take place in constant time, L_0 will be localized in a single constant-time step because all nodes contained therein are connected to the three beacons. Additionally, given L_i localized, ITP will localize L_{i+1} in a single constant-time step. Therefore, all layers will be localized in at most $O(\sqrt{\frac{n}{\log n}})$ steps and our claim is established. ■

Claim 4: For some $r \in O(\sqrt{\frac{\log n}{n}})$, with high probability, all sensors in $\mathbb{S}_n(r)$ can determine their positions with ITP and will have done so by expected time of $O(\sqrt{\log n})$ if beacons are placed on $[0, 1]^2$ by a Poisson point process of intensity $O(n/\log n)$.

Proof: We set r and partition $[0, 1]^2$ into square cells of area A as in the proof of Theorem 12. The Poisson point process places beacons into each cell at a rate $\lambda \propto nA/\log n \in O(1)$. Therefore, the probability that a cell contains at least three beacons is a constant p which is independent of n.

The probability that all cells contain less than three beacons is $q^{O(n/logn)}$, where $q = 1 - p$, so some cell contains at least three beacons with high probability, and consequently, all sensors can localize as in claim 3.

We now bound the expected time it takes for every sensor to localize given some cell contains three beacons. We say a cell is localized if every sensor it contains has determined its position. In a single constant-time step, ITP localizes a cell if it contains three beacons or if any of its neighbors

are localized. Because of this, in what follows we will refer to discretized time rather than steps.

The probability that a particular cell does not localize by time k is the probability that all cells within a square of cells with side $2k+1$ contain fewer than three beacons, $q^{(2k+1)^2}$. The probability that the last cell to localize does so after a certain time is the same as the probability that at least one of the cells localizes after that time. More formally, where t_i is the time at which square i localizes, since the total number of cells is $O(\frac{n}{\log n})$, the following is true,

$$Pr[max(t_i) > k] \in min(1, O(\frac{n}{\log n})q^{O(k^2)}).$$

Since the time to localize is a positive random variable, we can use the upper tail probabilities to determine its expected value,

$$E[max(t_i)] \in \sum_{k=0}^{\infty} min(1, O(\frac{n}{\log n})q^{O(k^2)}).$$

Observing that for some $k_0 \in O(\sqrt{\log n - \log\log n})$,

$$O(\frac{n}{\log n})q^{O(k^2)} > 1 \iff k < k_0,$$

we see that

$$E[max(t_i)] \in O(\sqrt{\log n}) + O(\frac{n}{\log n}) \sum_{k=k_0}^{\infty} q^{O(k^2)}.$$

In calculations we will not include here, it can be shown that $O(\frac{n}{\log n}) \sum_{k=k_0}^{\infty} q^{O(k^2)} \in O(1)$.

We have thus shown that with high probability, all sensors will localize in expected time of $O(\sqrt{\log n})$. ■

Claim 5: For some $r \in O(\sqrt{\frac{\log n}{n}})$, with high probability, all sensors in $\mathbb{S}_n(r)$ can determine their positions and will have determined their positions by $O(1)$ time if beacons are placed on $[0,1]^2$ by a Poisson point process of intensity $O(n)$.

Proof: If $r \in O(\sqrt{\frac{\log n}{n}})$, the Poisson point process places beacons in the sensing region of a sensor at rate $\lambda \propto nr^2 \propto \log n$. Since we expect $O(\log n)$ beacons connected to every sensor, with high probability, we will have $O(1)$ *i.e.*, at least three beacons connected to every sensor, and all sensors will localize in $O(1)$ time with high probability. ■

beacons	sensing radius	E[t_{loc}]
$O(1)$	$O(\sqrt{\frac{\log n}{n}})$	$O(\sqrt{\frac{n}{\log n}})$
$O(\frac{n}{\log n})$	$O(\sqrt{\frac{\log n}{n}})$	$O(\sqrt{\log n})$
$O(n)$	$O(\sqrt{\frac{\log n}{n}})$	$O(1)$

TABLE I

LOCALIZATION IN VARIOUS BEACON PLACEMENT SCHEMES.

VII. Simulation Study of Localization in Random Networks in 3-space

We simulate random geometric graphs in 3-space by generating points randomly in $[0, 1]^3$, placing four beacons in the center of the unit cube within sensing range of each other. We then simulate ITP by localizing nodes in computational rounds in which we determine positions for all nodes connected to four nodes with known position. We terminate the simulation when a round does not determine the position of any node. Note that that while these simulations are in 3-space, the theory of the previous section for 2-space is indicative of the 3-space results. In our first simulation, for three values of r, we track the percentage of nodes whose positions can be determined. We observe in Fig. 8 an increasingly sharp phase transition in the percentage of localizable nodes as we increase n.

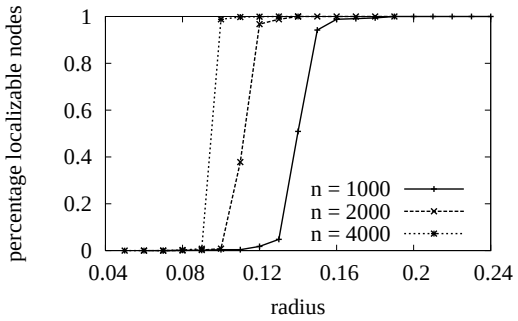

Fig. 8. Percentage of nodes localizable with 4-beacon ITP.

In our second simulation, we calculate the smallest radius at which the percentage of localizable nodes is greater than 95%. We see in Fig. 9 behavior similar to the analytical results of the plane in the preceding section. Note that the analytical asymptotic result more accurately models actual

behavior as n increases. The difference for small n is explained by the contribution of logarithmic terms in the localization probability that becomes significant when n is small.

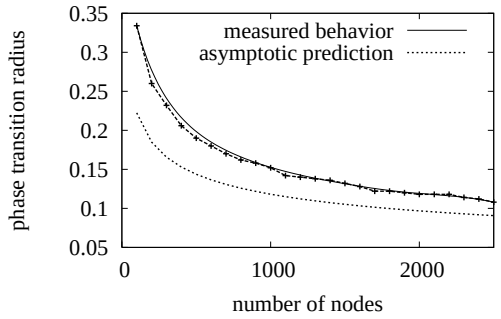

Fig. 9. Trilateration graph phase transition radius in $G_n(r)$.

Our last simulations investigate the number of computational rounds necessary to localize all nodes that can be localized. In Fig. 10, we observe for $n = 2000$ that the percentage of localized nodes at a given step increases dramatically with modest increases in sensing radius. Note that below the phase transition, at $r = 0.1$, the procedure fails to localize practically any nodes and completes in four steps. For r straddling the phase transition, Fig. 11 plots the number of steps before completion. The spike is due to a sudden increase in connectedness above the phase transition at which the radius is minimal for total localizability.

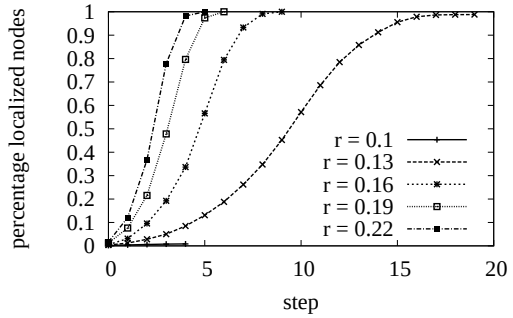

Fig. 10. Time-evolution of the number of localized nodes.

VIII. Related Work

Network localization is after four years of intense research very much still an active field. The previous approaches can be classified into two types: coarse-grained [10], [11], [28], [42], [50], [57]

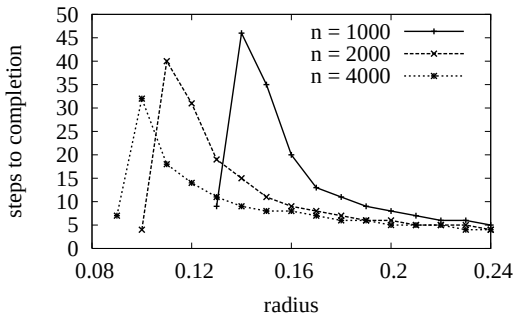

Fig. 11. Required steps for algorithm completion.

and fine-grained [1], [7], [12], [18], [37], [44], [46], [47], [49], [52], [53], [56]. The focus of this paper is fine-grained localization. As we discussed in the Introduction, the previous approaches were mainly heuristics, and a theoretical foundation was lacking. Eren *et al.* addressed this problem in a preliminary form in [20], providing the first theoretical analysis of network localization. Since that work was originally published, the authors in [44] applied our conditions to produce a localization algorithm based on trilateration graphs under noisy distance measurements which is effective in relatively dense networks. Our conditions for unique localizability are also applied in [49] in an algorithm which uses mobility to obtain distance measurements which result in globally rigid constraint structures. The algorithm presented in [7] uses semidefinite programming and is effective in relatively dense overconstrained networks. A technique called multidimensional scaling (MDS), which requires estimation of the complete distance matrix, is applied in [56], yielding good results in dense networks. Distances between nodes need not necessarily be used in estimation of the complete distance matrix, so this approach can be seen either as fine or coarse-grained localization. The work on fine-grained localization which inspired our research into the fundamental theory of the problem is [53]. The authors produced an innovative algorithm effective in practice for sufficiently dense networks, but left open the issues of unique localizability and complexity and did not necessarily localize all localizable nodes.

A related problem called molecular conformation has been studied in the chemistry community, *e.g.*, [2], [30], [45]. However, the focus of these studies is on 3-space. Also, since the structure of a molecule is given, they do not consider the network construction process.

One major building block of our analysis is rigidity theory and computational geometry. Rigidity has been long studied in mathematics and structural engineering (see for example [40], [60], [31],

[51], [61]) and has a surprising number of applications in many areas.

We formally analyzed the performance of network localization in networks of randomly placed nodes. Even though some researchers have studied random graphs in sensor networks, *e.g.*, [16], [17], [22], [39], the focus is mainly on routing but not on localization. Phase transitions for connectivity in random geometric graphs are explored in [41], and motivated some of our research on phase transitions for localizability in random sensor networks.

IX. Conclusion and Future Work

The unique localization of networks from distance measurements shares a number of features with work in several other active fields of study: rigidity and global rigidity in frameworks; the coordination of formations of automonous agents; and geometric constraints in CAD. In this paper, we have drawn on techniques and results from the first two fields, also combined in some previous joint work [21], as well as specific results on global rigidity [13], [34]. With these concepts, we were able to lay a coherent solid foundation for the underlying problem of when a network is uniquely localizable, for almost all configurations of the points. Specifically, we constructed a formation and then a graph for each network such that the localization problem for the network is uniquely solvable, almost always, if and only if the corresponding graph is generically globally rigid. From these connections, we drew specific results and showed that the trilateration networks are uniquely localizable for almost all initial locations.

It should be noted that global rigidity is a graph property linked with the unique localizability of an entire network. In work by Goldenberg *et al.* in [26], it was observed that even in networks with non-globally rigid grounded graphs, there may exist uniquely localizable nodes. A theoretical investigation of this phenomenon of *globally linked nodes* was later given by Jordan *et al.* in [35].

It also deserves note that as stated, the localization problem with precise distance is not in general numerically well posed since even if it is solvable with the given data, it may be unsolvable with data arbitrarily "close" to that which is given. In practical terms, this means that special attention must be paid to the computation process and to assessing the significance of "approximate solutions." It also means that only graphs which are generically globally rigid are capable of having computationally stable solutions for given data sets. This confirms our choice of conceptual framework for this problem. However, we comment that even approximate solutions are hard to compute due to the hardness of the localization problem.

Specifically, we have shown that the localization problem is NP-hard in the worst case for sparse graphs unless P=NP or RP=NP, if certain mild forms of approximation are permitted. This worst-case result for sparse graphs stands in contrast to results that show that localization is possible for dense graphs [7] or with high probability for random geometric graphs. The open questions that remain are where the boundary lies between our negative result and these positive results. In particular:

- Is there an efficient algorithm for *approximate* localization in sparse graphs, either by permitting moderate errors on distances or by permitting the algorithm to misplace some small fraction of the sensors?
- Given that the difficulty of the problem appears to be strongly affected by the density of nodes (and the resulting number of known distance pairs), what minimum density is necessary to allow localization in the worst case?
- How are these results affected by more natural assumptions about communications ranges, allowing different maximum distances between adjacent nodes or the possibility of placing small numbers of high-range beacons?
- How does the dimension (e.g., in the plane or in 3-space) affect the problem?

Answers to any of these questions would be an important step toward producing practical localization algorithms.

One potential direction to resolve the computational complexity issue is to introduce other modalities. In particular, other work such as [47] approaches network localization with angles, bearings and headings in addition to some distance constraints. Drawing on more general work on geometric constraints such as angles and directions in CAD such as in [55], we have further generic global uniqueness results that can give new insights where certain patterns of angles or headings are used [21], as well as insights into the complexity of general patterns of angle constraints. This will be explored further in a future paper.

ACKNOWLEDGMENT

We thank Joseph T. Chang, Stanley C. Eisenstat, and Jie (Archer) Lin for valuable suggestions. We thank Theodore Jewell for valuable comments on an early draft. We also thank the anonymous reviewers for their valuable comments and suggestions which improved the paper.

REFERENCES

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

[2] A. Y. Alfakih, A. Khandani, and H. Wolkowicz. Solving Euclidean distance matrix completion problems via semidefinite programming. *Computat. Optim. Appl.*, 12(1-3):13–30, 1999.

[3] B. Anderson, P. Belhumeur, T. Eren, D. Goldenberg, A. Morse, W. Whiteley, and R. Wang. Global properties of easily localizable sensor networks. Preprint Australian National University, 2005.

[4] M. Appel and R. Russo. The maximum vertex degree of a graph on uniform points in $[0,1]^2$. *Adv. Applied Probability*, 29:567–581, 1997.

[5] J. Aspnes, D. Goldenberg, and Y. R. Yang. On the computational complexity of sensor network localization. In *Proceedings of First International Workshop on Algorithmic Aspects of Wireless Sensor Networks*, Turku, Finland, July 2004.

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

[7] 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.

[8] B. Bollobás. *Random Graph Theory*. Academic Press, London, 1985.

[9] H. Breu and D. G. Kirkpatrick. Unit disk graph recognition is NP-hard. *Computational Geometry*, 9(1-2):3–24, 1998.

[10] 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.

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

[12] 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.

[13] R. Connelly. Generic global rigidity. Available at `http://www.math.cornell.edu/~connelly/`, Oct. 2003.

[14] R. Connelly. Generic global rigidity. *Discrete Comp. Geometry*, 33, 2005.

[15] G. Crippen and T. Havel. *Distance Geometry and Molecular Conformation*. John Wiley & Sons, 1988.

[16] J. Díaz, J. Petit, and M. Serna. A random graph model for optical networks of sensors. Technical Report LSI-02-72-R, Departament de Llenguatges i Sistemes Inform'atics, Universitat Polit'ecnica de Catalunya, December 2002.

[17] J. Díaz, J. Petit, and M. Serna. Random scaled sector graphs. Technical Report LSI-02-47-R, Departament de Llenguatges i Sistemes Inform'atics, Universitat Polit'ecnica de Catalunya, June 2002.

[18] 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.

[19] T. Eren, P. Belhumeur, B. Anderson, and A. Morse. A framework for maintaining formations based on rigidity. In *Proceedings of the 15th IFAC World Congress*, Barcelona, Spain, July 2002.

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

[21] T. Eren, W. Whiteley, A. Morse, and P. Belhumeur. Sensor network topologies of formations with distance - direction - angle constraints. In *Proceedings of the 42nd IEEE Conference on Decision and Control*, Maui, HI, USA, Mar. 2003.

[22] A. Farago. Scalable analysis and design of ad hoc networks via random graph theory. In *Proceedings of the 6th international workshop on Discrete algorithms and methods for mobile computing and communications*, pages 43–50. ACM Press, 2002.

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

[24] M. Garey and D. Johnson. *Computers and Intractability*. W.H. Freeman and Co., New York, NY, 1979.

[25] J. Gill. Computational complexity of probabilistic Turing machines. *SIAM Journal on Computing*, 6(4):675–695, 1977.

[26] D. Goldenberg, A. Krishnamurthy, W. Maness, Y. R. Yang, A. Young, A. S. Morse, A. Savvides, and B. Anderson. Network localization in partially localizable networks. In *Proceedings of IEEE INFOCOM '05*, Miami, FL, Apr. 2005.

[27] J. Graver, B. Servatius, and H. Servatius. *Combinatorial Rigidity*, volume 2 of *AMS Monographs*. American Math. Society, Providence, 1993.

[28] 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.

[29] B. Hendrickson. Conditions for unique graph realizations. *SIAM J. Comput.*, 21(1):65–84, 1992.

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

[31] L. Henneberg. *Die Graphische Statik der starren Systeme*. Johnson Reprint, 1911.

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

[33] J. E. Hopcroft and R. Tarjan. Dividing a graph into triconnected components. *SIAM J. Comput.*, 3:135–158, 1973.

[34] B. Jackson and T. Jordàn. Connected rigidity matroids and unique realizations of graphs. *J. Combinatorial Theory B*, 94:1–29, 2005.

[35] B. Jackson, T. Jordán, and Z. Szabadka. Globally linked pairs of vertices in equivalent realizations of graphs. Technical Report TR-2005-07, Egerv'ary Research Group, Budapest, 2005.

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

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

[38] 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.

[39] B. Krishnamachari, S. B. Wicker, and R. Bejar. Phase transition phenomena in wireless ad-hoc networks. In *Proceedings of Symposium on Ad-Hoc Wireless Networks, GlobeCom 2001*, Antonio, Texas, 2001.

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

[41] X.-Y. Li, P.-J. Wan, Y. Wang, and C.-W. Yi. Fault tolerant deployment and topology control in wireless networks. In *Proceedings of the Fourth ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, Annapolis, MD, June 2003.

[42] H. Lim and J. Hou. Localizatio for anisotropic sensor networks. In *Proceedings of IEEE INFOCOM '05*, Miami, FL, Apr. 2005.

[43] G. L. Miller and V. Ramachandran. A new graph triconnectivity algorithm and its parallelization. *Combinatorica*, 12:53–76, 1992.

[44] 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.

[45] J. More and Z. Wu. Global continuation for distance geometry problems. *SIAM Journal on Optimization*, 7:814–836, 1997.

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

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

[48] M. D. Penrose. On k-connectivity for a geometric random graph. *Random Structures and Algorithms*, 15(2):145–164, 1999.

[49] N. Priyantha, H. Balakrishnan, E. Demaine, and S. Teller. Mobile-assisted localization in wireless sensor networks. In *Proceedings of IEEE INFOCOM '05*, Miami, FL, Apr. 2005.

[50] 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.

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

[52] 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.

[53] 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.

[54] J. Saxe. Embeddability of weighted graphs in k-space is strongly NP-hard. In *Proceedings of the 17th Allerton Conference in Communications, Control and Computing*, pages 480–489, 1979.

[55] B. Servatius and W. Whiteley. Constraining plane configurations in cad: combinatorics of lengths and directions. *SIAM Journal on Discrete Mathematics*, 12:136–153, 1999.

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

[57] R. Stoleru, T. He, J. Stankovic, and D. Luebke. High-accuracy, low-cost localization system for wireless sensor network. In *Proceedings of The Third ACM Conference on Embedded Networked Sensor Systems (SenSys)*, Nov. 2005.

[58] L. G. Valiant and V. V. Vazirani. NP is as easy as detecting unique solutions. *Theoretical Computer Science*, 47(1):85–93, 1986.

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

[60] 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.

[61] W. Whiteley. Rigidity and scene analysis. In J. Goodman and J. O'Rourke, editors, *Handbook of Discrete and Computational Geometry*, pages 1327–1354. CRC Press, 2004.