Beyond Triangle Inequality: Sifting Noisy and Outlier Distance Measurements for Localization

ZHENG YANG, Tsinghua University and Hong Kong University of Science and Technology
LIRONG JIAN, Hong Kong University of Science and Technology
CHENSHU WU, Tsinghua University
YUNHAO LIU, Tsinghua University and Hong Kong University of Science and Technology

Knowing accurate positions of nodes in wireless ad hoc and sensor networks is essential for a wide range of pervasive and mobile applications. However, errors are inevitable in distance measurements and we observe that a small number of outliers can degrade localization accuracy drastically. To deal with noisy and outlier ranging results, triangle inequality, is often employed in existing approaches. Our study shows that triangle inequality has many limitations, which make it far from accurate and reliable. In this study, we formally define the outlier detection problem for network localization and build a theoretical foundation to identify outliers based on graph embeddability and rigidity theory. Our analysis shows that the redundancy of distance measurements plays an important role. We then design a bilateration generic cycles-based outlier detection algorithm, and examine its effectiveness and efficiency through a network prototype implementation of MicaZ motes as well as extensive simulations. The results show that our design significantly improves the localization accuracy by wisely rejecting outliers.

Categories and Subject Descriptors: C.2.2 [**Computer-Communication Networks**]: Network Protocols

General Terms: Design, Algorithms, Performance

Additional Key Words and Phrases: Wireless sensor networks, localization, outlier detection

ACM Reference Format:
Yang, Z., Jian, L., Wu, C., and Liu, Y. 2013 Beyond triangle inequality: Sifting noisy and outlier distance measurements for localization. ACM Trans. Sensor Netw. 9, 2, Article 26 (March 2013), 20 pages.
DOI: http://dx.doi.org/10.1145/2422966.2422983

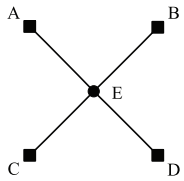
1. INTRODUCTION

Growing convergence among mobile computing devices and embedded technologies is stimulating the development and deployment of a wide range of location-aware applications, including smart space, intrusion detection, inventory management, mobile peer-to-peer computing, wireless sensor networks (WSNs) Surveillance, and so on. A number of localization approaches have been proposed in the literature and used in practice to locate wireless devices.

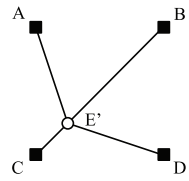
In recent years, novel schemes have been proposed for in-network localization, in which some special nodes (called beacons or anchors) know their global locations and

A preliminary version of this article appeared in *Proceedings of the 29th Annual International Conference on Computer Communications (IEEE-INFOCOM 2010)*.
This work is supported in part by the NSFC under grant 61171067 and 61170213, China Postdoctoral Science Foundation under grant 2011M500331, NSFC Major Program 61190110, the NSFC Distinguished Young Scholars Program under Grant 61125202, and National High-Tech R&D Program of China (863) under grant No. 2011AA010100.
Authors' addresses: email: {yang, jian, wu, yunhao}@greenorbs.com.

DOI: http://dx.doi.org/10.1145/2422966.2422983



(a) Ground truth.



(b) Estimated location.

Fig. 1. Multilateration with an outlier measurement, where A, B, C, and D are at corners of a square of $\sqrt{2}$, and E is at the center of that square.

the rest determine their locations by measuring the Euclidean distances to their neighbors. Several distance ranging methods, such as Radio Signal Strength (RSS) [Seidel and Rappaport 1992] and Time Difference of Arrival (TDoA) [Savvides et al. 2001], are adopted in practical systems. Based on these techniques, the ground truth of a wireless ad hoc network can be modeled as a weighted graph $G = \langle V, E, W \rangle$, where V is the set of wireless nodes, E is the set of links, and $W(u, v)$ is the distance measurement between a pair of nodes u and v. The problem of localization is to figure out the locations of other nodes based on internode distance measurements and global locations of beacon nodes.

In practice however, noise and outliers are inevitable in distance ranging. The outlier ranging results generally come from the following three sources.

Hardware malfunction or failure. Distance measurements will be meaningless when encountering ranging hardware malfunction. Besides, incorrect hardware calibration and configuration also deteriorate ranging accuracy, which has not been much emphasized in previous studies. For example, RSS suffers from transmitter, receiver, and antenna variability, and the inaccuracy of clock synchronization results in ranging errors for TDoA.

Environmental factors. RSS is sensitive to channel noise, interference, and reflection, all of which have significant impacts on signal amplitude. The irregularity of signal attenuation remarkably increases, especially in complex indoor environments. In addition, for the propagation time-based ranging measurements, the signal propagation speed often exhibits variability as a function of temperature and humidity, so we cannot assume the propagation speed is constant across a large field.

Adversary attacks. As location-based services become more and more prevalent, the localization infrastructure is becoming the target of adversary attacks. By reporting fake location or ranging results, an attacker, for example a compromised (malicious) node, can completely distort the coordinate system. Different from previous cases, the outliers here are intentionally generated by adversaries.

Ignoring their existence is not a suitable way to deal with outliers. Figure 1 shows an example of how outliers destroy localization accuracy. As shown in Figure 1, nodes A, B, C, and D (the black boxes) are beacons at known positions (four vertices of a square of length $\sqrt{2}$) and Node E (the white circle in the center of the square) is to be located. Suppose the accurate distance measurements are $|AB| = |BC| = |CD| = |AD| = \sqrt{2}$ and $|AE| = |BE| = |CE| = |DE| = 1$. Apparently, the calculated location of E is just the same as its real location when distance measurements are correct. Now suppose an outlier ranging result occurs: the distance between E and B is wrongly measured as 2 ($|BE| = 2$). In this case, if all ranging results are indiscriminately used to locate E, the estimated location E' (the white circle in Figure 1(b) calculated by multilateration

[Savvides et al. 2001; Yang and Liu 2010]) is away from the real location E. However, if we layout the outlier ranging, a better estimated location that is the same as the real location of E can be achieved.

To detect outliers, a straight forward solution is to judge graph embeddability based on triangle inequality. A graph-violating triangle inequality cannot be embeddable. However, triangle inequality has in the following two drawbacks.

Coarse granularity. In the case of Figure 1(b), the outlier ranging |BE| cannot be detected by triangle inequality since the triangle $\triangle$ABE (the distance between A and B is $\sqrt{2}$, implied by their locations) is still embeddable under the incorrect rangings. Actually in this case all triangles in Figure 1(b), including $\triangle$ABE, ACE, ADE, BCE, BDE and CDE, are embeddable and triangle inequality fails to detect outliers.

Identification. Triangle inequality just indicates the existence of outliers, but cannot identify them. Formally, for a triangle violating the inequality, we are sure that one or more distance measurements are incorrect but have no idea which they are.

Such limitations motivate us to design a novel approach to identify outlier distance measurements beyond triangle inequality. We notice that a rigid graph has discrete and finite realizations. A graph is called *rigid* if one cannot continuously deform the graph embedded in the plane while preserving distance constraints [Laman 1970]. Thus for a graph G and a pair of neighboring vertices u and v, if $G-(u,v)$ is rigid, the distance between them ($W(u,v)$) is accordingly discrete and finite under different realizations of $G-(u,v)$. Although we artificially exclude and ignore (u,v), we still have some idea of how to partially figure out $W(u,v)$. Such redundancy in internode distances is an excellent starting point to detect noise and outliers in distance ranging.

The main contributions of this work are as follows. We formally define the problem of outlier detection for localization and analyze the limitations of existing methods that are based on triangle inequality. Based on graph embeddability and rigidity, we establish the theoretical foundations for detecting outliers. An algorithm based on bilateration and generic cycles is accordingly designed to eliminate outliers during the localization process. To validate this design, a network prototype consisting of 25 MicaZ motes is constructed and extensive simulations are conducted to examine the effectiveness and efficiency of our solution.

The rest of this article is organized as follows. In Section 2, we build the network model and formulate the problem of outlier detection. Section 3 presents the theoretical foundation for identifying outliers and an algorithm based on graph embeddability and rigidity is designed in Section 4. Section 5 analyzes some practical issues of the algorithm. Sections 6 and 7 present the evaluation results from extensive simulations and field experiments. We summarize the related work in Section 8, and conclude the work in Section 9.

2. PROBLEM FORMULATION

In this section, we present the network model and formulate outlier detection for localization in wireless ad hoc and sensor networks.

2.1. Network Model

We assume that each node is located at a distinct physical location in some region of a plane and associated with a specific set of neighboring nodes. Let $\mathbb{N}$ be a network of n nodes labeled $v_1, v_2, \ldots, v_n$. Let $\pi(v_i)$ denote the ground truth position of v_i, and we suppose a small portion of nodes, called *beacons*, are at known locations.

We generate a *weighted grounded graph* [Saxe 1979; Eren et al. 2004] $G=\langle V, E, W\rangle$ to represent the geometric constraints of Network $\mathbb{N}$, where each vertex denotes a node in the network and each edge e indicates the neighborhood of its two endpoint nodes.

The distance between two neighboring nodes (or any pair of beacons) is available and defined by $W(e)$.

2.2. Error Model of Distance Ranging

Given a weighted grounded graph $G = \langle V, E, W\rangle$, W is determined by the observed information, such as measured internode distances and positions of beacons. We call W *measured distance* in the rest of this article. Because of the presence of noise, the measurements would be corrupted. Formally, for some edge $e = (v_i, v_j) \in E$, $W(e)$ is not necessarily equal to the *ground truth distance* between nodes v_i and v_j, that is $\|\pi(v_i) - \pi(v_j)\|$.

In general, there are two kinds of ranging errors in a localization system: *normal error* and *outlier error*. Coming from limitations of hardware and computation precision, normal errors are moderate and predictable, and have attracted a lot of research effort [Moore et al. 2004; Liu et al. 2006]. By contrast, outlier errors are much more severe and unpredictable. They are caused by hardware malfunction or failure, adversary attacks, and so on. We formulate two such kinds of errors as normal edges and outlier edges, respectively, as follows.

Definition 2.1. Given a weighted grounded graph $G = \langle V, E, W\rangle$, if an edge $e = (v_i, v_j) \in E$ is a *normal edge*, then $W(e) = \|\pi(v_i) - \pi(v_j)\|$; otherwise, e is an *outlier edge*, and $W(e) = \|\pi(v_i) - \pi(v_j)\| + \epsilon$, where ϵ is an arbitrary continuous random variable.

In the error model, we assume normal edges contain no ranging noise, which is a good starting point for addressing the outlier detection problem. This assumption and abstraction is also accepted in Addressing Kung et al. [2009], Berger et al. [1996], and Moore et al. [2004]. In Section 5, by introducing an error threshold, the proposed algorithm can handle a more practical error model, where normal edges have moderate ranging errors.

2.3. Outlier Detection Problem

Since outlier errors have seriously negative impacts, detecting and eliminating them is essential for highly accurate localization. We formulate the problem of outlier detection as follows.

Problem 2.2 (*Outlier Detection*). Given a weighted grounded graph $G = \langle V, E, W\rangle$ consisting of normal edges and outlier edges, identify the outlier edges in G.

Each edge $e \in E$ corresponds to a measured distance or a pair of beacons at known positions. An outlier edge indicates, in the former case, the distance measurement has a large error, while in the latter, the global positions of two corresponding beacons are corrupted.

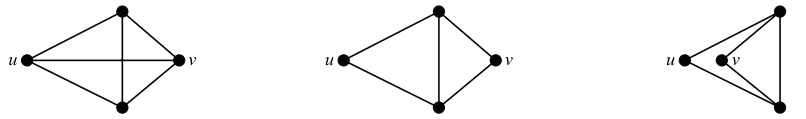
All normal edges are compatible, for example, satisfying triangle inequality, since they are telling the truth, while it is quite probable that an outlier edge is inconsistent with other normal edges and outlier edges, for example, violating triangle inequality, because it is just a random variable independent of the ground truth. This observation motivates us to detect outlier edges through exposing geometric inconsistency.

3. THEORETICAL FOUNDATION

In this section, we analyze the relationship between outlier edges and graph embeddability, and apply rigidity theory to build a theoretical foundation for outlier detection.

3.1. Embeddability of Weighted Graphs

In Saxe [1979], the problem of embeddability of weighted graphs is formally defined, and its computational complexity is also discussed.



(a) Ground truth (b) Realization 1 (c) Realization 2

Fig. 2. Remaining rigid after removing (u, v).

Definition 3.1. Given a weighted graph, $G = \langle V, E, W \rangle$, a *realization* (*embedding*) of G in the k-dimensional Euclidean space, $\mathbb{R}^k$, is a function r, mapping V into $\mathbb{R}^k$, such that for each edge $e = (v_i, v_j) \in E$, $W(e) = \|r(v_i) - r(v_j)\|$.

Problem 3.2 (*Embeddability*). Given a weighted graph, $G = \langle V, E, W \rangle$, determine whether G is *k-embeddable*, which means whether there is a realization (embedding) r of G in $\mathbb{R}^k$.

Note that, rather than planar embeddability, in this work embeddability means the existence of a realization of a graph in a 2D a plane that satisfies all internode measured distances. Since we focus on networks in a 2D plane, the term embeddable is used instead of 2-embeddable in the following discussion for simplicity. The following theorem reveals a close relationship between outlier existence and graph embeddability.

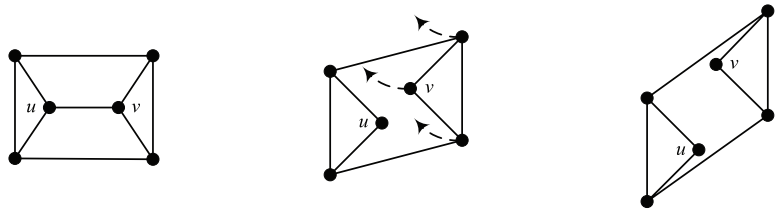
THEOREM 3.3. *Given a weighted grounded graph $G = \langle V, E, W \rangle$, if G is unembeddable, then E contains at least one outlier edge.*

PROOF. Assume that E contains no outlier edge, that is all the edges of G are normal edges. By the definition, for any normal edge $e = (v_i, v_j) \in E$, we have $W(e) = \|\pi(v_i) - \pi(v_j)\|$. Let $r(v_k) = \pi(v_k)$ for every vertex $v_k \in V, k = 1, 2, \ldots, n$, then $W(e) = \|r(v_i) - r(v_j)\|$. Thus, G is embeddable, contradicting that G is unembeddable. □

Theorem 3.3 has overcome the issue of coarse granularity associated with triangle inequality. Recall Figure 1(b), where triangle inequality fails to detect outliers. But Theorem 3.3 succeeds, since the underlying graph is unembeddable. However, similar to triangle inequality, it is also impossible for Theorem 3.3 to tell which edges are outliers or which are not outliers. Generally, under an arbitrary error model, for any given weighted grounded graph, the inverse of Theorem 3.3 doesn't hold, namely, even if G contains outliers, it can still be embeddable. One extreme example is illustrated as follows. Let $G = \langle V, E, W \rangle$, and for every edge $e = (v_i, v_j) \in E$, let $W(e) = \frac{1}{2}\|\pi(v_i) - \pi(v_j)\|$. Thus, every edge in E is an outlier, but apparently, G is embeddable. However, compared with an arbitrary error model, we are more interested in whether under the normal edge and outlier edge error model, there is some kind of weighted graph whose being embeddable would imply no outlier.

Suppose the graph G shown in Figure 2(a) is embeddable and we are interested in whether the edge $e = (u, v)$ is an outlier. After removing e, while preserving the remaining distance constraints, there are only two feasible options for the distance between u and v, corresponding to the two realizations illustrated in Figures 2(b), and 2(c), respectively. Thus, the embeddability of G suggests that $W(u, v)$ is one of the two options, and what's more, $e = (u, v)$ is a normal edge that is telling the truth, as it is with probability of 0 that the measured distance of an outlier edge is in a set consisting of two discrete values. This conclusion makes the quadrilateral structure play an important role in robust localization [Liu et al. 2005; Li et al. 2005] and phantom node filtering [Hwang et al. 2007].

As aforementioned, the inverse of Theorem 3.3 doesn't hold for all network structures. Suppose the graph G shown in Figure 3(a) is embeddable. After removing edge



(a) Ground truth (b) Flexing 1 (c) Flexing 2

Fig. 3. Flexing after removing (u, v).

$e = (u, v)$, while preserving the remaining distance constraints, the graph structure can continuously flex, as illustrated in Figures 3(b) and 3(c), which means the distance between u and v can continuously vary in an interval having a non-zero measurement. Similar to Figure 2, the embeddability of G implies that the measured distance of e falls into the feasible interval. However, since that interval has a non-zero measurement, it is with probability greater than 0 that e is an outlier edge and the measured distance of e falls into that interval. Thus, we cannot claim that e is not an outlier even knowing G is embeddable.

3.2. Redundant Rigidity

Previous studies have shown that the network localizability problem [Eren et al. 2004; Goldenberg et al. 2006] is closely related to graph rigidity. In this article, we will show that the outlier detection problem also has a close relationship with graph rigidity. A graph is *generically rigid* (or just called rigid) if it has no continuous deformation other than global rotation, translation, and reflection, while preserving distance constraints [Laman 1970]; otherwise, it is *flexible*. Here the word "generically" means the distances are algebraically independent, that is, no degeneracy. A graph is called *generically redundantly rigid* (or just called redundantly rigid) if it remains rigid after removing any single edge. Figure 3(a) shows a rigid but not redundantly rigid graph, since after removing $e = (u, v)$, this graph becomes flexible, as illustrated in Figures 3(b) and 3(c). It is also easy to check that after removing any edge of the graph shown in Figure 2(a), for example, $e = (u, v)$, the graph still has no continuous deformation, while preserving the remaining distance constraints. Thus, it is redundantly rigid.

Before presenting the relationship between the outlier detection problem and graph rigidity, we first give out the definition of *outlier disprovable* as follows.

Definition 3.4. Given a weighted graph $G = \langle V, E, W \rangle$, G is *outlier disprovable* if and only if the embeddability of G implies that it contains no outlier edge.

LEMMA 3.5. *Given a redundantly rigid weighted graph G, if it is embeddable, then with probability of 1, G contains no outlier edge.*

PROOF. Let $e = (u, v)$ be an outlier edge in G, and G is embeddable. Because of the redundant rigidity of G, after removing e, it is still rigid, which means there are finite discontinuous realizations up to congruence for $\tilde{G} = G - \{e\}$. In each realization of $\tilde{G}$, the distance between node u and node v is fixed. So, there are finite discrete values, denoted by set S, for the distance between u and v. As G is embeddable and $e \in E$, the measured distance of e is in S. However, as an outlier edge, the measured distance of e is a continuous random variable, which means that with probability of 0, $W(e) \in S$, since S has measure zero. So, with probability of 1, there is no outlier edge in G. □

Fig. 4. Redundantly rigid components in a graph with outlier edge (u, v).

To analyze the necessary condition for outlier disprovability, we present a proposition first proved by Hendrickson [1992].

PROPOSITION 3.6. *If a graph G is flexible, then for almost all realizations r of G, the flexing space of r contains a submanifold that is diffeomorphic to the circle.*

LEMMA 3.7. *Given a weighted graph G, if G is outlier disprovable, then G is redundantly rigid.*

PROOF. Assume the only interesting case is that G is rigid but not redundantly rigid. Thus, some edge $e = (u, v)$ exists, whose removal produces a flexible graph $\tilde{G}$. By Proposition 3.6, for almost all realizations r of $\tilde{G}$, the flexing space of r contains a submanifold diffeomorphic to the circle. The distance between nodes u and v will be a multivalued function for almost every point on this circle. Hence, the set of distances between u and v corresponding to r, denoted by $S_{d(u,v)}(r)$, has nonzero measure. So, even e is an outlier, with probability greater than 0, $W(e) \in S_{d(u,v)}(r)$, resulting in G is embeddable. Therefore, G is not outlier disprovable. □

Combining Lemma 3.5 and Lemma 3.7, we obtain the following theorem.

THEOREM 3.8. *Given a weighted graph G, G is outlier disprovable if and only if G is redundantly rigid.*

By Theorem 3.3 and Theorem 3.8, we present a framework for outlier detection, which is outlined by Algorithm 1. We explain the framework through an example shown in Figure 4, where solid lines denote normal edges and the outlier is denoted by the dashed line. Figure 4(a) is the ground truth, call it G. Besides Figure 4(a), there are three other redundantly rigid components, shown in Figures 4(b), 4(c), and 4(d), respectively. Intuitively, Figures 4(a), 4(b), and 4(c) are unembeddable, since they all contain $e = (u, v)$, the outlier edge. Following our algorithm, all the edges of G are marked outlier edges before checking the embeddability of Figure 4(d). As Figure 4(d) does not contain the outlier edge, it is embeddable. All the edges in Figure 4(d) are marked normal edges. Finally, we mark all the edges of G except $e = (u, v)$ as normal edges, leaving e marked as an outlier edge.

Algorithm 1 addresses the outlier detection problem from the perspective of components: if a redundantly rigid component is embeddable, then all its edges are normal edges. Now, let us look at the problem from another perspective: a single edge, call it e. Intuitively, by Theorem 3.8, through checking all redundantly rigid components containing e, if one is embeddable, then we know that e is a normal edge. The question is whether it is really necessary to check all the redundantly rigid components containing e, especially when e is actually an outlier edge.

Figure 5 illustrates an example for which we want to verify edge $e = (u, v)$. There are two redundantly rigid components containing e: the whole graph G and the component H highlighted by the dashed circle. If H is embeddable, then e is definitely marked as a normal edge; otherwise, G is also unembeddable as it contains H, and finally

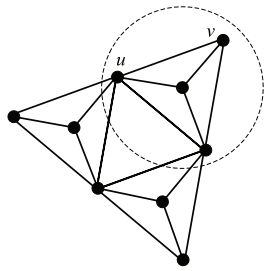
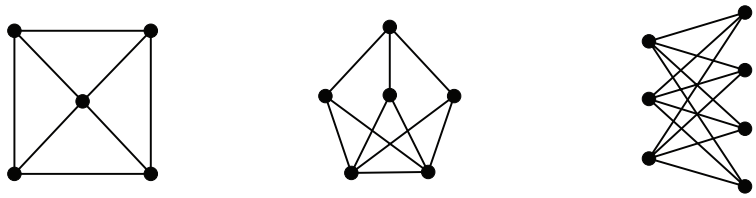


Fig. 5. One edge contained in more than one redundantly rigid component.



(a) (b) (c)

Fig. 6. Examples of generic cycles.

ALGORITHM 1: Component-based Outlier Detection.

```
1: for all redundantly rigid component H in G do
2:   if H is embeddable then
3:     mark every edge e ∈ H a normal edge.
4:   else
5:     for all edge e ∈ H not marked a normal edge do
6:       mark e an outlier edge.
7:     end for
8:   end if
9: end for
```

e is marked as an outlier. Thus, whether we can verify e, is only determined by the embeddability of H, and has nothing to do with G. Actually, for any edge in G, checking the embeddability of G is meaningless. This example suggests that for a single edge e, there are some minimally redundantly rigid components containing e such that the identity of e is only determined by the embeddabilities of these components.

3.3. Generic Cycle

Definition 3.9 (*[Jackson and Jordán 2005; Berg and Jordan 2002]*). A graph $G = \langle V, E\rangle$ with $|V| \geq 4$ is called a *generic cycle* if $|E| = 2|V| - 2$ and G satisfies

$$i(X) \leq 2|X| - 3 \text{ for all X} \subset \text{V with } 2 \leq |\text{X}| \leq |\text{V}| - 1, \tag{1}$$

where $i(X)$ denotes the number of edges induced by X in G.

Three instances of generic cycle are illustrated in Figures 6(a), 6(b), and 6(c). By the Laman condition [Laman 1970] and the definition of redundant rigidity, a generic cycle is minimally redundantly rigid. Here the word "minimally" means containing no redundantly rigid proper subgraph.

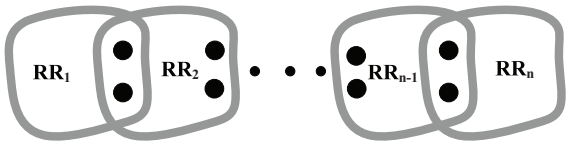


Fig. 7. A chain of n redundantly rigid components.

The following lemma shows the relationship between generic cycles and redundantly rigid graphs.

LEMMA 3.10 ([JACKSON AND JORDÁN 2005]). *A graph G is redundantly rigid if and only if G is rigid and each edge of G belongs to a generic cycle in G.*

THEOREM 3.11. *Let $G_1 = \langle V_1, E_1 \rangle$ and $G_2 = \langle V_2, E_2 \rangle$ be two redundantly rigid graphs with $|V_1 \cap V_2| \geq 2$. Then $G_1 \cup G_2$ is redundantly rigid.*

PROOF. By the Laman condition [Laman 1970] and the rigidity of G_1 and G_2, let H_1 and H_2 be a minimally rigid spanning subgraph of G_1 and G_2, respectively. Since $|V_1 \cap V_2| \geq 2$, then $H_1 \cup H_2$ is rigid [Jackson and Jordán 2005]. Furthermore, $H_1 \cup H_2$ is a spanning subgraph of $G_1 \cup G_2$. Thus, $G_1 \cup G_2$ is rigid. For any e in $G_1 \cup G_2$, without loss of generality, let $e \in E_1$, according to Lemma 3.10, there is a generic cycle $\tilde{H}$ in G_1 containing e. Apparently, $\tilde{H}$ is in $G_1 \cup G_2$. Therefore, $G_1 \cup G_2$ is redundantly rigid, by Lemma 3.10. □

By Theorem 3.11, Algorithm 1 suffers from a problem of computational prohibitiveness, termed as combinatorial explosion. Figure 7 shows an example of the inefficiency of Algorithm 1, where n basic redundantly rigid components, that is, RR_1, RR_2, ..., RR_{n-1}, and RR_n, compose a chain, and every pair of neighboring basic redundantly rigid components have more than one common vertex. According to Theorem 3.11, there are $n + (n-1) + \ldots + 1 = \frac{1}{2}n(n-1)$ redundantly rigid subgraphs by the combination of those n basic redundantly rigid components, all embeddabilities of which are checked by Algorithm 1. This is significantly inefficient, because as long as the embeddabilities of those n basic redundantly rigid subgraphs have been examined, others' embeddabilities can be implied and totally meaningless for outlier detection.

Based on Lemma 3.10, from the perspective of a single edge, we propose another framework for outlier detection as Algorithm 2, recall Figure 5. For every edge $e \in G$, through employing Algorithm 2, only the embeddability of the complete graph K_4 containing e, which is the smallest generic cycle, needs to be checked. And the embeddability of G would never be considered. By the redundant rigidity of generic cycles and Lemma 3.10, we obtain the following theorem, which shows that Algorithm 1 and Algorithm 2 are equivalent.

THEOREM 3.12. *Given a weighted graph $G = \langle V, E, W \rangle$, for any edge $e \in E$, e is marked a normal edge by Algorithm 1 if and only if e is also marked a normal edge by Algorithm 2.*

PROOF. *Sufficiency*. e is marked a normal edge by Algorithm 2 if and only if there is an embeddable generic cycle H containing e. Because of the redundant rigidity of a generic cycle, the embeddability of H is also checked in Algorithm 1 and e is marked a normal edge.

Necessity. e is marked a normal edge by Algorithm 1 if and only if there is a embeddable redundantly rigid component H containing e. By Lemma 3.10, there is a generic

ALGORITHM 2: Edge-based Outlier Detection.

```
1: for all e not marked in G do
2:   if there is a generic cycle H containing e is embeddable then
3:     mark any edge ∈ H, including e, a normal edge.
4:   else
5:     mark e an outlier edge.
6:   end if
7: end for
```

cycle $\tilde{H}$ of H containing e. As H is embeddable, $\tilde{H}$ is also embeddable. Therefore, e is also marked a normal edge by Algorithm 2. □

In Algorithm 2, each edge only cares about how to verify itself. Accordingly, in a wireless ad hoc network, the task of a node is to verify its incident edges. Thus, Algorithm 2 is actually a framework for distributed outlier detection. Since distributed methods are more appealing to large scale wireless ad hoc network applications than centralized methods, that is Algorithm 1, in the rest of this article, we focus on the implementation of Algorithm 2.

4. ALGORITHM DESIGN AND IMPLEMENTATION

Two key challenges exist in the implementation of Algorithm 2: how to find those generic cycles containing edge e and check their embeddabilities. Saxe [1979] proved that deciding embeddability of a given weighted graph with exact Euclidean norm edge lengths is strongly NP-hard in k dimensions, for any $k \geq 1$. In particular, even with the Unit Disk Graph (UDG) model, this problem is still weakly NP-hard [Bdoiu et al. 2004]. A basic approach to decide embeddability is to find a realization, namely, localization. Apparently, localization is not easier than deciding embeddability. However, some classes of graphs are computationally efficient to localize quadrilateration [Moore et al. 2004], trilateration [Eren et al. 2004], and bilateration [Goldenberg et al. 2006] graphs. Compared to quadrilateration and trilateration, bilateration has the following advantages.

—No trilateration graphs are generic cycles except K_4, but many of them are bilateration graphs.
—Trilateration requires the network to be dense, while bilateration can be carried out in sparse networks, as suggested in Goldenberg et al. [2006].
—Trilateration is an LS-based statistics method, which would hide outliers [Hampel et al. 2005]; by contrast, bilateration would effectively expose geometric inconsistency.

Based on this analysis, our target now becomes bilateration generic cycles. We define the problem as follows.

Problem 4.1 (*Bilateration Generic Cycles Search*). Given a bilateration ordering l of a graph G, find out those subgraphs H of G such that H is a generic cycle with a bilateration ordering $\tilde{l}$ that is a subsequence of l.

Before presenting our algorithm to find bilateration generic cycles, we first prove their properties.

LEMMA 4.2. *A bilateration graph $G = \langle V, E\rangle$ is a generic cycle if and only if there is a bilateration ordering $l = \langle v_1, v_2, v_3, \ldots, v_n\rangle$ of V, such that for all $1 \leq i \leq n$, the degree of v_i $d(v_i) \geq 3$, and for all $3 \leq i \leq (n-1)$, v_i is adjacent to exactly two distinct vertices v_j, with $j < i$, and v_n is adjacent to exactly three distinct vertices v_j with, $j < n$.*

ALGORITHM 3: Bilateration Generic Cycles Search

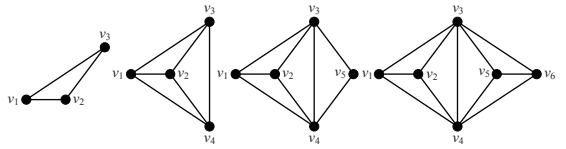
```
BGCS(List order)
1: for all vertex x with more than 2 neighbors earlier in order do
2:    List ancestors = (neighbors of x earlier in order)
3:    for all all triple (u,v,w) in ancestors do
4:       SortedList visited = new SortedList
5:       visited.add(u,v,w)
6:       Graph cycle = new Graph
7:       cycle.add(u,(u,x),v,(v,x),w,(w,x))
8:       Traceback(cycle, visited, order)
9:    end for
10: end for
```

```
Traceback(Graph cycle, SortedList visited, List order)
1: Node x = visited.pop()
2: List ancestors = (neighbors of x earlier in order)
3: if visited.length()==1 and x and visited[0] are neighbors then
4:    cycle.add((x,visited[0]))
5:    output cycle as a generic cycle
6:    ancestors.remove(visited[0])
7: end if
8: for all pair(u,v) in ancestors do
9:    visited.add(u,v)
10:   cycle.add(u,(u,x),v,(v,x))
11:   Traceback(cycle,visited,order)
12: end for
```

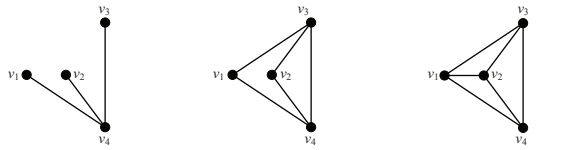
PROOF. *Sufficiency*. The number of edges of G is $2(n-3)+3+1 = 2n-2$. Assume there are some subsets S of V such that $|S| \geq 4$ and $i_G(S) \geq 2|S|-2$. Let X denote one such S with minimal cardinality, and $k = |X|$. Let $X = \langle v_{i_1}, v_{i_2}, \ldots, v_{i_k} \rangle$, where $1 \leq i_1 < i_2 < \cdots < i_k \leq n$. The only interesting case is where $k < n$. Let p denote the maximal i such that $v_i \notin X$. If $p = n$, according to the property of l, the degree of v_{i_k} in the subgraph induced by X in G, $d_{G(X)}(v_{i_k}) \leq 2$. Since $i_G(X) \geq 2|X| - 2$, $i_G(X - \{v_{i_k}\}) \geq 2|X - \{v_{i_k}\}| - 2$, which contradicts the minimal cardinality of X. Thus, $i_k = n$. We rewrite X in another way, $X = \langle v_{i_1}, v_{i_2}, \ldots, v_{i_{k-(n-p)}}, v_{p+1}, v_{p+2}, \ldots, v_n \rangle$. We show that $X' = \langle v_{i_1}, v_{i_2}, \ldots, v_{i_{k-(n-p)}} \rangle$ is satisfying $i_G(X') \geq 2|X'| - 2$, which also contradicts the minimal cardinality of X. Since $d(v_p) \geq 3$ in G, and v_p is adjacent to at almost two distinct vertices v_j with $j < p$ (since $p \neq n$), there must be some $j > p$ and v_j is adjacent to v_p. Thus, through removing $v_{p+1}, v_{p+2}, \ldots, v_n$ from X to get X', we have $i_G(X) - i_G(X') \leq 2(n-p-1)+3-1 = 2(n-p)$. Since $i_G(X) \geq 2|X| - 2$, $i_G(X') \geq i_G(X) - 2(n-p) \geq 2|X| - 2 - 2(n-p) = 2|X'| - 2$.

Necessity. Let G be a generic cycle with a bilateration ordering $l = \langle v_1, v_2, v_3, \ldots, v_n \rangle$. Thus, for all $1 \leq i \leq n$, $d(v_i) \geq 3$, and for all $3 \leq i \leq n$, v_i is adjacent to at least two distinct vertices v_j with $j < i$. Besides, there must be some v_i adjacent to at least three distinct vertices v_j with $j < i$. Let k denote the minimal of such i. Apparently, $k \geq 4$. Let $X = \langle v_1, v_2, v_3, \ldots, v_k \rangle$. Assume the only interesting case is that v_k is adjacent to exactly three distinct vertices v_j, with $j < k$ and for all $1 \leq i \leq k$, $d_{G(X)}(v_i) \geq 3$. By the first part of this proof, the subgraph $G(X)$ induced by X in G is a generic cycle. So, $G(X) = G$ and $X = V$. □

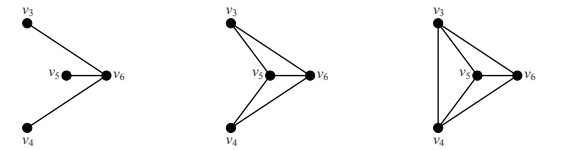
Based on Lemma 4.2, an algorithm for bilateration generic cycles search is given in Algorithm 3. Before proving the correctness of Algorithm 3, we give an example to illustrate it, shown in Figure 8, where Figure 8(a) shows the bilateration ordering $l = \langle v_1, v_2, v_3, v_4, v_5, v_6 \rangle$. Because v_4 has 3 neighbors, v_1, v_2, and v_3, earlier in



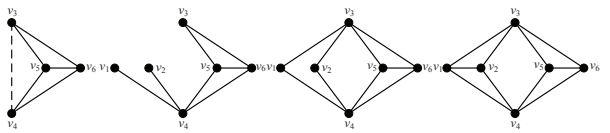
(a) Bilateration ordering



(b) First bilateration generic cycle



(c) Second bilateration generic cycle



(d) Third bilateration generic cycle

Fig. 8. Bilateration generic cycles search.

l, a bilateration generic cycles search procedure begins. At the first beginning, cycle $= (v_1, v_2, v_3, v_4, (v_1, v_4), (v_2, v_4), (v_3, v_4))$, and visited $= (v_3, v_2, v_1)$. Through checking v_3, we add (v_1, v_3) and (v_2, v_3) into the cycle, and update visited as (v_2, v_1). Since v_1 and v_2 are neighbors, after adding (v_1, v_2) into the cycle, we get a generic cycle, illustrated in Figure 8(b). Since there is no other neighbor of v_2 earlier in l, this search terminates. Another bilateration generic cycles search begins at v_6. Initially, cycle $= (v_3, v_4, v_5, v_6, (v_3, v_6), (v_4, v_6), (v_5, v_6))$ and visited $= (v_5, v_4, v_3)$. After tracing back to v_5, (v_3, v_5) and (v_4, v_5) are inserted into the cycle, and visited $= (v_4, v_3)$. Again, since v_3 and v_4 are neighbors, after inserting (v_3, v_4) into the cycle, we get the second generic cycle, shown in Figure 8(c). Besides v_3, v_4 still has the other two neighbors earlier in l, v_1 and v_2. After removing (v_3, v_4) from the cycle, the search procedure continues. And finally, this search procedure ends with the third generic cycle, shown in Figure 8(d). These three generic cycles are all the bilateration generic cycles in G with respect to $l = \langle v_1, v_2, v_3, v_4, v_5, v_6 \rangle$.

THEOREM 4.3. *Given a graph G with a bilateration ordering l, Algorithm 3 identifies all bilateration generic cycles in G with respect to l.*

PROOF. *Correctness*. Let $C = \langle V, E \rangle$ denote a cycle identified by Algorithm 3. Apparently, $m = |V| \geq 4$. Reversing the tracing back order in Algorithm 3, we get an ordering of vertices of C, $l' = \langle v_{i_1}, v_{i_2}, \ldots, v_{i_m} \rangle$, where $1 \leq i_1 < i_2 < \cdots < i_m \leq n$ such that v_{i_1} and v_{i_2} are neighbors and for all $3 \leq k \leq (m-1)$, v_{i_k} has exactly two neighbors earlier in l',

and v_{i_m} has exactly three neighbors earlier in l'. In particular, for all $1 \leq k \leq (m-1)$, there must be some j with $k < j \leq m$ and v_{i_j} and v_{i_k} are neighbors; otherwise, v_{i_k} would never be inserted into C. Thus, for all $3 \leq k \leq m$, $d_C(v_{i_k}) \geq 3$. Similarly, we can prove both $d_C(v_{i_1})$ and $d_C(v_{i_2}) \geq 3$. By Lemma 4.2, C is a bilateration generic cycle.

Optimality. By Lemma 4.2, every bilateration generic cycle C in G with respect to l has a bilateraion ordering $l' = \langle v_{i_1}, v_{i_2}, \ldots, v_{i_m} \rangle$, which is a subsequence of l, such that $m \geq 4$, and for all $1 \leq k \leq m$, $d_C(v_{i_k}) \geq 3$, and for all $3 \leq k \leq (m-1)$, v_{i_k} has exactly two neighbors earlier in l', and v_{i_m} has exactly 3 neighbors earlier in l'. From following Algorithm 3 and the first part of this proof, we can see that Algorithm 3 outputs all such bilateration orderings. □

Algorithm 3 solves the first issue, how to find generic cycles, in Algorithm 2. Through employing Sweeps [Goldenberg et al. 2006], we can check the embeddabilities of generic cycles based on localization, which addresses the second issue.

5. DISCUSSION

In this section, we discuss some practical issues of the proposed algorithm and solutions to address them.

5.1. Distributed Implementation

Since Algorithm 2 is a framework for outlier detection from the point of view of a single edge, it is not difficult to implement it in a distributed fashion. From every single edge, after exploring a bilateration ordering l, Algorithm 3 can identify all bilateration generic cycles with respect to l. In the exploring procedure, l uses the start edge as its ID, and every involved node remembers its order in l. Once a node finds that it has more than 2 neighbors earlier in l, a bilateration generic cycles search procedure is initiated. After identifying a bilateration generic cycle, a distributed solution, Sweeps [Goldenberg et al. 2006], can be implemented to test the embeddability of the generic cycle.

5.2. Practical Error Model

We assume normal edges are without any ranging errors. This abstraction is impractical in real world networks, where ranging errors always exist. Our basic algorithm fails in this situation, since it is almost impossible for those generic cycles identified by Algorithm 3 to be embeddable. We introduce an error threshold δ to handle this problem. Given a bilateration generic cycle $C = \langle V, E, W \rangle$, Sweeps computes all realizations r of C. Define *additive distortion* [Bdoiu et al. 2004] as

$$\epsilon = min_r\{max_{(u,v)}\{|W((u,v)) - \|r(u) - r(v)\||\}\} \tag{2}$$

for every $(u, v) \in E$. If $\epsilon \leq \delta$, then we say C is embeddable; otherwise, it is unembeddable. The value of δ has direct impact on the performance of the proposed algorithm, and should be determined according to the ranging accuracy. The introduction of δ, however, raises another issue: a generic cycle with outlier edges may be claimed embeddable; namely, there are false negative results. However, under such a model (normal edges containing mild errors), normal edges no longer mean that the measuring value is exactly equal to a real value. Instead, an edge with bounded error is treated as a normal edge. It is true that our solution will accept some measurements with mild errors, but they are actually defined to be normal when it is assumed that errors exist in normal edges. Hence, our solution is able to identify outlier measurements under this model, but cannot tell whether a normal edge is accurate or slightly erroneous (no probabilistically true conclusion on this point).

If false negative results are unacceptable in some scenarios, such as security applications, besides choosing a tighter δ, we can require that a normal edge be contained in more than one embeddable generic cycle.

Another issue induced by ranging errors is that the performance will decay rapidly by error accumulation. It is possible that the aggregation of mild errors conceals a gross error. Results from Bdoiu et al. [2004] show that even for a complete graph it is not an easy problem, let alone sparse graphs, that is, a generic cycle. One way to address this issue is to limit the size of a generic cycle. Of course, the most effective way to solve this problem is improving the ranging accuracy.

5.3. Computation Complexity

In the generic cycles search procedure, by Theorem 4.3, Algorithm 3 is an output-sensitive method, whose computation complexity is determined by the number of generic cycles and their sizes, that is, the size of the output. Due to the minimal redundant rigidity of generic cycles, from the perspective of verifying normal edges, it is necessary to check the embeddability of every generic cycle in the worst case. Thus, Algorithm 3 is optimal.

After identifying a generic cycle, Sweeps [Goldenberg et al. 2006] is employed to test the embeddability of that generic cycle. Although in the worst case, the requirement of computational time and space increases exponentially with the sizes of generic cycles, the results from Goldenberg et al. [2006] and our experiments show that Sweeps is effective in practice. What's more, even if we limit the maximal size of a generic cycle, that is, to 16 nodes, most normal edges can still be verified. Of course, it is a trade-off between theoretical guarantees and computational complexity, which should be determined by practical application requirements and resource constraints of wireless devices.

6. EVALUATION

6.1. Normal Edges with Precise Distance Measurements

We first evaluate the performance of the proposed algorithm when normal edges have precise distance measurements. We generate random networks of 400 nodes in a square region $[0, 1]^2$. We do not consider outlier edges, as here we are only interested in how many normal edges can be verified, which is only determined by the topological structure of the network consisting of normal edges. We aggregate the results from 100 network instances at each sensing radius and compute 95th-percentile confidence intervals for the quantity of interest.

Figure 9 plots the performance of the proposed algorithm, which is based on Bilateration Generic Cycles, and a Quadrilateral-based speculative filtering algorithm [Hwang et al. 2007], the state of the art outlier detection algorithm, in which all edges in a quadrilateral would be verified. By Theorem 3.8, through identifying all the redundantly rigid components using the Pebble Game algorithm [Jacobs and Hendrickson 1997], we can figure out all the edges that are theoretically possible to verify. We find that the quadrilateral-based speculative filtering algorithm always verifies fewer edges than the proposed method, because a quadrilateral is the smallest bilateration generic cycle. Specifically, the proposed algorithm verifies at least 80% of theoretically verifiable edges when the network density varies widely. For comparison, the quadrilateral-based algorithm guarantees verification of only 50% of edges. We limit the size of the largest bilateration generic cycle (at most 16 nodes) to ensure the needed computational space and time doesn't exceed the capacity of off-the-shelf sensor nodes, such as *Telos* or *Mica* motes; otherwise, the performance gap would be even larger. The result suggests that in scenarios where networks are sparse or networks are dense but with a large

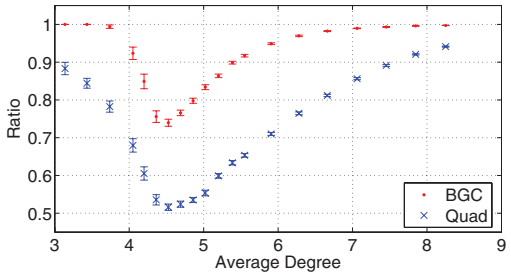


Fig. 9. Ratios of the number of normal edges verified to the number of normal edges that are theoretically possible to verify.

fraction of outlier edges, the proposed algorithm outperforms the quadrilateral-based algorithm.

6.2. Normal Edges with Moderate Ranging Errors

To examine the effectiveness of the practical error model, where normal edges have moderate ranging errors, we implement the proposed outlier detection on the data trace collected from the ongoing ecological surveillance system GreenOrbs [Mo et al. 2009], as illustrated in Figure 13. In this system, commercial off-the-shelf sensor motes are programmed, enclosed, and deployed in the forest. With each node equipped with multiple sensors, such as light, temperature, and humidity, GreenOrbs supports various ecological applications. Localization is one of the most important issues in the project since sensing data without location information is almost meaningless.

Based on the data collected on June 15, 2009, we extract the network topology, shown in Figure 14. Besides light, temperature, and humidity information, each packet also contains the first 10 discovered neighbors of one node in the latest measurement interval. To decrease the uncertainty caused by dynamic environmental factors, for each pair of neighboring nodes (u, v) depicted in Figure 14, we require that there are at least three packets of u containing v as their neighbor, and vice versa. Nevertheless, the resulting topology still greatly differs from the UDG model, because of the complicated forest environment. Based on this network topology, we conduct extensive simulations to investigate how outlier detection can improve localization accuracy. The ranging errors of outlier edges are 10 times that of normal edges. We integrate the results from 100 instances.

The proposed outlier detection algorithm doesn't serve as a localization algorithm, but an intermediate component between measuring internode distances and localizing wireless devices based on such ranging information. We utilize AHLoS [Savvides et al. 2001] as the basic localization method, and compare the localization performance of three strategies towards outliers: sifting, ignoring (the basic AHLoS approach), and eliminating completely (the idealized case). Note that our outlier detection algorithm works well with arbitrary localization methods. We choose AHLoS since it delegates a major class of widely used multilateration (trilateration)-based localization techniques.

Figure 10 reports the result for $\sigma = 5\%$ outlier edges. For all strategies, the average location error increases linearly with increasing ranging error. In the idealized case, labeled as Without Outlier, the location errors are much less than the ignoring case labeled as No Sifting, while employing Algorithm 2 to sift outliers achieves a median location accuracy, which is close to the idealized case and also much better than the basic AHLoS approach. In particular, when the distance ranging error is 0.25 m, the proposed algorithm improves the localization accuracy from 19 m to 14 m, which is close to 12 m, the result of the idealized case.

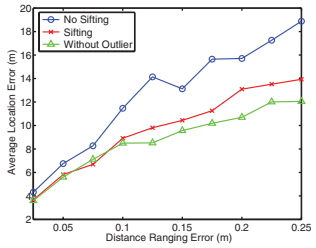


Fig. 10. The impact of ranging error on accuracy.

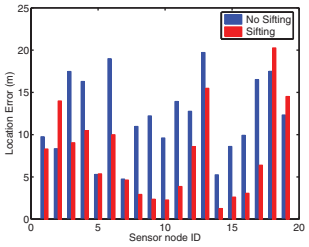


Fig. 11. Localization errors of sensor nodes.

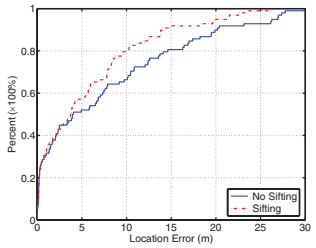


Fig. 12. Empirical cumulative function of location errors.

We also study the typical localization errors for every single node, illustrated in Figure 11. For all 19 nodes, the location accuracy has been improved by sifting outliers. Figure 12 plots the characteristics of error propagation. As shown, nearly 90% of nodes have at most 15 m error and 80% have at most 10 m error by sifting outliers, while for the basic AHLoS algorithm, only 80% of nodes have less than 15 m error and 65% of nodes have less than 10 m error.

7. FIELD EXPERIMENT AND RESULTS

We conduct a localization experiment using *MicaZ* motes in an indoor environment. In the experiment, a network prototype consisting of 25 MicaZ motes is constructed in a 5×5 grid on the floor, with grid spacing of 2 feet. Each MicaZ mote is augmented with a ranging sensor board that detects distance between two nodes by measuring the time of flight of specified acoustic pulses. Due to the reflections from walls and ceilings, which gives rise to null regions, the network connectivity is unexpectedly irregular and unsymmetrical, as illustrated in Figure 15(a).

7.1. Pairwise Ranging Measurements

Each ranging sensor board has a sounder, which can generate 4kHz fixed frequency acoustic signals, and a microphone that detects the tone produced by the sounder. In

Fig. 13. The deployment and enclosure of motes.

Fig. 14. The network topology

our experiment, even after employing robust statistical methods to filter some outliers caused by unexpected events, we find that qualities of pairwise ranging measurements still vary. Most are stable, with variances less than 4 cm, while some are highly uncertain, with variances as large as 25 cm, two of which are shown in Figure 15(a), colored red.

7.2. Localization Experiment Results

We have done six sets of experiments using the 5×5 setting with the connectivity graph as in Figure 15(a). For each dataset, we have performed the following four types of AHLoS [Savvides et al. 2001] localization.

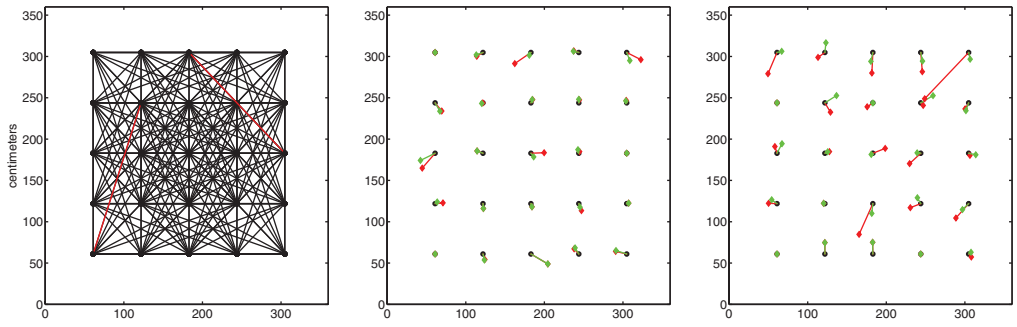
—NSNM: no outlier sifting, no malicious beacon;
—SNM: with outlier sifting, no malicious beacon;
—NSM: no outlier sifting, with a malicious beacon;
—SM: with outlier sifting, with a malicious beacon.

For NSNM and SNM, there are three honest beacons, which broadcast their true locations, and for NSM and SM, we have four beacons, one of which is malicious and fakes its location. As dicussed in Section 2, the proposed method is capable of detecting malicious beacons. Table I presents the average location errors of these scenarios. Regardless of whether there is a malicious beacon, the proposed algorithm largely improves the location accuracy by rejecting outliers during the localization process. We further provide two examples to demonstrate the effectiveness of the proposed outlier detection method. Figure 15(b) shows the location results of NSNM and SNM, and

Table I. Localization Results

Data Sets	1	2	3	4	5	6
NSNM	15.73	12.43	10.64	10.80	9.78	11.42
SNM	12.52	11.95	8.72	9.54	9.74	12.57
NSM	18.95	26.24	20.79	18.29	14.03	14.70
SM	10.94	16.00	11.18	10.20	13.34	12.21

The reported data is in centimeters.



(a) Connectivity graph (b) NSNM vs. SNM (c) NSM vs. SM

Fig. 15. The experimental network prototype and results, where black circles are true locations, and red and green diamonds represent the estimates of NSNM (NSM) and SNM (SM), respectively.

Figure 15(c) gives those of NSM and SM, where the node at the top right corner is a malicious beacon.

8. RELATED WORK

Location information is essential for a wide range of pervasive and mobile applications [Li and Liu 2009; Mo et al. 2009; Li et al. 2011; Wang et al. 2011]. In the literature of wireless ad hoc and sensor networks, many localization methods have been proposed and used in real-world positioning systems. In general, we can classify these methods into two major categories: range-based and range-free. Range-based methods usually assume sensor nodes have the capability to measure distances, based on Radio Signal Strength (RSS) [Seidel and Rappaport 1992] and Time Difference of Arrival (TDOA) [Savvides et al. 2001; Priyantha et al. 2000], and/or relative orientations of neighbor nodes, that is, Angle of Arrival (AOA) [Niculescu and Nath 2003; Chang et al. 2008]. By contrast, range-free methods use only node connectivity and hop-count [He et al. 2003; Lederer et al. 2009; Li and Liu 2010]. To build a full theoretical foundation, some works [Eren et al. 2004; Goldenberg et al. 2005] show that network localizability has a close relationship with graph rigidity theory. Recent work [Yang et al. 2010] employs wheel structure to study node localizability.

Because of the inevitability of noise in practice, how to control error accumulation in localization also attracts a lot of research. Error management [Liu et al. 2006] uses error registries to select nodes that participate in the localization procedure based on their relative contribution to the localization accuracy, and Moore et al. [2004] requires "robust quadrilaterals" to prevent large location errors introduced by flip ambiguities. From the perspective of security, outlier-resistant localization has also been studied. Minimum Mean Square Estimation (MMSE) is used to identify and remove malicious nodes in Liu et al. [2005], and Least Median of Squares (LMS), an estimator with high breakdown point, is adopted in Li et al. [2005]. A speculative filtering algorithm is

designed in Hwang et al. [2007] to detect phantom nodes. Deriving inspiration from robust statistics, the recent work, SISR [Kung et al. 2009], uses a residual shaping influence function to deemphasize the bad nodes and bad links during the localization procedure. Besides requiring dense links, SISR is a centralized algorithm, which may prevent it from applying to large-scale deployments.

Different from those methods that require dense networks, the proposed outlier detection algorithm pays more attention to exploring and utilizing the topological structure, and thus works properly in networks that have moderate connectivity, which to the best of our knowledge, cannot be realized by existing approaches.

9. CONCLUSION

We have shown that noisy and outlier distance measurements greatly degrade the location accuracy of existing localization approaches. Hence, outlier detection serves as an essential and prior component for all range-based localization approaches. Based on graph embeddability and rigidity theory, we build the theoretical foundation for identifying outliers and accordingly design a bilateration generic cycle-based algorithm. The results of a network prototype and extensive simulations show that the proposed algorithm largely improves the location accuracy by modestly and wisely rejecting outliers during the localization process.

REFERENCES

Bdoiu, M., Demaine, E., Hajiaghayi, M., and Indyk, P. 2004. Low-dimensional embedding with extra information. In *Proceedings of the 20th Annual Symposium on Computational Geometry*. 320–329.

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

Berger, B., Kleinberg, J., and Leighon, T. 1996. Reconstructing a three-dimensional model with arbitrary errors. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing*. 449–458.

Chang, H., Tian, J., Lai, T., Chu, H., and Huang, P. 2008. Spinning beacons for precise indoor localization. In *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems.* 127–140.

Eren, T., Goldenberg, O., Whiteley, W., Yang, Y., Morse, A., Anderson, B., and Belhumeur, P. 2004. Rigidity, computation, and randomization in network localization. In *Proceedings of IEEE INFOCOM*. Vol. 4. 2673–2684.

Goldenberg, D., Bihler, P., Cao, M., Fang, J., Anderson, B., Morse, A., and Yang, Y. 2006. Localization in sparse networks using sweeps. In *Proceedings of the 12th Annual International Conference on Mobile Computing and Networking*. 110–121.

Goldenberg, D., Krishnamurthy, A., Maness, W., Yang, Y., Young, A., Morse, A., and Savvides, A. 2005. Network localization in partially localizable networks. In *Proceedings of IEEE INFOCOM*. vol. 1, 313–326.

Hampel, F., Ronchetti, E., Rousseeuw, P., and Stahel, W. 2005. *Robust Statistics: The Approach Based on Influence Functions*. Wiley Series in Probability and Statistics.

He, T., Huang, C., Blum, B. M., Stankovic, J. A., and Abdel Zaher, T. 2003. Range-free localization schemes for large scale sensor networks. In *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking (MobiCom).* 81–95.

Hendrickson, B. 1992. Conditions for unique graph realizations. *Siam J. Comput. 21*, 1, 65–84.

Hwang, J., He, T., and Kim, Y. 2007. Detecting phantom nodes in wireless sensor networks. In *Proceedings of IEEE INFOCOM.* 2391–2395.

Jackson, B. and Jordán, T. 2005. Connected rigidity matroids and unique realizations of graphs. *J. Combin. Theory B 94*, 1, 1–29.

Jacobs, D. and Hendrickson, B. 1997. An algorithm for two-dimensional rigidity percolation: The pebble game. *J. Comput. Phys. 137*, 2, 346–365.

Kung, H., Lin, C., Lin, T., and Vlah, D. 2009. Localization with snap-inducing shaped residuals (SISR): Coping with errors in measurement. In *Proceedings of the 15th Annual International Conference on Mobile Computing and Networking*. 333–344.

Laman, G. 1970. On graphs and rigidity of plane skeletal structures. *J. Eng. Math. 4*, 4, 331–340.

Lederer, S., Wang, Y., and Gao, J. 2009. Connectivity-based localization of large-scale sensor networks with complex shape. *ACM Trans. Sen. Net. 5*, 4, 1–32.

LI, M., CHENG, W., LIU, K., HE, Y., LI, X., AND LIAO, X. 2011. Sweep coverage with mobile sensors. *IEEE Trans. Mobile Comput. 10*, 11, 1534–1545.

LI, M. AND LIU, Y. 2009. Underground coalmine monitoring with wireless sensor networks. *ACM Trans. Sen. Net. 5*, 2.

LI, M. AND LIU, Y. 2010. Rendered path: Range-free localization in anisotropic sensor networks with holes. *ACM Trans. Network. 18*, 1, 320–332.

LI, Z., TRAPPE, W., ZHANG, Y., AND NATH, B. 2005. Robust statistical methods for securing wireless localization in sensor networks. In *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks*. 91–98.

LIU, D., NING, P., AND DU, W. 2005. Attack-resistant location estimation in sensor networks. In *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks*.

LIU, J., ZHANG, Y., AND ZHAO, F. 2006. Robust distributed node localization with error management. In *Proceedings of the 7th ACM International Symposium on Mobile Ad Hoc Networking and Computing*. 250–261.

MO, L., HE, Y., LIU, Y., ZHAO, J., TANG, S., LI, X., AND DAI, G. 2009. Canopy closure estimates with greenorbs: Sustainable sensing in the forest. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*. 99–112.

MOORE, D., LEONARD, J., RUS, D., AND TELLER, S. 2004. Robust distributed network localization with noisy range measurements. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*. 50–61.

NICULESCU, D. AND NATH, B. 2003. Ad hoc positioning system (APS) using AOA. In *Proceedings of IEEE INFOCOM*. 1734–1743.

PRIYANTHA, N., CHAKRABORTY, A., AND BALAKRISHNAN, H. 2000. The cricket location-support system. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*. 32–43.

SAVVIDES, A., HAN, C., AND STRIVASTAVA, M. 2001. Dynamic fine-grained localization in ad hoc networks of sensors. In *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*. 166–179.

SAXE, J. 1979. Embeddability of weighted graphs in k-space is strongly NP-hard. Carnegie-Mellon University, Department of Computer Science.

SEIDEL, S. AND RAPPAPORT, T. 1992. 914 MHz path loss prediction models for indoor wireless communications in multifloored buildings. *IEEE Trans. Antennas Propag. 40*, 2, 207–217.

WANG, X., FU, L., TIAN, X., BEI, Y., PENG, Q., GAN, X., YU, H., AND LIU, J. 2011. Converge-cast: On the capacity and delay tradeoffs. *IEEE Trans. Mobile Comput. 99*, 1.

YANG, Z. AND LIU, Y. 2010. Quality of trilateration: Confidence-based iterative localization. *IEEE Trans. Parallel Distrib. Syst. 21,* 5, 631–640.

YANG, Z., LIU, Y., AND LI, X. 2010. Beyond trilateration: On the localizability of wireless ad hoc networks. *IEEE/ACM Trans. Network. 18*, 6, 1806–1814.

Received September 2011; revised January 2012; accepted March 2012