Distributed Mobility Management for Target Tracking in Mobile Sensor Networks

Yi Zou, *Member*, *IEEE*, and Krishnendu Chakrabarty, *Senior Member*, *IEEE*

Abstract—Mobility management is a major challenge in mobile ad hoc networks (MANETs) due in part to the dynamically changing network topologies. For mobile sensor networks that are deployed for surveillance applications, it is important to use a mobility management scheme that can empower nodes to make better decisions regarding their positions such that strategic tasks such as target tracking can benefit from node movement. In this paper, we describe a distributed mobility management scheme for mobile sensor networks. The proposed scheme considers node movement decisions as part of a distributed optimization problem which integrates mobility-enhanced improvement in the quality of target tracking data with the associated negative consequences of increased energy consumption due to locomotion, potential loss of network connectivity, and loss of sensing coverage.

Index Terms—Wireless mobile sensor networks, mobility management, target tracking, Bayesian, distributed system.

1 Introduction

Mobility management has long been recognized as a major challenge in mobile ad hoc networks (MANETs) [3], [35]. As discussed in [3], a MANET generally has the following characteristics:

1. new members can join and leave the network any time,
2. no base station is available to provide connectivity to backbone hosts or to other mobile hosts,
3. it is difficult to implement sophisticated scheme for handover and location management,
4. each node acts as a router, forwarding packets from others nodes, and
5. communication connectivity is usually "weak" in the sense that it is easily broken due to node movement.

We focus on the mobility management problem for mobile sensor networks in this paper. Mobility management in sensor networks is different from that in mobile ad hoc networks because the movement of sensor nodes here is not random; rather, the movement of sensor nodes is purposeful, e.g., to actively and better track an intruder. In such scenarios, it is important to have an efficient mobility management scheme to ensure that sensor node mobility is exploited in the best possible way, e.g., to improve the quality of target tracking. At the same time, the mobility management strategy should avoid inefficient usage of scarce resources, such as energy and network bandwidth. Furthermore, the mobility management scheme should also take into account the potential negative consequences of node movement, e.g., loss of area coverage, loss of connectivity, and degradation of network performance. In addition, node movement also involves locomotion energy and routing overhead, especially the need to reestablish routes. Therefore, a practical mobility management scheme should empower a node with the ability to determine whether it should move and where it should move to such that the movement can enhance tracking quality without depleting scarce resources or significantly compromising coverage and network connectivity.

Note that, for wireless sensor networks with scarce energy resources, it is not always favorable for nodes to move during field operation because the energy required for locomotion energy is often much higher than that for sensing and communication [20], [24]. However, as shown in [19], [26], when nodes can afford the energy cost associated with mobility, it is important to have a network management scheme that can make effective use of mobility to facilitate application objectives. For example, multiple mobile robots can be deployed in a battlefield for target tracking without human intervention [20]. These mobile robots can form an ad hoc sensor network for monitoring the region of interest. To ensure better tracking quality for a moving target, it is beneficial to dynamically move nodes to advantageous locations.

In this paper, we present an efficient mobility management scheme that can be implemented in a fully distributed manner. The proposed mobility management scheme is a general framework that incorporates both the positive and negative consequences of node movement; it allows a node to autonomously decide whether it should move and where it should move to. It is based on concepts taken from Bayesian estimation theory [2], [4], [32].

The rest of the paper is organized as follows: Section 2 presents background material and some related prior work. In Section 3, we formulate the problem of finding the best move for a node to improve the quality of target tracking. In Section 4, we estimate the negative consequences due node movement. Section 5 presents cost evaluation and selection

- *Y. Zou is with Unitrends, 9 Science Court, Suite 300, Columbia, SC 29203. E-mail: yzou@unitrends.com.*
- *K. Chakrabarty is with the Department of Electrical and Computer Engineering, Duke University, Durham, NC 27708. E-mail: krish@ee.duke.edu.*

Manuscript received 18 Jan. 2006; revised 29 June 2006; accepted 23 Oct. 2006; published online 7 Feb. 2007.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-0017-0106.
Digital Object Identifier no. 10.1109/TMC.2007.1005.

rules that allow a node to make an autonomous decision regarding its movement. Simulation results are presented in Section 6. Section 7 concludes the paper and presents future research directions.

2 Related Prior Work

Mobile ad hoc networks have received considerable attention in the literature [7]. Most existing methods for mobility management focus on communication issues arising from dynamically changing topologies due to node mobility [7], [13], [15], especially in personal communication services (PCS) such as cellular phone networks [25].

Research on centralized target tracking has been carried on for many years, originating from early work on target tracking by radar during World War II [9]. Common tracking techniques include Kalman filtering, Bayesian estimation methods, and their variants [4], [9], [12]. The unique constraints of wireless sensor networks, such as limited energy due to battery-based power supply, limited storage capacity for time-series data, scalability for network management, and distributed sensing and data processing, pose a number of new challenges for target tracking.

Recent research efforts on target tracking in wireless sensor networks [1] have focused on collaborative sensing [9], [32], energy-efficient routing and management [5], [8], [14], [28], [29], and sensor node deployment [26], [27], [33]. Collaborative sensing and signal processing provide raw sensory data from the low-level sensing units on sensor nodes. In many cases, cheap sensors such as omnidirectional acoustic sensors [9], [32] are used since alternatives such as CCD cameras generally require more resources for power, memory, bandwidth, and computation. Although the target information from a single node is generally limited, more useful information can be obtained through data exchange and aggregation between multiple nodes, based upon which higher-level strategic decisions can be made [16].

Routing in ad hoc sensor networks has received a lot attention and is considered a great challenge for ad hoc sensor networks [1]. Many efforts have been made to achieve energy-efficient routing in data aggregation, especially for target tracking applications. For a brief review, we discuss several typical examples as follows: The LEACH protocol [14] forms a clustered hierarchy in sensor networks, where the cluster head will be responsible for transmitting sensor data for its cluster members. The energy savings is achieved because the data is consolidated through such clusterization. SPAN [8] is another energy-efficient routing protocol where sensor nodes are selected to operate on off-duty and on-duty cycles for sensor nodes. By switching between off-duty and on-duty states, the energy level of all nodes are averaged, resulting in an extended sensor network lifetime. In [29], the authors introduced routing fidelity as a measure to evaluate the routing cost in the sense of energy consumption. This is used to adaptively tune the routing, resulting in energy consumption reduction. This, however, may not perform well due to its dependency on geographic information of the network. Another effort for energy-efficient routing is the rumor routing protocol proposed in [5], where routing is based on reaction to events in the network. This frees the routing protocol from depending on any geographic information of sensor nodes when a coordinate system is not available. Backbone-based routing using a connected-dominating-set (CDS) is proposed in [28]. After the backbone is established, routing and querying can be achieved via the backbone nodes, leaving none-backbone nodes in a energy-saving state.

Relatively less attention has been devoted to the problem of mobility management for mobile sensor networks. Obviously, the network topology changes when nodes move, and this change in topology affects both sensing coverage and communication connectivity. For static sensor networks, Shakkottai et al. [23] provide an elegant analysis of the interrelated problems of sensing coverage and communication connectivity. In mobile sensor networks, however, these issues are complicated due to the changing topology, typically resulting in a node being disconnected, the network becoming partitioned, or loss of sensing coverage in some regions. In addition, the mechanical energy consumption due to node mobility is generally higher than the energy required for sensing, communication, and computation [20], [24].

In [26], Wang et al. used limited mobility to achieve a better topology that considers both sensing and communication. The mobility investment is traded off with the improved topology, which subsequent operations can benefit from. The "dynamic enclose cell" routing protocol is introduced in [31] to reduce the overhead for routing due to increased complexity in mobile sensor networks. However, this work is focused on the adaptation of static network routing protocols rather than protocol design from the perspective of a mobile sensor network. The idea of "virtual coordinates" is proposed in [21], where virtual coordinates are based on node connectivity; this forms an abstract layer that existing geometry-based routing protocols can use. In [17], Jea et al. explored the use of the mobile element, i.e., data mule in the paper's context, to perform data collection. The control mobility is achieved by setting rules based on where the mobile element goes as well as how long it is expected to take. Recently, in [24], Rao and Kesidis have shown that purposeful mobility can be used to achieve energy savings for routing in the sense of an amortized cost measure. It is also shown in [19] that mobility is helpful for maintaining sensing coverage for both static and mobile targets, particularly when the number of nodes is not sufficient for covering the complete area. This idea is similar to repositioning schemes for sensor node deployment as proposed in [26], [33].

The above methods have shown that managed mobility leads to improved network topologies, which can in turn facilitate subsequent data collection and processing operations in sensor networks. The approach proposed in this paper differs from prior work in several aspects. First, we focus on mobility management for the specific objective of target tracking, which is a typical application scenario for mobile sensor networks. In such cases, e.g., mobile robots deployed in a battlefield environment, mobility is affordable from cost considerations, but nodes should be carefully controlled to improve tracking quality. Second, we present

an analysis that evaluates the risks of losing connectivity and sensing coverage from the perspective of a mobile sensor network with an inherently dynamic topology. We introduce a mobility management framework that unifies tracking quality, sensing coverage, network connectivity, and energy consumption. Finally, we present a distributed algorithm for implementing the proposed mobility management scheme.

3 Tracking Quality Improvement Due to Node Movement

To improve the quality of target tracking, a node can decide to move to another location at the next time instant. These locations are referred to as candidate locations. In the following discussion, we first describe the tracking problem based on standard target estimation theory [2], [4]. We then formulate the problem of selecting the best candidate location for a node in a fully distributed manner.

3.1 Preliminaries

The target state is denoted by $\mathbf{x}_t \in \mathbb{R}^{d_x}$, where t is a discrete time sequence. For example, for target tracking in a 2D field, we can define $\mathbf{x}_t$ as a column vector of $[x; y; x_a; y_a]$, where x and y are the target speeds and x_a and y_a are the corresponding accelerations in the X and Y coordinates. The parameter $\mathbf{x}_t$ is described by the system model as $\mathbf{x}_t = \mathbf{f}_t(\mathbf{x}_{t-1}, \mathbf{w}_{t-1})$, where $\mathbf{f}_t : \mathbb{R}^{d_x} \times \mathbb{R}^{d_w} \rightarrow \mathbb{R}^{d_x}$ and $\mathbf{w}_{t-1}$ represents i.i.d. process noise. The parameters d_x and d_w denote the dimensions of $\mathbf{x}_t$ and $\mathbf{w}_t$, respectively. Note that $\mathbf{x}_t$ is estimated recursively from sensor measurements given by the observation model, i.e., $\mathbf{z}_t = \mathbf{h}_t(\mathbf{x}_t, \mathbf{v}_t)$, where $\mathbf{h}_t : \mathbb{R}^{d_x} \times \mathbb{R}^{d_v} \rightarrow \mathbb{R}^{d_z}$ and $\mathbf{v}_t$ represents i.i.d. measurement noise. The parameters d_z and d_v denote the dimensions of $\mathbf{z}_t$ and $\mathbf{v}_t$, respectively. The statistics for both $\mathbf{w}_t$ and $\mathbf{v}_t$ are assumed to be known. In Bayesian estimation theory, $\mathbf{x}_t$ is estimated recursively by incorporating the new measurements to modify the prior and thereby obtain the posterior [2], [4], [32]. We denote the estimated target state at time t as $\hat{\mathbf{x}}_t$. The Bayesian estimation is given by

$$p(\hat{\mathbf{x}}_t|\hat{\mathbf{z}}_{1:t-1}) = \int p(\hat{\mathbf{x}}_t|\hat{\mathbf{x}}_{t-1}) p(\hat{\mathbf{x}}_{t-1}|\hat{\mathbf{z}}_{1:t-1}) d\hat{\mathbf{x}}_{t-1}, \quad (1)$$

$$p(\hat{\mathbf{x}}_t|\mathbf{z}_{1:t}) = \frac{p(\hat{\mathbf{z}}_t|\hat{\mathbf{x}}_t) p(\hat{\mathbf{x}}_t|\hat{\mathbf{z}}_{1:t-1})}{p(\hat{\mathbf{z}}_t|\hat{\mathbf{z}}_{1:t-1})}, \quad (2)$$

$$p(\hat{\mathbf{z}}_t|\hat{\mathbf{z}}_{1:t-1}) = \int p(\hat{\mathbf{z}}_t|\hat{\mathbf{x}}_t) p(\hat{\mathbf{x}}_t|\hat{\mathbf{z}}_{1:t-1}) d\hat{\mathbf{x}}_t, \quad (3)$$

where the likelihood is given by $p(\mathbf{z}_t|\hat{\mathbf{x}}_t)$. Equation (1) is the prediction step, (2) updates the prior using the new measurement $\mathbf{z}_t$ to obtain the posterior $p(\hat{\mathbf{x}}_t|\mathbf{z}_{1:t})$, and (3) is the normalizing factor. From these equations, we can see that sensor measurements must be forwarded to a processing center for data integration where the prior information is already available. In sensor networks, this can be implemented either in a centralized manner by designating one of the sensor nodes as the processing center [12], [16] or a cluster-based approach [32]. The first approach normally uses a node with more powerful computation capabilities for centralized processing; all other nodes forward their collected sensor data. The second approach depends on a dynamic clustering algorithm to select one of the nodes as the cluster head, i.e., the node that performs sensor fusion. When the cluster head is changed, usually in accordance with the estimated target track, it needs to pass the prior information to the next cluster head for continuous tracking [32].

The above scenario becomes more complicated when mobile sensor nodes need to make decisions locally about their movements to better track the target. Below, we list some challenges encountered in the mobility management for sensor nodes in such scenarios:

- Nodes must make decisions on where to move in a timely manner. Nodes may not be able to afford to wait for the posterior from the fully integrated sensor measurements on the processing center or the cluster head.
- In the prediction stage as shown in (1), $p(\hat{\mathbf{x}}_{t-1}|\hat{\mathbf{z}}_{1:t-1})$ is assumed to be known to the processing center or the cluster head from previous estimation at $t-1$. This implies that if each mobile node needs to make a decision on its movement based on the result from the complete sensor integration, this information must also be available. However, because mobility management should be autonomous as well as distributed, it is difficult to decide to which nodes this information should be forwarded. Furthermore, continuously forwarding the prior based on all nodes' measurements will cause considerable burden on the communication bandwidth and it will increase energy consumption.
- Node movements result in topology changes in the sensor network. This implies that the set of neighbor nodes for each node also changes.

To ensure that a node is able to make a local decision using only current local knowledge, we assume that every node has the capability to perform sensor integration locally. A node that has local sensor measurement exchanges sensor measurements within its own one-hop neighborhood. It also performs estimation using the Bayesian approach with possibly incomplete sensor measurements, i.e., $\mathbf{z}_t$ does not necessarily contain sensor measurements from all sensor nodes that have detected the target at time sequence t, but only nodes within the one-hop neighborhood. Note that sensor measurements can still be delivered to the designated processing server node for an estimation based on current complete sensor measurements.

Fig. 1 illustrates the proposed method. Consider a mobile sensor node s_i, as shown in Fig. 1a. In order to improve the quality of target tracking data, node s_i moves to a location that leads to improved sensor measurement. In other words, s_i will expect a higher signal-to-noise S/N ratio at its location at time instant $t+1$ compared to its sensor measurement at time instant t. Note that topology in a mobile sensor network is dynamic. Apart from the extra energy a node spends in movement, a node also faces risks of losing communication connectivity to its neighbors, as well as losing sensing coverage in certain areas. For the node s_i in this example, these are shown by Fig. 1b and Fig. 1c, respectively.

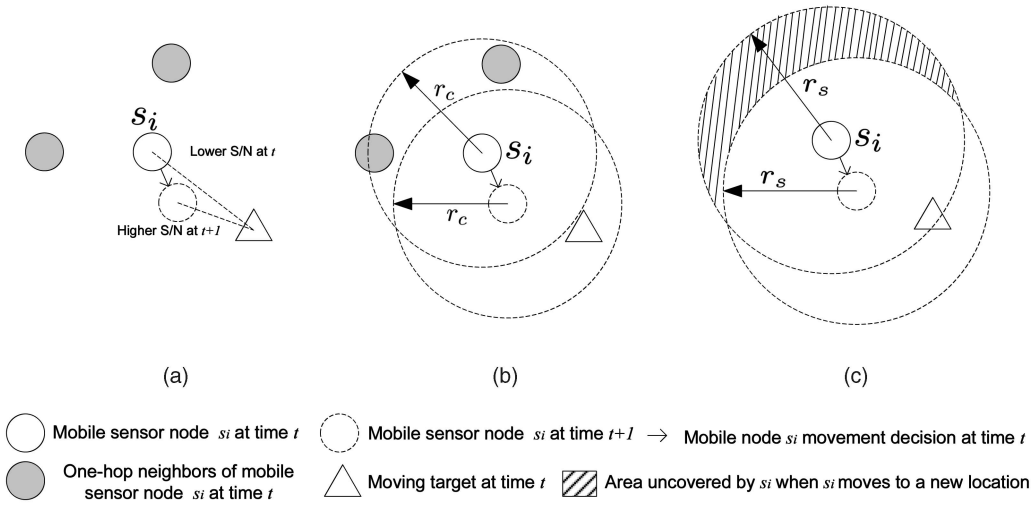

Fig. 1. An illustration of the proposed mobility management method. (a) To improve tracking quality, mobile node s_i chooses to move to a location that is expected to have a higher signal-to-noise ratio. (b) Movement of s_i may break the communication connectivity with its neighbor nodes. (c) Movement of s_i may cause some area under coverage at t to become uncovered at $t+1$.

The method we propose here is related to the Information Driven Sensor Query (IDSQ) method described in [32]. IDSQ is used to select the best sensor measurement from a set of fixed sensor nodes that have currently reported a target, i.e., sensor nodes that have new sensor measurements. The selection is based on the estimated information gain from the sensor measurement on the candidate sensor node, where the information gain is evaluated using well-defined rules.

In mobility management, the movement decision for a node is based on whether the new location will improve tracking quality. Since a node does not know *a priori* the quality of sensor measurements it will get at the new location, we first predict all possible sensor measurements corresponding to all possible candidate locations that the node might choose to move to. We then treat these predicted sensor measurements as true measurements, as if they were from nodes currently located at these candidate locations. Thus, the problem of making decisions on where to move is viewed as the problem of selecting one of the predicted measurements that are expected to best improve the quality of tracking data. In this sense, this problem is similar to the sensor selection strategy in [32]. However, in [32], the sensor measurements at time t are already available locally at those nodes to be selected, whereas, in our case, we focus on predicted measurements corresponding to candidate locations that a node has to decide to choose as its next location at time $t+1$. Moreover, [32] does not consider mobile sensor nodes.

3.2 Assumptions

To simplify the discussion, we make the following assumptions for the sensor network:

1. In this paper, we assume that both sensor nodes and the target are moving at constant speeds. This is justified since Bayesian estimation is not limited by this assumption.
2. We assume that the sampling interval of all sensor nodes is small enough such that there is no drastic change in sensor measurements of the target state.
3. All nodes have the same number of candidate locations where they can move. This is justified for a gridded sensing region.
4. Node s_i considers movement and carries out the evaluation process only if it detects a target.
5. A node uses the prior of its current location to predict the sensor measurements at its candidate locations.
6. A node uses the current sensor measurements from its current one-hop neighbor nodes.
7. When node s_i performs evaluation for movement decision at time instant t, we assume that the neighbor nodes of s_i at time instant $t+1$ are the same as at t.
8. When node s_i performs evaluation for movement decision at time instant t, we assume that s_i has collected sensor measurements from its one-hop neighbors that have also detected the target.
9. When node s_i performs evaluation for movement decision at time instant t, we assume that s_i has complete knowledge about the candidate locations of its neighboring nodes.
10. When node s_i performs evaluation for movement decision at time instant t, we only consider costs associated with locomotion, loss of communication connectivity, and loss of sensing coverage.

Consider a node s_i located at $\mathbf{l}_t^i$ at time t. The prior from previous estimation, denoted by $p(\hat{\mathbf{x}}_{t-1}|\hat{\mathbf{z}}_{1:t-1})$, is different for each sensor node since a node can start this process at any time when necessary and the sensor measurements are only from its one-hop neighborhood. This neighborhood might constantly change due to node movement. Let $\mathcal{N}_t^i$ be a function that maps from the discrete time t to a set of nodes that are one-hop neighbors of node s_i at t. We denote the sensor measurements available for s_i at time t as $\mathbf{z}_t^i$, i.e., $\mathbf{z}_t^i$ contains the measurement z_t^i from s_i and z_t^j from all nodes in $\mathcal{N}_t^i$. We use $\hat{\mathbf{x}}_t^i$ to denote the target estimate on node s_i at time t that is derived using $\mathbf{z}_t^i$. Fig. 2 illustrates

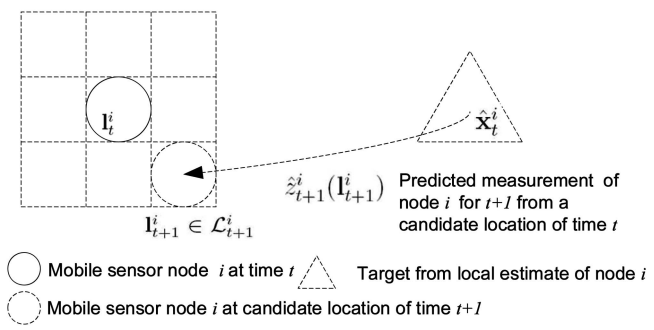

Fig. 2. Node s_i predicts its measurement at a candidate location based on its current target estimate.

how a node uses the predicted measurement to decide its next movement.

To simplify the discussion, we assume that there are only a limited number of locations that a node can move to from its current position; these are referred to as candidate locations. Candidate locations can be determined from the speed of the node and the sampling frequency of the sensor on the node. At time t, let $\mathcal{L}_{t+1}^i$ be the set of candidate locations for node s_i at time $t+1$, i.e., $\mathbf{l}_{t+1}^i \in \mathcal{L}_{t+1}^i$. Our goal is then to find the location $\mathbf{l}_{t+1}^i$ such that the tracking quality is best among all candidate locations. Note that we have $\mathbf{l}_t^i \in \mathcal{L}_{t+1}^i$ to include the case that a node may decide to stay in its current location. For a given grid point in the surveillance area, $\mathcal{L}_{t+1}^i$ can include locations that are just one step away from the current location, corresponding to due-east, north-east, due-north, north-west, due-west, south-west, due-south, south-east, and the current location, respectively. This is shown in Fig. 2 as the dotted squares where node s_i can move.

We further denote the predicted sensor measurement for s_i at $\mathbf{l}_{t+1}^i$ as $\hat{z}_{t+1}^i(\mathbf{l}_{t+1}^i)$. We use $\hat{\mathbf{z}}_{t+1}^i(\mathbf{l}_{t+1}^i)$ to denote the vector containing all predicted sensor measurements within a one-hop neighborhood of s_i when s_i is located at $\mathbf{l}_{t+1}^i$ at time $t+1$. Note that we use $\hat{\mathbf{x}}_t^i$ instead of $\hat{\mathbf{x}}_t$ due to the fact that s_i may not be able carry out target estimation to obtain $\hat{\mathbf{x}}_t$ based on all sensor measurements at time t; some nodes with the target data may not be within the one-hop neighborhood of s_i. Moreover, we are unable to use the target estimate at time $t+1$, i.e., $\hat{\mathbf{x}}_{t+1}^i$, since sensor measurements at $t+1$ are not yet available. This implies that the error in the predicted sensor measurement is expected to be large if there is a drastic change in the target state, e.g., a sudden acceleration of the target.

Note also that $\hat{\mathbf{z}}_{t+1}^i(\mathbf{l}_{t+1}^i)$ contains predicted sensor measurements from the neighbor nodes of s_i at $t+1$. However, for each candidate location $\mathbf{l}_{t+1}^i \in \mathcal{L}_{t+1}^i$, each neighbor node $s_j \in \mathcal{N}_t^i$ also has $|\mathcal{L}_{t+1}^j|$ candidate locations to choose from, which means that s_i has to calculate $|\mathcal{L}_{t+1}^j|$ predicted sensor measurements for each neighbor node $s_j \in \mathcal{N}_t^j$. This in turn implies that a complete computation on node s_i for all possible predicted sensor measurements requires a total of $|\mathcal{L}_{t+1}^i| \prod_{s_j \in \mathcal{N}_t^i} |\mathcal{L}_{t+1}^j|$ calculations. In this way, s_i is assumed to have complete knowledge of $\mathcal{L}_{t+1}^j$ for each $s_j \in \mathcal{N}_t^i$.

Furthermore, since neighbor nodes of s_i may also move, $\mathcal{N}_t^i$ and $\mathcal{N}_{t+1}^i$ are not necessarily the same. Since $\mathcal{N}_{t+1}^i$ is not yet available for s_i at t, we assume that $\mathcal{N}_{t+1}^i$ is the same as $\mathcal{N}_t^i$; the latter can be obtained by exchanging neighbor sensor measurements at current time instant t. We further simplify the discussion by assuming that s_i uses current sensor measurements from its current neighbor nodes in $\mathcal{N}_t^i$, i.e., for any two $\mathbf{l}_{t+1}^{i_1}, \mathbf{l}_{t+1}^{i_2} \in \mathcal{L}_{t+1}^i$, $\hat{\mathbf{z}}_{t+1}^i(\mathbf{l}_{t+1}^{i_1})$ and $\hat{\mathbf{z}}_{t+1}^i(\mathbf{l}_{t+1}^{i_2})$ only differ from each other in $\hat{z}_{t+1}^i(\mathbf{l}_{t+1}^{i_1})$ and $\hat{z}_{t+1}^i(\mathbf{l}_{t+1}^{i_2})$ and all other elements are the same as z_t^j, $\forall s_j \in \mathcal{N}_t^i$. These assumptions are valid because we assume that the interval between two consecutive discrete time instants is small; since the maximum displacement from node movement is also correspondingly small, we expect that there is no drastic change in sensor measurements.

Table 1 presents a list of the notation used throughout the paper.

3.3 Probability of Node Movement to a New Location

Fig. 3 illustrates how the predicted sensor measurement at candidate locations is used to obtain the predicted target estimate on the basis of Bayesian estimation, as described in Section 3.1. Next, we rewrite the Bayesian estimation equations introduced earlier to calculate the target estimate based on the predicted sensor measurements. The node can then make a decision on where to move, i.e., select the best $\mathbf{l}_{t+1}^i \in \mathcal{L}_{t+1}^i$, by evaluating the estimated improvement in the target estimate for the next time instant $t+1$. This is shown as follows:

$$p(\hat{\mathbf{x}}_{t+1}^i|\hat{\mathbf{z}}_{1:t}^i) = \int p(\hat{\mathbf{x}}_{t+1}^i|\hat{\mathbf{x}}_t^i) p(\hat{\mathbf{x}}_t^i|\hat{\mathbf{z}}_{1:t}^i) d\hat{\mathbf{x}}_t^i, \tag{4}$$

$$p(\hat{\mathbf{x}}_{t+1}^i|\hat{\mathbf{z}}_{1:t}^i, \hat{\mathbf{z}}_{t+1}^i(\mathbf{l}_{t+1}^i)) = \frac{p(\hat{\mathbf{z}}_{t+1}^i(\mathbf{l}_{t+1}^i)|\hat{\mathbf{x}}_{t+1}^i) p(\hat{\mathbf{x}}_{t+1}^i|\hat{\mathbf{z}}_{1:t}^i)}{p(\hat{\mathbf{z}}_{t+1}^i(\mathbf{l}_{t+1}^i)|\hat{\mathbf{z}}_{1:t}^i)}, \tag{5}$$

$$p(\hat{\mathbf{z}}_{t+1}^i(\mathbf{l}_{t+1}^i)|\hat{\mathbf{z}}_{1:t}^i) = \int p(\hat{\mathbf{z}}_{t+1}^i(\mathbf{l}_{t+1}^i)|\hat{\mathbf{x}}_{t+1}^i) p(\hat{\mathbf{x}}_{t+1}^i|\hat{\mathbf{z}}_{1:t}^i) d\hat{\mathbf{x}}_{t+1}^i, \tag{6}$$

where $\hat{\mathbf{z}}_{t+1}^i$ is the vector containing predicted sensor measurements at time $t+1$ from node s_i, $\mathbf{z}_{1:t}^i$ is the vector containing previous measurements of node s_i and its one-hop neighbors, $\hat{\mathbf{x}}_t^i$ is the previous target estimate for node s_i, $\hat{\mathbf{x}}_{t+1}^i$ is the target estimate based on the predicted measurements $\hat{\mathbf{z}}_{t+1}^i$ at time $t+1$ from node s_i, and $p(\hat{\mathbf{z}}_{t+1}^i(\mathbf{l}_{t+1}^i)|\hat{\mathbf{x}}_{t+1}^i)$ is the likelihood based on the predicted measurements. Note that the above equations represent the Bayesian estimation method based on both predicted sensor measurements $\hat{\mathbf{z}}_{t+1}^i$ from node s_i and possibly incomplete previous target estimation $\hat{\mathbf{x}}_t^i$ from node s_i. However, if there exists a central processing node, it can still eventually send the posterior $p(\hat{\mathbf{x}}_t|\mathbf{z}_{1:t})$ to node s_i to replace $\hat{\mathbf{x}}_t^i$ for improving the local target estimate. In this way, even though s_i is able to make its movement decision based on local knowledge, future decisions can be improved when the target estimate based on complete sensor measurements is available to it.

After we obtain the target estimate based on predicted sensor measurements, i.e., $p(\hat{\mathbf{x}}_{t+1}^i|\mathbf{z}_{1:t}^i, \hat{\mathbf{z}}_{t+1}^i(\mathbf{l}_{t+1}^i))$, we can use

TABLE 1
List of Notations Used Throughout the Paper

Notation	Description
n	Total number of sensor nodes
s_i	Sensor node i
r_c, r_s	Sensor node communication and sensing radius
X, Y	The x- and y-dimension of the 2D sensing grid
$g_k, \mathcal{G}$	Grid point with index k and the set of all grid points
$\mathbf{x}_t, \hat{\mathbf{x}}_t$	Target state vector and estimated target state vector at t
$\mathbf{z}_t$	Sensor measurement vector at t
$\mathbf{v}_t, \mathbf{w}_t$	Process and measurement noise vector at t
d_x, d_z, d_w, d_v	Dimensions of $\mathbf{x}_t, \mathbf{z}_t, \mathbf{w}_t, \mathbf{v}_t$
$z_t^i, \mathbf{z}_t^i$	Measurement from s_i and measurements available for s_i at t
m	Total number of candidate locations
$\mathbf{l}_t^i$	Candidate location vector for s_i at t
$\mathcal{L}_t^i$	The set of all candidate location vectors for s_i at t
$\mathcal{N}_t^i$	One-hop neighbors of node s_i at t
d_{t+1}^{ij}	The distance between node s_i and s_j
$\hat{z}_{t+1}^i(\mathbf{l}_{t+1}^i)$	The predicted measurement for s_i at $\mathbf{l}_{t+1}^i$
$\hat{\mathbf{z}}_{t+1}^i(\mathbf{l}_{t+1}^i)$	All predicted measurements for s_i at $\mathbf{l}_{t+1}^i$
ψ	Information utility function
δr_c	Communication radius threshold
c_k^i	Sensing coverage probability at g_k from s_i
$c_k^i(\mathbf{l}_t^i)$	Coverage probability of g_k from s_i at $\mathbf{l}_t^j$
S_k	The set of nodes that can detect g_k
c_k	The collective coverage probability of g_k from S_k
$\mathcal{S}_t^k$	The set of nodes that detect grid point g_k at time t
c_t^k	The collective coverage probability of g_k from $\mathcal{S}_t^k$
p_{th}	The required sensing coverage threshold
A_i	The sensing area of node s_i, $A_i \in \mathcal{G}$
$\mathcal{A}(\mathbf{l}_t^i)$	The sensing area of node s_i located at $\mathbf{l}_t^i$ at t

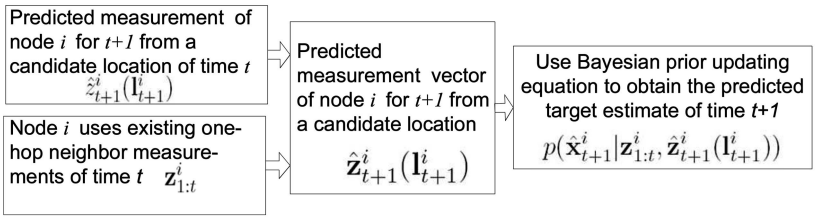

Fig. 3. An illustration of the use of the predicted measurement for updating the prior in Bayesian estimation.

similar rules as proposed in [32] to select the best predicted sensor measurements $\hat{\mathbf{z}}_{t+1}^i(\mathbf{l}_{t+1}^i)$, which subsequently gives us the best predicted sensor measurement $\hat{z}_{t+1}^i(\mathbf{l}_{t+1}^i)$ from node s_i. Thus, the corresponding candidate location $\mathbf{l}_{t+1}^i$ that is expected to best improve the tracking quality can be found from $\mathcal{L}_{t+1}^i$. In this paper, we use the definition of the information utility function ψ described in [32]. The parameter ψ is defined as $\psi : \mathcal{P}(\mathbb{R}^{d_l}) \rightarrow \mathbb{R}$, where d_l is the dimension of node location $\mathbf{l}_t^i$ and $\mathcal{P}(\mathbb{R}^{d_l})$ is a class of probability distributions. In our case, $\mathcal{P}(\mathbb{R}^{d_l})$ corresponds to all posteriors calculated from predicted sensor measurements from all candidate nodes locations at $t+1$ in $\mathcal{L}_{t+1}^i$. Since the output of the utility function ψ is a real number, ψ maps the posterior based on the predicted measurement from the candidate location to a real number representing the improvement in tracking quality from this candidate location. In standard estimation theory, the trace or determinant of the estimation error covariance matrix is commonly used as a measure of the tracking quality [4], [32]. Let $\hat{\mathbf{e}}_{t+1}^i(\mathbf{l}_{t+1}^i)$ be the target estimate error from local estimation on s_i based on predicted measurements and $\mathbf{R}_{t+1}^i(\mathbf{l}_{t+1}^i)$ be the corresponding covariance matrix of $\hat{\mathbf{e}}_{t+1}^i(\mathbf{l}_{t+1}^i)$, i.e., $\mathbf{R}_{t+1}^i(\mathbf{l}_{t+1}^i) = E\{(\hat{\mathbf{e}}_{t+1}^i) \cdot (\hat{\mathbf{e}}_{t+1}^i)'\}$, where $'$ denotes the transpose of a matrix. Suppose the posterior is Gaussian. One way of defining the utility function is given by $\psi(p(\hat{\mathbf{x}}_{t+1}^i|\hat{\mathbf{z}}_{t+1}^i(\mathbf{l}_{t+1}^i)) \equiv -\mathrm{trace}(\mathbf{R}_{t+1}^i(\mathbf{l}_{t+1}^i))$, where trace is the trace of a matrix. Some alternative approaches can also be used to evaluate the improvement in tracking quality [4], [32]. In general, the selection of a candidate location should maximize the utility function, i.e.,

$$\bar{\mathbf{l}}_{t+1}^i = \max_{\mathbf{l}_{t+1}^i \in \mathcal{L}_{t+1}^i} \psi(p(\hat{\mathbf{x}}_{t+1}^i|\hat{\mathbf{z}}_{t+1}^i(\mathbf{l}_{t+1}^i)). \quad (7)$$

Since $\psi(p(\hat{\mathbf{x}}^i_{t+1}|\hat{\mathbf{z}}^i_{t+1}(\mathbf{l}^i_{t+1}))$ is associated with the posterior based on the predicted sensor measurements on candidate locations, we can also use it to express the effectiveness of candidate locations. Thus, we can also express the probability of a node making a decision to move to $\mathbf{l}^i_{t+1}$ at $t+1$ as follows:

$$p(\mathbf{l}^i_{t+1}) \equiv \Pr(s_i \text{ at } \mathbf{l}^i_{t+1} \text{ at } t+1) = \frac{\psi(p(\hat{\mathbf{x}}^i_{t+1}|\hat{\mathbf{z}}^i_{t+1}(\mathbf{l}^i_{t+1}))}{\sum_{\mathbf{l}^i_{t+1}\in\mathcal{L}^i_{t+1}} \psi(p(\hat{\mathbf{x}}^i_{t+1}|\hat{\mathbf{z}}^i_{t+1}(\mathbf{l}^i_{t+1}))}, \tag{8}$$

where $p(\mathbf{l}^i_{t+1})$ represents the probability of node s_i being at $\mathbf{l}^i_{t+1}$ at $t+1$ to improve the quality of target tracking data. Note that, in (8), we assume that, by definition, $\psi(p(\hat{\mathbf{x}}^i_{t+1}|\hat{\mathbf{z}}^i_{t+1}(\mathbf{l}^i_{t+1}))$ yields a nonnegative real value. As shown in the next section, $p(\mathbf{l}^i_{t+1})$ is used to evaluate the integrated cost when other factors are also considered in the proposed mobility management scheme. From (8), the computational complexity for obtaining $p(\mathbf{l}^i_{t+1})$ depends on the complexity of the estimation algorithm as well as the size of set of candidate locations $\mathcal{L}^i_{t+1}$.

4 Estimation of Negative Consequences

As discussed in Section 1, we need to consider the negative consequences of node movement, e.g., additional energy consumption, connectivity loss, and coverage loss. Additional risks include the need for reestablishing the route, the potentially higher rate of node failures due to node movement. In this paper, we only focus on the energy, connectivity, and coverage issues, and leave the rest for future work. In this section, we derive the probabilities associated with the above-mentioned negative consequences when a node chooses to move to a candidate location. These probabilities are then used in an integrated cost evaluation described in the next section.

4.1 Energy Consumption

Obviously, nodes have to spend additional energy for movement. Even though sensor nodes on mobile platforms can carry more battery supplies, it is important to ensure that the available energy is properly used to best serve the purpose of surveillance tasks. We assume a simplified dynamics model for the sensor node movement; this model is similar to the one used in [24]. We assume that all nodes move at the same constant speed. We ignore the energy consumption for acceleration when the node starts to move as well as for deceleration when the node stops to move. We also assume that the node always moves along a straight line, i.e., the distance that a node has moved during the interval between two consecutive time instants is the distance between the old location and the new location of the node.

Consider an arbitrarily chosen node s_i located at $\mathbf{l}^i_t$ at time instant t. Let $\mathcal{E}^i_{t+1}(\mathbf{l}^i_{t+1})$ be a mapping from $\mathbf{l}^i_{t+1}$ to a real number representing the energy consumption on s_i when s_i decides to move to $\mathbf{l}^i_{t+1}$. We relate the energy consumption to the distance that the node has moved from time t to $t+1$ as follows:

$$\mathcal{E}^i_{t+1}(\mathbf{l}^i_{t+1}) = \theta\|\mathbf{l}^i_t - \mathbf{l}^i_{t+1}\|, \tag{9}$$

where θ is a constant in units of Joules per meter and $\|\mathbf{l}^i_t - \mathbf{l}^i_{t+1}\|$ is the euclidean distance between $\mathbf{l}^i_t$ and $\mathbf{l}^i_{t+1}$. Note that (9) indicates a linear relationship between the energy consumption and the distance moved because we consider a simplified energy consumption evaluation model in this paper.

Since s_i may have multiple candidate locations to choose from as its next location, we define $p^i_{t+1}(\mathrm{E}|\mathbf{l}^i_{t+1})$ as the weighted probability for node s_i to move to $\mathbf{l}^i_{t+1}$ at $t+1$, where the weight indicates the energy consumption associated with this movement. The probability $p^i_{t+1}(\mathrm{E}|\mathbf{l}^i_{t+1})$ is given by

$$p^i_{t+1}(\mathrm{E}|\mathbf{l}^i_{t+1}) = \frac{\bar{\mathcal{E}} - \mathcal{E}^i_{t+1}(\mathbf{l}^i_{t+1})}{\sum_{\mathbf{l}^i_{t+1}\in\mathcal{L}^i_{t+1}} (\bar{\mathcal{E}} - \mathcal{E}^i_{t+1}(\mathbf{l}^i_{t+1}))}, \tag{10}$$

where $\bar{\mathcal{E}}(\bar{\mathcal{E}} \geq \mathcal{E}^i_{t+1}(\mathbf{l}^i_{t+1}))$ is a known constant representing the maximum amount of energy that the node can afford for making the one-step movement. The parameter $\bar{\mathcal{E}}$ usually depends on the available battery and the operational lifetime requirement. Obviously, (10) yields the highest probability for a candidate location with the minimum amount of energy consumption.

4.2 Probability of a Node Being Disconnected

Once again, we consider node s_i. We focus on analyzing the risk that the node becomes disconnected due to its possible movement for the next time instant $t+1$. We assume that the network is connected at current time instant t. The current work that analyzes connectivity in wireless sensor networks deals only with the relationship between the node density and the probability of the network being connected or disconnected [18], [30]. While these results are useful in random sensor deployment, they cannot be directly applied to mobile sensor networks where the topology is dynamically changing. In this paper, to simplify the discussion, we only consider the probability that node s_i is disconnected from all other nodes at the next time instant, i.e., the probability that $\mathcal{N}^i_{t+1} = \phi$, where $\mathcal{N}^i_{t+1}$ is defined earlier as the set of neighbor nodes for s_i at time $t+1$. Let d^{ij}_{t+1} be the distance between node s_i and s_j at $t+1$ and let r_c be the communication radius for a node. Then the probability that s_i is disconnected at $t+1$ is given by

$$\Pr(\mathcal{N}^i_{t+1} = \phi) \equiv \Pr(d^{ij}_{t+1} > r_c, \forall s_j \in S \setminus \{s_i\}),$$

which requires the testing of connectivity from s_i to all other nodes in S since knowledge about $\mathcal{N}^i_{t+1}$ is generally not available until s_i has moved to the new location $\mathbf{l}^i_{t+1}$. To ensure that the proposed scheme is suitable for a distributed implementation, we restrict the calculation to the probability that s_i is disconnected from nodes only in the current neighbor set $\mathcal{N}^i_t$. Since it is possible that $\mathcal{N}^i_t \cap \mathcal{N}^i_{t+1} = \phi$ and $\mathcal{N}^i_{t+1} \setminus \mathcal{N}^i_t \neq \phi$, the above restriction is a conservation one in evaluating the probability of loss of connectivity. In this way, the calculation requires only current one-hop knowledge. Let $p^i_{t+1}(\mathrm{C}) \equiv \Pr(\mathcal{N}^i_{t+1} = \phi)$ be the probability that s_i is disconnected at time $t+1$. Then, $p^i_{t+1}(\mathrm{C})$ is given by

$$p^i_{t+1}(\mathrm{C}) \equiv \sum_{\mathbf{l}^i_{t+1}\in\mathcal{L}^i_{t+1}} \Pr(\mathcal{N}^i_{t+1}=\phi|\mathbf{l}^i_{t+1})p(\mathbf{l}^i_{t+1}) = \sum_{\mathbf{l}^i_{t+1}\in\mathcal{L}^i_{t+1}} \Pr(d^{ij}_{t+1}>r_c, \forall s_j\in\mathcal{N}^i_t|\mathbf{l}^i_{t+1})p(\mathbf{l}^i_{t+1}), \tag{11}$$

where $p(\mathbf{l}^j_{t+1})$ is defined by (8) as the probability that neighbor node s_j is located at $\mathbf{l}^j_{t+1}$ at $t+1$. It can be argued whether $p(\mathbf{l}^j_{t+1})$ is indeed available to s_i. The value of $p(\mathbf{l}^j_{t+1})$ can be requested by s_i from its neighbors before it makes any movement at the same time when it is receiving the sensor measurements. However, since $p(\mathbf{l}^j_{t+1})$ is not available until the procedure described in Section 3 is completed, this implies that all nodes have to first wait for their neighbors to finish the evaluation of improvement in target tracking data. This also requires additional bandwidth. If retrieval of $p(\mathbf{l}^j_{t+1})$ by s_i is not feasible, s_i does not have any a priori knowledge about how its neighbors are going to move. In this case, we can simply let $p(\mathbf{l}^j_{t+1})=\frac{1}{|\mathcal{L}^j_{t+1}|}$, where $|\mathcal{L}^j_{t+1}|$ is a constant, and the same for each node s_j.

In (11), we let

$$p^i_{t+1}(\mathrm{C}|\mathbf{l}^i_{t+1}) \equiv \Pr(d^{ij}_{t+1}>r_c, \forall s_j\in\mathcal{N}^i_t|s_i \text{ at } \mathbf{l}^i_{t+1}),$$

which is the probability that s_i is disconnected at $t+1$, given that it moves to $\mathbf{l}^i_{t+1}$. Therefore, we have

$$p^i_{t+1}(\mathrm{C}) = \sum_{\mathbf{l}^i_{t+1}\in\mathcal{L}^i_{t+1}} P^i_{t+1}(\mathrm{C}|\mathbf{l}^i_{t+1})p(\mathbf{l}^i_{t+1}) = \sum_{\mathbf{l}^i_{t+1}\in\mathcal{L}^i_{t+1}} \left(\prod_{s_j\in\mathcal{N}^i_t} \Pr(d^{ij}_{t+1}>r_c|\mathbf{l}^i_{t+1})\right)\cdot p(\mathbf{l}^i_{t+1}), \tag{12}$$

where $\Pr(d^{ij}_{t+1}>r_c|\mathbf{l}^i_{t+1})$ represents the probability that s_i is disconnected from its neighbor s_j, given that s_i moves to $\mathbf{l}^i_{t+1}$ at $t+1$. The probability $\Pr(d^{ij}_{t+1}>r_c|\mathbf{l}^i_{t+1})$ can be obtained by

$$\Pr(d^{ij}_{t+1}>r_c|\mathbf{l}^i_{t+1}) = \sum_{\mathbf{l}^j_{t+1}\in\mathcal{L}^j_{t+1}} \Pr(d^{ij}_{t+1}>r_c|\mathbf{l}^j_{t+1},\mathbf{l}^i_{t+1})p(\mathbf{l}^j_{t+1}). \tag{13}$$

Now, $\Pr(d^{ij}_{t+1}>r_c|\mathbf{l}^j_{t+1},\mathbf{l}^i_{t+1})$, the probability that s_i is disconnected at time $t+1$ from a neighbor s_j, given that s_i is at $\mathbf{l}^i_{t+1}$ and s_j is at $\mathbf{l}^j_{t+1}$, is either 1 or 0 when $\mathbf{l}^i_{t+1}$ and $\mathbf{l}^j_{t+1}$ are given. However, to also include the differences in the distances between s_i and its neighbors in $\mathcal{N}^i_t$, we can refine the definition of this probability in several ways. Let $p^{ij}_{t+1}(\mathrm{C}) \equiv \Pr(d^{ij}_{t+1}>r_c|\mathbf{l}^i_{t+1},\mathbf{l}^j_{t+1})$. Several ways of determining $p^{ij}_{t+1}(\mathrm{C})$ are listed below.

1. Use predicted distance d^{ij}_{t+1} between s_i and s_j directly:

$$p^{ij}_{t+1}(\mathrm{C}) \equiv \frac{d^{ij}_{t+1}}{\sum_{\mathbf{l}^j_{t+1}\in\mathcal{L}^j_{t+1}} d^{ij}_{t+1}}. \tag{14}$$

   Therefore, if s_i is at $\mathbf{l}^i_{t+1}$, it will have the highest probability of being disconnected from s_j when their mutual distance is the maximum among the candidate locations of s_j.

2. Alternatively, if we want to be more conservative, i.e., to inform the node of the possibility of being disconnected in advance of this event actually occurring at the next time instant, we can define $p^{ij}_{t+1}(\mathrm{C})$ as

$$p^{ij}_{t+1}(\mathrm{C}) \equiv \begin{cases} \dfrac{\frac{1}{\delta r_c - d^{ij}_{t+1}}}{\sum_{\mathbf{l}^j_{t+1}\in\mathcal{L}^j_{t+1}} \left[\frac{1}{\delta r_c - d^{ij}_{t+1}}\right]}, & \text{if } d^{ij}_{t+1} < \delta r_c, \\ 1, & \text{if } d^{ij}_{t+1} \geq \delta r_c, \end{cases} \tag{15}$$

   where δr_c is a fraction of r_c representing an acceptable threshold on how far away the neighbor s_j can be to s_i, e.g., $\delta r_c = 0.9 r_c$. Equation (15) can be used in situations where connectivity is given high priority; a smaller value of δr_c imposes a stricter connectivity constraint on node movement.

Next, we rewrite (12) for $p^i_{t+1}(\mathrm{C})$ as follows:

$$p^i_{t+1}(\mathrm{C}) = \sum_{\mathbf{l}^i_{t+1}\in\mathcal{L}^i_{t+1}} p^i_{t+1}(\mathrm{C}|\mathbf{l}^i_{t+1})p(\mathbf{l}^i_{t+1}) = \sum_{\mathbf{l}^i_{t+1}\in\mathcal{L}^i_{t+1}} \left(\prod_{s_j\in\mathcal{N}^i_t} \left(\sum_{\mathbf{l}^j_{t+1}\in\mathcal{L}^j_{t+1}} p^{ij}_{t+1}(\mathrm{C})p(\mathbf{l}^{ij}_{t+1})\right)\right)p(\mathbf{l}^j_{t+1}). \tag{16}$$

Note that (16) requires only local knowledge from a one-hop neighborhood which then can be implemented in a distributed manner. Also note that, from a general perspective of the connectivity in mobile sensor network, movement of nodes causes the partitioning of the network. As shown in the above discussion, the partitioning is avoided by evaluating the probability of the node being disconnected, which again is from the predicated target estimate. The connectivity problem in this case is integrated with the application goal for target tracking.

4.3 Potential Loss of Sensing Coverage

Another potential risk arising from node movement is the possible loss of sensing coverage in certain regions of the sensor field. When nodes move in the sensor field, the sensing area that is originally covered by these mobile nodes may not be covered by any other nodes. This implies that there may be some "holes" in the coverage over the sensor field. If these holes are not covered by any other nodes, a target that appears at the same time in this area will remain undetected. Therefore, the movement of nodes has to be managed in a way such that no such "hole" is formed.

We represent the sensor field as a 2D grid with dimension X by Y. Let there be a total of $n_g = XY$ grid points in the set $\mathcal{G}$ and let g_k be a grid point with index k. Let c^i_k be the probability that g_k is covered by node s_i and $c_k \equiv p(S_k)$ be the mapping from the set of nodes S_k that detect the grid point g_k to the probability that g_k is covered by the set of nodes in S_k. When nodes in S_k move, c_k changes due to the fact that locations of nodes in S_k change as well; thus, S_k may also change. Let $c^i_k(\mathbf{l}^i_t)$ map $\mathbf{l}^i_t$ to a probability representing the coverage probability of g_k due to s_i when s_i is located at $\mathbf{l}^i_t$. Let $\mathcal{S}^k_t$ be a mapping from time instant t to a set of nodes that detect grid point g_k at time t. We then

define $c_t^k \equiv p(\mathcal{S}_t^k)$ as the mapping from a set of nodes $\mathcal{S}_t^k$ that detect g_k at t to the coverage probability at time t for g_k from nodes in $\mathcal{S}_t^k$. We assume that the sensing coverage requirement is initially achieved by the sensor deployment algorithm, i.e., after the sensor deployment, $\forall g_k \in \mathcal{G}$, $c_k \geq p_{th}$, where p_{th} is a given sensor deployment control parameter representing the required sensing coverage threshold. One way for calculating c_k and c_t^k is shown in [34] as $c_k = 1 - \prod_{s_i \in S_k}(1 - c_k^i)$ and $c_t^k = 1 - \prod_{s_j \in \mathcal{S}_k^t}(1 - c_k^j)$.

Consider an arbitrarily chosen node s_i at time t. Suppose that s_i moves from its current location $\mathbf{l}_t^i$ to $\mathbf{l}_{t+1}^i$. To simplify the analysis, we assume that the sensing radius r_s of a node remains constant when it moves. Therefore, the area that s_i is able to cover at any time instant is fixed. However, the global sensing coverage may be affected by node movement. The coverage in the sensing area centered at the current location of s_i, i.e., $\mathbf{l}_t^i$, may be reduced, e.g., there may not be enough nodes to provide coverage for the area centered at $\mathbf{l}_t^i$ after s_i has moved to $\mathbf{l}_{t+1}^i$. On the other hand, s_i may even improve the sensing coverage at its new location $\mathbf{l}_{t+1}^i$. A thorough evaluation of the loss (or gain) of global sensing coverage requires an exchange of global information for the current topology of the sensor network. Due to the need for a distributed implementation, we only use the knowledge of a limited number of hops for local sensing coverage evaluation, i.e., $\lceil \frac{2r_s}{r_c} \rceil$-hops neighbor information [34].

Let the sensing area of node s_i be $A_i \in \mathcal{G}$, i.e., A_i is the set of grid points in $\mathcal{G}$ covered by s_i. Let $\mathcal{A}(\mathbf{l}_t^i)$ be a mapping from $\mathbb{R}^{d_l}$ to the set of grid points, i.e., $\mathcal{A}(\mathbf{l}_t^i)$ is the set of grid points corresponding to the sensing area centered at $\mathbf{l}_t^i$. We denote the set of grid points that will not be covered by s_i after s_i moves to $\mathbf{l}_{t+1}^i$ as $\Delta\mathcal{A}(\mathbf{l}_{t+1}^i)$, which is given by

$$\Delta\mathcal{A}(\mathbf{l}_{t+1}^i) = \mathcal{A}(\mathbf{l}_{t+1}^i) \cup \mathcal{A}(\mathbf{l}_t^i) \setminus \mathcal{A}(\mathbf{l}_{t+1}^i). \tag{17}$$

To ensure that there is no hole in the sensing area originally covered by s_i at t, the following condition must be satisfied:

$$c_{t+1}^k \geq p_{th}, \forall g_k \in \Delta\mathcal{A}(\mathbf{l}_{t+1}^i). \tag{18}$$

We need to find the expected coverage for g_k at time $t+1$, i.e, $E\{c_{t+1}^k\}$. Obviously, $E\{c_{t+1}^k\}$ requires the knowledge of $\mathcal{S}_{t+1}^k$, which is not available to node s_i at t. However, we can restrict the calculation of $E\{c_{t+1}^k\}$ on nodes in $\mathcal{S}_t^k$ only, which contains information about the probability that any $s_j \in \mathcal{S}_t^k$ is not in $\mathcal{S}_{t+1}^k$. Since it is possible that $\mathcal{S}_{t+1}^k$ may include other nodes that are not in $\mathcal{S}_t^k$, the evaluation is more conservative. The value of $E\{c_{t+1}^k\}$ for grid point $g_k \in \Delta\mathcal{A}(\mathbf{l}_{t+1}^i)$ can be obtained as follows:

$$\begin{aligned} E\{c_{t+1}^k\} &\equiv E\{c_{t+1}^k(\mathcal{S}_t^k)\} \\ &= 1 - \prod_{s_j \in \mathcal{S}_t^k} \left(1 - \sum_{\mathbf{l}_{t+1}^j \in \mathcal{L}_{t+1}^j} c_k^j(\mathbf{l}_{t+1}^j) p(\mathbf{l}_{t+1}^j)\right), \end{aligned} \tag{19}$$

where $p(\mathbf{l}_{t+1}^j)$ is given by (8). Note that, since nodes make movement decisions based on local knowledge only, we have assumed that the movement decisions on all nodes are independent. Hence, the probability of the appearance of a hole in $\Delta\mathcal{A}(\mathbf{l}_{t+1}^i)$ can be obtained as:

$$\Pr(\Delta\mathcal{A}(\mathbf{l}_{t+1}^i) \text{ has a hole}) \equiv \Pr(\exists g_k \in \Delta\mathcal{A}(\mathbf{l}_{t+1}^i) | E\{c_{t+1}^k\} < p_{th}). \tag{20}$$

It is easy to see that (20) yields either 1 or 0 because $\Pr(\Delta\mathcal{A}(\mathbf{l}_{t+1}^i) \text{ has a hole})$ denotes a binary outcome on the existence of a hole. Note that there may be more than one g_k in $\Delta\mathcal{A}(\mathbf{l}_{t+1}^i)$ that satisfies $c_{t+1}^k < p_{th}$. Furthermore, different choice of $\mathbf{l}_{t+1}^i$ may yield different numbers of such grid points as g_k. To describe more accurately the loss of sensing coverage in $\Delta\mathcal{A}(\mathbf{l}_{t+1}^i)$ due to the movement of node s_i, we introduce $p_{t+1}^i(\mathrm{S}|\mathbf{l}_{t+1}^i)$ as the probability of the loss of sensing coverage for s_i, given that s_i is located at $\mathbf{l}_{t+1}^i$ at time $t+1$. The probability $p_{t+1}^i(\mathrm{S}|\mathbf{l}_{t+1}^i)$ can be defined in several ways as listed below.

1. A straightforward method for obtaining $p_{t+1}^i(\mathrm{S}|\mathbf{l}_{t+1}^i)$ is to count the number of grid points that fail to satisfy the coverage requirement. Therefore,

$$p_{t+1}^i(\mathrm{S}|\mathbf{l}_{t+1}^i) \equiv \frac{|\Delta\mathcal{A}_0(\mathbf{l}_{t+1}^i)|}{|\Delta\mathcal{A}(\mathbf{l}_{t+1}^i)|}, \tag{21}$$

where $\Delta\mathcal{A}_0(\mathbf{l}_{t+1}^i) \subseteq \Delta\mathcal{A}(\mathbf{l}_{t+1}^i)$ is defined as

$$\Delta\mathcal{A}_0(\mathbf{l}_{t+1}^i) \equiv \{g_k | g_k \in \Delta\mathcal{A}(\mathbf{l}_{t+1}^i), E\{c_{t+1}^k\} \leq p_{th}\}. \tag{22}$$

Equation (22) gives the highest probability to the candidate location $\mathbf{l}_{t+1}^i$ that will have the highest number of grid points in the corresponding $\Delta\mathcal{A}(\mathbf{l}_{t+1}^i)$ whose coverage are below the threshold p_{th}. Note that this approach does not consider the exact coverage on grid points.

2. To also include the drop of coverage on grid points that fail to meet the coverage threshold requirement, $p_{t+1}^i(\mathrm{S}|\mathbf{l}_{t+1}^i)$ can be refined to express the coverage loss more precisely. Thus,

$$\begin{aligned} p_{t+1}^i(\mathrm{S}|\mathbf{l}_{t+1}^i) &\equiv \Pr(\text{coverage drop of } g_k \in \Delta\mathcal{A}(\mathbf{l}_{t+1}^i) | E\{c_{t+1}^k\} \leq p_{th}) \\ &\equiv \frac{\sum_{\forall g_k \in \Delta\mathcal{A}(\mathbf{l}_{t+1}^i), E\{c_{t+1}^k\} < p_{th}} (\, p_{th} - E\{c_{t+1}^k\}\,)}{\sum_{\forall g_k \in \Delta\mathcal{A}(\mathbf{l}_{t+1}^i)} E\{c_{t+1}^k\}}. \end{aligned} \tag{23}$$

3. Alternatively, since c_t^k is available to s_k at t, we can then define $p_{t+1}^i(\mathrm{S}|\mathbf{l}_{t+1}^i)$ to reflect the expected absolute loss of coverage per grid point. Let $\Delta c_{t+1}^k \equiv c_{t+1}^k - c_t^k$. Obviously, $\Delta c_{t+1}^k < 0$ represents the sensing coverage loss on grid point g_k. Since we only have $E\{c_{t+1}^k\}$, we use the expectation of Δc_{t+1}^k, i.e, $E\{\Delta c_{t+1}^k\}$. Therefore,

$$\begin{aligned} p_{t+1}^i(\mathrm{S}|\mathbf{l}_{t+1}^i) &\equiv \Pr(\text{coverage loss of } g_k \in \Delta\mathcal{A}(\mathbf{l}_{t+1}^i) | E\{\Delta c_{t+1}^k\} < 0) \\ &\equiv \frac{\sum_{\forall g_k \in \Delta\mathcal{A}(\mathbf{l}_{t+1}^i), E\{\Delta c_{t+1}^k\} < 0} |E\{\Delta c_{t+1}^k\}|}{\sum_{\forall g_k \in \Delta\mathcal{A}(\mathbf{l}_{t+1}^i)} |E\{\Delta c_{t+1}^k\}|}, \end{aligned} \tag{24}$$

where $|E\{\Delta c_{t+1}^k\}|$ gives the absolute value of $E\{\Delta c_{t+1}^k\}$. Equation (24) shows that if a candidate

location $\mathbf{l}^i_{t+1}$ gives the maximum total absolute coverage loss on all grid points, it has the highest probability of loss of sensing coverage $p^i_{t+1}(\mathrm{S}|\mathbf{l}^i_{t+1})$. Note that the denominator of (24) also includes the possible coverage gain on all grid points in $\Delta\mathcal{A}(\mathbf{l}^i_{t+1})$. For example, for two candidate locations with the same amount of coverage loss given by the numerator in (24), the one with larger coverage gain in the denominator has the lower probability of loss of sensing coverage.

Note that $\mathbf{l}^i_t \in \mathcal{L}^i_{t+1}$, which implies that $\Delta\mathcal{A}(\mathbf{l}^i_{t+1}) = \phi$ when $\mathbf{l}^i_{t+1} = \mathbf{l}^i_t$. So, we define $p^i_{t+1}(\mathrm{S}|\mathbf{l}^i_{t+1} = \mathbf{l}^i_t) = 0$. With $p^i_{t+1}(\mathrm{S}|\mathbf{l}^i_{t+1})$, we can obtain the probability of loss of sensing coverage, denoted by $p^i_{t+1}(\mathrm{S})$, as follows:

$$p^i_{t+1}(\mathrm{S}) = \sum_{\forall \mathbf{l}^i_{t+1} \in \mathcal{L}^i_{t+1}} p^i_{t+1}(\mathrm{S}|\mathbf{l}^i_{t+1}) p(\mathbf{l}^i_{t+1}), \tag{25}$$

where $p^i_{t+1}(\mathrm{S}|\mathbf{l}^i_{t+1})$ can be obtained by one of the definitions described above.

5 Decision on Node Movement

In Section 3 and 4, we have derived the probabilities associated with tracking quality improvement, additional energy consumption, loss of connectivity, and loss of coverage. Next, we investigate the cost evaluation based on using these probabilities.

5.1 Cost Evaluation

Recall that the optimal candidate location in the sense of tracking quality improvement is given by (7). However, this selection does not consider the negative consequences described in Section 4. Next, we present the selection rule based on the cost evaluation that takes into account all negative consequences due to node movement. Let constants C_e, C_c, and C_s be the individual costs corresponding to energy consumption due to movement, loss of connectivity, and loss of coverage, respectively. To simplify the discussion, we assume that C_e, C_c, and C_s are already properly normalized. Note that C_e, C_c, and C_s have nonnegative values to indicate the costs associated with the energy consumption in movement and risks of losing connectivity and coverage. Let $\mathcal{C}^i_{t+1}(\mathbf{l}^i_{t+1})$ be the total cost for node s_i when s_i moves to $\mathbf{l}^i_{t+1}$ at $t+1$. We define $\mathcal{C}^i_{t+1}(\mathbf{l}^i_{t+1})$ as

$$\begin{aligned}\mathcal{C}^i_{t+1}(\mathbf{l}^i_{t+1}) \equiv\ & p^i_{t+1}(\mathrm{E}|\mathbf{l}^i_{t+1})\mathrm{C}_e w_e + p^i_{t+1}(\mathrm{C}|\mathbf{l}^i_{t+1})\mathrm{C}_c w_c \\ & + p^i_{t+1}(\mathrm{S}|\mathbf{l}^i_{t+1})\mathrm{C}_s w_s,\end{aligned} \tag{26}$$

where w_e, w_c, and w_s are normalized weighting factors for energy consumption, connectivity, and sensing coverage, respectively. In various types of application scenarios, w_e, w_c, and w_s can be used to reflect different priorities on these costs. Based on (26), the expected total cost for s_i when s_i moves to $\mathbf{l}^i_{t+1}$ can be found as

$$E\{\mathcal{C}^i_{t+1}\} \equiv \sum_{\mathbf{l}^i_{t+1} \in \mathcal{L}^i_{t+1}} \mathcal{C}^i_{t+1}(\mathbf{l}^i_{t+1}) p(\mathbf{l}^i_{t+1}). \tag{27}$$

Equation (27) can be used as an extension to the proposed scheme in this paper for group mobility management, where nodes can exchange their expected total cost and decide who should move. Similarly, if there is a need to impose a centralized control over node movement, the expected total cost given by (27) can be sent to the base station to guide the node movement. Hence, the proposed scheme is flexible in various types of applications.

5.2 Decision on Movement

When the total cost is obtained for all candidate locations, the optimal selection of the candidate location for node s_i can be obtained by considering both positive and negative consequences. In practice, the decision on node movement depends on the actual requirement for particular operations in the sensor network. Considering both the positive and negative consequences, we define the selection rule for the candidate location as a two-step selection process described below.

- **Selection Step 1**. We first find locations that are expected to improve the target tracking data. This is given by

$$\begin{aligned}\bar{\mathcal{L}}^i_{t+1} = \{\mathbf{l}^i_{t+1} | \mathbf{l}^i_{t+1} \in \mathcal{L}^i_{t+1} \text{ and } \psi(\mathbf{l}^i_{t+1}) \\ \geq \delta_\psi \max_{\mathbf{l}^i_{t+1} \in \mathcal{L}^i_{t+1}} \psi(\mathbf{l}^i_{t+1})\},\end{aligned} \tag{28}$$

  where $\psi(\mathbf{l}^i_{t+1}) \equiv \psi(p(\hat{\mathbf{x}}^i_{t+1}|\hat{\mathbf{z}}^i_{t+1}(\mathbf{l}^i_{t+1}))$ is described in Section 3.3 and $\delta_\psi (0 \leq \delta_\psi \leq 1)$ is a given parameter which represents how many candidate locations are considered good enough for the improvement of target tracking data. Obviously, $\bar{\mathcal{L}}^i_{t+1} \subseteq \mathcal{L}^i_{t+1}$. The next step then selects the candidate location in $\bar{\mathcal{L}}^i_{t+1}$ that has the minimum cost.
- **Selection Step 2**.

$$\bar{\mathbf{l}}^i_{t+1} = \min_{\mathbf{l}^i_{t+1} \in \bar{\mathcal{L}}^i_{t+1}} \mathcal{C}^i_{t+1}(\mathbf{l}^i_{t+1}), \tag{29}$$

  where $\mathcal{C}^i_{t+1}(\mathbf{l}^i_{t+1})$ is given by (26).

Note that, when $\delta_\psi = 1$, we have

$$\bar{\mathcal{L}}^i_{t+1} = \{\bar{\mathbf{l}}^i_{t+1} | \bar{\mathbf{l}}^i_{t+1} = \arg_{\mathbf{l}^i_{t+1} \in \mathcal{L}^i_{t+1}} \max \psi(\mathbf{l}^i_{t+1})\},$$

which is the same as the selection given by (7). This corresponds to the case where we highly prioritize the target tracking improvement regardless of the associated costs; Selection Step 2 can then be omitted. On the other hand, when $\delta_\psi = 0$, we have $\bar{\mathcal{L}}^i_{t+1} = \mathcal{L}^i_{t+1}$, which implies that we are more concerned about the costs associated with all candidate locations. Selection Step 1 can then be omitted. Also, note that, as discussed in Sections 3 and 4, all evaluations require local knowledge, therefore, the proposed mobility management scheme can be implemented in a distributed manner.

5.3 Analysis of Time Complexity

Next, we analyze the time complexity for the proposed mobility management algorithm. Fig. 4 shows pseudocode for the distributed mobility management procedure. The computational complexity for the proposed procedure illustrated in Fig. 4 depends on complexities of the individual computation for target estimation, movement probability, connectivity, coverage loss, and cost evaluation.

Input: node s_i

01 **For** $\forall \mathrm{l}_{t+1}^i \in \mathcal{L}_{t+1}^i$ **Do**
02 Calculate the predicted measurement $\hat{z}_{t+1}^i(\mathrm{l}_{t+1}^{i_1})$ and construct $\hat{\mathbf{z}}_{t+1}^i(\mathrm{l}_{t+1}^i)$
03 Perform local target estimation using $\hat{\mathbf{z}}_{t+1}^i(\mathrm{l}_{t+1}^i)$
04 Evaluate the tracking improvement using Equation (8)
05 Calculate probability of movement $p(\mathrm{l}_{t+1}^i)$
06 Evaluate the $\mathcal{E}_{t+1}^i(\mathrm{l}_{t+1}^i)$, $p_{t+1}^i(\mathrm{C}|\mathrm{l}_{t+1}^i)$, and $p_{t+1}^i(\mathrm{S}|\mathrm{l}_{t+1}^i)$
07 **End**
08 **For** $\forall \mathrm{l}_{t+1}^i \in \mathcal{L}_{t+1}^i$ **Do**
07 Evaluate the $p_{t+1}^i(\mathrm{C}|\mathrm{l}_{t+1}^i)$ and the cost $\mathcal{C}_{t+1}^i(\mathrm{l}_{t+1}^i)$
09 **End**
10 Evaluate the expected total cost $E\{\mathcal{C}_{t+1}^i\}$
11 Move to the determined location

Fig. 4. Pseudocode for distributed mobility management.

Note that, for any two nodes s_i and $s_j (i \neq j)$, $|\mathcal{L}_{t+1}^i| = |\mathcal{L}_{t+1}^j|$. Let $m = |\mathcal{L}_{t+1}^j|$. Assume that the target tracking algorithm has a complexity of $O(T)$, where T reflects the complexity of the estimation algorithm. For example, in Kalman filtering, $O(T) = O(2d_x^2 d_z) + O(2d_x d_z^2) + O(d_x^3) + O(d_x^3)$ [11]. As shown in Section 3 from (7) and (8), for each candidate location l_{t+1}^i, node s_i has to perform a local target estimation using the predicted sensor measurements to obtain $p(\mathrm{l}_{t+1}^i)$. Note that the node has to integrate all available sensor data within its one-hop neighborhood, which, for the worst case, may be as many as $n-1$. Therefore, the computational complexity for node movement probability is $O(mnT)$. From (10), evaluation of $\mathcal{E}_{t+1}^i(\mathrm{l}_{t+1}^i)$ takes $O(1)$ time. Evaluation of $p_{t+1}^i(\mathrm{C}|\mathrm{l}_{t+1}^i)$, i.e., the probability for s_i being disconnected, requires the calculation of distances between s_i and all of its neighbors at the current time instant for all their candidate locations. Assume that there are a total of n nodes, this procedure takes $O(mn)$ time. For coverage loss evaluation, the complexity depends on grid dimensions X and Y representing the sensing area. It takes $O(mnXY)$ time for coverage loss evaluation. From Section 5, the cost evaluation and movement decision take $O(m)$ time. Therefore, the mobility management procedure takes $O(mnT) + O(mn) + O(mnXY) + O(m) = O(mnXY) + O(mnT)$ time.

Note that in the proposed mobility management algorithm, there is no communication required at the time for a node to make its movement. This is especially important and useful because a node can react to targets in a timely manner. Though the node still needs the prior to obtain the posterior, as mentioned in Section 3.3, the target estimate based on complete sensor measurements can be forwarded to nodes at a later stage after node makes its decision on its movement to dynamically track the mobile target. In mobile sensor networks, where topology is constantly changing, a node depends on techniques such as periodical HELLO message for current neighborhood information. The HELLO message can be easily extended for exchange of sensor measurements and target estimate. The mobility management algorithm proposed in this paper imposes virtually no communication overhead.

6 Simulation Studies

We simulated the proposed mobility management scheme using MatLab. The setup of the simulation contains a wireless sensor network with 20 homogeneous nodes with a communication radius of 18 m and a sensing radius of 9 m. Nodes are randomly deployed in a $20\ \mathrm{m} \times 20\ \mathrm{m}$ sensor field represented by a 20×20 grid. We consider a linear system model in the simulation and use Kalman filtering for target estimation. The system model and measurement model are given as

$$\mathbf{x}_t = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \mathbf{x}_{t-1} + \begin{bmatrix} \Delta t & 0 \\ 0 & \Delta t \end{bmatrix} \mathbf{u} + \mathbf{w}_{t-1}$$

and

$$\mathbf{z}_t = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \mathbf{x}_t + \mathbf{v}_t,$$

where Δt is the sampling interval and $\mathbf{u} = [\mu_x; \mu_y]$ represents the constant speeds of the target in the x and y directions, respectively. The state vector $\mathbf{x}_t$ contains the x and y coordinates of the target. We assume that $\mu_x = \mu_y$. We assume that, initially, $\hat{\mathbf{x}}_0 = [0; 0]$ for all nodes. The sensing model that we use for coverage evaluation is given by $c_k^i = e^{-\alpha d_k^i}$, where c_k^i is the coverage probability, d_k^i is the distance between the grid point g_k and the node s_i, and α represents the physical characteristics of the sensor [34]. In the simulations, we have chosen α such that $\alpha = -\frac{\ln p_{th}}{r_s}$.

6.1 Static Sensor Network versus Mobile Network with Mobility Management

In Fig. 5 and Fig. 6, we compare the proposed approach with a baseline case of fixed nodes, i.e., nodes remain at their original locations throughout the simulation. In addition to the localized approach, we also consider a centralized target estimation approach which is based on all the sensor measurements. The target speed is set as $\mu_x = \mu_y = 1\ \mathrm{m/s}$ and nodes have the same speed as the target. The sampling interval Δt is 1 sec. The target is initially placed at [2; 2]. The selection of candidate locations for target tracking data improvement is based on the trace of the error covariance matrix. Fig. 5 shows the comparison of the trace of the error covariance matrix and position estimation error in a $\log_{10}$ scale. It clearly shows that the error for a mobile sensor network is less than that for the static network. Another good and well-accepted metric for evaluating the tracking quality is the norm of the position

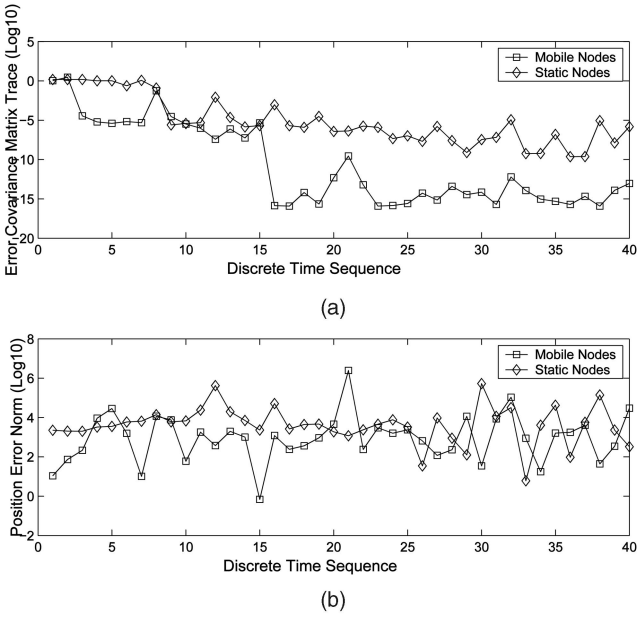

Fig. 5. (a) Trace of estimation error covariance matrix: mobility management versus static network. (b) Position error norm: mobility management versus static network.

error [4], [12], [32], which is shown at the bottom of Fig. 5. The norm of the position error for the mobile network is roughly 72.5 percent less during the time that target is moving through the sensor field.

Fig. 6a shows the average of the expected probability that a mobile node is disconnected, i.e., $p^i_{t+1}(\mathrm{C}|\mathbf{l}^i_{t+1})$, and the average distance moved by the mobile nodes. Note that node movement causes an increase in the expected probability of nodes being disconnected, but since the sensor network is densely deployed and node movement is not drastic per time step, the probability of disconnection is still very small. Fig. 6b illustrates the average of the expected probability of coverage loss, i.e., $p^i_{t+1}(\mathrm{S}|\mathbf{l}^i_{t+1})$ for all mobile nodes, and the average global coverage difference between the mobile network and the static network. The average global coverage is defined as the sum of individual grid points coverage on individual grid points over the total number of grid points on the example sensor field. Similar to Fig. 6a, $p^i_{t+1}(\mathrm{S}|\mathbf{l}^i_{t+1})$, which is shown in the top graph of Fig. 6b is very small. However, as shown by the bottom graph in Fig. 6b, the mobility management scheme improves the global coverage compared to the static network.

6.2 Random Mobile Sensor Network versus Mobile Network with Mobility Management

Fig. 7 illustrates the effects of mobility management for mobile sensor networks. In one case, the nodes in the mobile network use the proposed mobility management algorithm to make movement decisions. In the baseline case, nodes randomly select a candidate location for the next move. Fig. 7a shows that the mobility management algorithm achieves the selection of candidate locations that can provide better tracking quality. Fig. 7b illustrates the effect of mobility management for improved sensing coverage, which is advantageous for surveillance. This benefit is not available for mobile sensor networks that use a random mobility management scheme.

6.3 Localized Versus Centralized Implementations

Next, we compare the localized implementation with a centralized implementation. For the centralized approach, we assume that here exists a base station that acts as the processing center. All nodes that have detected the target forward the data to the base station for processing. The base station uses the same evaluation rule as described in previous sections. However, the base station has global knowledge of the location of all the nodes and it can integrate sensor data from all the sensor data. It is expected that the centralized approach requires more bandwidth and

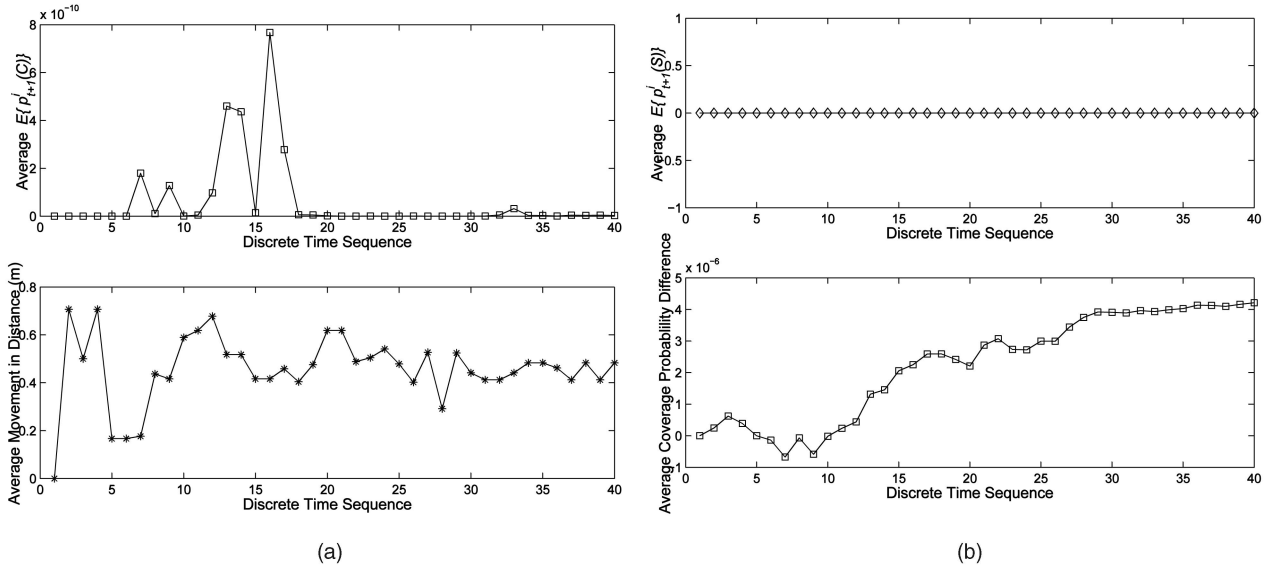

Fig. 6. (a) Top: Average expected probability of a node being disconnected. Bottom: Average distance moved by a node. (b) Top: Average expected probability of loss of sensing coverage. Bottom: Average global coverage difference between the example mobile sensor network and the static sensor.

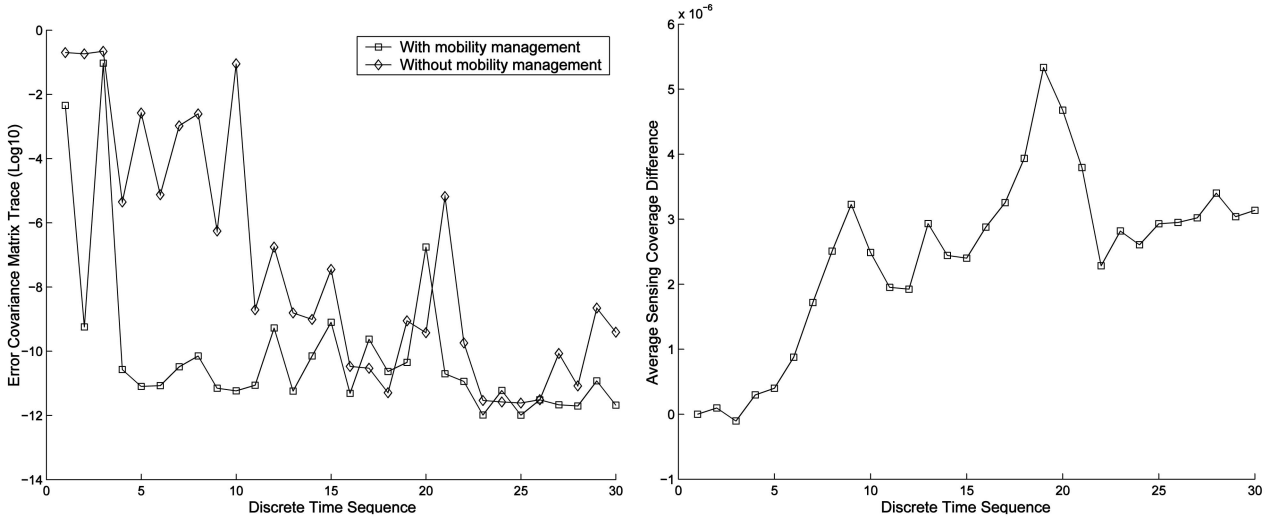

Fig. 7. (a) Trace of estimation error covariance matrix: with mobility management versus without mobility management. (b) Average global coverage difference between the mobile sensor network with mobility management and without mobility management.

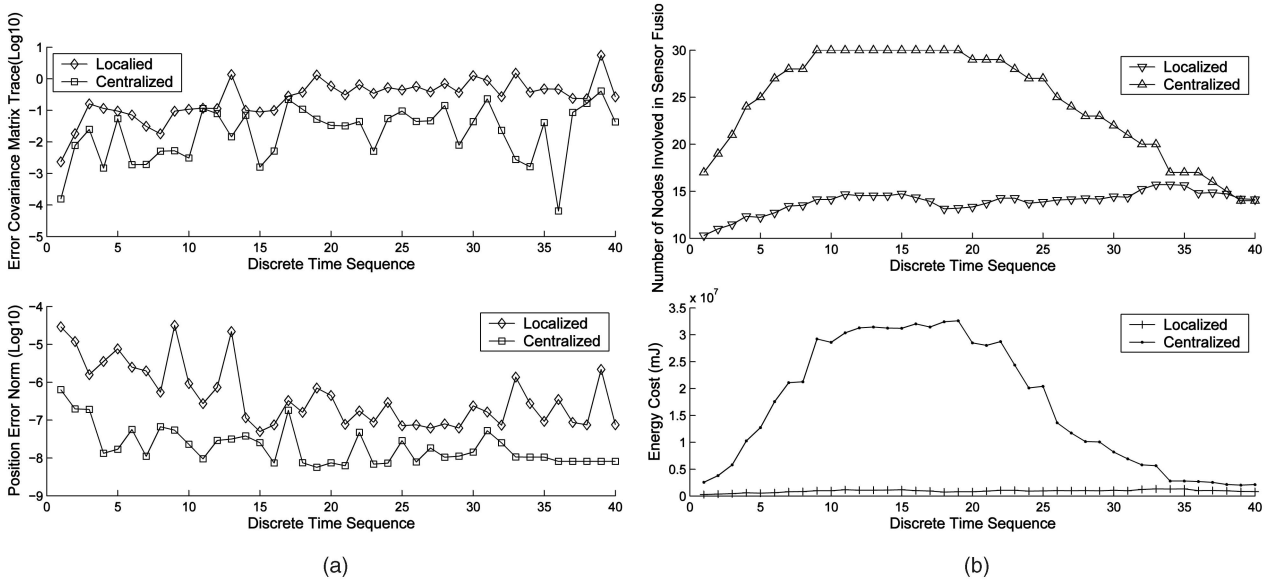

Fig. 8. (a) Mobility management: localized versus centralized. Top: Trace of estimation error covariance matrix. Bottom: Position error norm. (b) Mobility management: localized versus centralized. Top: Number of nodes involved in evaluation. Bottom: Energy cost evaluation.

energy for communication. Fig. 8 shows the simulation results. For simplicity, we assume that the base station is located at the center of the sensing region.

Fig. 8a shows the trace of estimation error covariance matrix and position error norm for the localized and centralized implementations. As shown in the top graph of Fig. 8a, the centralized approach outperforms localized implementation in tracking quality because in the localized algorithm nodes only have local knowledge. This is evident by the bottom of Fig. 8a, where the estimated error covariance matrix of localized implementation is always larger than that for the centralized case.

On the other hand, the energy consumption is considerably higher for the centralized method; see the bottom part of Fig. 8b. Equation (30) is used to calculate the energy consumption at time instant t for node s_i:

$$e(t,i) \;=\; e_{comm}(t,i) + e_{comp}(t,i) + e_{move}(t,i), \qquad (30)$$

where $e_{comm}(t,i)$, $e_{comp}(t,i)$, and $e_{move}(t,i)$ account for energy consumption due to communication, computation, and movement, respectively. Note that sensing energy consumption is not considered here because it is not affected by whether the proposed mobility management scheme is implemented as a localized or a centralized algorithm. Let $\hat{\mathcal{N}}_t^i$ be the set of nodes that have detected the target and are also the one-hop neighbors of node s_i. Let $d_{ij}(t)$ be the distance between node s_i and s_j at time t. Let $\hat{d}_i(t) \equiv \max d_{ij}(t),\ \forall s_j \in \hat{\mathcal{N}}_t^i$, be the maximum distance

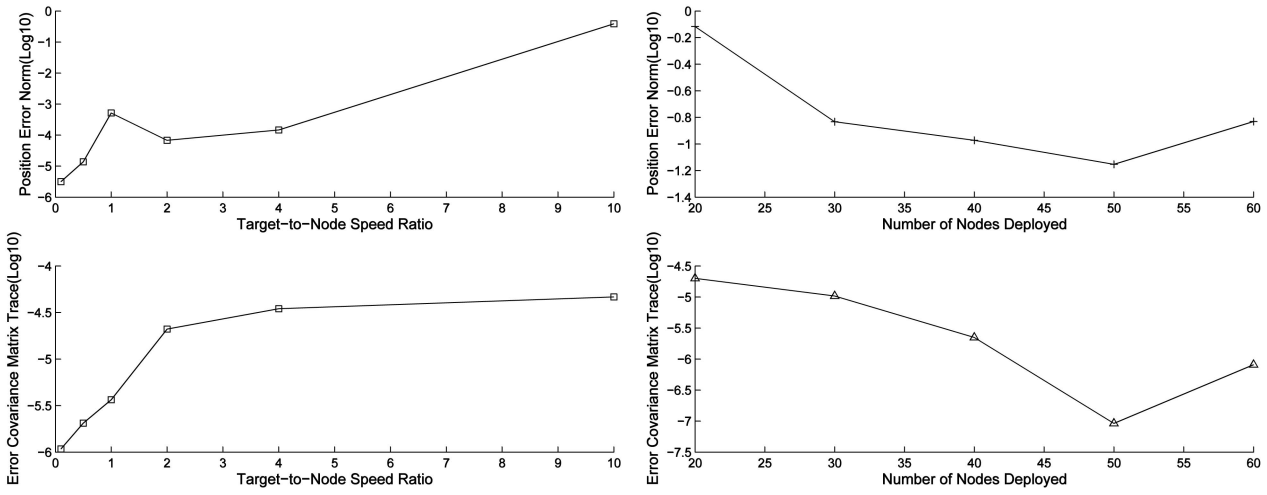

Fig. 9. (a) Mobility management for target at different speeds. Target-to-node speed ratio is 0.1, 0.5,1, 2, 4, and 10. (b) Mobility management for different node densities. Random sensor deployment with 20, 30, 40, 50, and 60 nodes.

between s_i and its neighbors. Let $\Delta d_i(t)$ be the distance that s_i moves at time t. The measures $e_{comm}(t,i)$, $e_{comp}(t,i)$, and $e_{move}(t,i)$ are evaluated as follows:

$$e_{comm}(t,i) = \hat{d}_i(t)^{\alpha} \times (\mathrm{E}_{send} + |\hat{\mathcal{N}}_t^i| \times \mathrm{E}_{recv}), \tag{31}$$

$$e_{comp}(t,i) = (|\hat{\mathcal{N}}_t^i| + 1) \times m \times \mathrm{E}_{comp}, \tag{32}$$

$$e_{move}(t,i) = \Delta d_i(t) \times \mathrm{E}_{move}, \tag{33}$$

where E_{send}, E_{recv}, E_{comp}, and E_{move} are power constants for communication, transmitting, computation, and locomotion, respectively. α is the attenuation constant. We use $\alpha = 2.5$, $\mathrm{E}_{send} = \mathrm{E}_{recv} = 1\ \mathrm{mJ/m}$, and $\mathrm{E}_{move} = 50\ \mathrm{mJ/m}$, which are based on metrics presented in [24]. Note that E_{recv} and E_{send} are also related to the size of the data packet. In this simulation, we assume that a 100-byte packet is sufficient for a node to send its sensor response as well as its current location to its neighbors. We assume that $\mathrm{E}_{comp} = 0.1\ \mathrm{mJ}$. The number of candidate locations m, which is defined in Section 5.3, is set to 9 in this simulation.

As shown in top of Fig. 8b, since only local knowledge is used, the number of nodes involved in data integration for the localized algorithm is much lower than that for the centralized case. As a result, the tracking quality is better for the centralized case, as shown in Fig. 8a. However, the centralized approach requires considerably more energy, as shown in the bottom of Fig. 8b. The results presented here can be used as a guideline to select an appropriate implementation strategy to trade off energy consumption with tracking accuracy. Moreover, from the top of Fig. 8b, we can see that the localized algorithm is scalable since the number of nodes involved in data integration does not increase with the total number of deployed nodes.

6.4 Discussion

The mobility management scheme proposed in this paper determines the probability that a node moves to a certain location, i.e., $p(\mathrm{l}_{t+1}^i)$. As shown in Section 3, $p(\mathrm{l}_{t+1}^i)$ is obtained by making use of the predicted sensor measurement at the candidate location. Alternative techniques can be used to achieve the same purpose, for example, the distributed probability inference method as described in [22], where a distributed architecture is presented for message exchange among sensor nodes to solve problems such as probability inferencing. However, this architecture is not designed to handle mobility management for target tracking as described in this paper. As discussed in Section 3, we use the predicted measurement from a node's candidate locations. Therefore, we expect further improvement for target tracking in mobile sensor networks by integrating the proposed mobility management scheme with the efficient probability inferencing technique based on the distributed architecture described in [22]. In view of the inherent complexity and the dynamic nature of mobile sensor networks, we design the algorithm with only localized communication requirements. Every node can make its movement decision in a timely manner for dynamic target tracking without lengthy negotiation with neighbors for maintaining connectivity and sensing coverage. This ensures a flexible distributed implementation for mobility management in mobile sensor networks for target tracking.

Note that the performance of the proposed scheme is also related to factors such as the speed of the nodes, the target speed, and the number of nodes deployed in the sensing region. Fig. 9 shows the position error norm as well as the error covariance matrix trace for target tracking using the proposed mobility management method for different target speeds and various numbers of nodes. Fig. 9a illustrates the tracking performance when the target-to-node speed ratio is varied from 0.1 to 10. It is expected that, when the target is moving slowly, nodes are able to track more accurately, as is evident from the increase in the error covariance matrix trace. In Fig. 9b, when the number of nodes deployed increases, the target tracking quality is improved because more sensor data are available within the one-hop neighborhood for the node to make movement decisions based on local knowledge.

Note that, due to the complexity of the mobility management problem, we have made a number of assumptions as described in Section 3.1. These assumptions simplify the mobility management problem, but they also highlight some of the limitations of the proposed method. One issue is that, when a target is moving at a much higher speed than all sensor nodes, the difference between two consecutive measurements from a node is too large to be useful for the calculation of the predicted sensor measurements, as described in Section 3.3. However, this problem can be addressed at the time when the sensor network is deployed, i.e., the sampling rate of the chosen sensor nodes should be fast enough to match the target speed. On the other hand, if the target is moving at a much lower speed, a single step of node movement will cause a drastic change in a node's two consecutive measurements. In this case, a node has to adjust its speed before it starts moving to avoid unnecessary locomotion energy consumption. We have also not modeled the communication energy needed for node management, motivated in part by the fact that locomotion energy consumption is relatively larger in magnitude. Nevertheless, the proposed sensor node mobility management algorithm provides useful insights into the problem and it can be extended for more general scenarios.

7 Conclusions

In this paper, we have introduced a new mobility management scheme for mobile sensor networks. This scheme considers target tracking quality, connectivity breakage, loss of sensing coverage, and energy consumption due to node movement. The constantly changing topology due to node movement makes mobility management difficult for mobile sensor networks. The proposed mobility management scheme is designed for a distributed implementation, where only knowledge of the one-hop (or limited numbers of hops) neighborhood is required. The cost evaluation technique allows us to trade off target tracking quality improvement with the negative consequences of energy consumption, loss of connectivity and coverage. We are currently investigating the routing problem in mobility management, e.g., quantifying the additional cost of reestablishing routes when previous routes become invalid due to node movement.

Acknowledgments

This research was supported by DARPA and administered by the US Army Research Office under Emergent Surveillance Plexus MURI Award No. DAAD19-01-1-0504. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the sponsoring agencies. This work was carried out while Yi Zou was working as a postdoctoral research associate in the Department of Electrical and Computer Engineering at Duke University.

References

[1] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A Survey on Sensor Networks," *IEEE Comm. Magazine,* pp. 102-114, Aug. 2002.

[2] M.S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking," *IEEE Trans. Signal Processing,* vol. 50, pp. 174-188, 2002.

[3] F. Baker, "An Outsider's View of MANET," IETF Internet draft, work in progress, June 2006.

[4] Y. Bar-Shalom, *Multitarget/Multisensor Tracking: Applications and Advances-Volume III.* Artech House, 2000.

[5] D. Braginsky and D. Estrin, "Rumor Routing for Sensor Networks," *Proc. ACM Int'l Workshop Wireless Sensor Networks and Applications,* pp. 22-31, 2002.

[6] R.R. Brooks, C. Griffin, and D. Friedlander, "Distributed Target Classification and Tracking in Sensor Networks," *Proc. IEEE,* pp. 1163-1171, 2003.

[7] T. Camp, J. Boleng, and V. Davies, "A Survey of Mobility Models for Ad Hoc Network Research," *J. Wireless Comm. Mobile Computing,* vol. 2, pp. 483-502, 2002.

[8] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An Energy-Effecient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks," *Proc. MobiCom,* pp. 85-96, 2001.

[9] J.C. Chen, K. Yao, and R.E. Hudson, "Source Localization and Beamforming," *IEEE Signal Processing Magazine,* vol. 19, pp. 30-39, 2002.

[10] A.J. Coulson, A.G. Williamson, and R.G. Vaughan, "A Statistical Basis for Lognormal Shadowing Effects in Multipath Fading Channels," *IEEE Trans. Comm.,* pp. 494-502, 1998.

[11] M.J. Goris, D.A. Gray, and I.M.Y. Mareels, "Reducing the Computational Load of a Kalman Filter," *IEE Electronics Letters,* vol. 33, pp. 1539-1541, 1997.

[12] D.L. Hall and J. Llinas, *Handbook of Multisensor Data Fusion.* CRC Press, 2001.

[13] Z.J. Haas and B. Liang, "Ad Hoc Mobility Management with Uniform Quorum Systems," *IEEE/ACM Trans. Networking,* vol. 7, pp. 228-240, 1999.

[14] W.R. Heizelman, A. Chandrakasan, and H. Balakrishnan, "Energy Efficient Communication Protocol for Wireless Micro Sensor Networks," *Proc. Int'l Conf. System Sciences Sciences,* pp. 1-10, 2000.

[15] X. Hong, M. Gerla, G. Pei, and C. Chiang, "A Group Mobility Model for Ad Hoc Wireless Networks," *Proc. ACM Int'l Workshop Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM '99),* pp. 53-60, 1999.

[16] S.S. Iyengar and R.R. Brooks, *Handbook on Distributed Sensor Networks.* Chapman and Hall/CRC Press, July 2003.

[17] D. Jea, A.A. Somasundara, and M.B. Srivastava, "Multiple Controlled Mobile Elements (Data Mules) for Data Collection in Sensor Networks," *Proc. IEEE Int'l Conf. Distributed Computing in Sensor Systems (DCOSS '05),* pp. 244-257, 2005.

[18] X.Y. Li, P.J. Wan, Y. Wang, and C.W. Yi, "Fault Tolerant Deployment and Topology Control in Wireless Networks," *Proc. MobiHoc,* pp. 117-128, 2003.

[19] B.Y. Liu, P. Brass, O. Dousse, P. Nain, and D. Towsley, "Mobility Improves Coverage of Sensor Network," *Proc. MobiHoc,* pp. 300-308, 2005.

[20] Y.G. Mei, Y.H. Lu, Y.C. Hu, and C.S. George Lee, "Deployment Strategy for Mobile Robots with Energy and Timing Constraints," *Proc. IEEE Int'l Conf. Intelligent Robots and Systems,* 2005.

[21] T. Moscibroda, R. O'Dell, M. Wattenhofer, and R. Wattenhofer, "Virtual Coordinates for Ad Hoc and Sensor Networks," *Proc. IEEE Foundations of Mobile Computing Workshop,* pp. 8-16, 2004.

[22] M. Paskin, C. Guestrin, and J. McFadden, "A Robust Architecture for Distributed Inference in Sensor Networks," *Proc. IEEE Int'l Conf. Information Processing in Sensor Networks (IPSN '05),* 2005.

[23] S. Shakkottai, R. Srikant, and N.B. Shroff, "Unreliable Sensor Grids: Coverage, Connectivity and Diameter," *Proc. IEEE INFOCOM,* pp. 1073-1083, 2003.

[24] R. Rao and G. Kesidis, "Purposeful Mobility for Relaying and Surveillance in Mobile Ad-Hoc Sensors Networks," *IEEE Trans. Mobile Computing,* vol. 3, pp. 225-231, 2004.

[25] A. Roy, S.K. Das, and A. Misra, "Exploiting Information Theory for Adaptive Mobility and Resource Management in Future Cellular Networks," *IEEE Wireless Comm. Magazine,* vol. 11, pp. 59-65, 2004.

[26] G. Wang, G. Cao, and T. La Porta, "Movement-Assisted Sensor Deployment," *IEEE Trans. Mobile Computing,* vol. 5, pp. 640-652, 2006.

[27] X.R. Wang, G.L. Xing, Y.F. Zhang, C.Y. Lu, R. Pless, and C. Gill, "Integrated Coverage and Connectivity Configuration in Wireless Sensor Networks," *Proc. ACM Conf. Embedded Networked Sensor Systems (SenSys '03)*, pp. 28-39, 2003.
[28] J. Wu, "Extended Dominating-Set-Based Routing in Ad Hoc Wireless Networks with Unidirectional Links," *IEEE Trans. Parallel and Distributed Computing*, pp. 327-340, 2002.
[29] Y. Xu, J. Heidemann, and D. Estrin, "Geography-Informed Energy Conservation for Ad Hoc Routing," *Proc. MobiCom*, pp. 70-84, 2001.
[30] F. Xue and P.R. Kumar, "The Number of Neighbors Needed for Connectivity of Wireless Networks," *Wireless Networks*, vol. 10, no. 2, pp. 169-181, 2004.
[31] Y. Yang, D. Lee, M. Park, and H. Peter, "Dynamic Enclose Cell Routing in Mobile Sensor Networks," *Proc. IEEE Software Eng. Conf.*, pp. 736-737, 2004.
[32] F. Zhao, J. Liu, J. Liu, L. Guibas, and J. Reich, "Collaborative Signal and Information Processing: An Information Directed Approach," *Proc. IEEE*, pp. 1199-1209, 2003.
[33] Y. Zou and K. Chakrabarty, "Sensor Deployment and Target Localization Based on Virtual Forces," *Proc. INFOCOM*, pp. 1293-1303, 2003.
[34] Y. Zou and K. Chakrabarty, "A Distributed Coverage- and Connectivity-Centric Technique for Selecting Active Nodes in Wireless Sensor Networks," *IEEE Trans. Computers*, vol. 54, pp. 978-991, 2005.
[35] IETF Mobile Ad Hoc Networks (MANET) Charter, http://www.ietf.org/html.charters/manet-charter.html, June 2006.

Yi Zou received the BE degree in electrical engineering from Si Chuan University, People's Republic of China, in 1997, the ME degree in electrical engineering from Nanyang Technological University, Singapore, in 2000, and the PhD degree in computer engineering from Duke University in 2004. Upon completion of the PhD degree, he worked as a research associate at Duke University. Since August 2005, he has been with Unitrends Software Corporation, Columbia, South Carolina, where he is responsible for developing networked backup/restore systems. Before his PhD program, he was working as a development engineer for VPN products at CE-Infosys Pte. Ltd., Singapore. In the summer of 2003, he was an intern at Xerox PARC, Palo Alto, California, where he was working on target tracking in wireless sensor networks. His research interests include mobile ad hoc networks, wireless communication, embedded system computing, and robotics. He is a coinventor of a pending US patent on wireless sensor networks. He has published five journal papers, 13 conference papers, and contributed to five books. He has also served as an organization committee member for the IEEE WICON '05 Workshop and as a technical committee member for IEEE ICWMC '06. He is a member of the IEEE.

Krishnendu Chakrabarty received the BTech degree from the Indian Institute of Technology, Kharagpur, in 1990, and the MSE and PhD degrees from the University of Michigan, Ann Arbor, in 1992 and 1995, respectively, all in computer science and engineering. He is now a professor of electrical and computer engineering at Duke University. Dr. Chakrabarty is a recipient of the US National Science Foundation Early Faculty (CAREER) award and the US Office of Naval Research Young Investigator award. His current research projects include testing and design-for-testability of system-on-chip integrated circuits, microfluidic biochips, microfluidics-based chip cooling, and wireless sensor networks. He has authored four books—*Microelectrofluidic Systems: Modeling and Simulation* (CRC Press, 2002), *Test Resource Partitioning for System-on-a-Chip* (Kluwer, 2002), *Scalable Infrastructure for Distributed Sensor Networks* (Springer, 2005), and *Digital Microfluidics Biochips: Synthesis, Testing, and Reconfigutaion Techniques* (CRC Press, 2006)—and edited the book volumes *SOC (System-on-a-Chip) Testing for Plug and Play Test Automation* (Kluwer, 2002) and *Design Automation Methods and Tools for Microfluidics-Based Biochips* (Springer, 2006). He is also an author of the forthcoming book *Adaptive Cooling of Integrated Circuits Using Digital Microfluidics* (Artech House, April 2007). He has contributed over a dozen invited chapters to book volumes and published more than 240 papers in archival journals and refereed conference proceedings. He holds a US patent in built-in self-test and is a coinventor of a pending US patent on sensor networks. He is a recipient of best paper awards at the 2007 IEEE International Conference on VLSI Design, the 2005 IEEE International Conference on Computer Design, and the 2001 IEEE Design, Automation and Test in Europe (DATE) Conference. He is also a recipient of the Humboldt Research Fellowship, awarded by the Alexander von Humboldt Foundation, Germany, and the Mercator Visiting Professorship, awarded by the Deutsche Forschungsgemeinschaft, Germany. Dr. Chakrabarty is a distinguished visitor of the IEEE Computer Society for 2006-2007 and a distinguished lecturer of the IEEE Circuits and Systems Society for 2006-2007. He is an associate editor of the *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, the *IEEE Transactions on VLSI Systems*, the *IEEE Transactions on Circuits and System I*, the *IEEE Transactions on Biomedical Circuits and Systems*, and the *ACM Journal on Emerging Technologies in Computing Systems*. He is also an editor of *IEEE Design & Test of Computers* and the *Journal of Electronic Testing: Theory and Applications* (JETTA). He is a member of the editorial board for the *Microelectronics Journal*, *Sensor Letters*, and the *Journal of Embedded Computing* and serves as a subject area editor for the *International Journal of Distributed Sensor Networks*. In the recent past, he has also served as an associate editor of the *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*. He is a senior member of the IEEE, a senior member of the ACM, and a member of Sigma Xi. He serves as the chair of the emerging technologies subcommittee for the IEEE International Conference on Computer Aided Design (2005-2007), the subcommittee for new, emerging, and specialized technologies for the IEEE/ACM Design Automation Conference (2006-2007), and the subcommittee on BIST/DFT for the IEEE/ACM Design, Automation and Test in Europe (DATE) Conference (2008). He served as the tutorials chair for the 2005 IEEE International Conference on VLSI Design, as program chair for the 2005 IEEE Asian Test Symposium, and as program chair for the CAD, Design, and Test Conference for the 2007 IEEE Symposium on Design, Integration, Test, and Packaging of MEMS/MOEMS (DTIP 2007). He delivered keynote talks at the International Conference & Exhibition on Micro Electro, Opto, Mechanical Systems and Components (Munich, Germany, October 2005), the International Conference on Design and Test of Integrated Systems (Tunis, Tunisia, September 2006), the IEEE Workshop on Design and Diagnostics of Electronic Circuits and Systems (Krakow, Poland, April 2007), as well as invited talks on biochips CAD at several other conferences.