Attack-Resistant Location Estimation in Sensor Networks

Donggang Liu and Peng Ning
North Carolina State University
{dliu, pning}@ncsu.edu

Wenliang Kevin Du
Syracuse University
wedu@ecs.syr.edu

Abstract— **Many sensor network applications require sensors' locations to function correctly. Despite the recent advances, location discovery for sensor networks in *hostile environments* has been mostly overlooked. Most of the existing localization protocols for sensor networks are vulnerable in hostile environments. The security of location discovery can certainly be enhanced by authentication. However, the possible node compromises and the fact that location determination uses certain physical features (e.g., received signal strength) of radio signals make authentication not as effective as in traditional security applications. This paper presents two methods to tolerate malicious attacks against beacon-based location discovery in sensor networks. The first method filters out malicious beacon signals on the basis of the "consistency" among multiple beacon signals, while the second method tolerates malicious beacon signals by adopting an iteratively refined voting scheme. Both methods can survive malicious attacks even if the attacks bypass authentication, provided that the benign beacon signals constitute the majority of the "consistent" beacon signals. This paper also presents the implementation of these techniques on MICA2 motes running TinyOS, and the evaluation through both simulation and field experiments. The experimental results demonstrate that the proposed methods are promising for the current generation of sensor networks.**

I. Introduction

Recent technological advances have made it possible to develop distributed sensor networks consisting of a large number of low-cost, low-power, and multi-functional sensor nodes that communicate in short distances through wireless links [1]. Such sensor networks are ideal candidates for a wide range of applications such as health monitoring, data acquisition in hazardous environments, and military operations. The desirable features of distributed sensor networks have attracted many researchers to develop protocols and algorithms that can fulfill the requirements of these applications.

Sensors' locations play a critical role in many sensor network applications. Not only do applications such as environment monitoring and target tracking require sensors' location information to fulfill their tasks, but several fundamental techniques developed for wireless sensor networks also require sensor nodes' locations. For example, in geographical routing protocols (e.g., GPSR [15] and GEAR [28]), sensor nodes make routing decisions at least partially based on their own and their neighbors' locations. Indeed, many sensor network applications will not work without sensors' location information.

A number of location discovery protocols (e.g., [2], [5], [9], [18]–[21], [26], [27]) have been proposed for wireless sensor networks in recent years. These protocols share a common feature: They all use some special nodes, called *beacon nodes*, which are assumed to know their own locations (e.g., through GPS receivers or manual configuration). These protocols work in two stages. In the first stage, non-beacon nodes receive radio signals called *beacon signals* from the beacon nodes. The packet carried by a beacon signal, which we call a *beacon packet*, usually includes the location of the beacon node. The non-beacon nodes then estimate certain measurements (e.g., distance between the beacon and the non-beacon nodes) based on features of the beacon signals (e.g., received signal strength indicator (RSSI), time difference of arrival (TDoA)). We refer to such a measurement and the location of the corresponding beacon node collectively as a *location reference*. In the second stage, a sensor node determines its own location when it has enough number of location references from different beacon nodes. A typical approach is to consider the location references as constraints that a sensor node's location must satisfy, and estimate it by finding a mathematical solution that satisfies these constraints with minimum estimation error. Existing approaches either employ *range-based* methods [5], [19], [20], [26], [27], which use the exact measurements obtained in stage one, or *range-free* ones [2], [9], [16], [18], [21], which only need the existences of beacon signals in stage one.

Despite the recent advances, location discovery for wireless sensor networks in *hostile environments*, where there may be malicious attacks, has been mostly overlooked. Most of existing location discovery protocols become become vulnerable in the presence of malicious attacks. For example, an attacker may provide incorrect location references by replaying the beacon packets intercepted in different locations. Moreover, an attacker may compromise a beacon node and distribute malicious location references by lying about the beacon node's location or manipulating the beacon signals (e.g., changing the signal strength if RSSI is used to estimate the distance). In either of these cases, non-beacon nodes will determine their locations incorrectly.

The location verification technique proposed in [25] can verify the relative distance between a verifying node and a beacon node, and the technique proposed in [16] can protect location discovery using sectored antennae at beacon nodes. However, neither of them can ensure correct location discovery when beacon nodes are compromised. A robust location detection is developed in [24]. However, it cannot be directly applied in resource constrained sensor networks due to its high computation and storage overheads.

Without protection, an attacker may easily mislead the location estimation at sensor nodes and subvert the normal operation of sensor networks. The security of location discovery can certainly be enhanced by authentication. Specifically, each beacon packet should be authenticated with a cryptographic key only known to the sender and the intended receivers, and a non-beacon node accepts a beacon signal only when the beacon packet carried by the beacon signal can be authenticated. However, authentication does not guarantee the security of location discovery, either. An attacker may forge beacon packets with keys learned through compromised nodes, or replay beacon signals intercepted in different locations. Indeed, our experiment in Section II-B shows that an attacker can introduce substantial location estimation errors by forging or replaying beacon packets.

In this paper, we develop two attack-resistant location estimation techniques to tolerate the malicious attacks against range-based location discovery in wireless sensor networks. Our first technique, named *attack-resistant Minimum Mean Square Estimation*, is based on the observation that malicious location references introduced by attacks are intended to mislead a sensor node about its location, and thus are usually inconsistent with the benign ones. To exploit this observation, our method identifies malicious location references by examining the inconsistency among location references (indicated by the mean square error of estimation), and defeats malicious attacks by removing such malicious data. Our second technique, a *voting-based location estimation* method, quantizes the deployment field into a grid of cells and has each location reference "vote" on the cells in which the node may reside. Moreover, we develop a method that allows iterative refinement of the "voting" results so that it can be executed in resource constrained sensor nodes.

We have implemented the proposed schemes on MICA2 motes [4] running TinyOS [10], and evaluated the performance through simulation and field experiments. It shows that the proposed schemes are promising for the current generation of sensor networks.

The rest of the paper is organized as follows. Section II presents the proposed approaches for attack-resistant location estimation as well as the simulation evaluation and the field test of the proposed techniques. Section III discusses related work. Section IV concludes this paper and points out some future research directions.

II. ATTACK-RESISTANT LOCATION ESTIMATION

In this section, we present two approaches to dealing with malicious attacks against location discovery in wireless sensor networks. The first approach is extended from the minimum mean square estimation (MMSE). It uses the mean square error as an indicator to identify and remove malicious location references. The second one adopts an iteratively refined voting scheme to tolerate malicious location references introduced by attackers.

Our techniques are purely based on a set of location references. The location references may come from beacon nodes that are either single hop or multiple hops away, or from those non-beacon nodes that already estimated their locations. We do not distinguish these location references, though the effect of "error propagation" may affect the performance of our techniques due to the estimation errors at non-beacon nodes. We consider such investigations as possible future work. Since our techniques only utilize the location references from beacon nodes. There is no extra communication overhead involved when compared to the previous localization schemes.

A. Assumptions and Threat Model

We assume all beacon nodes are uniquely identified. In other words, a non-beacon node can identify the original sender of each beacon packet based on the cryptographic key used to authenticate the packet. This can be easily achieved with a pairwise key establishment scheme [3], [6], [7] or a broadcast authentication scheme [22].

We assume each non-beacon node uses at most one location reference derived from the beacon signals sent by each beacon node. As a result, even if a beacon node is compromised, the attacker that has access to the compromised key can only introduce at most one malicious location reference to a given non-beacon node by impersonating the compromised node.

For simplicity, we assume the distances measured from beacon signals (e.g., with RSSI or TDoA [26]) are used for location estimation. (Our techniques can certainly be modified to accommodate other measurements such as angles.) For the sake of presentation, we denote a location reference obtained from a beacon signal as a triple $\langle x, y, \delta \rangle$, where (x, y) is the location of the beacon declared in the beacon packet, and δ is the distance measured from its beacon signal.

We assume an attacker may change any field in a location reference. In other words, it may declare a wrong location in its beacon packets, or carefully manipulate the beacon signals to affect the distance measurement by, for example, adjusting the signal

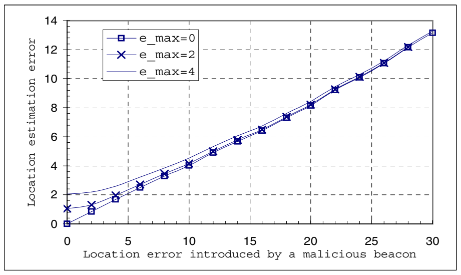


Fig. 1. Location estimation error. Unit of measurement for x and y axes: meter

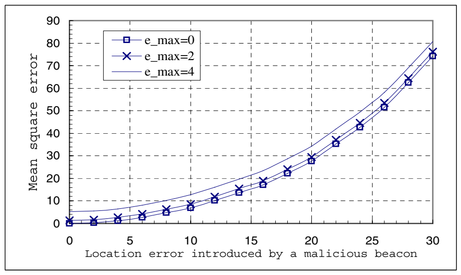


Fig. 2. Mean square error ς^2. Unit of measurement for x-axis: meter

strength when RSSI is used for distance measurement. We also assume multiple malicious beacon nodes may collude together to make the malicious location references appear to be "consistent". Our techniques can still defeat such colluding attacks as long as the majority of location references are benign.

B. Attack-Resistant Minimum Mean Square Estimation

Intuitively, a location reference introduced by a malicious attack is aimed at misleading a sensor node about its location. Thus, it is usually "different" from benign location references. When there are redundant location references, there must be some "inconsistency" between the malicious location references and the benign ones. (An attacker may still have a location reference consistent with the benign ones after changing both the location and the distance values. However, such a location reference will not generate significantly negative impact on location determination.) To take advantage of this observation, we propose to use the "inconsistency" among the location references to identify and remove the malicious ones.

We assume a sensor node uses a MMSE-based method (e.g., [5], [19]–[21], [26], [27]) to estimate its own location. Thus, most current range-based localization methods can be used with this technique. To harness this observation, we first estimate the sensor's location with the MMSE-based method, and then assess if the estimated location could be derived from a set of consistent location references. If yes, we accept the estimation result; otherwise, we identify and remove the most "inconsistent" location reference, and repeat the above process. This process may continue until we find a set of consistent location references or it is not possible to find such a set.

We use the mean square error ς^2 of the distance measurements based on the estimated location as an indicator of the degree of inconsistency, since all the MMSE-based methods estimate a sensor node's location by (approximately) minimizing this mean square error. Other indicators are possible but need further investigation.

Definition 1: Given a set of location references $\mathcal{L} = \{\langle x_1, y_1, \delta_1 \rangle, \langle x_2, y_2, \delta_2 \rangle, ..., \langle x_m, y_m, \delta_m \rangle\}$ and a location $(\tilde{x}_0, \tilde{y}_0)$ estimated based on $\mathcal{L}$, the *mean square error of this location estimation* is $\varsigma^2 = \sum_{i=1}^{m} \frac{(\delta_i - \sqrt{(\tilde{x}_0 - x_i)^2 + (\tilde{y}_0 - y_i)^2})^2}{m}$.

Intuitively, the more inconsistent a set of location references is, the greater the corresponding mean square error should be. To gain further understanding, we performed an experiment through simulation with the MMSE-based method in [26]. We assume the distance measurement error is uniformly distributed between $-e_{max}$ and e_{max}. We used 9 honest beacon nodes and 1 malicious beacon node evenly deployed in a $30m \times 30m$ field. The node that estimates location is positioned at the center of the field. The malicious beacon node always declares a false location that is x meters away from its real location, where x is a parameter in our experiment.

Figures 1 and 2 show the location estimation error (i.e., the distance between a sensor's real location and the estimated location) and the mean square error ς^2 when x increases. As these figures show, if a malicious beacon node increases the location estimation error by introducing greater errors, it also increases the mean square error ς^2 at the same time. This further demonstrates that the mean square error ς^2 is potentially a good indicator of inconsistent location references.

In this paper, we choose a simple, threshold-based method to determine if a set of location references is consistent. Specifically, a set of location references $\mathcal{L} = \{\langle x_1, y_1, \delta_1\rangle, \langle x_2, y_2, \delta_2\rangle, ..., \langle x_m, y_m, \delta_m\rangle\}$ obtained at a sensor node is *τ-consistent w.r.t. a MMSE-based method* if the method gives an estimated location $(\tilde{x}_0, \tilde{y}_0)$ such that the mean square error of this location estimation $\varsigma^2 = \sum_{i=1}^{m} \frac{(\delta_i - \sqrt{(\tilde{x}_0 - x_i)^2 + (\tilde{y}_0 - y_i)^2})^2}{m} \leq \tau^2$.

The threshold τ is clearly a critical parameter. We will discuss how to determine τ in Section II-B.1. We now describe the attack-resistant MMSE method, assuming τ is already set properly.

Since the MMSE-based methods can deal with measurement errors better if there are more benign location references, we should keep as many benign location references as possible when the malicious ones are removed. This implies we should get the largest set of consistent location references.

Given a set $\mathcal{L}$ of n location references and a threshold τ, a naive approach to computing the largest set of τ-consistent location references is to check all subsets of $\mathcal{L}$ with i location references about τ-consistency, where i starts from n and drops until a subset of $\mathcal{L}$ is found to be τ-consistent or it is not possible to find such a set. Thus, if the largest set of consistent location references consists of m elements, a sensor node has to use the MMSE method at least $1 + \binom{n}{m+1} + \cdots + \binom{n}{n}$ times to find out the right one. If $n = 10$ and $m = 5$, a node needs to perform the MMSE method for at least 387 times. It is certainly desirable to reduce the computation for resource constrained sensor nodes.

To reduce the computation on sensor nodes, we adopt a greedy algorithm, which is simple but suboptimal. This greedy algorithm works in rounds. It starts with the set of all location references in the first round. In each round, it first verifies if the current set of location references is τ-consistent. If yes, the algorithm outputs the estimated location and stops. Optionally, it may also output the set of location references. Otherwise, it considers all subsets of location references with one fewer location reference, and chooses the subset with the least mean square error as the input to the next round. This algorithm continues until it finds a set of τ-consistent location references or when it is not possible to find such a set (i.e., there are only 3 remaining location references).

The greedy algorithm significantly reduces the computational overhead in sensor nodes. To continue the earlier example, a sensor node only needs to perform MMSE operations for about 50 times (instead of 387 times) using this algorithm. In general, a sensor node needs to use a MMSE-based method for at most $1 + n + (n-1) + \cdots + 4 = 1 + \frac{(n-3)(n+4)}{2}$ times.

1) Determining Threshold τ: The determination of threshold τ depends on the measurement error model, which is assumed to be available for us to perform simulation off-line and determine an appropriate τ. The threshold is stored on each sensor node. Usually, the movement of sensor nodes (beacon or non-beacon nodes) does not have significant impact on this threshold, since the measurement error model will not change significantly in most cases. However, when the error model changes frequently and significantly, the performance of our techniques may be affected. In this paper, we assume the measurement error model will not change.

Note that the malicious beacon signals usually increase the variance of estimation. Thus, having a lower bound (e.g., Cramer-Rao bound) is not enough for us to filter malicious beacon signals. In fact, the upper bound or the distribution of the mean square error are more desirable. In this paper, we study the distribution of the mean square error ς^2 when there are no malicious attacks, and use this information to help determine the threshold τ.

Since there is no other error besides the measurement error, a benign location reference $\langle x, y, \delta\rangle$ obtained by a sensor node at (x_0, y_0) must satisfy: $|\delta - \sqrt{(x-x_0)^2 + (y-y_0)^2}| \leq \epsilon$, where ϵ is the maximum measurement error.

All the localization techniques are aimed at estimating a location as close to the sensor's real location as possible. Thus, we may assume the estimated location is very close to the real location when there are no attacks. Next, we derive the distribution of the mean square error ς^2 using the real location as the estimated location, and compare it with the distribution obtained through simulation when there are location estimation errors.

The measurement error of a benign location reference $\langle x_i, y_i, \delta_i\rangle$ can be computed as $e_i = \delta_i - \sqrt{(x_0 - x_i)^2 + (y_0 - y_i)^2}$, where (x_0, y_0) is the real location of the sensor node. Assuming the measurement errors introduced by different benign location references are independent, we can get the distribution of the mean square error through the following Lemma.

Lemma 1: Let $\{e_1, ..., e_m\}$ be a set of independent random variables, and μ_i, σ_i^2 be the mean and the variance of e_i^2, respectively. If the estimated location of a sensor node is its real location, the probability distribution of ς^2 is $\lim_{m\to\infty} \mathrm{F}[\varsigma^2 \leq \varsigma_0^2] = \Phi(\frac{m\varsigma_0^2 - \mu'}{\sigma'})$, where $\mu' = \sum_{i=1}^{m} \mu_i$, $\sigma' = \sqrt{\sum_{i=0}^{m} \sigma_i^2}$, and $\Phi(x)$ is the probability of a standard normal random variable being less than x.

Proof: Obviously, the mean square error can be computed by $\varsigma^2 = \sum_{i=1}^{m} \frac{e_i^2}{m}$. Thus, the cumulative distribution function can be calculated by $F(\varsigma^2 \leq \varsigma_0^2) = F(\sum_{i=1}^{m} e_i^2 \leq m\varsigma_0^2)$. Since $\{e_1^2, e_2^2, \cdots, e_m^2\}$ are independent, according to the central limit theorem, we have $\lim_{m\to\infty} P(\frac{S_m - \mu'}{\sigma'} \leq x) = \Phi(x)$, where $S_m = \sum_{i=0}^{m}(e_i^2)$. Thus, we have $\lim_{m\to\infty} F(\varsigma^2 \leq \varsigma_0^2) = \lim_{m\to\infty} F(S_m \leq m\varsigma_0^2) = \lim_{m\to\infty} P(\frac{S_m - \mu'}{\sigma'} \leq \frac{m\varsigma_0^2 - \mu'}{\sigma'}) = \Phi(\frac{m\varsigma_0^2 - \mu'}{\sigma'})$. ■

Lemma 1 describes the probability distribution of ς^2 based on a sensor's real location. Though it is different from the probability distribution of ς^2 based on a sensor's estimated location, it can be used to approximate such distribution in most cases.

Let us further assume a simple model for measurement errors, where the measurement error is evenly distributed between $-\epsilon$ and ϵ. Then the mean and the variance for e_i are 0 and $\frac{\epsilon^2}{3}$, respectively, and the mean and the variance for any e_i^2 are $\frac{\epsilon^2}{3}$ and $\frac{4\epsilon^4}{45}$, respectively. Let $c = \frac{\varsigma_0}{\epsilon}$, we have $F(\varsigma^2 \leq (c \times \epsilon)^2) = \Phi(\frac{\sqrt{5m}(3c^2 - 1)}{2})$.

Figure 3 shows the probability distribution of ς^2 derived from Lemma 1 and the simulated results using sensors' estimated locations. We can see that when the number of location references m is large (e.g., $m = 9$) the theoretical result derived from Lemma 1 is very

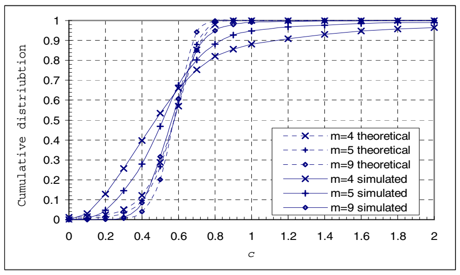


Fig. 3. Cumulative distribution function $F(\varsigma^2 \leq \varsigma_0^2)$. Let $c = \frac{\varsigma_0}{\epsilon}$.

close to the simulation results. However, when m is small (e.g., $m = 4$), there are observable differences between the theoretical results and the simulation. The reasons are twofold. First, our theoretical analysis is based on the central limit theorem, which is only an approximation of the distribution when m is a large number. Second, we used the MMSE-based method proposed in [26] in the simulation, which estimates a node's location by only *approximately* minimizing the mean square error. (Otherwise, the value of ς^2 for benign location references should never exceed ϵ^2.)

Figure 3 gives three hints about the choice of the threshold τ. First, when there are enough number of benign location references, a threshold less than the maximum measurement error is enough. For example, when $m = 9$, $\tau = 0.8\epsilon$ can guarantee the nine benign location references are considered consistent with high probability. Besides, a large threshold may lead to the failure to filter out malicious location references. Second, when m is small (e.g. 4), the cumulative probability becomes flatter and flatter when $c > 0.8$. This means that setting a large threshold τ for small m may not help much to guarantee the consistency test for benign location references; instead, it may give an attacker high chance to survive the detection. Third, the threshold cannot be too small; otherwise, a set of benign location references has high probability to be determined as a non-consistent reference set.

Based on the above observations, we propose to choose the value for τ with a hybrid method. Specifically, when the number of location references is large (e.g., more than 8), we determine the value of τ based on Lemma 1. Specifically, we choose a value of τ corresponding to a high cumulative probability (e.g., 0.9). When the number location references is small, we perform simulation to derive the actual distribution of the mean square error, and then determine the value of τ accordingly. Since there are only a small number of simulations to run, we believe this approach is practical.

C. Voting-Based Location Estimation

In this approach, we have each location reference "vote" on the locations at which the node of concern may reside. To facilitate the voting process, we quantize the target field into a grid of cells, and have each sensor node determine how likely it is in each cell based on each location reference. We then select the cell(s) with the highest vote and use the "center" of the cell(s) as the estimated location. To deal with the resource constraints on sensor nodes, we further develop an iterative refinement scheme to reduce the storage overhead, improve the accuracy of estimation, and make the voting scheme efficient on resource constrained sensor nodes.

1) The Basic Scheme: After collecting a set of location references, a sensor node should determine the target field. The node does so by first identifying the minimum rectangle that covers all the locations declared in the location references, and then extending this rectangle by R_b, where R_b is the maximum transmission range of a beacon signal. This extended rectangle forms the target field, which contains

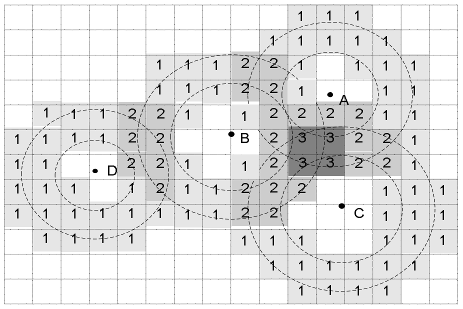


Fig. 4. The voting-based location estimation

all possible locations for the sensor node. The sensor node then divides this rectangle into M small squares (cells) with the same side length L, as illustrated in Figure 4. (The node may further extend the target field to have square cells.) The node then keeps a voting state variable for each cell, initially set to 0.

Consider a benign location reference $\langle x, y, \delta \rangle$. The node that has this location reference must be in a ring centered at (x, y), with the inner radius $\max\{\delta - \epsilon, 0\}$ and the outer radius $\delta + \epsilon$. For the sake of presentation, we refer to such a ring a *candidate ring (centered) at location* (x, y). For example, in Figure 4, the ring centered at point A is a candidate ring at A, which is derived from the location reference with the declared location at A.

For each location reference $\langle x, y, \delta \rangle$, the sensor node identifies the cells that overlap with the corresponding candidate ring, and increments the voting variables for these cells by 1. After the node processes all the location references, it chooses the cell(s) with the highest vote, and uses its (their) geometric centroid as the estimated location of the sensor node.

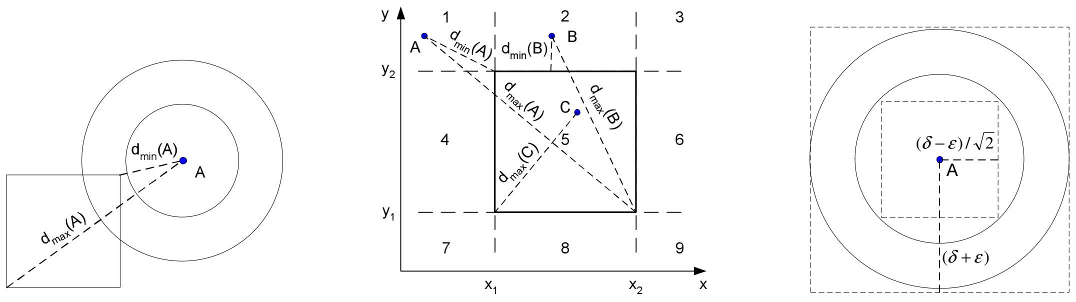
2) Overlap of Candidate Rings and Cells: A critical problem in the voting-based approach is to determine if a candidate ring overlaps with a cell. We discuss how to determine this efficiently below.

Suppose we need to check if the candidate ring at A overlaps with the cell shown in Figure 5(a). Let $d_{min}(A)$ and $d_{max}(A)$ denote the minimum and maximum distances from a point in the cell to point A, respectively. We can see that the candidate ring does not overlap with the cell only when $d_{min}(A) > r_o$ or $d_{max}(A) < r_i$, where $r_i = \max\{0, \delta - \epsilon\}$ and $r_o = \delta + \epsilon$ are the inner and the outer radius of the candidate ring, respectively.

To compute d_{min} and d_{max}, we divide the target field into 9 regions based on the cell, as shown in Figure 5(b). It is easy to see that given the center of any candidate ring, we can determine the region in which it falls with at most 6 comparisons between the coordinates of the center and those of the corners of the cell. When the center of a candidate ring is in region 1 (e.g., point A in Figure 5(b)), it can be shown that the closest point in the cell to A is the upper left corner, and the farthest point in the cell from A is the lower right corner. Thus, $d_{min}(A)$ and $d_{max}(A)$ can be calculated accordingly. These two distances can be computed similarly when the center of a candidate ring falls into regions 3, 7, and 9.

Consider point B in region 2. Assume the coordinate of point B is (x_B, y_B). We can see that $d_{min}(B) = y_B - y_2$. Computing $d_{max}(B)$ is a little more complex. We first need to check if $x_B - x_1 > x_2 - x_B$. If yes, the farthest point in the cell from B must be the lower left corner of the cell. Otherwise, the farthest point in the cell from B should be the lower right corner of the cell. Thus, we have $d_{max}(B) = \sqrt{(\max\{x_B - x_1, x_2 - x_B\})^2 + (y_B - y_1)^2}$. These two distances can be computed similarly when the center of a candidate ring falls into regions 4, 6, and 8.

Consider a point C in region 5. Obviously, $d_{min}(C) = 0$ since



(a) Overlap of a ring and a cell (b) Computing d_{min} and d_{max} (c) Limiting the examinations of cells

Fig. 5. Determine whether a ring overlaps with a cell

point C itself is in the cell. Assume the coordinate of point C is (x_c, y_c). The farthest point in the cell from C must be one of its corners. Similarly to the above case for point B, we may check which point is farther away from C by checking $x_c - x_1 > x_2 - x_c$ and $y_c - y_1 > y_2 - y_c$. As a result, we get $d_{max}(C) = \sqrt{(\max\{x_c - x_1, x_2 - x_c\})^2 + (\max\{t_c - y_1, y_2 - y_c\})^2}$.

Based on the above discussion, we can determine if a cell and a candidate ring overlap with at most 10 comparisons and a few arithmetic operations. To prove the correctness of the above approach only involves elementary geometry, and thus is omitted.

For a given candidate ring, a sensor node does not have to check all the cells for which it maintains voting states. As shown in Figure 5(c), with simple computation, the node can get the outer bounding box centered at A with side length $2(\delta + \epsilon)$. The node only needs to consider the cells that intersect with or fall inside this box. Moreover, the node can get the inside bounding box with simple computation, which is centered at A with side length $\sqrt{2}(\delta - \epsilon)$, and all the cells that fall into this box need not be checked.

3) Iterative Refinement: The number of cells M (or equivalently, the quantization step L) is a critical parameter for the voting-based algorithm. It has several implications to the performance of our approach. First, the larger M is, the more state variables a sensor node has to keep, and thus the more storage is required. Second, the value of M (or L) determines the precision of location estimation. The larger M is, the smaller each cell will be. As a result, a sensor node can determine its location more precisely based on the overlap of the cells and the candidate rings.

However, due to the resource constraints on sensor nodes, the granularity of the partition is usually limited by the memory available for the voting state variables on the nodes. This puts a hard limit on the accuracy of location estimation. To address this problem, we propose an *iterative refinement* of the above basic algorithm to achieve fine accuracy with reduced storage overhead.

In this version, the number of cells M is chosen according to the memory constraint in a sensor node. After the first round of the algorithm, the node may find one or more cells having the largest vote. To improve the accuracy of location estimation, the sensor node then identifies the smallest rectangle that contains all the cells having the largest vote, and performs the voting process again. For example, in Figure 4, the same algorithm will be performed in a rectangle which exactly includes the 4 cells having 3 votes. Note that in a later iteration of the basic voting-based algorithm, a location reference does not have to be used if it does not contribute to any of the cells with the highest vote in the current iteration.

Due to a smaller rectangle to quantize in a later iteration, the size of cells can be reduced, resulting in a higher precision. Moreover, a malicious location reference will most likely be discarded, since its candidate ring usually does not overlap with those derived from benign location references. For example, in Figure 4, the candidate ring centered at point D will not be used in the second iteration.

The iterative refinement process should terminate when a desired precision is reached or the estimation cannot be refined. The former condition can be tested by checking if the side length L of each cell is less than a predefined threshold S, while the latter condition can be determined by checking whether L remains the same in two consecutive iterations. The algorithm then stops and outputs the estimated location obtained in the last iteration. It is easy to see that the algorithm will fall into either of these two cases, and thus will alway terminate. In practice, we may set the desired precision to 0 in order to get the best precision.

D. Security Analysis

Both proposed techniques remove the effect of the malicious location references from the final location estimation when there are more benign location references than the malicious ones. To defeat the attack-resistant MMSE approach, the attacker has to distribute to a victim node more malicious location references than the benign ones, and control the declared locations and the physical features (e.g., signal strength) of beacon signals so that the malicious location references are considered consistent. To defeat the voting-based approach, the attacker needs similar efforts so that the cell containing the attacker's choice gets more votes than those containing the sensor's real location.

An attacker has two ways to satisfy the above conditions (in order to defeat our techniques). First, the attacker may compromise beacon nodes and then generate malicious beacon signals. Since all beacon packets are authenticated, and a sensor node uses at most one location reference derived from the beacon signals sent by each beacon node, the attacker needs to compromise more beacon nodes than the benign beacon nodes from which a target sensor node may receive beacon signals, besides carefully crafting the forged beacon signals.

Second, the attacker may launch wormhole attacks [13] (or replay attacks) to tunnel benign beacon signals from one area to another. In this case, the attacker does not have to compromise any beacon node, though he/she has to coordinate the wormhole attacks. To deal with such threats, we may use wormhole detection methods such as packet leashes [13] or directional antennae [12]. As a result, it is difficult for an attacker to launch such attacks without being detected.

Our techniques certainly have a limit. In an extreme case, if all the beacon nodes are compromised, our techniques will fail. However, the proposed techniques offer a graceful performance degradation as more malicious location references are introduced. In contrast, an attacker may introduce arbitrary location error with a single malicious location reference in the previous schemes. To further improve the security of location discovery, other complementary mechanisms (e.g., detection of malicious beacon nodes) should be used.

E. Simulation Evaluation

This subsection presents the simulation results for both proposed schemes. The evaluation focuses on the improvement on the accuracy of location estimation in hostile environments.

Three attack scenarios are considered. The first scenario considers a single malicious location reference that declares a wrong location e meters away from the beacon node's real location. (An attacker may also modify the distance component δ in a location reference, which will generate a similar impact.) In the second scenario, there are multiple non-colluding malicious location references, and each of them independently declares a wrong location that is e meters away from the beacon node's real location. In the third scenario, multiple colluding malicious location references are considered. In this case, the malicious location references declare false locations by coordinating with each other to create a virtual location e meters away from the sensor's real location. Thus, the malicious location references may appear to be consistent to a victim node.

In all simulations, a set of benign beacon nodes and a few malicious beacon nodes are evenly deployed in a $30m \times 30m$ target field. The non-beacon sensor node is located at the center of this target field. We assume the maximum transmission range of beacon signals is $R_b = 22m$, so that the non-beacon node can receive the beacon signal from every beacon node located in the target field. We assume the entire deployment field is much larger than this target field so that an attacker can create a very large location estimation error inside the deployment field. Each malicious beacon node declares a false location according to the three attack scenarios discussed above. We assume a simple distance measurement error model. That is, the distance measurement error is uniformly distributed between $-\epsilon$ and ϵ, where the maximum distance measurement error ϵ is set to $\epsilon = 4$m.

Due to the space limit, we only show some evaluation results below. More results are included in our extented version [17].

Evaluation of Attack-Resistant MMSE: In the simulation, we use the MMSE-based method proposed in [26], which we call the *basic MMSE method*, to perform the basic location estimation. Our attack-resistant MMSE method is then implemented on the basis of this method, as discussed in Section II-B. We set $\tau = 0.8\epsilon$ according to Figure 3, which guarantees 9 benign location references are considered consistent with probability of 0.999.

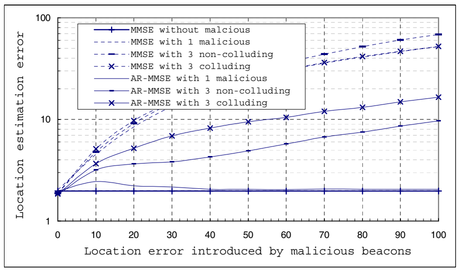


Fig. 6. Performance of attack-resistant MMSE. $\tau = 0.8\epsilon$. Unit of measurement for x and y axes: meter

Figure 6 shows the performance of the attack resistant MMSE method and the basic MMSE-based method when there are malicious location references. It indicates that the attack-resistant MMSE reduces the location estimation error significantly compared with the basic MMSE-based method. It is worth noting that the performance becomes worse when there are multiple malicious location references. This is because multiple malicious location references, especially when they collude together, make the filtering of malicious location references more difficult. It is possible that a few benign location references are removed.

Evaluation of Voting-Based Scheme: In the simulation, we set $M = 100$, which implies 100 Bytes memory for the voting variables, and $S = 0$ to get the minimum location estimation error achievable by this method. Figure 7 compares the accuracy of the basic MMSE method and our voting-based scheme under different types of attacks. We can clearly see that the accuracy of location estimation is improved significantly in our scheme. In addition, unlike the attack-resistant MMSE scheme, the voting-based scheme can tolerate multiple (colluding or non-colluding) malicious location references more effectively.

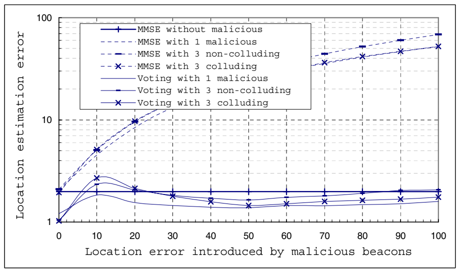


Fig. 7. Performance of the voting-based scheme ($M = 100, S = 0$). Unit of measurement for x and y axes: meter

Note that the curves for the voting-based scheme in Figure 7 have a bump when the location error introduced by malicious location references is around 10m. This is because the malicious location references are not significantly different from the benign location references around this point, and our scheme cannot completely shield the effect of malicious location references. Nevertheless, the attacker will not be able to introduce large location estimation errors by simply creating large location errors. As a result, the location estimation errors are always bounded even if there are malicious attacks. In addition, we also note that the performance of voting-based scheme under attacks is usually better than the performance of MMSE scheme without attacks. This is because we used the MMSE-based method in [26] in the simulation, which estimates a node's location by only *approximately* minimizing the mean square error.

F. Implementation and Field Experiments

We have implemented both schemes on TinyOS [10], an operating system for networked sensors. These implementations are targeted at MICA2 motes [4] running TinyOS. The attack-resistant MMSE is implemented based on the basic MMSE method proposed in [26]. However, our implementation of the basic MMSE method is simplified by only using the location coordinates (without the ultrasound propagation speed, which is not necessary in our study).

Scheme	ROM (bytes)	RAM (bytes)
MMSE	2034	286
AR-MMSE	3226	396
Voting-Based	4488	174

TABLE I
CODE SIZE (ASSUME 12 LOCATION REFERENCES; $M = 100$)

Table I gives the code size (ROM and RAM) for these implementations on MICA2 platform. Table I is obtained by assuming at most 12 location references. More location references will increase the RAM size of the program, but the increased RAM is only required to save the additional location references.

Figure 8 shows the average execution time of the basic MMSE, the attack-resistant MMSE, and the voting-based schemes on real MICA2 motes. These data are collected by counting the numbers of CPU clock cycles spent on location estimation. The location

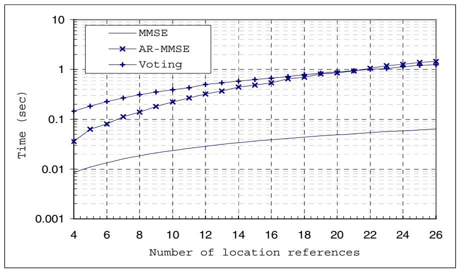


Fig. 8. Average execution time on MICA2 motes ($\epsilon = 4$m, $\tau = 0.8\epsilon$, $M = 100$ and $S = 0$)

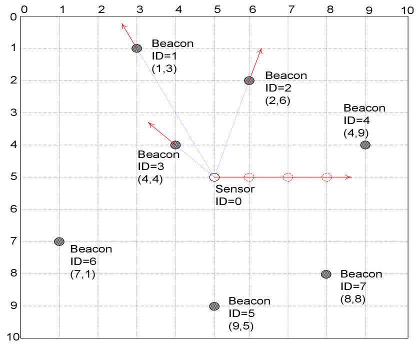


Fig. 9. Target area of field experiment.

references used in the experiment are generated from the simulation in Section II-E. We can see that the basic MMSE method has the least execution time. The attack-resistant MMSE scheme has less computational cost than the voting-based scheme when the number of location references is small; however, when there are large numbers of location references (e.g., 20), it takes the voting-based method less time to finish than the attack-resistant MMSE method. From Table I and Figure 8, we conclude that our proposed techniques are practical for the current generation of sensor networks in terms of the storage and the computation overheads, especially when the locations of sensor nodes do not change frequently.

To further study the feasibility of our techniques, we performed an outdoor field experiment. In this experiment, eight MICA2 motes were deployed in a 10×10 target field, where each unit of distance is 4 feet, as shown in Figure 9. The sensor node with ID 0 is configured as a non-beacon node, which is located at the center of the field. All the other sensor nodes are configured as beacon nodes.

We considered three attack scenarios in this experiment. In the first scenario, beacon node 1 is configured as a malicious beacon node that always declares a location e feet away from its real location in the direction away from the non-beacon node. In the second scenario, beacon nodes 1, 2 and 3 are configured as malicious beacon nodes. Each of these three nodes declares a location e feet away from its real location in the directions away from the non-beacon node. In the third scenario, three malicious beacon nodes 1, 2, and 3 work together to create a virtual location. Each of these three nodes declares a false location by increasing its horizontal coordinate by e feet. This actually creates a virtual location in the horizontal axis e feet away from the non-beacon node's real location. This is illustrated in Figure 9 by the horizontal arrow starting from the non-beacon node.

To measure the distance (δ) between sensor nodes, we use a simple RSSI based technique. Note that the Active Message protocol in TinyOS provides a reading in the *strength* field for the MICA2 platform. This value is returned in every received packet, and can be used to compute the signal strength. Thus, we performed an experiment before the actual field experiment to estimate the relationship between the values of this field and the distance between two nodes. For each given distance, we computed the average of this values on 20 observations. We then built a table that contains distances and the corresponding average readings. During the field experiments, when a sensor node receives 20 packets from a beacon node, it computes the average of the *strength* values, and estimates the distance with interpolation according to this table. For example, if the average reading v falls in between two adjacent points (v_i, d_i) and (v_{i+1}, d_{i+1}) in the table, the sensor computes the distance $d = d_i + \frac{(v-v_i)\times(d_{i+1}-d_i)}{v_{i+1}-v_i}$. We set ϵ to 4 feet, which is the maximum distance measurement error observed in the experiment.

Figure 10 shows the performance of the proposed methods and the basic MMSE method in the field experiment. For the first two attack scenarios, we can see that the proposed methods can tolerate malicious location references quite effectively. The performance in the third scenario is worse than the first two cases. The reason is that the non-beacon nodes has only 4 benign location references, but 3 colluding location references. However, we still see that the location estimation error drops when the location errors introduced by the malicious attacks are above certain thresholds. Overall, the location estimation errors caused by malicious attacks are bounded when the proposed techniques are used, while the errors can be arbitrarily large when the basic MMSE method is used.

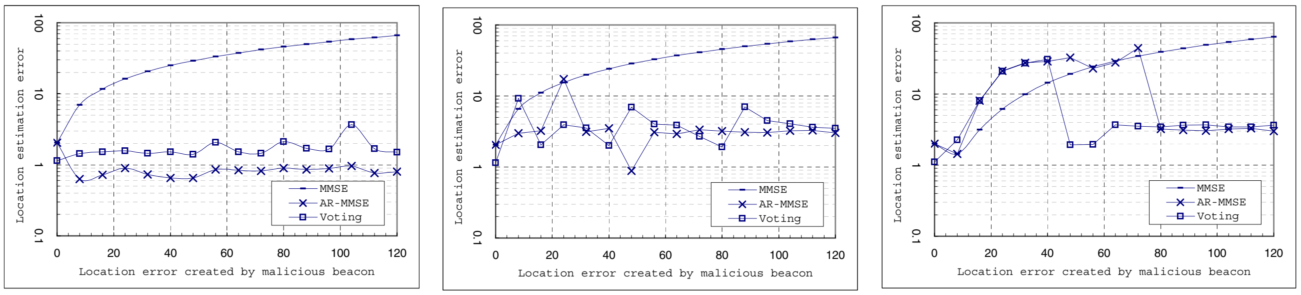
The field experiment further shows that our methods are efficient and effective in tolerating malicious attacks. It also indicates that our methods are promising for the current generation of sensor networks.

III. Related Work

Many range-based localization schemes have been proposed for sensor networks [5], [19], [20], [26], [27]. Savvides et al. developed AHLoS protocol based on Time Difference of Arrive [26], which was extended in [27]. Doherty et al. presented a localization scheme based on connectivity constraints and relative signal angles between neighbors [5]. Angle of Arrival is used to develop localization scheme in [20] and [19]. Range-free schemes are proposed to provide localization services for the applications with less precision requirements [2], [9], [18], [21]. Bulusu, Heidemann and Estrin proposed to estimate a sensor's location as the centroid of all locations in the received beacon signals [2]. Niculescu and Nath proposed to use the minimum hop count and the average hop size to estimate the distance between nodes and then determine sensor nodes' locations accordingly [21]. None of these schemes will work properly when there are malicious attacks. The techniques developed in [25] and [16] can deal with malicious attacks to a certain extent. However, neither of them can ensure correct location discovery when beacon nodes are compromised. The techniques proposed in this paper address this problem by tolerating malicious beacon signals.

A robust location detection is developed in [24]. However, it cannot be directly applied in sensor networks due to its high computation and storage overheads. A voting-based scheme named Cooperative Location Sensing (CLS) was proposed in [8]. However, CLS is designed for powerful nodes (e.g., PDAs), while our scheme further uses iterative refinement to improve the performance with small storage overhead. Therefore, our technique can be implemented and executed efficiently on resource constrained sensor nodes.

Security in sensor networks has attracted a lot of attention in the past several years. To provide practical key management, researchers have developed key pre-distribution techniques [3], [6], [7]. To enable broadcast authentication, a protocol named μTESLA has been explored to adapt to resource constrained sensor networks [22]. Security of sensor data has been studied in [11], [23]. Attacks against



(a) A single malicious loc. ref. (b) 3 non-colluding malicious loc. ref. (c) 3 colluding malicious loc. ref.

Fig. 10. Results of the field experiment. Assume $M = 100$ and $S = 0$ for voting-based scheme; $\tau = 0.8\epsilon = 3.2$ feet for attack-resistant MMSE. Unit of measurement for x and y axes: feet

routing protocols in sensor networks and possible counter measures were investigated in [14]. The research in this paper addresses another fundamental security problem that has not drawn enough attention.

IV. Conclusion and Future Work

In this paper, we proposed an attack-resistant MMSE-based location estimation and a voting-based location estimation technique to deal with attacks in localization schemes. We have implemented the proposed techniques on MICA2 motes [4] running TinyOS [10], and evaluated them through both simulation and field experiments. Our experiences indicate that the proposed techniques are promising solutions for securing location discovery in wireless sensor networks.

Our future research is two-fold. First, we will study how to combine the proposed techniques with other protection mechanisms such as wormhole detection. Second, our simulations and experiments in this paper are conducted in small scales. It is very interesting to study the performance in a large scale.

Acknowledgment The authors would like to thank the anonymous reviewers and the shepherd Dr. Adrian Perrig for their valuable comments. This work is supported by the National Science Foundation (NSF) under grants CNS-0430223 and CNS-0430252.

References

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

[2] N. Bulusu, J. Heidemann, and D. Estrin. GPS-less low cost outdoor localization for very small devices. In *IEEE Personal Communications Magazine*, pages 28–34, October 2000.

[3] H. Chan, A. Perrig, and D. Song. Random key predistribution schemes for sensor networks. In *IEEE Symposium on Research in Security and Privacy*, pages 197–213, 2003.

[4] Crossbow Technology Inc. Wireless sensor networks. `http://www.xbow.com/Products/Wireless_Sensor_Networks.htm`. Accessed in February 2004.

[5] L. Doherty, K. S. Pister, and L. E. Ghaoui. Convex optimization methods for sensor node position estimation. In *Proceedings of INFOCOM'01*, 2001.

[6] W. Du, J. Deng, Y. S. Han, and P. Varshney. A pairwise key predistribution scheme for wireless sensor networks. In *Proceedings of 10th ACM Conference on Computer and Communications Security (CCS'03)*, pages 42–51, October 2003.

[7] L. Eschenauer and V. D. Gligor. A key-management scheme for distributed sensor networks. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, pages 41–47, November 2002.

[8] C. Fretzagias and M. Papadopouli. Cooperative location-sensing for wireless networks. In *Second IEEE International conference on Pervasive Computing and Communications*, 2004.

[9] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. F. Abdelzaher. Range-free localization schemes in large scale sensor networks. In *Proceedings of ACM MobiCom 2003*, 2003.

[10] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D.E. Culler, and K. S. J. Pister. System architecture directions for networked sensors. In *Architectural Support for Programming Languages and Operating Systems*, pages 93–104, 2000.

[11] L. Hu and D. Evans. Secure aggregation for wireless networks. In *Workshop on Security and Assurance in Ad Hoc Networks*, January 2003.

[12] L. Hu and D. Evans. Using directional antennas to prevent wormhole attacks. In *Proceedings of the 11th Network and Distributed System Security Symposium*, pages 131–141, February 2003.

[13] Y.C. Hu, A. Perrig, and D.B. Johnson. Packet leashes: A defense against wormhole attacks in wireless ad hoc networks. In *Proceedings of INFOCOM 2003*, April 2003.

[14] C. Karlof and D. Wagner. Secure routing in wireless sensor networks: Attacks and countermeasures. In *Proceedings of 1st IEEE International Workshop on Sensor Network Protocols and Applications*, May 2003.

[15] B. Karp and H. T. Kung. GPSR: Greedy perimeter stateless routing for wireless networks. In *Proceedings of ACM MobiCom 2000*, 2000.

[16] L. Lazos and R. Poovendran. Serloc: Secure range-independent localization for wireless sensor networks. In *ACM workshop on Wireless security (ACM WiSe 2004)*, Philadelphia, PA, October 1 2004.

[17] D. Liu, P. Ning, and W.K. Du. Attack-resistant location estimation in wireless sensor networks. Technical Report TR-2004-29, North Carolina State University, Department of Computer Science, 2004.

[18] R. Nagpal, H. Shrobe, and J. Bachrach. Organizing a global coordinate system from local information on an ad hoc sensor network. In *IPSN'03*, 2003.

[19] A. Nasipuri and K. Li. A directionality based location discovery scheme for wireless sensor networks. In *Proceedings of ACM WSNA'02*, September 2002.

[20] D. Niculescu and B. Nath. Ad hoc positioning system (APS) using AoA. In *Proceedings of IEEE INFOCOM 2003*, pages 1734–1743, April 2003.

[21] D. Niculescu and B. Nath. DV based positioning in ad hoc networks. In *Journal of Telecommunication Systems*, 2003.

[22] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and D. Tygar. SPINS: Security protocols for sensor networks. In *Proceedings of Seventh Annual International Conference on Mobile Computing and Networks*, July 2001.

[23] B. Przydatek, D. Song, and A. Perrig. SIA: Secure information aggregation in sensor networks. In *Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys '03)*, Nov 2003.

[24] S. Ray, R. Ungrangsi, F. D. Pellegrini, A. Trachtenberg, and D. Starobinski. Robust location detection in emergency sensor networks. In *Proceedings of IEEE INFOCOM 2003*, April 2003.

[25] N. Sastry, U. Shankar, and D. Wagner. Secure verification of location claims. In *ACM Workshop on Wireless Security*, 2003.

[26] A. Savvides, C. Han, and M. Srivastava. Dynamic fine-grained localization in ad-hoc networks of sensors. In *Proceedings of ACM MobiCom '01*, pages 166–179, July 2001.

[27] A. Savvides, H. Park, and M. Srivastava. The bits and flops of the n-hop multilateration primitive for node localization problems. In *Proceedings of ACM WSNA '02*, September 2002.

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