An Efficient Gradient Descent Approach to Secure Localization in Resource Constrained Wireless Sensor Networks

Ravi Garg, *Student Member, IEEE*, Avinash L. Varna, *Member, IEEE*, and Min Wu, *Fellow, IEEE*

***Abstract*—Many applications of wireless sensor networks require precise knowledge of the locations of constituent nodes. In these applications, it is desirable for the nodes to be able to autonomously determine their locations before they start sensing and transmitting data. Most localization algorithms use anchor nodes with known locations to determine the positions of the remaining nodes. However, these existing techniques often fail in hostile environments where some of the nodes may be compromised by adversaries and used to transmit misleading information aimed at preventing accurate localization of the remaining sensors. In this paper, a computationally efficient secure localization algorithm that withstands such attacks is described. The proposed algorithm combines iterative gradient descent with selective pruning of inconsistent measurements to achieve high localization accuracy. Results show that the proposed algorithm utilizes fewer computational resources and achieves an accuracy better than or comparable to that of existing schemes. The proposed secure localization algorithm can also be used in mobile sensor networks, where all nodes are moving, to estimate the relative locations of the nodes without relying on anchor nodes. Simulations demonstrate that the proposed algorithm can find the relative location map of the entire mobile sensor network even when some nodes are compromised and transmit false information.**

***Index Terms*—Gradient descent, mobile sensor networks (MSNs), secure localization, wireless sensor networks (WSNs).**

I. Introduction

RECENT technological advances in microelectromechanical systems (MEMS) and wireless communications have enabled the development of small, low-cost, low-power sensor nodes capable of sensing data of importance, processing it, and transmitting it to other nodes or a base station through wireless medium. Multiple sensors deployed in a given area form a network and are referred to as a wireless sensor network (WSN). These WSNs are expected to form the backbone of future intelligent networks for a broad range of applications such as underwater surveillance [1], military surveillance, traffic monitoring [2], habitat monitoring [3], forest fire detection [4], and flood detection [5].

Manuscript received July 21, 2011; revised November 11, 2011; accepted November 29, 2011. Date of publication January 12, 2012; date of current version March 08, 2012. This work was supported in part by the National Science Foundation under Grant #0824081 and in part by a University of Maryland Litton Graduate Fellowship. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Wade Trappe.

The authors are with the Department of Electrical and Computer Engineering, University of Maryland, College Park, MD 20742 USA (e-mail: ravig@umd.edu; varna@umd.edu; minwu@umd.edu.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIFS.2012.2184094

In such applications as forest fire and flood detection [4], [5], sensors alert the base station to any changes in parameters that indicate a potential for forest fire or flooding. The base station needs to know the locations of the nodes transmitting the data, so that appropriate actions may be taken at the relevant site to prevent disasters or to provide early response and contain the damage. In many applications, static sensors may be randomly deployed within an area using helicopters or road vehicles. As a result, the sensor locations are not known *a priori* and need to be determined after deployment.

Location information is also important for correctly interpreting the sensed data during communication of sensing measurements among nodes and the base station. Each node may not be within the communication range of every other node or the base station. If a node needs to transmit a message to another node with which it does not share a direct communication link, the message should be routed through intermediate nodes in the network. To minimize the communication cost, the shortest route between the transmitter and the recipient should be used. Such situations also arise in cooperative communications applications [6]. Many routing algorithms rely on the locations of the nodes in the network to determine the shortest path for transmitting the message [7].

Additionally, sensors should be optimally deployed to provide maximum coverage in a given area at a low communication cost. This can be achieved with the help of mobile nodes [8]. The algorithms used for such mobility assisted efficient deployment require sensors to be location aware. In robotics applications such as distributed formation and coordination, where mobile robots with limited communication range coordinate to achieve a common task, the location information of the robots is needed to ensure connectivity in the network [9]. Thus, we see that location information is important in both static and mobile sensor networks.

Node locations can be obtained by using GPS devices on the nodes. However, equipping each sensor with a GPS may not be feasible for large scale networks with small low-cost sensors. As a result, an important first step in setting up a sensor network is to accurately determine the position of each individual node through a process called *localization*. Most localization schemes rely on a set of beacon or anchor nodes with known location information to identify the positions of the remaining nodes. In these schemes, anchor nodes transmit a beacon signal which contains their own location, using which other nodes can estimate their distances from the anchors. Commonly used distance metrics are received signal strength (RSS) [10], time of

arrival (ToA), time difference of arrival (TDoA) [11], angle of arrival (AOA) [12], and hop count [13]. Once the non-anchor nodes have a sufficient number of distance measurements, they can determine their location by triangulation or trilateration.

In hostile environments, an adversary may wish to prevent accurate localization of the nodes and thus prevent the entire network from functioning properly. The adversary may compromise some nodes and thereby gain access to the secret keys and other data stored on the node. This information can then be used to provide misleading information to the base station and other nodes in the network. Incorrect location references may also be provided by intercepting and replaying the packets containing measurements transmitted by anchor nodes. Without effective approaches to filter out or nullify the effect of incorrect measurements, localization would result in a wrong estimate of the sensor position. Hence, there is a strong need to design secure localization algorithms that are robust to such intentional attacks and accurately determine the positions of sensors in the presence of adversaries. At the same time, as the sensors have limited memory, computational, and energy resources, these secure localization algorithms should be resource efficient.

A. Prior Work

A related problem of location verification has been explored in the literature, where the focus is on developing strategies to verify that a node is indeed located at the claimed position. Methods such as verifiable multilateration, location verification using mobile base stations, and several other distance bounding protocols have been proposed to withstand attacks in secure location verification problems [14]–[16].

The problem of secure localization in WSNs in the presence of malicious adversaries has also attracted attention in the research community. A greedy approach to find the location consistent with the largest number of measurements from anchor nodes was explored in [17]. A voting-based scheme was also proposed, in which the localization area is divided into a grid and the vote count of each grid point is incremented if its distance from an anchor node is approximately equal to the distance measurement obtained from that anchor. A similar voting approach with the help of sectored antennas and beacon nodes was proposed in [18]. From a signal processing point of view, the voting based scheme is similar in spirit to the Hough transform used for detecting objects with certain shapes in computer vision and image processing literature [19]. In the Hough transform, a voting procedure is carried out in a parameter space, from which candidate parameters for objects are determined as local maxima of accumulated votes. Similarly, in the voting-based scheme for secure localization, the location with the maximum votes is identified as the position of the node.

A least median square (LMdS) approach was proposed in [20] to solve the localization problem for scenarios where less than 50% of the nodes are malicious. This method shares similarities with the random sample consensus (RANSAC) algorithm [21], as it uses several subsets of nodes to identify candidate locations, and then chooses the solution that minimizes the median of the residues. Most of these existing methods localize the nodes with small error as long as the fraction of malicious nodes is not too large. However, the memory requirement and computational cost of running these algorithms is still high and can be difficult to meet in resource limited applications.

In contrast to static sensor networks, very little work has been done on secure localization in mobile sensor networks. A two stage Monte Carlo based approach for localization was proposed in [22]. In the first stage, using the current estimate of the location, a fixed number of candidate sample locations that satisfy a constraint on the maximum velocity of the nodes are randomly generated. In the second stage of filtering, samples that are inconsistent with the measurements obtained from anchor nodes are filtered out, and a final estimate of location is found by averaging the remaining samples. The localization accuracy of the algorithm in [22] was improved in [23] using a box shaped region to sample particles in the prediction phase and eliminate inconsistent particles in the filtering stage. These algorithms did not consider the presence of malicious anchor nodes in the network.

The Monte Carlo algorithm was extended to incorporate security by modifying the filtering stage in [24]. Instead of identifying points that are consistent with all measurements, the position consistent with the maximum number of measurements from anchors is determined. This approach is similar to the voting-based approach [17] for secure localization in static sensor networks explained previously and suffers from the same drawback of high computational and storage requirements. Algorithms proposed in [25] use the hop count information and communication range information of sensor nodes to find a feasible region for the node position and use this information to estimate the location. These prior works assume the presence of some anchor nodes that are used to determine the position of the mobile nodes, and cannot be applied to mobile networks without anchor nodes.

In this paper, we develop an iterative technique for secure localization that is applicable to both static and mobile networks. In terms of the vector interpretation for iterative updates, the proposed algorithm has similarities to the robust localization algorithm inspired by self organizing maps proposed in [26]. The algorithm in [26] considered noise, but was not designed to withstand attacks by active adversaries, whereas we develop an algorithm for localization that can filter out malicious measurements obtained from nodes compromised by adversaries.

B. Overview of This Work

In this paper, we propose a computationally efficient method to solve the problem of secure localization based on gradient descent. The main idea behind the algorithm is to minimize a suitable cost function involving the position of the localizing node and the available measurements using an iterative gradient descent approach. The cost function is dynamically updated to remove inconsistent measurements arising from malicious nodes. The algorithm operates in two stages. In the first stage, the cost function involves data from all anchor nodes. In the second stage, selective pruning of inconsistent measurements is performed to mitigate the effect of malicious nodes on the solution. The second stage of the algorithm is similar in spirit to the approach employed in [27] to find the least trimmed squares (LTS) solution to data containing outliers [28]. We show that the

proposed algorithm can be used for secure localization in both static and mobile sensor networks, and can achieve localization accuracy better than or comparable to existing algorithms in a computationally efficient manner.

This paper is organized as follows. Section II describes the problem setup in the simple case where the distance between the localizing node and the anchor nodes is measured directly. The proposed algorithm for secure localization is then described. In Section III, the gradient descent approach for secure localization is applied to the scenario when only time difference of arrival (TDoA) measurements are available. Section IV considers the case of secure localization of mobile sensors. Section V summarizes the contributions and concludes the paper.

II. Secure Localization in Static Sensor Networks

In this section, we consider the secure localization problem for sensor networks where direct measurements of the distance between the localizing node and the anchor nodes are available. These measurements may be obtained through different techniques such as hop count and ToA measurements. When ToA is used to obtain the distance measurements, each anchor node transmits a beacon signal that includes a timestamp and its own location. The localizing node determines its distance from the anchor node based on the embedded timestamp and the time at which it receives the beacon signal. The scenarios where direct measurements of the distance are not available, such as when TDoA is used for localization, will be considered in Section III.

A. Problem Formulation

Let N be the number of anchor nodes whose locations are known. These may represent nodes that are deployed at known locations and serve to bootstrap the localization of the other sensors in the network. Once a node has determined its own location, it can function as an anchor node for localizing the remaining nodes. Let us denote the true position of the localizing node by $\mathbf{P} = [x_{\text{true}}, y_{\text{true}}]^T$. The localizing node receives the location of each of the anchor nodes $\mathbf{P_k} = [x_k, y_k]^T$ and an estimate of the distance between the anchor node and itself, which may be obtained using techniques such as ToA. These distance measurements may be noisy in practice, and we model the measurement errors as additive Gaussian noise with zero mean and variance σ^2. Given the set of noisy measurements $\{(\mathbf{P_k}, d_k)\}, k = 1, 2, \ldots, N$, an estimate for the node's location $\hat{\mathbf{P}} = [\hat{x}, \hat{y}]^T$ can be obtained by solving the following overdetermined system of equations in a least square (LS) sense:

$$\|\mathbf{P}_k - \hat{\mathbf{P}}\| - d_k = 0 \quad k = 1, 2, \ldots, N. \tag{1}$$

Nodes compromised by adversaries may intentionally report wrong information in their $(\mathbf{P}_k, d_k)$ measurements. In these cases, the LS estimate may be quite far from the true location. Thus, we need secure localization algorithms that are resilient to such attacks.

In the static nodes setting we consider two types of adversaries with different objectives and resources. The first kind of adversaries are able to compromise multiple nodes, but have limited communication and computational resources to coordinate the attacks launched by these nodes. We refer to these attacks as *non-coordinated attacks*. The second type of adversaries have more resources and want to not only prevent the network from precisely locating the nodes, but also try to shift the location estimates to some desired position. We refer to these attacks as *coordinated attacks*. Detailed formulations of these attacks are as follows.

1) Noncoordinated Attacks: In noncoordinated attacks, the adversary is assumed to act independently at each compromised node and aims to prevent accurate localization by perturbing the distance estimates reported to the localizing node. Without loss of generality, we assume that each malicious node modifies the d_k value of the $(\mathbf{P_k}, d_k)$ measurements, since modifying any other parameter can be transformed into an equivalent modification of the d_k value. We model the noncoordinated attack by adding independent uniformly distributed perturbations to the actual distance estimates from each malicious node and provide this information to the localizing node. Let $\text{dist}_k = \|\mathbf{P_k} - \mathbf{P}\|$ be the actual distance between the localizing node and the anchor. Define

$$d_k^{(nc)} = \begin{cases} \text{dist}_k + u_k + n_k, & \text{if node } k \text{ is malicious,} \\ \text{dist}_k + n_k, & \text{otherwise} \end{cases}$$

where the u_k are independent zero-mean uniform random variables with variance σ^2_{attack} that model the perturbation introduced from noncoordinated attacks, and the n_k are independent $\mathcal{N}(0, \sigma^2)$ Gaussian variables representing the measurement noise. Under the noncoordinated attack setting, the localizing node receives the measurements $\{(\mathbf{P_k}, d_k^{(nc)})\}, k = 1, 2, \ldots, N$ from N anchor nodes, and uses this information to determine its position.

2) Coordinated Attacks: A stronger attack against the network can be launched by multiple compromised nodes acting together to make a localizing node estimate its position as $\mathbf{P_{mal}} = [x_{\text{mal}}, y_{\text{mal}}]^T$, which is some arbitrary point determined by the attackers. We model this scenario by reporting the distance between the anchor node position $\mathbf{P_k}$ and $\mathbf{P_{mal}}$ as the measurement from the malicious anchor. Specifically, let $d_k^{(c)}$ be defined as

$$d_k^{(c)} = \begin{cases} \|\mathbf{P_k} - \mathbf{P_{mal}}\| + n_k, & \text{if node } k \text{ is malicious} \\ \text{dist}_k + n_k, & \text{otherwise} \end{cases}$$

where dist_k is the actual distance between the kth anchor and the localizing node and n_k represents measurement noise as before. The localizing node receives the measurements $\{(\mathbf{P_k}, d_k^{(c)})\}, k = 1, 2, \ldots, N$ from the anchor nodes and uses this information to determine its position. The strength of the coordinated attack is characterized in terms of the distance, $d_a = \|\mathbf{P_{mal}} - \mathbf{P}\|$, between the actual position and the position reported by malicious nodes.

B. Proposed Method for Secure Localization

In this subsection, we propose an iterative secure localization algorithm by combining gradient descent with a selection stage to filter out the malicious measurements [29]. We first consider the likelihood of the measurements given the true position of the

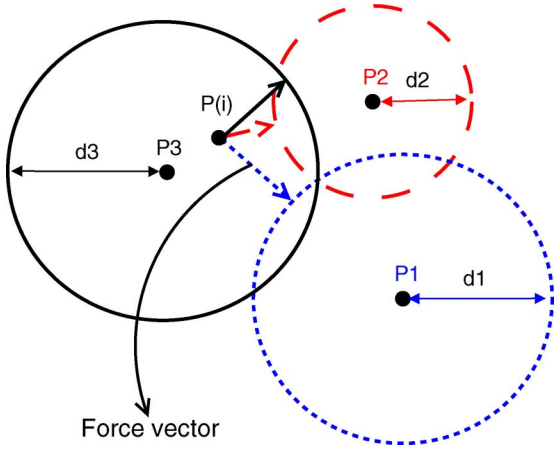

Fig. 1. Force vector representation of terms contributing to the gradient.

localizing node. When there is no malicious node, and the measurement noise is Gaussian, the likelihood of the measurements given the true position of the localizing node $\mathbf{P}$ is

$$\Pr\left(\{d_k\}_{k=1}^N \middle| \mathbf{P}, \{\mathbf{P_k}\}_{k=1}^N\right) = \frac{1}{(2\pi\sigma^2)^{N/2}} \exp\left\{-\frac{1}{2\sigma^2}\sum_{k=1}^N (\|\mathbf{P_k}-\mathbf{P}\|-d_k)^2\right\}. \quad (2)$$

The maximum-likelihood (ML) estimate $\hat{\mathbf{P}}$ for the true position can then be found by maximizing the likelihood of the measurements, or equivalently, by minimizing the negative of the exponent

$$\begin{aligned}\hat{\mathbf{P}} &= \arg\max_{\mathbf{P}} \Pr\left(\{d_k\}_{k=1}^N \middle| \mathbf{P}, \{\mathbf{P_k}\}_{k=1}^N\right) \\ &= \arg\min_{\mathbf{P}} \frac{1}{2}\sum_{k=1}^N (\|\mathbf{P_k}-\mathbf{P}\|-d_k)^2 \\ &= \arg\min_{\mathbf{P}} f(\mathbf{P}) \end{aligned} \quad (3)$$

where $f(\mathbf{P})$ denotes the cost function in (3) and corresponds to the negative of the exponent in (2). The ML estimate $\hat{\mathbf{P}}$ is thus identical to the LS estimate obtained by solving (1) in a least squares sense.

In the proposed secure localization algorithm, we adopt an iterative gradient descent algorithm to first search for the LS solution. The algorithm starts by randomly initializing the estimate $\hat{\mathbf{P}}(0)$ to some point in the deployment area. At the ith step of the iteration, the gradient of the cost function $f(\mathbf{P})$ is evaluated at the current estimate $\hat{\mathbf{P}}(i-1)$, and the estimate is then updated by moving it one step in the direction of the negative of the gradient. Let $\mathbf{g}(i)$ denote the negative of the gradient of the cost function at the current estimate of the position

$$\mathbf{g}(i) = -\nabla_{\mathbf{P}}(f(\mathbf{P}))|_{\mathbf{P}=\hat{\mathbf{P}}(i-1)}$$

where $\nabla_{\mathbf{P}}(\cdot)$ denotes the derivative with respect to $\mathbf{P}$. The estimate is then updated by moving it one step in the direction of the negative of the gradient as

$$\hat{\mathbf{P}}(i) = \hat{\mathbf{P}}(i-1) + \delta(i) \times \frac{\mathbf{g}(i)}{\|\mathbf{g}(i)\|}$$

where $\delta(i)$ is the step size at the ith iteration and $(\mathbf{g}(i))/(\|\mathbf{g}(i)\|)$ is the unit vector in the direction of the negative of the gradient. The negative gradient $\mathbf{g}(i)$ is found to be

$$\begin{aligned}\mathbf{g}(i) &= -\nabla_{\mathbf{P}} f(\mathbf{P})|_{\mathbf{P}=\hat{\mathbf{P}}(i-1)} \\ &= -\nabla_{\mathbf{P}}\left(\frac{1}{2}\sum_{k=1}^N (\|\mathbf{P_k}-\mathbf{P}\|-d_k)^2\right)\Bigg|_{\mathbf{P}=\hat{\mathbf{P}}(i-1)} \\ &= \sum_{k=1}^N (\|\mathbf{P_k}-\hat{\mathbf{P}}(i-1)\|-d_k) \times \frac{\mathbf{P_k}-\hat{\mathbf{P}}(i-1)}{\|\mathbf{P_k}-\hat{\mathbf{P}}(i-1)\|} \\ &= \sum_{k=1}^N \mathbf{g_k}(i) \end{aligned} \quad (4)$$

where we define the term $\mathbf{g_k}(i)$ as

$$\mathbf{g_k}(i) = (\|\mathbf{P_k}-\hat{\mathbf{P}}(i-1)\|-d_k) \times \frac{\mathbf{P_k}-\hat{\mathbf{P}}(i-1)}{\|\mathbf{P_k}-\hat{\mathbf{P}}(i-1)\|}.$$

Conceptually, as shown in Fig. 1, the gradient component $\mathbf{g_k}(i)$ can be visualized as a "force vector" with direction along the line joining the current estimate of the location $\hat{\mathbf{P}}(i-1)$ and the position of the anchor node $\mathbf{P_k}$ and magnitude equal to the distance between the current estimate and the circle of radius d_k around the anchor node. The sum of these force vectors gives the overall gradient.

Each iteration results in a new estimate that has a higher probability of being the true location of the node. This gradient descent algorithm eventually converges to the ML estimate which is the same as the LS estimate when in the absence of the malicious nodes. As described previously, due to malicious measurements from adversaries, the LS estimate can have large errors. Hence, once the gradient descent algorithm converges to the LS solution, we switch to a selection stage in which some force vectors are pruned as discussed next.

Selection Stage: In the noncoordinated attack case, the independent perturbations added by various malicious nodes tends to average out and the LS solution is close to the true position. In the coordinated attack case whereby less than 50% of the nodes are malicious, the LS estimate obtained from the first stage of the algorithm is closer to the true position $\mathbf{P}$ than to the position $\mathbf{P_{mal}}$ chosen by the malicious nodes. This is because in such situations, the true position satisfies more equations in (1) than the position reported by the malicious nodes. So the LS solution tends to be closer to the true position than $\mathbf{P_{mal}}$.

As a result, when the estimate of the node's position $\hat{\mathbf{P}}(i)$ approaches the LS estimate, the residues corresponding to the terms arising from the malicious nodes tend to be larger than those from the honest nodes. We update the cost function to exclude the terms with large residues, which are likely to correspond to the measurements from the malicious nodes. In our algorithm, we achieve this by pruning out a fraction of the force vectors with large magnitudes and using the remaining vectors to compute the gradient. The estimate is updated by moving it one step in the direction of this modified gradient at each iteration. The final algorithm is shown in Algorithm 1, and Fig. 2 shows a flowchart of the proposed algorithm.

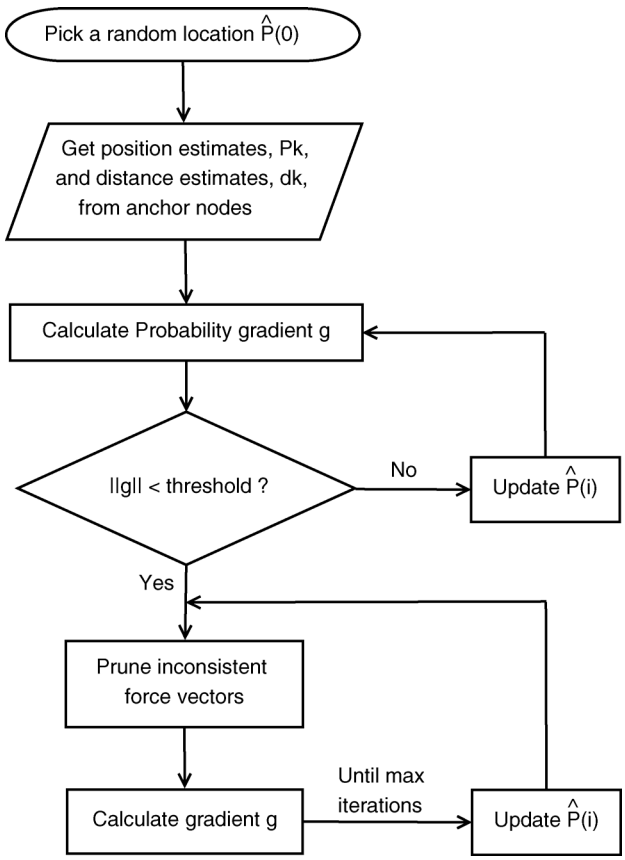

Fig. 2. Flow chart for localization algorithm.

Algorithm 1 Proposed algorithm for secure localization

Input: N=number of anchor nodes; M= number of iterations; $\delta(.)$=step size function (constant or non-increasing) S={all anchor nodes}; $\{(\mathbf{P_k}, d_k)\}$, where $\mathbf{P_k} = [x_k, y_k]^T, k = 1, 2, \ldots, N$ be the position claimed by k^{th} anchor node, and d_k be the distance reported to the localizing node.

Output: Estimated coordinates $\hat{\mathbf{P}}(M) = [\hat{x}, \hat{y}]^T$

Initialization: a random point $\hat{\mathbf{P}}(0)$; $stage = 1$

for $i = 1 : M$ **do**

  for $k = 1 : N$ **do**

    $\mathbf{g_k}(i) = (\|\mathbf{P_k} - \hat{\mathbf{P}}(i-1)\| - d_k) \times \frac{\mathbf{P_k} - \hat{\mathbf{P}}(i-1)}{\|\mathbf{P_k} - \hat{\mathbf{P}}(i-1)\|}$

  end

  $\mathbf{g}(i) = \sum_{k=1}^{N} \mathbf{g_k}(i)$; //gradient

  if $(\|\mathbf{g}(i)\| < threshold) or (stage == 2)$ **then**

    $stage$ = 2; //enter the selection stage

    S={set of $\frac{N}{2}$ anchor nodes whose corresponding force vectors are the smallest}

  end

  $\mathbf{g}(i) = \sum_{k \epsilon S} \mathbf{g_k}(i)$; //update gradient

  Update: $\hat{\mathbf{P}}(i) = \hat{\mathbf{P}}(i-1) + \delta(i) \times \frac{\mathbf{g}(i)}{\|\mathbf{g}(i)\|}$

end

C. Comparison of Computational Complexity

We now compare the computational complexity of the proposed gradient descent algorithm to the voting scheme [17] and the LMdS scheme [20]. Table I shows the computational complexity for each algorithm and the average run time for a set of experiments conducted on MATLAB platform. From this table, we see that for the voting scheme, the complexity increases with the square of the grid size. To obtain better localization accuracy, the grid needs to be quantized more finely, leading to a higher number of cells in the grid. Alternatively, grids can be coarsely quantized in the beginning and the localization resolution can be improved by conducting multiple stages of the voting algorithm, with each stage using progressively finer cells in the areas of the grid which receive high number of votes in the previous stage. This leads to high computational requirements in the voting scheme.

TABLE I
COMPARISON OF RUN TIME COMPLEXITY OF DIFFERENT ALGORITHMS

Method	Complexity	Run time(in ms)
Least Median Square	$\Theta(M_1 N)$	24.8
Voting based scheme	$\Theta(n_1^2 N)$	14.8
Gradient Descent	$\Theta(MN)$	3.7

The LMdS approach requires a certain minimum number of subsets of nodes M_1, which increases as the percentage of malicious nodes increases, in order to ensure that one estimate is the correct estimate with very high probability. An LS estimate needs to be found for each of these subsets, which is computationally expensive. The computation complexity associated with the LMdS method is calculated using the linear least squares (LLS) algorithm described in [30]. LMdS algorithm first performs M_1 LLS on different subsets of size n giving a computational complexity of $\Theta(M_1 n)$. After finding each LLS solution, a consistency check with measurements from all N nodes is performed, which has a computational complexity of $\Theta(M_1 N)$ for all M_1 rounds. In the final step, another LS estimate is found using the maximum size subset of nodes that have passed the consistency test. The computation complexity of this final operation is smaller than that of the first two steps. The overall computation complexity of LMdS is $\Theta(M_1(N+n))$, which can be represented as $\Theta(M_1 N)$, due to $n < N$.

In contrast, the computational complexity of the proposed scheme is independent of the number of malicious nodes and the grid size, although it increases linearly with the number of iterations. The number of iterations can be reduced by choosing variable step size to increase the convergence rate of the algorithm [31]. At each iteration, our proposed algorithm calculates only the distance of the current estimate from the anchor nodes and requires less computation. Thus, the gradient descent algorithm is computationally simpler than the voting-based scheme and the LMdS method.

In Fig. 3, we plot the run time required to achieve a desired localization accuracy for different secure localization algorithms considered in this paper. From this plot, we see that the run time required to achieve a given localization accuracy for our proposed method is approximately eight times lower as compared to the LMdS method. Comparing the voting scheme and gradient descent based scheme, we can see that they have similar run time at settings that result in medium to high location error; Location accuracy for the voting scheme can be improved by increasing the grid density, which leads to a quadratic increase of run time and memory use. The accuracy for our proposed gradient descent scheme can be controlled by the terminating

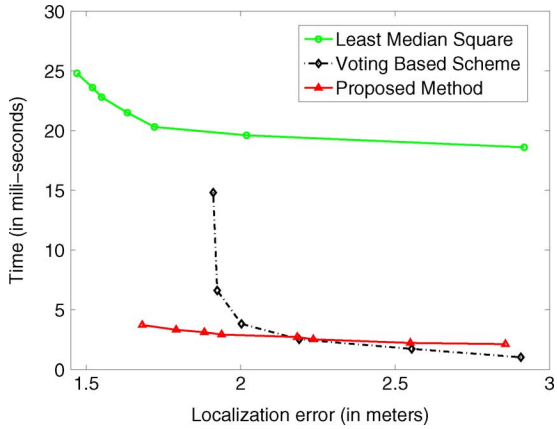

Fig. 3. Comparison of run-time for different localization schemes for a fixed localization error.

condition of the iteration, and the run time for higher location accuracy increases quite moderately and is considerably lower than the other two schemes.

D. Simulation Results

We experimentally compare our proposed algorithm with two existing secure localization methods, namely, the voting scheme and the LMdS algorithm. The simulation parameters are similar to those in [17] to allow for comparison of the results. Thirty anchor nodes are randomly deployed in an area of size 60 m × 60 m. The measurement noise standard deviation is set to be $\sigma = 2$ m. For the LMdS method, the number of subsets is set to be $M_1 = 20$ and the number of nodes in each subset is chosen to be $n = 4$. For the voting scheme, the region of deployment is divided into a square grid with each cell of size 1 m × 1 m, so that $n_1 = 60$. We use the algorithm described in [17] to find the votes for each cell. For the proposed algorithm, in the selection stage, we prune 50% of the force vectors with the largest magnitude and the number of iterations $M = 200$. The threshold for switching to selection stage is determined experimentally by varying its value between 0.01 and 0.1 and choosing the value that gives the best localization performance. For our simulations, we determine the threshold value to be 0.9. The results shown are obtained by averaging over 1500 runs of simulations.

We compare the performance of our proposed method when a variable step size and a fixed step size are used for the gradient descent based method, respectively. In the fixed step size version of the algorithm, $\delta(i) = 0.5$ for all iterations $i = 1, 2, \ldots, M$. For the variable step size algorithm, we adopt the following step size that is linearly decreasing:

$$\delta(i) = 15 - \frac{15(i-1)}{M}. \tag{5}$$

1) Noncoordinated Attacks: The localization accuracy achieved by various secure localization algorithms under noncoordinated attacks with different parameters is shown in Fig. 4. In particular, Fig. 4(a) shows the localization error as a function of the noise standard deviation σ_{attack} added by the malicious nodes when 30% of the nodes are compromised; and

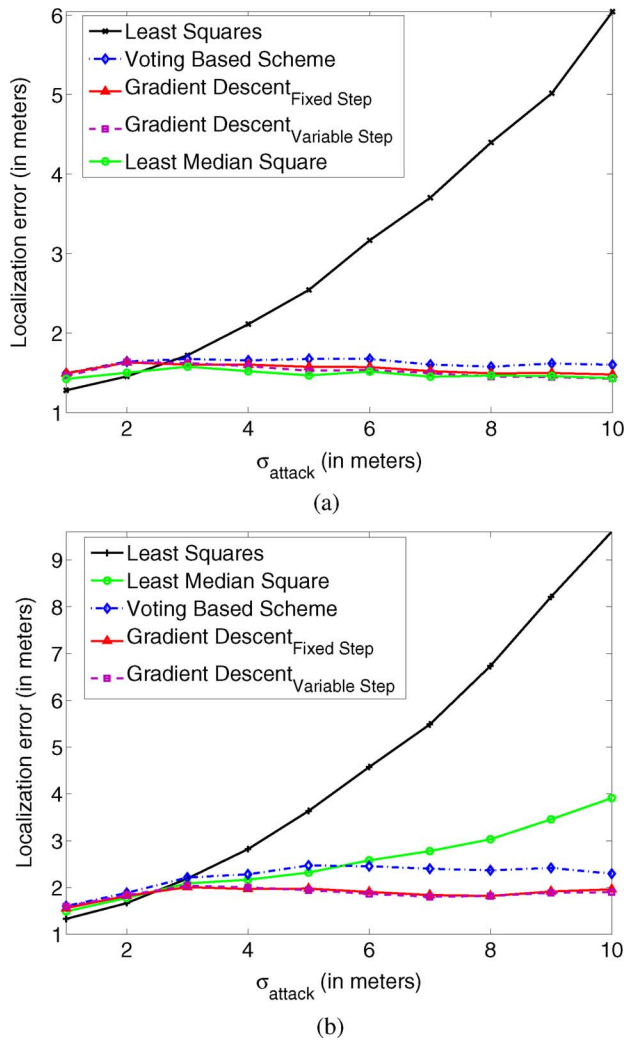

Fig. 4. Comparison of localization schemes for noncoordinated attacks. (a) 30% malicious nodes. (b) 60% malicious nodes.

Fig. 4(b) shows the corresponding results when 60% of the nodes are compromised.

From Fig. 4(a) we observe that the localization error using our method is comparable to the other schemes when the fraction of malicious nodes is less than 50%. For 60% malicious nodes, the LMdS method results in a localization error that increases with attack strength, as it cannot tolerate attacks by more than 50% of the nodes, but the proposed method can still localize the node with high accuracy. The independent distance random perturbations by the malicious nodes result in randomly oriented force vectors, which have a mutually canceling effect when summed up to compute the overall gradient. As a result, the proposed algorithm is robust against noncoordinated attacks, and the average localization error does not increase as the attack noise variance σ^2_{attack} increases. The voting-based scheme also gives good localization accuracy, but the error is slightly higher than the gradient descent method because of the discrete nature of the grid points. The localization accuracy in the voting based scheme can be increased by finely quantizing the grid points at a cost of higher computation and memory.

While the average localization error is a useful indicator of the accuracy of the algorithm, it may be dominated by the cases

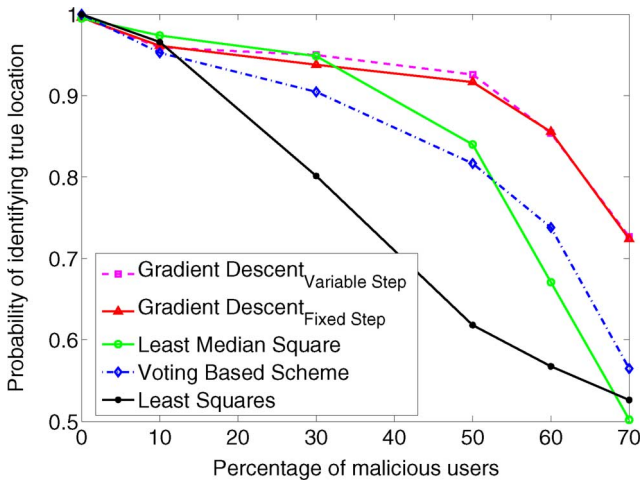

Fig. 5. Probability of converging to the correct estimate for different localization schemes under noncoordinated attacks for $\sigma_{\text{attack}} = 4$ m.

where the algorithm does not converge to the true position. To obtain a different perspective on the accuracy, we compare the probability that the algorithm correctly identifies the true position. Due to the presence of noise, finite grid size, and step size, we consider that the algorithm has converged to the correct location if the final estimate is within a distance of σ meters from the true location. Fig. 5 compares the probability of converging to the correct estimate as a function of the fraction of malicious nodes participating in the attack, for different localization schemes under noncoordinated attacks. We see that when the fraction of malicious nodes is less than 50%, all the secure localization algorithms except the simple LS method have similar performance and converge to the correct estimate about 90% of the time. However, if the fraction of nodes participating in the attack is more than 50%, the proposed scheme outperforms the existing algorithms. For example, when the fraction of attacking nodes is 60% or 70%, the gradient descent approach has approximately 10% higher probability of converging to the true position.

2) Coordinated Attacks: Fig. 6(a) shows the localization error under coordinated attack by 30% of the nodes. The x-axis represents the distance d_a between the true location of the sensor and the point $\mathbf{P_{mal}}$ chosen by the malicious nodes at random. From the figure, we observe that when the fraction of malicious nodes is 30%, the localization accuracy for all the methods except LS is almost the same. We obtained similar results when the fraction of malicious nodes is 35%. The localization error for the proposed gradient descent method is slightly higher than the other techniques under this setting. The reason for this behavior is that as the percentage of malicious nodes increases, even a few uncompromised anchor nodes whose distances from malicious position and true position are approximately the same can cause received data from anchor nodes to be more consistent with malicious positions. This phenomenon will be discussed in detail in Section II-E.

Fig. 6(b) compares the probability of converging to the true location for various algorithms under coordinated attacks when $d_a = 22$ m. From this figure, we see that all the secure localization schemes have similar performance and converge to the correct estimate about 90% of the time for coordinated attacks

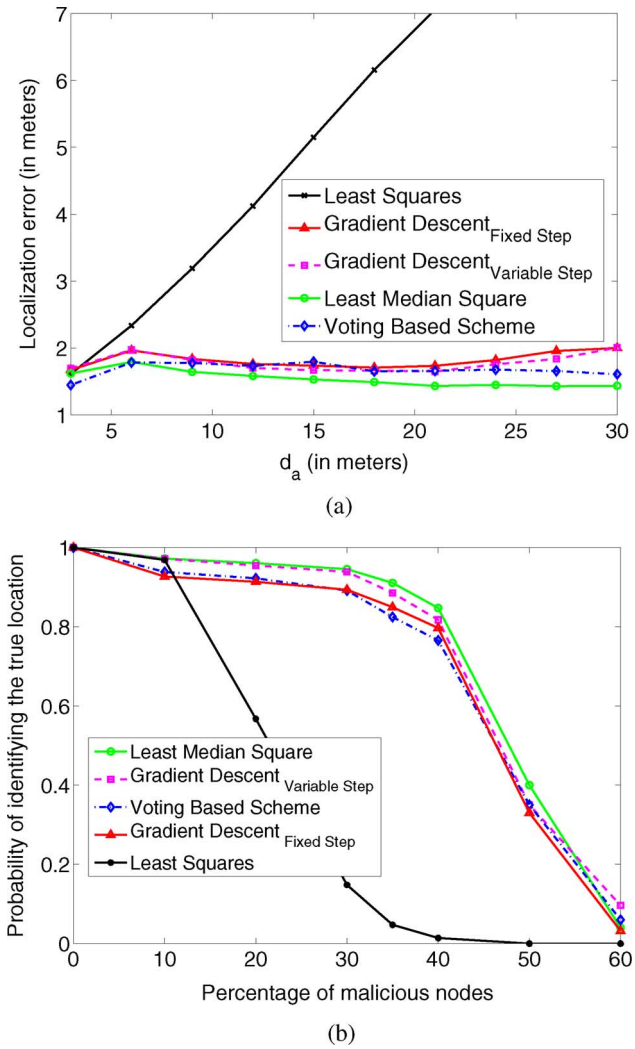

Fig. 6. Performance of the secure localization schemes under coordinated attacks by 30% of the nodes. (a) Average localization error and (b) probability of correctly identifying the true position for $d_a = 22$ m.

by less than 30% of the nodes. For attacks by a larger fraction of nodes, the probability of converging to the true position is slightly lower for the gradient descent algorithm with variable step size when compared to the LMdS algorithm. This is again due to the honest nodes that are at approximately the same distance from both the true position and the position reported by the malicious nodes.

Measurement noise also has a major impact on the performance of localization algorithms. In our simulations, we observed that the localization error is approximately same for a fixed σ, and increases linearly with an increase in σ for all three secure localization methods considered in this paper.

E. Discussions

In the first stage of our algorithm, we find the LS estimate of the location in an iterative manner. After convergence in the first stage, our algorithm switches to the second stage to prune outliers. Modeling the secure localization problem in such an iterative framework helps us see similarities between our proposed algorithm and the iterative LTS algorithm, and understand the robustness of our proposed algorithm. In particular, our pruning stage can be modeled similar to the iterative approach used to

solve the LTS problem proposed in the literature for robust estimation of the parameters of observations containing outliers [27].

To demonstrate this similarity, each term inside the summation in (3) can be considered as the residual error in estimating the true location $\mathbf{P}$. We can rewrite (3) in the following form:

$$\hat{\mathbf{P}} = \arg\min_{\mathbf{P}} \sum_{k=1}^{N} r_k^2 \tag{6}$$

where $r_k = (\|\mathbf{P_k} - \mathbf{P}\| - d_k)$ denotes the residual in estimating the true location. Let $(r^2)_{1:n} \leq (r^2)_{2:n} \leq \cdots \leq (r^2)_{n:n}$ be the ordered squared residuals of the set $\{r_1, r_2, \ldots, r_n\}$. The LTS method seeks to minimize the following cost function:

$$\sum_{i=1}^{h} (r^2)_{i:n} \tag{7}$$

where h is the number of residues used to evaluate the LTS cost function. An efficient iterative method to solve the LTS problem was proposed in [27]. In this iterative approach, parameters estimated in the $(i-1)$th iteration are used to calculate the residues, r_k's in the ith iteration. The residues are then arranged in an ascending order of their magnitudes and an estimate of parameters is obtained for the ith iteration by finding the LS estimate using the h smallest residue points. Our proposed method is similar to this general statistics approach, and the magnitude of our force vectors, $\mathbf{g_k}(i), i = 1, 2, \ldots, n$, reflect the magnitude of the residues, $r_k, k = 1, 2, \ldots, n$. However, in the iterative LTS method, an LS estimate is obtained at each iteration, which requires rather high computation power. Instead of following the conventional iterative LTS method [27], we iteratively update the location estimate by a step size in the direction of the decreasing cost function. We see from experiments that our algorithm converges even after relaxing the parameter update criteria of the conventional iterative LTS algorithm.

The breakdown point of the iterative LTS algorithm, i.e., the number of outliers that the algorithm is guaranteed to tolerate, is shown in the literature to be approximately 50%, which is the same as that of the LMdS method. As the proposed algorithm shares the spirit of the iterative LTS algorithm, the proposed algorithm has the same breakdown point. However, in noncoordinated attack case, our algorithm can tolerate more than 50% malicious nodes because of the statistically canceling nature of the attacks as discussed in Section II-D1. In coordinated attacks, depending on the topology of the sensor nodes, the breakdown point can be slightly less than 50% as discussed next.

Fundamentally under coordinated attacks, it is impossible for any scheme to perform secure localization using only the location and the distance information if the number of coordinating malicious nodes is more than the number of honest nodes. In these cases, there are more consistent equations satisfied by the location $(\mathbf{P_{mal}})$ reported by the malicious nodes than those satisfied by the true node location $(\mathbf{P})$. Hence, without any additional information to authenticate, it is impossible to distinguish between these two locations, and robustness against coordinated attacks should be focused on the situation when less than 50% of the nodes are compromised. This scenario was analyzed in [32] and it was shown that if the total number of nodes, N, in the network is more than $2L + 2$, where L is the number of malicious nodes, then the position of the localizing node can be computed with a bounded error.

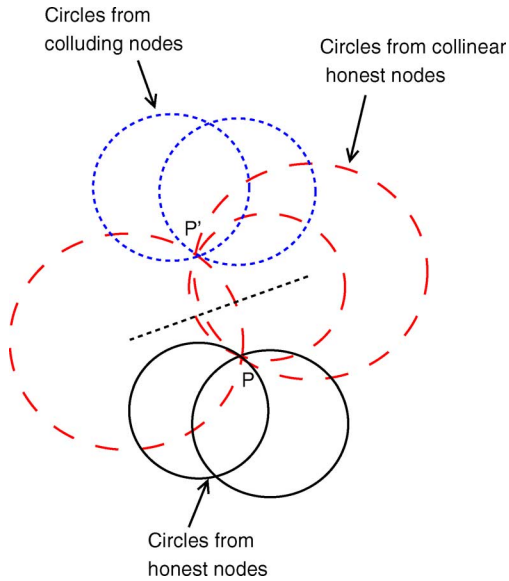

Fig. 7. Geometry for the bound on the maximum number of colluding nodes.

In practical scenarios, adversaries can create more consistent measurements for the malicious location even when they are unable to compromise a majority of the nodes, by carefully choosing the distance measurement to report, as shown in Fig. 7. In this figure, $\mathbf{P}$ denotes the location of the localizing node, while $\mathbf{P'}$ denotes the position chosen by malicious nodes to shift the estimate. Each circle is drawn with position reported by anchor nodes as center and distance measured by localizing node from corresponding anchor node as radius. The circles intersecting at both locations $\mathbf{P}$ and $\mathbf{P'}$ correspond to the collinear anchor nodes as centers and are shown using the dashed line. The colluding nodes can take advantage of the fact that the measurements reported by the collinear honest nodes are also consistent with their second point of intersection $\mathbf{P'}$. When nodes are randomly deployed, the probability that three or more nodes are exactly collinear is negligible. As such, in the absence of measurement noise, we then require that $N > 2L+2$ for secure localization. In practice, however, the presence of measurement noise requires a localization algorithm to equip with some tolerance capability. Therefore, being merely close to collinear in the noisy case can have the inevitable effect to aid the adversary in the same way as what the exact collinear situation does for the noise-free case. Since the probability of nodes being close to collinear is nontrivial, the required lower bound on N is considerably larger than $2L+2$. In our experiments, we have observed that such occurrences of almost collinear nodes with an intersection point close to $\mathbf{P_{mal}}$ account for a large fraction of the cases where the secure localization algorithms do not correctly identify the true position of the node. Because of increase in the probability of nodes that are collinear, the proposed scheme can tolerate fewer malicious nodes—about 40% in our study as shown in Fig. 6(b), for coordinated attack case. Beyond that, the probability of correctly identifying the location drops sharply.

In LMdS, localization is performed using multiple subsets of four nodes and the estimated location will be correct as long

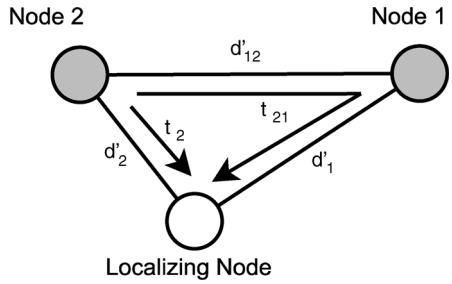

Fig. 8. Diagram representing the basic TDoA protocol.

as one of these M_1 subsets contains all innocent nodes. Therefore, the LMdS algorithm performs moderately better than the gradient descent and voting algorithms when the percentage of malicious nodes is higher, although this gain is achieved at the cost of much higher computational resources. The difference in localization error between the voting-based scheme and our scheme can be attributed to resolution accuracy of grid used in voting scheme and to the step size in the gradient descent based scheme. Overall, we see a tradeoff between the localization accuracy and resource use. Our gradient descent based algorithm requires a significantly less amount of computational and memory resources, at a cost of slightly higher localization error for coordinated attacks launched by a high percentage of colluding malicious nodes.

III. Gradient Descent Approach Applied to TDoA Measurements

In the previous section, we showed that the proposed gradient descent algorithm with selective pruning can be used to securely localize nodes in hostile scenarios. We assumed that a direct measurement of the distance between the localizing node and the anchor nodes is available. This distance measurement may be obtained using ToA of the beacon signals, and requires synchronization between the transmitter and the receiver. A small synchronization error can cause a large error in the spatial localization as time is multiplied by the speed of light or sound. Time difference of arrival (TDoA) is used as one way to mitigate these synchronization issues [33], [34].

The setting for obtaining one TDoA measurement is shown in Fig. 8. In this example, the localizing node wants to obtain a TDoA measurement with the help of anchor nodes 1 and 2, which already know their position. Node 2 transmits its position coordinates and a timestamp to node 1 and the localizing node. Node 1 receives the signal from node 2 and forwards it to the localizing node after including node 1's own position coordinates. The forwarding delay in this process is assumed to be known in advance, as it depends on the processing speed at the node and may be known *a priori*. In order to take account of possible minor variations, we model the forwarding delay to be normally distributed around a known mean value. In practical applications, additional delays associated with queueing may be introduced depending on the routing protocols. These additional delays can be taken into account by incorporating the queueing delay distribution into the cost function. The localizing node receives the signal from nodes 1 and 2 and finds the difference in the time of arrival of the signal after subtracting the known mean value of processing time at the forwarding node.

Let the positions of nodes 1, 2, and the localizing node be $\mathbf{P}_1$, $\mathbf{P}_2$, and $\mathbf{P}$, respectively. Denote the distances between the nodes by d'_{12}, d'_1, and d'_2, respectively, as shown in Fig. 8. Let the time at which the node 2 transmits its position coordinates to localizing node and node 1 be t_0. Localizing node receives the signal transmitted directly from node 2 at time t_2 and the forwarded signal through node 1 at time t_{21}. Then we have

$$\begin{aligned} d'_2 &= c(t_2 - t_0) \\ d'_{12} + d'_1 &= c\,(t_{21} - t_0) \\ \Rightarrow d'_1 - d'_2 &= c(t_{21} - t_2) - d'_{12} \triangleq \Delta_{21} \end{aligned} \tag{8}$$

where c is the speed of the signal in the medium. This can be the speed of light for radio signals or the speed of sound for ultrasonic signals. Thus, given the time difference of arrival $t_{21} - t_2$, the localizing node position lies on a hyperbola with foci at $\mathbf{P}_1$ and $\mathbf{P}_2$.

A. Secure Localization Problem for TDoA

With this background on TDoA, we can now set up the secure localization problem when TDoA measurements are available. Suppose that we have N anchor nodes with known position coordinates, out of which L nodes are malicious and launch coordinated attacks. We need to determine the position of an unknown node using the time difference of arrival method to estimate the distance between the localizing node and the anchor nodes. Each pair of anchor nodes gives rise to one equation of a hyperbola. If we assume that every pair of anchor nodes is used to obtain one TDoA measurement, we have $\binom{N}{2}$ measurements to determine the position of the localizing node, which increases as $O(N^2)$. In practical resource constrained networks, obtaining such a large number of measurements and solving the corresponding equations can consume a lot of resources.

To simplify the problem, we assume that there is one tamper-proof trusted anchor node in the network (say node 1) that will be used to help localize the other nodes. Each of the remaining anchor nodes transmit the beacon signal with the timestamp to the localizing node. Upon receiving the signal, the trusted node 1 forwards it to the localizing node, which thus obtains one TDoA measurement. Under this assumption, we have a manageable number of $N-1$ equations for the node location, which may be solved in practical resource constrained networks. Let $\mathbf{P}_k$ denote the location of the kth anchor node. We need to determine the point $\mathbf{P}$ that satisfies

$$\|\mathbf{P} - \mathbf{P_1}\| - \|\mathbf{P} - \mathbf{P_k}\| = \Delta_{k1}, \quad k = 2, 3, \ldots, N \tag{9}$$

where Δ_{k1} is the TDoA measurement obtained through anchor node k and the trusted node 1 as described in (8). Fig. 9 shows an example where four anchor nodes are located at $\mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3$, and $\mathbf{P}_4$, and there are no malicious nodes or measurement noise. The hyperbolas correspond to the loci of points that are consistent with one TDoA measurement. The common intersection of the

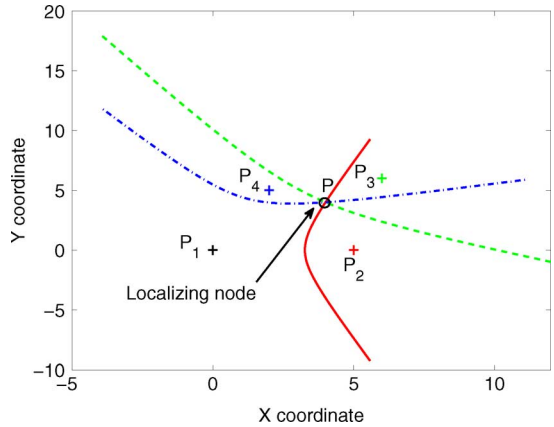

Fig. 9. Intersection of three hyperbolas gives the location of a node when TDoA is used.

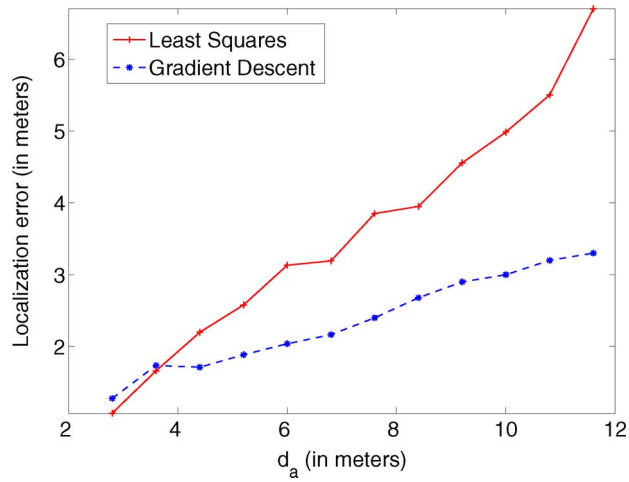

Fig. 10. Localization accuracy for coordinated attacks by 30% of the nodes using TDoA measurements.

hyperbolas, denoted by point $\mathbf{P}$, is the position of the localizing node.

In the presence of measurement noise alone, (9) can be solved in the least squares sense by finding the solution to the following LS equation:

$$\hat{\mathbf{P}} = \arg\min_{\mathbf{P}} \sum_{k=2}^{N} (\|\mathbf{P}-\mathbf{P_1}\| - \|\mathbf{P}-\mathbf{P_k}\| - \Delta_{k1})^2 = \arg\min_{\mathbf{P}} f_{td}(\mathbf{P}). \quad (10)$$

In the presence of malicious nodes, this LS solution may not be accurate. The malicious node may collude together to prevent the accurate localization of other nodes. Based on our assumption that a tamper-proof trusted node is used to help localize the node, the attacker cannot successfully launch an attack by modifying the timestamp alone. Any changes in the timestamp corresponding to the time of transmission will not affect the distance measurement, which only depends on the difference in the time of arrival of the two signals. Instead, the strategy of attacker will be to modify the transmitted position coordinates of kth node to $\mathbf{P'_k}$ in an intelligent way. Suppose that the attacker knows the position $\mathbf{P_1}$ of the trusted node 1 and the position $\mathbf{P}$ of the localizing node. Based on this knowledge, the attacker can estimate the t_k and t_{k1}-time instants at which the localizing node receives the direct beacon signal from kth node and forwarded the beacon signal from node 1. Denote by $\mathbf{P_{mal}}$ the position where the attacker wants to shift the estimate. We then have the following relations:

$$ct_k = \|\mathbf{P_{mal}} - \mathbf{P'_k}\| \quad (11)$$

$$ct_{k1} = d'_{1k} + \|\mathbf{P_{mal}} - \mathbf{P_1}\| \quad (12)$$

where $d'_{1k} = \|\mathbf{P}_1 - \mathbf{P}'_k\|$. The attackers can determine a suitable value of P_{mal} and P'_k for the nodes that are compromised such that (10) and (11) are satisfied.

Our gradient descent based approach with selective pruning described in the previous section can be extended to perform secure localization in this case. The algorithm starts by randomly initializing the LS estimate $\hat{\mathbf{P}}(0)$. At the ith step of the iteration, the gradient of the cost function $f_{td}(\mathbf{P})$ is evaluated at the current estimate $\hat{\mathbf{P}}(i-1)$, and the estimate is updated by moving it one step in the direction of the negative of the gradient, denoted by $\mathbf{g_{td}}(i)$:

$$\mathbf{g_{td}}(i) = -\nabla_{\mathbf{P}} f_{td}(\mathbf{P})|_{\mathbf{P}=\hat{\mathbf{P}}(i-1)}.$$

The geometric interpretation of the gradient descent for TDoA is similar to the previous case for ToA, where at each iteration, a step in the direction of the negative of the gradient moves the current estimate of the location towards the intersection of the hyperbolas. At every iteration, we compute the gradient corresponding to each term in (10) and then sum them up to find the overall gradient. In the pruning stage, we discard a fraction of the terms with large gradient magnitude and sum up the remaining terms to obtain the gradient direction.

We perform simulations using the same settings as before. A total of $N = 30$ nodes are distributed uniformly in a grid of size 60 m $\times$ 60 m. The measurement noise is assumed to be Gaussian with zero mean and $\sigma = 2$ m. Fig. 10 shows the localization error under coordinated attacks by 30% of the nodes. The x-axis represents the distance d_a between the position reported by the malicious nodes and the true location. The dashed line represents the localization accuracy using the proposed method, while the solid line represents the localization accuracy using the least squares solution. From the figure, we see that the localization error using our gradient descent algorithm is less than the error obtained using the least square method under coordinated attacks, although the localization error increases with an increase in attack distance, d_a, for both approaches. The reason for this behavior can be explained by examining the probability of identifying the true location as a function of the percentage of malicious nodes. From the results shown in Fig. 11 for $d_a = 22$ m, we see that the probability of converging to the true estimate under TDoA for our scheme is approximately 10% lower than that of the ToA case of Fig. 6(b). When the algorithm does not converge to the correct estimate, it converges close to the position reported by the malicious nodes, which corresponds to local minimum of the cost function and incurs an error that grows with the strength of the attack. As a result, the average error increases as the distance of attack, d_a, increases. However, as compared to using the baseline LS solution, the probability of

converging to the correct position is 20%–30% higher for the gradient descent algorithm.

To summarize, in this section and the previous section, we have described a computationally efficient method for localization in static wireless sensor networks in adversarial scenarios when the distance measurements are obtained using different techniques such as ToA and TDoA. The localization accuracy of the proposed method is better than or comparable to that of existing algorithms for secure localization in static networks. In the next section, we consider the case of mobile sensor networks and describe how the gradient descent algorithm can be used for secure localization in such a case.

IV. Secure Localization for Mobile Sensor Networks

The localization problem in mobile sensor networks (MSNs), where the individual nodes are moving, involves determining the location of each node at a series of time instants. Prior work that addresses the problem of localization in mobile networks was reviewed in Section I-A. These prior works assume the presence of anchor nodes in the network that are used to localize the mobile nodes. In contrast, we consider in this section a more challenging case where all the nodes are moving and the network may not have any anchor nodes. Our proposed algorithm operates in a fully distributed manner so that each node can localize itself, without the need for centralized processing.

Many applications involving mobile sensors rely on the relative positions of the nodes. The knowledge of the absolute locations may not always be critical for the functioning of the network. Instead, a map of relative locations that preserves the distances and neighborhood relations between the nodes is usually sufficient. Examples include distributed control algorithms such as leader-following [9], and direction-based routing algorithms [35], [36]. Even in applications where the absolute locations of the nodes need to be determined, the relative map can be used as an intermediate step in the localization process. As it preserves pairwise distances, the set of relative locations is only a possible rotation and translation of the absolute locations. Once the set of relative positions is obtained, the rotation and translation parameters can be determined with the help of the absolute known positions of any three nodes, and thus the absolute locations of the remaining nodes [30] are obtained. In this paper, we only consider the problem of determining the relative locations of the nodes in a mobile sensor network at each time instant. To the best of our knowledge, this is the first work addressing the secure localization of mobile sensor networks in the absence of anchor nodes.

A. Problem Formulation

Denote the location of the ith node in the network at time instant t by $\mathbf{P}_i(t)$, and let $\mathbf{S}(t) = \{\mathbf{P}_1(t), \mathbf{P}_2(t), \ldots, \mathbf{P}_N(t)\}$ be the set of node positions. Let $d_{ij}(t) = \|\mathbf{P}_i(t) - \mathbf{P}_j(t)\|, j \neq i$ be the distance between nodes i and j at time t. At a given instant, each node i obtains an estimate of $d_{ij}(t)$, $j = \{1, 2, \ldots, N\}/i$, using ToA or other distance estimation methods along with the current estimate of node j's location. The problem of estimating the relative location map at time instant t involves finding a set of location estimates $\hat{\mathbf{S}}(t) = \{\hat{\mathbf{P}}_1(t), \hat{\mathbf{P}}_2(t), \ldots, \hat{\mathbf{P}}_N(t)\}$ such that the inter-node distances $\hat{d}_{ij}(t) = \|\hat{\mathbf{P}}_i(t) - \hat{\mathbf{P}}_j(t)\|$ are approximately the same as the true inter-node distances $d_{ij}(t)$. The process of obtaining such a relative map can be considered as an embedding of the locations of the nodes into two-dimensional space.

Multidimensional scaling (MDS) is a classic approach to embedding higher dimensional data into a lower dimensional space [37]. Given a matrix of dissimilarities or distance metrics between objects, MDS finds a set of locations in a specified high-dimensional space that best approximates the input distance matrix. MDS has been used in information visualization and data analysis as a dimensionality reduction technique in combination with other tools [38], and solve the localization problem in static sensor networks [30]. This approach has a high computational complexity due to the use of singular value decomposition (SVD) whose complexity is $O(N^3)$, where N is the number of nodes in the network. MDS algorithm requires centralized processing, as it needs knowledge of inter-node distances between all the nodes. We adapt the computationally efficient gradient descent approach described in previous sections to find a relative location map of the entire network in an iterative manner. To apply this algorithm in a distributed manner, each node needs to iteratively obtain and update the current estimates of the relative position of other nodes and its own distance from other nodes.

B. Attack Model

Malicious nodes independently falsify the timestamp of their signals to provide erroneous information to the other nodes. We model this scenario by adding a random value u_{ik} uniformly distributed in $(0, d_{\max}]$ and constant over time, to the distance estimate provided to the ith localizing node by the kth node. A similar attack model was used in [24] to model noncoordinated attacks in mobile sensor networks. The distance estimate obtained by localizing node i from node k at time instant t can then be written as

$$d_{ik}^{(nc)}(t) = \begin{cases} d_{ik}(t) + u_{ik} + n_{ik}(t), & \text{if node } k \text{ is malicious} \\ d_{ik}(t) + n_{ik}(t), & \text{otherwise} \end{cases}$$

where $d_{ik}(t)$ is the actual distance between the nodes i and k and $n_{ik}(t)$ is the measurement noise.

The above attack model corresponds to the noncoordinated attacks. Launching coordinated attacks in mobile sensor networks is not as straightforward as in static sensor networks. In order to successfully implement fully coordinated attacks to change the estimated position/path of the localizing node to a desired location/path chosen by an adversary, each malicious node needs to have an exact estimate of the position and the speed of all the remaining nodes at each time instant to consistently mislead them. If the position reported at the next time instant is not consistent with the motion of the localizing node, the attacks from malicious nodes can be easily detected by the localizing node using velocity constraints. As a result, an adversary needs significantly more information to successfully launch a coordinated attack in a mobile sensor network. We only examine noncoordinated attacks in the case of mobile sensor networks in this paper. The problem of launching smart attacks in mobile sensor networks will be considered in future work.

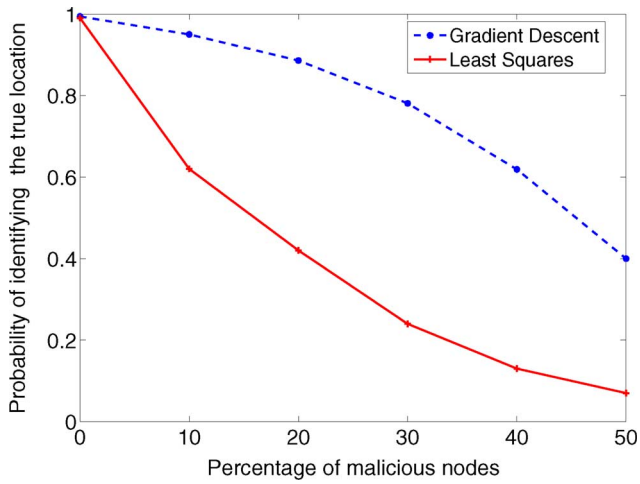

Fig. 11. Probability of converging to the correct estimate under coordinated attacks for TDoA, $d_a = 22$ m.

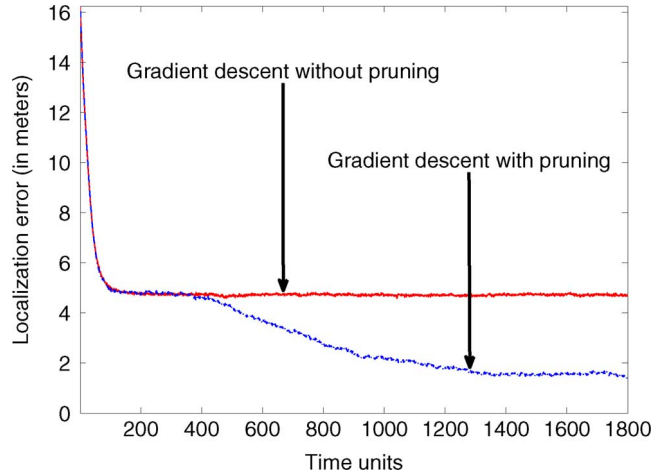

Fig. 12. Localization error, $E(t)$, as a function of time for estimating the relative locations in MSNs when 50% nodes are malicious.

C. Gradient Descent-Based Approach

For the case of localization in static networks discussed in Section II, the localizing node obtain a measurement of its distance from each of the anchor nodes. The gradient descent approach with selective pruning is then used to find the node position. In the case of mobile sensor networks without anchor nodes, as all the nodes are moving, each node obtains an estimate of its distance from every other node at each time instant. We modify the gradient descent algorithm to be applicable in this setting as described next.

Each node i randomly initializes its estimate for the current position $\hat{\mathbf{P}}_i(0)$. At each subsequent time instant t, the ith node obtains measurements $\{\hat{\mathbf{P}}_k(t-1), d_{ik}^{(nc)}(t)\}$ for $k = 1, 2, \ldots, N; i \neq k$ from the remaining nodes, and formulates a least squares problem similar to (1). The cost function for the ith node at time instant t is given by

$$f_i^{(t)}(\mathbf{P}_i(t)) = \sum_{k=1\ldots N, k\neq i} \left(\|\mathbf{P}_i(t) - \hat{\mathbf{P}}_k(t-1)\| - d_{ik}^{(nc)}(t) \right)^2. \tag{13}$$

Node i evaluates the gradient of the cost function in (13) at the estimate of its current position $\hat{\mathbf{P}}_i(t-1)$, and then updates the estimate by adding one step in the direction of the negative of the gradient

$$\mathbf{g}_i'(t) = -\nabla_{\mathbf{P}} \left(f_i^{(t)}(\mathbf{P}) \right) \Big|_{\mathbf{P}=\hat{\mathbf{P}}_i(t-1)}$$
$$\hat{\mathbf{P}}_i(t) = \hat{\mathbf{P}}_i(t-1) + \delta(t) \times \frac{\mathbf{g}_i'(t)}{\|\mathbf{g}_i'(t)\|}.$$

After the magnitude of the gradient falls below a threshold, the selection stage of the algorithm is activated by pruning a fraction of the terms in the cost function that have large gradient magnitudes. As we shall see next through simulations, the estimates of the relative positions converge to the correct solution, and we can accurately track the positions of the nodes as they move.

D. Simulation Results

In this subsection, we demonstrate experimentally the accuracy of the proposed method for localization in mobile sensor networks under noncoordinated attacks. Thirty sensors are randomly deployed in a 60 m $\times$ 60 m area. The velocity of the nodes at each instant is a random variable with x and y components V_x and V_y, uniformly distributed on $[0, V_{\max}]$. This mobility model is similar to the random way-point model used commonly for modeling mobile and ad-hoc networks [22], [39]. The measurement noise, $n_{ik}(t)$, is assumed to be additive Gaussian with mean 0 and $\sigma = 2$ m. The maximum error introduced by a malicious node into the distance measurements is $d_{\max} = 30$ m. In the selection stage of the gradient descent algorithm, we prune 50% of the force vectors as in the previous case of static sensor networks.

The estimation accuracy of the estimated relative location map is measured by comparing the actual inter-node distances $d_{ij}(t)$ with estimated inter-node distances $\hat{d}_{ij}(t)$, where $\hat{d}_{ij}(t) = \|\hat{\mathbf{P}}_i(t) - \hat{\mathbf{P}}_j(t)\|$. The localization error $E(t)$ is defined as the sum of the absolute difference between $d_{ij}(t)$ and corresponding $\hat{d}_{ij}$ at each time instant for all i and j:

$$E(t) = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} |d_{ij}(t) - \hat{d}_{ij}(t)|. \tag{14}$$

Low value of $E(t)$ implies that the algorithm can accurately estimate the inter-node distances and can provide a relative location map that satisfies the inter-node distance constraints. The estimated relative location map can then be used to find the absolute locations of all the nodes in the network if true locations of three nodes are known.

We first evaluate the accuracy of the gradient descent algorithm for a fixed maximum velocity, $V_{\max} = 1$ m per unit time. A constant step size of $\delta(t) = (V_{\max})/(\sqrt{2}) = (1)/(\sqrt{2})$ is used, which is approximately the average distance that a node can move in unit time. The plot of error $E(t)$ as a function of time when 50% of the nodes are malicious is shown in Fig. 12. The dashed line represents the error using the proposed gradient descent approach, while the solid line represents the error when

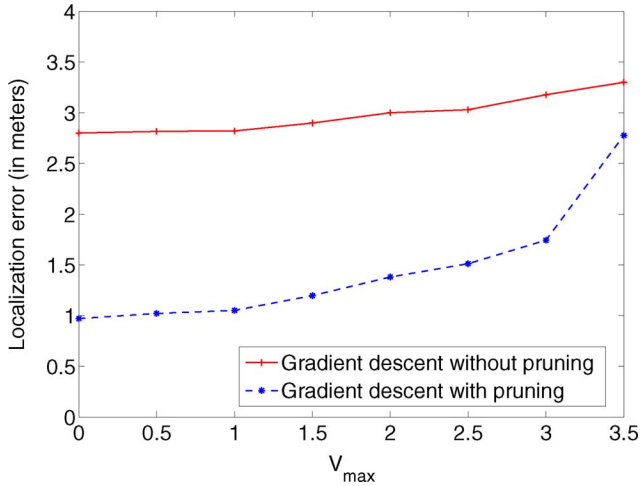

Fig. 13. Effect of velocity on the error in estimating the map of relative locations.

the selection stage of the algorithm is not used. The latter approach is basically the LS solution. We can see that the value of $E(t)$ is quite high during initialization of the algorithm as each node initializes its position estimate randomly. The localization error decreases during subsequent time instants as the algorithm iteratively updates the estimate of the position. Applying the second stage of the algorithm to prune out the observations due to malicious nodes further reduces the average error to less than 1.5 m.

We also examine the effect of the node velocity on the localization accuracy. We fix the value of $d_{\max}$ to 20 m and determine the error after convergence for different maximum velocities $V_{\max}$. Fig. 13 compares the localization accuracy of the gradient descent algorithm with and without pruning as a function of the velocity. The step size of the gradient descent algorithm is chosen to be $(V_{\max})/(\sqrt{2})$ as described previously. The point corresponding to $V_{\max} = 0$ denotes the special case of determining relative location map in the static network in the absence of any anchor node. A small step size is used to update the estimates at each iteration for the case of $V_{\max} = 0$. From this figure, we observe that as long as the velocity is small, the error in estimating the relative locations remains small. As the node velocity increases, the localization error also increases. The increase in localization error is more in the gradient descent approach with pruning than that for without pruning. At high velocities, each node can move quite far from its previous position and the gradient descent approach may not be able to track the position of node accurately. Applying multiple iterations in each time unit can alleviate this problem at the expense of increased computational complexity.

V. Conclusion

In this paper, we propose a secure and computational efficient algorithm for localization in wireless sensor networks. The proposed algorithm utilizes a gradient descent approach combined with a pruning stage that filters out inconsistent measurements to determine the location of nodes. We demonstrated the effectiveness of the algorithm when distance estimates between anchor nodes and non-anchor nodes are obtained using time of arrival and time difference of arrival measurements. Simulation results show that under coordinated attacks the proposed method has localization accuracy comparable to that of existing methods. For noncoordinated attacks, we showed an improvement by approximately 1 m in localization accuracy for a deployment region of size 60 m $\times$ 60 m, in the presence of Gaussian measurement noise when more than 50% nodes are compromised. Computation requirements and run time in the proposed method was shown to be lower than for existing methods. We also demonstrated that the algorithm can be used for localization in mobile sensor networks with malicious nodes to find the relative location map of each node in the network even in the absence of anchor nodes. The proposed method can track the mobile nodes with small localization error when nodes are moving slowly. The average localization error in the relative location map was less than 1.5 m for a deployment region of size 60 m $\times$ 60 m when up to 50% of the nodes are malicious, and nodes are moving with a maximum velocity of 3 meters per second.

References

[1] E. Cayirci, H. Tezcan, Y. Dogan, and V. Coskun, "Wireless sensor networks for underwater surveillance systems," *Ad Hoc Netw.*, vol. 4, no. 4, pp. 431–446, 2006.

[2] T. He, S. Krishnamurthy, J. A. Stankovic, T. Abdelzaher, L. Luo, R. Stoleru, T. Yan, L. Gu, J. Hui, and B. Krogh, "Energy-efficient surveillance system using wireless sensor networks," in *Proc. 2nd ACM Int. Conf. Mobile Syst., Applicat., Services (MobiSys)*, Boston, MA, 2004, pp. 270–283.

[3] R. Szewczyk, E. Osterweil, J. Polastre, M. Hamilton, A. Mainwaring, and D. Estrin, "Habitat monitoring with sensor networks," *Commun. ACM*, vol. 47, no. 6, pp. 34–40, Jun. 2004.

[4] L. Yu, N. Wang, and X. Meng, "Real-time forest fire detection with wireless sensor networks," in *Proc. Int. Conf. Wireless Commun., Netw., Mobile Comput.*, Maui, HI, Sep. 2005, vol. 2, pp. 1214–1217.

[5] E. A. Basha, S. Ravela, and D. Rus, "Model-based monitoring for early warning flood detection," in *Proc. 6th ACM Conf. Embedded Netw. Sens. Syst. (SenSys)*, Raleigh, NC, 2008, pp. 295–308.

[6] K. J. R. Liu, A. K. Sadek, W. Su, and A. Kwasinski, *Cooperative Communications and Networking*. Cambridge, U.K.: Cambridge Univ. Press, 2008.

[7] Y. B. Ko and N. H. Vaidya, "Location-aided routing (LAR) in mobile ad hoc networks," in *Proc. 4th Annu. ACM/IEEE Int. Conf. Mobile Comput. Netw.*, Dallas, TX, 1998, pp. 66–75.

[8] G. Wang, G. Cao, and T. F. L. Porta, "Movement assisted sensor deployment," *IEEE Trans. Mobile Comput.*, vol. 5, no. 6, pp. 640–652, Jun. 2006.

[9] N. Michael, M. Zavlanos, V. Kumar, and G. Pappas, "Distributed multi-robot task assignment and formation control," in *Proc. IEEE Int. Conf. Robot. Autom.*, Pasadena, CA, May 2008, pp. 128–133.

[10] P. Bahl and V. Padmanabhan, "Radar: An in-building RF-based user location and tracking system," in *Proc. 19 th Annu. Joint Conf. IEEE Comput. Commun. Soc. (INFOCOM)*, Tel Aviv, Israel, 2000, vol. 2, pp. 775–784.

[11] A. Savvides, C.-C. Han, and M. B. Strivastava, "Dynamic fine-grained localization in ad-hoc networks of sensors," in *Proc. 7th ACM Annu. Int. Conf. Mobile Comput. Netw. (MobiCom)*, Rome, Italy, 2001, pp. 166–179.

[12] D. Niculescu and B. Nath, "Ad hoc positioning system (APS) using AOA," in *Proc. 22nd Annu. Joint Conf. IEEE Comput. Commun. (INFOCOM)*, San Francisco, CA, Mar. 2003, vol. 3, pp. 1734–1743.

[13] D. Niculescu and B. Nath, "DV based positioning in ad-hoc networks," *Telecomm. Syst.*, vol. 22, no. 1–4, pp. 267–280, 2003.

[14] J. T. Chiang, J. J. Haas, and Y.-C. Hu, "Secure and precise location verification using distance bounding and simultaneous multilateration," in *Proc. 2nd ACM Conf. Wireless Netw. Security*, Zurich, Switzerland, 2009, pp. 181–192.

[15] S. Capkun, K. Rasmussen, M. Cagalj, and M. Srivastava, "Secure location verification with hidden and mobile base stations," *IEEE Trans. Mobile Comput.*, vol. 7, no. 4, pp. 470–483, Apr. 2008.

[16] S. Capkun and J.-P. Hubaux, "Secure positioning in wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 2, pp. 221–232, Feb. 2006.
[17] D. Liu, P. Ning, A. Liu, C. Wang, and W. K. Du, "Attack-resistant location estimation in wireless sensor networks," *ACM Trans. Inf. Syst. Security*, vol. 11, no. 4, pp. 1–39, 2008.
[18] L. Lazos and R. Poovendran, "SeRLoc: Secure range-independent localization for wireless sensor networks," in *Proc. 3rd ACM Workshop Wireless Security (WiSe)*, Philadelphia, PA, 2004, pp. 21–30.
[19] R. O. Duda and P. E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," *Commun. ACM*, vol. 15, no. 1, pp. 11–15, 1972.
[20] Z. Li, W. Trappe, Y. Zhang, and B. Nath, "Robust statistical methods for securing wireless localization in sensor networks," in *Proc. 4th Int. Symp. Inf. Process. Sens. Netw. (IPSN)*, Los Angeles, CA, 2005, p. 12.
[21] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.
[22] L. Hu and D. Evans, "Localization for mobile sensor networks," in *Proc. 10th ACM Annu. Int. Conf. Mobile Comput. Netw. (MobiCom)*, Philadelphia, PA, 2004, pp. 45–57.
[23] A. Baggio and K. Langendoen, "Monte-Carlo localization for mobile wireless sensor networks," in *Proc. Conf. Mobile Ad-Hoc Sens. Netw. (MSN)*, HongKong, 2006.
[24] Y. Zeng, J. Cao, J. Hong, S. Zhang, and L. Xie, "SecMCL: A secure monte carlo localization algorithm for mobile sensor networks," in *Proc. IEEE 6th Int. Conf. Mobile Adhoc Sens. Syst. (MASS)*, Macau, China, Oct. 2009, pp. 1054–1059.
[25] S. Misra, S. Bhardwaj, and X. Guoliang, "ROSETTA: Robust and secure mobile target tracking in a wireless ad hoc environment," in *Proc. IEEE Military Commun. Conf.*, Washington, DC, Oct. 2006, pp. 1–7.
[26] Y. Takizawa, P. Davis, M. Kawai, H. Iwai, A. Yamaguchi, and S. Obana, "Self-organizing location estimation method using received signal strength," *IEICE Trans. Commun.*, vol. B89-B, no. 10, pp. 2687–2695, 2006.
[27] P. Rousseeuw and K. Driessen, "Computing LTS regression for large data sets," *Data Mining Knowl. Disc.*, vol. 12, no. 1, pp. 29–45, 2006.
[28] P. J. Rousseeuw and A. M. Leroy, *Robust Regression and Outlier Detection*. New York: Wiley, 1987.
[29] R. Garg, A. L. Varna, and M. Wu, "Gradient descent approach for secure localization in resource constrained wireless sensor networks," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Dallas, TX, Mar. 2010, pp. 1854–1857.
[30] S. Yi, R. Wheeler, Y. Zhang, and M. Fromherz, "Localization from mere connectivity," in *Proc. ACM Int. Symp. Mobile Ad-Hoc Netw. Comput.*, 2003, pp. 201–212.
[31] R. Kwong and E. Johnston, "A variable step size LMS algorithm," *IEEE Trans. Signal Process.*, vol. 40, no. 7, pp. 1633–1642, Jul. 1992.
[32] S. Zhong, M. Jadliwala, S. Upadhyaya, and C. Qiao, "Towards a theory of robust localization against malicious beacon nodes," in *Proc. 27th IEEE Int. Conf. Comput. Commun. (INFOCOM)*, Los Angeles, CA, 2008, pp. 1391–1399.
[33] N. Patwari, "Location Estimation in Sensor Networks," Ph.D. dissertation, Univ. of Michigan, Ann Arbor, 2005.
[34] D. Munoz, F. Bouchereau, C. Vargas, and R. Enriquez, *Position Location Techniques and Applications*. Amsterdam, The Netherlands: Elsevier, 2009.
[35] M. Mauve, A. Widmer, and H. Hartenstein, "A survey on position-based routing in mobile ad hoc networks," *IEEE Network*, vol. 15, no. 6, pp. 30–39, Nov. 2001.
[36] S. Ramanathan and M. Steenstrup, "A survey of routing techniques for mobile communications networks," *Mobile Netw. Applicat.*, vol. 1, no. 2, pp. 89–104, 1996.
[37] J. B. Kruskal and M. Wish, *Multidimensional Scaling*. Thousand Oaks, CA: Sage, 1978.
[38] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, pp. 2319–2323, 2000.
[39] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," *Wireless Commun. Mobile Comput. (WCMC): Special Iss. Mobile Ad Hoc Network.: Res., Trends, Applicat.*, vol. 2, pp. 483–502, 2002.

Ravi Garg (S'10) received the B.Tech. degree in electrical engineering from the Indian Institute of Technology, Madras, in 2008. He is currently pursuing the Ph.D. degree in the Department of Electrical and Computer Engineering, University of Maryland, College Park.

His current research interests include wireless sensor networks and security, multimedia forensics, image processing, and speech processing.

Mr. Garg received the A. James Clark School of Engineering Distinguished Graduate Fellowship in 2008, and the Distinguished Teaching Assistant award in 2009, both from the University of Maryland. His first-author paper at the 2011 ACM Multimedia Conference won a Best Student Paper Award.

Avinash L. Varna (S'06–M'12) received the B.Tech. degree in electrical engineering from the Indian Institute of Technology, Madras, in 2005 and the Ph.D. degree in electrical engineering from the University of Maryland, College Park, in 2011.

He is currently with Intel, Chandler, AZ. His research interests include digital rights management, information forensics, and multimedia security.

Dr. Varna received a Silver Medal in the International Chemistry Olympiad 2001. He was awarded the Distinguished Dissertation Fellowship by the Department of Electrical and Computer Engineering and the Litton Fellowship for academic excellence by the University of Maryland. His coauthored paper at the 2011 ACM Multimedia Conference received a Best Student Paper Award.

Min Wu (S'95-M'01-SM'06-F'11) received the B.E. degree in electrical engineering and the B.A. degree in economics from Tsinghua University in Beijing, China (both with the highest honors), in 1996 and the Ph.D. degree in electrical engineering from Princeton University, Princeton, NJ, in 2001.

Since 2001, she has been with University of Maryland, College Park, where she is currently a Professor. She leads the Media and Security Team (MAST) at the University of Maryland, with main research interests on information security and forensics and multimedia signal processing. She has coauthored two books and holds eight U.S. patents on multimedia security and communications.

Dr. Wu is a corecipient of two Best Paper Awards from the IEEE Signal Processing Society and EURASIP, respectively. She also received a NSF CAREER award in 2002, a TR100 Young Innovator Award from the MIT Technology Review Magazine in 2004, an ONR Young Investigator Award in 2005, a Computer World 40 Under 40 IT Innovator Award in 2007, and an IEEE Mac Van Valkenburg Early Career Teaching Award in 2009. She has been elected to serve as Vice President—Finance of the IEEE Signal Processing Society (2010–2012) and Chair of the IEEE Technical Committee on Information Forensics and Security (2012–2013). She is an IEEE Fellow for contributions to multimedia security and forensics. [URL: http://www.ece.umd.edu/minwu/].