A Statistical Approach to Contour Line Estimation in Wireless Sensor Networks With Practical Considerations

Pei-Kai Liao, *Member, IEEE*, Min-Kuan Chang, *Member, IEEE*, and C.-C. Jay Kuo, *Fellow, IEEE*

***Abstract*—Contour line estimation in a monitored physical phenomenon using wireless sensor networks (WSNs) with cross-layer considerations is investigated in this paper. Herein, we present a statistical approach to locate contour points for contour line construction in the presence of irregular sensor deployment with the assistant of Delaunay triangular meshes. The proposed contour-estimation algorithm has certain error-correction capability, and it is robust to the data degradation coming from sensing and/or communication noise. To ensure the success of the proposed algorithm, we further analyze three sources of signal distortion: sensing noise, data quantization error, and data communication noise. We find that choosing a proper data quantization level can balance the communication cost and the desired signal quality. Meanwhile, the impact of channel impairment can be mitigated by the appropriate data-fusion mechanism. In this paper, an adaptive data-fusion mechanism is proposed to avoid excessive packet retransmissions. The correctness of the proposed approach is verified through simulation, and the effects of different system parameters on the overall system performance are also given to show the robustness of the proposed algorithm. This paper also offers some guidelines on the design of the WSN system.**

***Index Terms*—Contour estimation, contour lines, Delaunay triangulation, environmental monitoring, triangular meshes, Voronoi diagram, wireless sensor networks (WSNs).**

I. Introduction

WIRELESS sensor networks (WSNs) have become an area of active research due to their wide range of applications and their ability of providing real-time information. Environmental monitoring is one of the important applications in WSNs, where sensors are deployed to observe natural phenomena in real time. The collected sensor data provide a set of sampled values of the underlying 2-D field. However, in applications such as monitoring the density of a certain chemical such as dioxin, the variation of carbon dioxide in a forest, etc., only a set of the sampled values cannot provide the researchers or authority agencies enough information to promptly deal with a monitored event. Adopting the notion of contour line helps to visually correlate the measured data with the geographical information and to locally or globally estimate the impact or the implication of a monitored event. Hence, it is more than desirable to use a contour line to reconstruct the 2-D scalar field based on these sampled values. This is, however, a challenging problem due to the irregular topology of the deployed sensors and the local communication constraint among sensors. In this paper, we examine the contour line-estimation problem using WSNs in the presence of irregular sensor deployment.

There has been much research activity on the monitoring of natural phenomena. Nowak and Mitra [1] proposed a boundary estimation scheme using a sensor network with a hierarchical topology. Chintalapudi and Govindan [2] proposed several localized methods to detect the edge nodes of the monitored physical phenomena. More recently, a statistical decision-fusion approach to edge node detection was proposed in [3] to further improve the detection performance and reduce the sensitivity to sensor localization errors. However, these algorithms have focused on the existence of a phenomenon (i.e., in the ON- or OFF-state). Their generalization to the contour estimation problem is, however, not straightforward due to the many special constraints associated with the WSN.

More recently, localized methods were proposed in [4] and [5] to directly extract the information of contour lines from noisy sensor measurements. These methods require the deployment of sensor nodes of two types with a two-layer hierarchical architecture. That is, several expensive sensor nodes form a coarse-layer network on top of the conventional sensor network. To build up and maintain this network hierarchy, additional wireless communication and protocol processing are needed, which means increased energy consumption. This makes their approaches less attractive in WSNs. Thus, the objective of this paper is to develop a new contour line-extraction algorithm when sensor nodes involved in the contour line extraction are of the same type.

We propose a statistical approach to estimate the contour line by exploiting the concept of Delaunay triangular meshes, which have been applied to WSNs in many areas such as network graph construction [6], deployment [7], coverage [8], [9], ranging [10], [11], and so on. The main advantage is that it can localize true contour lines with irregular (or random) sensor deployment with no predefined hierarchical network

Manuscript received August 19, 2007; revised May 13, 2008, October 4, 2008, January 14, 2009, and February 18, 2009. First published March 6, 2009; current version published August 14, 2009. The review of this paper was coordinated by Dr. O. B. Akan.

P.-K. Liao is with MediaTek Inc., Hsinchu 30078, Taiwan (e-mail: liao.peikai@gmail.com).

M.-K. Chang is with the Department of Electrical Engineering and the Graduate Institute of Communication Engineering, National Chung Hsing University, Taichung 402, Taiwan (e-mail: minkuanc@dragon.nchu.edu.tw).

C.-C. J. Kuo is with Ming Hsieh Department of Electrical Engineering and the Signal and Image Processing Institute, University of Southern California, Los Angeles, CA 90089 USA (e-mail: ckuo@sipi.usc.edu).

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

Digital Object Identifier 10.1109/TVT.2009.2016666

topology. In addition, to make the contour line possible, the identification of contour points, which are connected to form a contour line and will formally be defined later, is necessary. The locations of the estimated contour points could be erroneous due to several reasons, including sensing noise, channel communication noise, and scarcity of sensor nodes. To address this problem, an error compensation scheme is developed to correct the detection errors to some extent so that the fidelity of the final estimation results can be enhanced. As compared with the previous works in [4] and [5], the proposed triangular-mesh-based method is more flexible and robust.

Furthermore, we are also interested in a cross-layer solution that takes realistic environment settings into account. For example, to enhance the communication power efficiency, we may want to avoid excessive data packet retransmission at the cost of slightly worse system performance (i.e., the accuracy of extracted contour lines). Another example is that we may apply the forward error correction (FEC) scheme, which increases the system complexity in the physical layer, to save transmission power. Clearly, there exists a tradeoff among system performance, communication power, and system complexity. The proper choice depends on system requirements. Several cross-layer design issues are examined. Simulation results are given to demonstrate the effects of various parameters on the overall system performance. The insights obtained from this paper may provide good guidelines to system design in the future.

The rest of this paper is organized as follows. The system model and the model parameters are described in Section II. A contour line-estimation algorithm based on a triangular mesh structure in the application layer is described in Section III. Cross-layer considerations in system design are discussed and examined in Section IV. Proper parameter settings are discussed, and simulation results are provided to demonstrate the system performance in Section V. Finally, concluding remarks are given in Section VI.

II. System Model

A. Functionalities of Sensor Nodes, Sinks, and Remote Base Station

Consider a WSN deployed over a large geographical extent to monitor a physical phenomenon. Depending on the computing capability and processing power, two types of nodes are deployed in the WSN, namely, the sensor nodes and the sinks. The roles and functionalities of the sensor nodes and the sinks are described in the following.

1) Sensor Nodes: Each sensor node that is equipped with an omnidirectional antenna is mainly responsible for data collection, initial data processing, information dissemination, and data fusion. Each sensor node also acts as a fusion center to fuse the collected data from the sensor nodes within the probing range and makes an initial decision. In this paper, the probing range is defined as the maximum spacing between the fusion center and the sensor node from which this fusion center can collect data, as indicated in Fig. 1. In addition, we assume that the sensor nodes are static.

2) Sinks: The sinks have better computing capability and processing power than the sensor nodes. They act as bridges between the remote base station and the sensor nodes. In addition, we assume that both the sinks and the remote base station have the globe view of the entire WSN. That is, the location information of sensor nodes is assumed to be known by either the sinks or the remote base station. Either the sinks or the remote base station helps the sensor nodes to determine their neighbor nodes by performing Delaunay triangulation [12], [13] based on the location information sent by the sensor nodes. Delaunay triangulation will be discussed in detail in Section III. When the number of the sensor nodes a sink has to handle is reasonable, the sinks perform Delaunay triangulation.

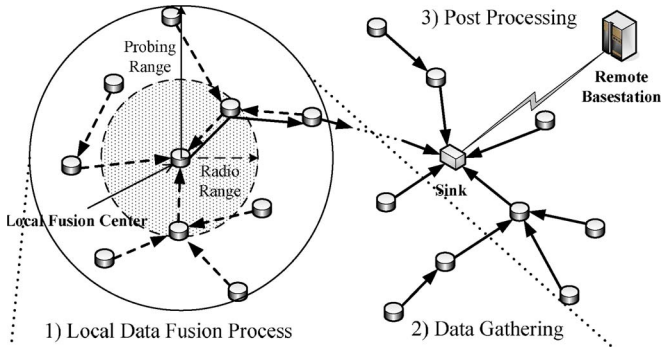

Fig. 1. Measurement collection and information processing procedure in a WSN.

3) Remote Base Station: The remote base station is mainly responsible for coordinating sensor nodes and sinks and performs the final data processing. Delaunay triangulation can be done at the remote base station if this task cannot be handled by the sinks. It is worthwhile to point out that whether the Delaunay triangulation is done at the sinks or at the remote base station, the remote base station always has the overall result of Delaunay triangulation. This will help the remote base station perform the contour line-connection algorithm described later.

B. Measurement Collection and Information Processing

The procedure of measurement collection and information processing is illustrated in Fig. 1. In this paper, this procedure consists of three steps: 1) local data processing/fusion, 2) data gathering, and 3) data postprocessing.

1) Local Data Processing/Fusion: Each sensor node is responsible for monitoring a physical phenomenon and measuring the signal related to this monitored phenomenon. The measured data collected by the sensor nodes are locally exchanged. After the exchange of the local measured data among all the local sensor nodes, each local sensor acts as a fusion center to fuse the measured data from its sensor nodes within the probing range and to extract useful local information to help construct the contour lines of the monitored physical phenomenon.

2) Data Gathering: In the second step, the processed/fused results are forwarded to a sink or a remote base station via a data-gathering tree. The data-gathering tree can be constructed beforehand with the help of the geographic routing protocols [14]–[16].

Another purpose of the data-gathering tree is to coordinate the data collection and the data exchange among sensor nodes. The remote base station sends a command to all the sensor

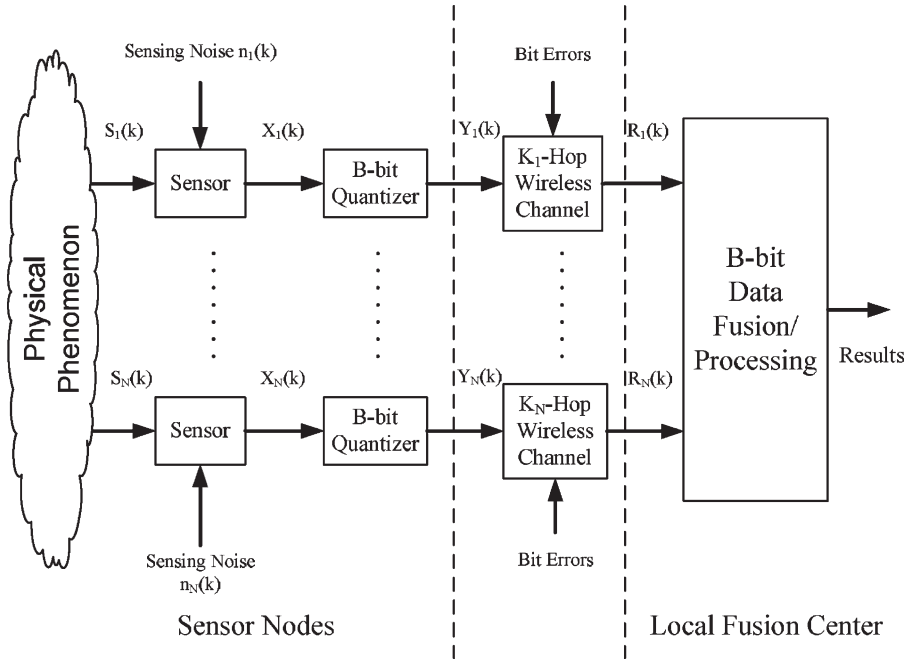

Fig. 2. Block diagram of the local data-fusion process.

nodes via the data gathering tree to ask them to measure the monitored physical phenomenon. After the measurement period is over, the sampled data are exchanged among sensor nodes. The procedure of the sensor nodes exchanging their sampled data is similar to the procedure shown in [17]. The data exchange starts with those sensor nodes at the very bottom of the data-gathering tree. Those sensor nodes then ask the sensor nodes within the probing range to forward the data samples to themselves. Each sensor node can ask the sensor nodes within the probing range to forward their data samples only after all of its child nodes have done so. This procedure continues until it reaches the sinks.

3) Data Postprocessing: Data postprocessing is the responsibility of the remote base station. In this step, the remote base station constructs the contour lines based on the processed and fused data from the sensor nodes via the data-gathering tree.

C. Local Data-Fusion Process

The block diagram of the local data-fusion process in this paper is depicted in Fig. 2. This process is conducted at each sensor node in the WSN, regardless of whether it is a leave or an intermediate node in the data-gathering tree. Highlights of Fig. 2 are given in the following.

1) Local Data Measurement and Quantization: To ensure that the data samples measured by the sensor nodes are sufficient to describe the monitored phenomenon, it is assumed that the largest distance between two adjacent sensor nodes over the sensor field is smaller than one half of the period of the highest frequency component of the monitored phenomenon, i.e., the Nyquist sampling criterion is met. The received measurement from the ith sensor node within the probing range of a local fusion center at time k can be written as

$$R_i(k) = Q_B\left[X_i(k)\right] + n_{c,i}(k) \tag{1}$$

where $Q_B[\cdot]$ is a quantization function with 2^B levels, where the number of quantization bits B may vary according to channel conditions and residual battery power, $n_{c,i}(k)$ is the equivalent noise component induced by communication errors, $X_i(k)$ is the data sample measured at time k at the ith sensor node within the probing range of a local fusion center, which consists of the sampled signal value of the monitored signal $S_i(k)$ by the ith sensor node within the probing range of a local fusion center and the sensing error $n_i(k)$ at time k, and

$$X_i(k) = S_i + n_i(k). \tag{2}$$

In this paper, we assume that the strength of the monitored signal is uniformly distributed over $[S_{\min}, S_{\max}]$, and the measured data sample $X_i(k)$ is quantized to $Y_i(k) = S_{\min} + (\Delta/2) + (j-1)\times\Delta$ if its value falls between $S_{\min}+(j-1)\times\Delta$ and $S_{\min} + j \times \Delta$, where $j = 1, \ldots, 2^B$ is the quantization level, and Δ is the quantization step.

2) Data Forwarding: Since the communication range of each sensor node is limited, it may require multiple hops for the measured data of a sensor node to reach a local fusion center. For the ith sensor node within the probing range of a local fusion center, a K_i-hop wireless communication scheme is used to transmit data collected at this sensor node to this local fusion center, as shown in Fig. 2. Due to the fading channel and wireless channel noise, the received data at the local fusion center might be corrupted. Thus, a couple of transmissions may be needed to meet the data quality requirement at the receiver side. In Section IV, we will discuss related design considerations in a K_i-hop wireless communication scheme. After receiving the measured data, the local fusion center performs data fusion to extract the essential information, which is beneficial for contour line detection.

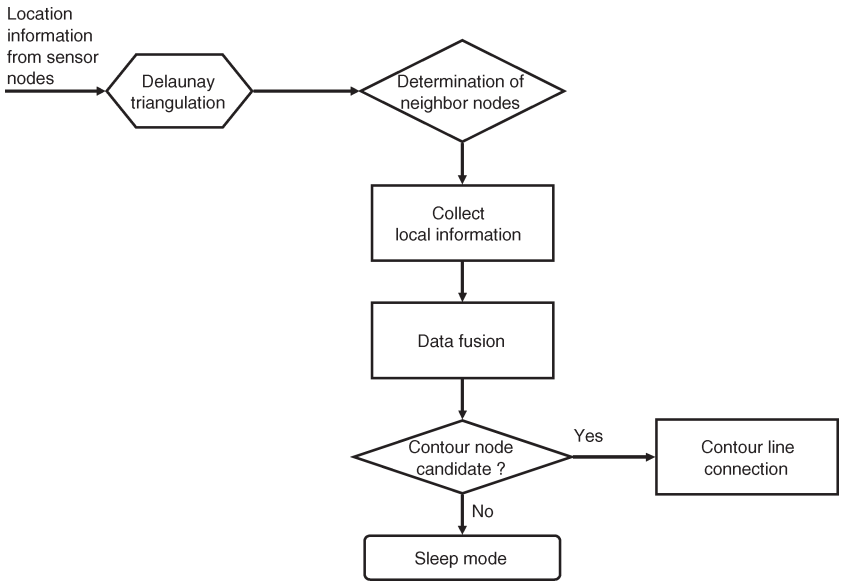

Fig. 3. High-level flowchart of the proposed algorithm.

III. Procedures of Contour Line Extraction

This task consists of three steps: 1) triangulation from randomly deployed nodes; 2) contour point determination; and 3) contour line connection. An overview of the proposed contour line estimation algorithm is given in order according to the high-level flowchart of the proposed algorithm in Fig. 3, which only highlights the important blocks of the procedures of contour line extraction.

First, using the Delaunay triangulation, the sinks or the remote base station forms a triangular mesh based on the location information of the sensor nodes in the system. Generally, the triangulation is conducted only at the beginning of the monitoring process for static networks. The result of the triangulation will be stored in the memory for later usage. After the triangulation, the neighbor nodes of each sensor node in the monitored field can be identified with ease. The information of the neighbor nodes of each sensor node helps the remote base station depict the contour line. In the second step, the local fusion center determines whether it is the contour point candidate for a chosen contour line based on statistical inference from the measured data collected by the sensor nodes within its probing range. The contour point candidates are those who have the largest average likelihood value along the direction and the reverse direction of the gradient, as compared with the sensor nodes within the probing range. In the third step, the remote base station connects the detected contour points with lines to construct the estimated contour line. However, due to the effects of sensing noise, channel noise, and scarcity of sensor nodes, detection results might be erroneous. To lessen these effects, an error-correction scheme is incorporated in this paper as well. Details of these three steps are given in the following sections.

A. Step 1: Triangular Mesh Construction From Randomly Deployed Sensor Nodes

Given a set of randomly deployed sensor nodes, we would like to connect them by a triangular mesh in which each pair of neighbor nodes is connected by exactly one edge. The triangular meshes in this paper serve as good references to determine the location of contour lines of interest. Due to the limited computation capability of each sensor node, the triangular meshes are constructed by either the sink or the remote base station via the Delaunay triangulations. To begin with, we define the neighbor nodes as follows.

Definition 1—Neighbor Nodes: Two nodes are called neighbor nodes if their Dirichlet regions are adjacent to each other, where the Dirichlet region of a sensor node located at $\vec{p}_i$ is defined as

$$D_{\vec{p}_i}(P) = \left\{\vec{z} \in \mathbb{R}^2 : d(\vec{z}, \vec{p}_i) \leq d(z, \vec{p}_j), j \neq i, \vec{p}_j \in P\right\}$$

where $P = \{\vec{p}_1, \vec{p}_2, \ldots, \vec{p}_M\}$ are the locations of sensor nodes of consideration, and $d(z, \vec{p}_i)$ is the Euclidean distance between point $\vec{z}$ and the sensor node at $\vec{p}_i$.

The determination of neighbor nodes assists the remote base station to connect the contour points, which are defined later, and draw the contour line. In addition, the information of the neighbor nodes of a sensor node also helps make the final judgement of whether this sensor node is crucial to the construction of the contour line.

Based on this definition, one can determine neighbor nodes using the Voronoi diagram. It is well known that the Delaunay triangulation is a geometry dual of the Voronoi diagram. Thus, a direct way to construct a triangular mesh and, hence, determine the neighbor nodes is to apply the Delaunay triangulation. This algorithm, which was originally proposed by Green and Sibson [12] and further modified by Boissonnat and Teillaud [13], is given in Algorithm 1. See [13] for more details. The average computation time of the Delaunay triangulation is shown in Fig. 4. From Fig. 4, we can see that the average computation time of the Delaunay triangulation is reasonable. Thus, the sinks in the base station are able to handle the computational load caused by the Delaunay triangulation (see Fig. 5).

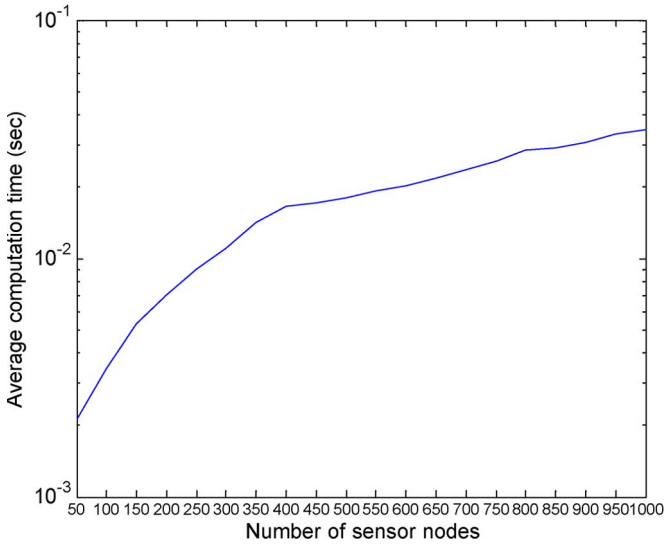

Fig. 4. Average computation time of the Delaunay triangulation.

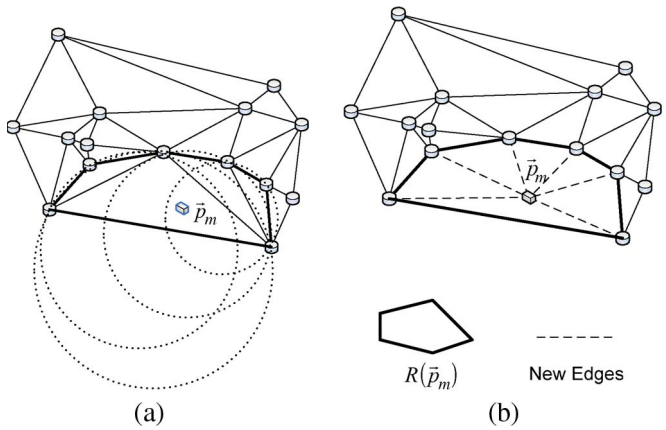

Fig. 5. Delaunay triangulation algorithm. (a) Point increment in Step 2. (b) Triangle update in Step 3.

Algorithm 1 Algorithm for Constructing the Delaunay Triangulations [13]

Let $P = \{\vec{p}_1, \vec{p}_2, \ldots, \vec{p}_M\}$ be a set of points in a 2-D plane.

Step 1. Initialization:

Pick any three points, $\vec{p}_i$, $\vec{p}_j$, and $\vec{p}_k$, in set P to form an initial triangle.

Step 2. Point Increment:

To add point $\vec{p}_m$, which is not in the point set of the current graph, to the graph, we have to verify if there is any triangle in $R(\vec{p}_m)$ whose Delaunay circle, which is the circumcircle of this triangle, contains $\vec{p}_m$. If there is, go to Step 3. This is illustrated in Fig. 5(a).

Step 3. Triangle Update:

Remove all inner edges of region $R(\vec{p}_m)$, and connect the vertices of polygon $R(\vec{p}_m)$ to point $\vec{p}_m$ such that several new triangles are formed with $\vec{p}_m$ as the common vertex. This is illustrated in Fig. 5(b).

Step 4. Iteration:

Iterate Steps 2 and 3 until all points in P have been added to the graph.

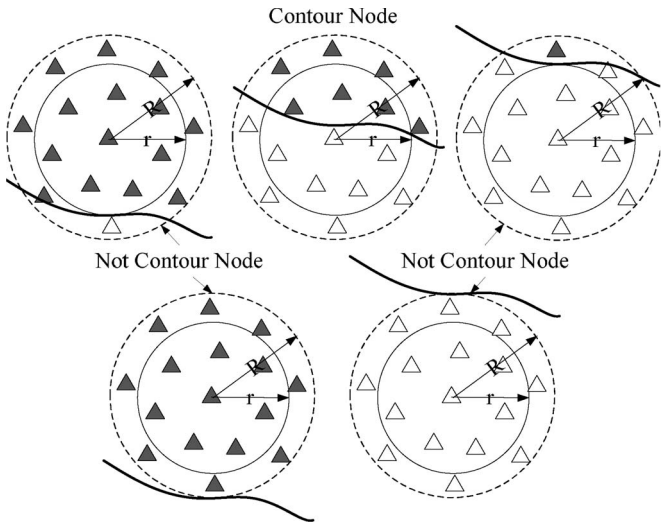

Fig. 6. Contour line crossing the neighborhood of a target sensor node, where a contour point candidate is shown in the middle of the top row, whereas the others are not contour point candidates.

B. Step 2: Determination of Contour Point Candidates

In this section, we will elaborate how the data are fused at the local fusion sensor. Let us start with defining the contour point. A contour point can be viewed as a generalized edge node [3] and defined as follows.

Definition 2—Contour Point: If the distance between a contour line of interest and a sensor node is less than r, then the sensor node is said to be a contour point for this contour line, where r is called the tolerance range.

With this definition, we can develop a statistical method to infer the probability for a sensor node of being a contour point, which is called the contour point probability. This probability also enables us to determine when a sensor node should enter the sleep mode and when it should be awake. Those sensor nodes with lower contour point probabilities can switch to sleep mode for power saving, and they will not participate in any further calculation until the next estimation cycle begins. On the other hand, those sensor nodes with higher contour point probabilities stay awake and help estimate the contour line.

Five possible scenarios for a contour line to cross the neighborhood of a sensor node are illustrated in Fig. 6, where R is the probing range (or data collection range), r is the tolerance range, and the gray and white triangles represent sensor nodes with their signal values smaller and larger than that of the contour line of interest (called the target contour line), respectively. As we can see from Fig. 6, a sensor node can be a contour point candidate if and only if those sensor nodes inside the tolerance range and inside the probing range can be divided into two separate and distinct groups by the target contour line, as illustrated in the middle part of the top row in that figure. In that illustration, one group consists of those sensor nodes with signal values that are larger than that of the target contour line, and the rest fall into the other group. There is no overlapping between these two groups. This observation motivates us to formulate and calculate the contour

point probability of a sensor node, as derived in the following sections.

1) Derivation of Contour Point Probability: Let E_1 be the event that a certain sensor node (which is called a target sensor node hereafter) is a contour point, and let $E_{0,r}$ (or $E_{0,R}$) be the event that the contour line of interest does not cross the tolerance range (or the probing range). Assume that there are N sensor nodes within the probing range of a target sensor node with N observations. These N observations are then sorted according to the distances between sensor nodes and the target sensor node in ascending order and are denoted by $Y_1(k), \ldots, Y_N(k)$. Recalling from Fig. 2, we have

$$Y_i(k) = Q_B\left[X_i(k)\right] \tag{3}$$

where $X_i(k)$ is defined in (2). Given $Y_1(k), \ldots, Y_N(k)$, the conditional probability for a target sensor node to be a contour point can be calculated by

$$\begin{aligned} &P\left\{E_1|Y_1(k),\ldots,Y_N(k)\right\} \\ &\quad = \left[1 - P\left\{E_{0,r}|Y_1(k),\ldots,Y_{N_r}(k)\right\}\right] \\ &\qquad \times \left[1 - P\left\{E_{0,R}|Y_{N_r+1}(k),\ldots,Y_N(k)\right\}\right] \\ &\quad = \left[1 - \prod_{i=1}^{N_r} P\left\{S_i(k) > \mu|Y_i(k)\right\}\right. \\ &\qquad \left. - \prod_{i=1}^{N_r} P\left\{S_i(k) < \mu|Y_i(k)\right\}\right] \\ &\qquad \times \left[1 - \prod_{i=N_r+1}^{N} P\left\{S_i(k) > \mu|Y_i(k)\right\}\right. \\ &\qquad \left. - \prod_{i=N_r+1}^{N} P\left\{S_i(k) < \mu|Y_i(k)\right\}\right] \end{aligned} \tag{4}$$

where μ is the signal value of the target contour line, N_r is the number of sensor nodes inside the tolerance range of a target sensor node, and

$$\begin{aligned} P\left\{S_i(k) > \mu|Y_i(k)\right\} &= \int_{\mu}^{\infty} f\left(S|Y_i(k)\right) dS \\ &= \left.\frac{\int_{\mu}^{\infty} P(Y|S) f(S) dS}{\int_{-\infty}^{\infty} P(Y|S) f(S) dS}\right|_{Y=Y_i(k)} \end{aligned} \tag{5}$$

$$\begin{aligned} P\left\{S_i(k) < \mu|Y_i(k)\right\} &= \int_{-\infty}^{\mu} f\left(S|Y_i(k)\right) dS \\ &= \left.\frac{\int_{-\infty}^{\mu} P(Y|S) f(S) dS}{\int_{-\infty}^{\infty} P(Y|S) f(S) dS}\right|_{Y=Y_i(k)}. \end{aligned} \tag{6}$$

Recall that $S_i(k)$ is uniformly distributed over $[S_{\min}, S_{\max}]$. Hence, (5) and (6) reduce to

$$P\left\{S_i(k) > \mu|Y_i(k)\right\} = \frac{\int_{\mu}^{S_{\max}} P\left(Y_i(k)|S\right) dS}{\int_{S_{\min}}^{S_{\max}} P\left(Y_i(k)|S\right) dS} \tag{7}$$

$$P\left\{S_i(k) < \mu|Y_i(k)\right\} = \frac{\int_{S_{\min}}^{\mu} P\left(Y_i(k)|S\right) dS}{\int_{S_{\min}}^{S_{\max}} P\left(Y_i(k)|S\right) dS}. \tag{8}$$

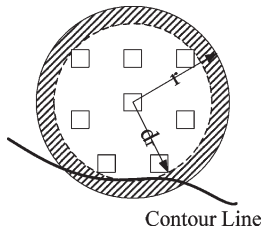

Fig. 7. Example of a missed contour point.

Further simplification of (7) and (8) requires the conditional probability $P(Y_i(k)|S)$. Since $Q_B[\cdot]$ is not a linear function, the statistical properties of $Y_i(k)$ cannot easily be obtained from the linear combination of $X_i(k)$ directly. However, recall that $X_i(k)$ is quantized to $Y_i(k) = S_{\min} + (\Delta/2) + (i-1) \times \Delta$ if its value falls between $S_{\min} + (i-1) \times \Delta$ and $S_{\min} + j \times \Delta$, where $i = 1, \ldots, 2^B$ is the quantization level, and Δ is the quantization step. Based on this, we are able to compute the conditional probability $P(Y_i(k)|S)$. Let $n_i(k)$ be the zero-mean Gaussian white noise with variance σ_n. Then, $P(Y_i(k)|S)$ can be calculated by

$$\begin{aligned} &P\left(Y_i(k) = \left(S_{\min} + \frac{\Delta}{2}\right) + (j-1) \times \Delta \,\middle|\, S\right) \\ &\quad = P\left\{S_{\min} + (j-1) \times \Delta < X_i(k)\right. \\ &\qquad \left. \le S_{\min} + j \times \Delta \,|\, S\right\} \\ &\quad = P\left\{S_{\min} + (j-1) \times \Delta - S < n_i(k)\right. \\ &\qquad \left. \le S_{\min} + j \times \Delta - S \,|\, S\right\} \\ &\quad = Q\left(\frac{\left(Y_i(k) - \frac{\Delta}{2}\right) - S}{\sigma_n}\right) - Q\left(\frac{\left(Y_i(k) + \frac{\Delta}{2}\right) - S}{\sigma_n}\right) \end{aligned} \tag{9}$$

where $Q(\cdot)$ is the tail probability of a standard Gaussian distribution. The value of σ_n can locally be estimated using long-term noise power estimation.

By calculating (5)–(9), we can determine the probability for a target sensor node to be a contour point of signal level T based on quantized measurements $Y_i(k)$ via (4). If the quality of the received data $Y_i(k)$ is poor, the estimation performance may degrade due to data distortion. In this situation, the data diversity (i.e., more data obtained from different sensor nodes) can be used to enhance the performance. In addition, note that if $n_i(k)$ is white but non-Gaussian, then $P(Y_i(k)|S)$ can be computed by

$$\begin{aligned} &P\left(y = \left(S_{\min} + \frac{\Delta}{2}\right) + (i-1) \times \Delta \,|\, S\right) \\ &\qquad = \int_{S_{\min} + (i-1) \times \Delta - S}^{S_{\min} + i \times \Delta - S} f_n(t)\, dt \end{aligned} \tag{10}$$

where $f_n(t)$ is the probability density function of this non-Gaussian noise.

2) Correction Due to Sparse Deployment: When the sensor deployment is sparse, there is a possibility that a contour point may not be detected, even if the contour line of interest does cross its tolerance range, as shown in Fig. 7, where r is the tolerance range, and d_r is the distance between the target sensor node and its farthest sensor node inside the tolerance range.

In this scenario, the target node fails to detect the contour line of interest, because the contour line passes through the slashed area, and those sensor nodes within the tolerance range in this scenario cannot be separated into two distinct groups by the contour line. To address this problem, the probability in (4) has to be modified. The probability of the existence of a contour line in the tolerance range is modified to the probability that the contour line does not pass the slashed region in the tolerance range, and not all sensor nodes within this range have signal values lower than T. Hence, this probability is

$$\left(1 - P\left\{E_{0,r} \,|\, Y_1(k), \ldots, Y_{N_r}(k)\right\}\right) \times \left(1 - P_{\overline{P},r}(d_r)\right)$$

where

$$P_{\overline{P},r}(d_r) = \left(\frac{d_r}{r}\right)^{2N_r} \times \left(\frac{r - d_r}{r}\right). \tag{11}$$

Since $d_r < r$, $P_{\overline{P},r}(d_r)$ becomes smaller when N_r is larger. In other words, the missed probability can be reduced if the sensor density is higher.

A similar scenario could happen in the probing range of a target sensor node. In such a scenario, the probability of the existence of a contour line in the probing range is modified to

$$\left(1 - P\left\{E_{0,R} \,|\, Y_{N_r+1}(k), \ldots, Y_N(k)\right\}\right) \times \left(1 - P_{\overline{P},R}\left(d_R\right)\right)$$

where d_R is distance between the target sensor node and the farthest sensor node in the probing range, and

$$P_{\overline{P},R}(d_R) = \left(\frac{d_R}{R}\right)^{2N} \times \left(\frac{R - d_R}{R}\right). \tag{12}$$

Consequently, the probability for a sensor node to be a contour node can be modified and shown as

$$\begin{aligned}
&\tilde{P}\left\{E_1 \,|\, Y_1(k), \ldots, Y_N(k)\right\} \\
&\quad = \left[\left(1 - P\left\{E_{0,r} \,|\, Y_1(k), \ldots, Y_{N_r}(k)\right\}\right) \times \left(1 - P_{\overline{P},r}\left(d_r\right)\right)\right] \\
&\qquad \times \left[\left(1 - P\left\{E_{0,R} \,|\, Y_{N_r+1}(k), \ldots, Y_N(k)\right\}\right)\left(1 - P_{\overline{P},R}\left(d_R\right)\right)\right].
\end{aligned} \tag{13}$$

It is worthwhile to point out that when $d_r = r$ and $d_R = R$, (4) and (13) coincide.

3) Further Screening of Contour Point Candidates: Sensor nodes with high contour point probabilities, as calculated in (4) or (13), serve as good contour point candidates. We can further screen these candidates. Before that, first, we define that the spatial average log-likelihood estimate of the sensor node $\chi(\vec{p}_0)$ located at $\vec{p}_0$ can be computed via

$$L_{\vec{p}_0}(k) = C_1 - \frac{C_2}{N+1} \sum_{\ell=0}^{N} \left(X_{\chi(\vec{p}_\ell)}(k) - \mu\right)^2 \tag{14}$$

where $\chi(\cdot)$ is a function that maps the location of a sensor node to its index, and its C_1 and C_2 are arbitrary values, which would not affect the system performance, $X_{\chi(\vec{p}_l)}(k)$ is the data sample measured by the ℓth sensor node at time k, $\ell = 0, 1, \ldots, N$, and μ is the signal value of the contour line of interest.

From the definition of the spatial average log-likelihood estimate, when a sensor node is close to the contour line of interest, its spatial average log-likelihood estimate should be large. Hence, from this observation, a target sensor node is a true contour point if it possesses the largest spatial average log-likelihood estimate, as compared with its neighbor nodes along the positive and negative gradient directions. In other words, we can compute the gradient direction and the spatial average log-likelihood estimates for sensor nodes along this direction to screen these candidates.

Suppose there are N sensor nodes within the probing range of a target sensor node, where N is dependent on the deployment of sensor nodes. The gradient direction at a target sensor node with location $\vec{p}_0$ can be estimated via

$$\widehat{\vec{\theta}}(\vec{p}_0) = \left[H(\vec{p}_0, N)^T H(\vec{p}_0, N)\right]^{-1} H(\vec{p}_0, N)^T Z_d(\vec{p}_0, N) \tag{15}$$

where $Z_d(\vec{p}_0, N)$ is an $N \times 1$ vector whose ith element is $(X_{\chi(\vec{p}_i)}(k) - X_{\chi(\vec{p}_0)}(k))/|\vec{p}_i - \vec{p}_0|$, $H(\vec{p}_0, N)$ is also an $N \times 1$ vector whose ith element is $(\vec{p}_i - \vec{p}_0)/|\vec{p}_i - \vec{p}_0|$, $\vec{p}_i$'s, $i = 1, 2, \cdots, N$ are the spatial coordinates of those sensor nodes within the probing range of the target sensor node $\vec{p}_0$ within the probing range, $X_{\chi(\vec{p}_0)}(k)$ is the data sample of the target sensor, and $X_{\chi(\vec{p}_i)}(k)$ for $i = 1, 2, \ldots, N$ are data samples of the sensor nodes within the probing range of the target sensor node. Our goal now is to find two special neighbor nodes along the positive and negative gradient directions at this target sensor node and compare their spatial average log-likelihood estimates with the target sensor node. With the knowledge of the Delaunay triangulation structure, we can obtain the information of the neighbor nodes of the target sensor node. Let $\vec{p}_{a_i}$ be the coordinate of the ith neighbor node for $i = 1, 2, \ldots, L$, where L is the total number of neighbor nodes of the target sensor node.

Then, we can calculate the inner product of the estimated unit gradient vector and the following unit vector:

$$\frac{\vec{p}_{a_i} - \vec{p}_0}{|\vec{p}_{a_i} - \vec{p}_0|}, \qquad i = 1, \ldots, L \tag{16}$$

and choose neighbor nodes resulting in the largest or smallest values of this inner product along the direction and the reverse direction of the gradient at this target sensor node, respectively. After finding these two neighbor nodes, we compare their spatial average log-likelihood estimates with that of the target sensor node. If the target sensor node possesses the largest value, then it is more likely to be the desired contour point. The overall contour point candidate algorithm is outlined in Algorithm 2.

Algorithm 2 Contour Point Candidate Algorithm

Step 1. Initialization.

A. Construct the data-gathering tree.

B. The sinks/remote base station acquires location information of sensor nodes.

C. Sinks/remote base station constructs the Delaunay triangulations.
D. Neighbor nodes of each sensor node are determined.

Step 2. Collection data.
A. The remote base station sends a command to initiate the data-collection process.
B. Each sensor node starts to measure the monitored physical phenomenon.
C. Sensor nodes at the very bottom of the data gathering tree begin ask the sensor nodes within its probing range to forward the data samples to themselves. A sensor node can ask the sensor nodes within its probing range to forward the data samples only after all its child nodes along the data gathering tree have done so.
D. This process continues until all the sensor nodes have collected their necessary data.

Step 3. Contour point candidates.
A. Calculation of contour point probability of each sensor node.
Case 1: Dense deployment of sensor nodes.
The contour point probability of a sensor node can be calculated via (4)–(9).
Case 2: Sparse deployment of sensor nodes.
The contour point probability of a sensor node can be calculated via (5)–(9) and (11)–(13).
B. Determination of contour point candidates. When the contour point probability of a sensor node is greater than a certain threshold, this sensor node is classified as a contour point candidate.

Step 4. Screening of contour point candidates.
A. Use (15) to calculate the unit gradient vector at a target sensor node.
B. Use (16) to compute the vectors between the target sensor and its neighbor nodes.
C. Find the inner products between the unit gradient vector and vectors found in 4.B.
D. Search two neighbor nodes yielding the largest and the smallest values found in 4.C.
E. Find the likelihood values (14) of the target sensor node and those two nodes found in 4.D.
F. A target sensor node is a true contour point candidate if and only if it has the largest likelihood value compared to those two sensor nodes found in 4.D.

Step 5. Termination.
Information of contour point candidates is sent back to the remote base station via the data-gathering tree for contour line connection.

C. Step 3: Contour Line Connection

In the third step, we attempt to connect the detected contour points with lines. The basic idea can be stated as follows. To initialize, we choose a node from the list of unprocessed contour points with the largest average likelihood value and look for its neighbor nodes in the unprocessed list. Among all the neighbor nodes, we choose the one that has a connection direction that is perpendicular to the estimated local gradient direction as much as possible. The above connection process is repeated until all the nodes in the list of unprocessed contour points are connected or until no further connection can be made. Sometimes, the current node does not have any neighbor node in the unprocessed list but has a two-hop neighbor node in the list. Then, we accommodate this case and make the connection as well.

A detailed description of the contour line connection algorithm is given in Algorithm 3. With the proposed contour line connection algorithm, errors resulting from the two-hop missing (disconnections) and the one-hop false alarm (improper branches) can be corrected if the detected contour points do not contain two successive errors and the estimation error of local gradients is small. This algorithm can yield two or more contour lines if they are separated by at least three hops away from each other.

Algorithm 3 Contour Line Connection Algorithm

Step 1. Initialization.
Choose a node with the largest average log-likelihood value from the list of unprocessed contour points, which is denoted by $L_{\rm ucn}$, as the head point of the current contour line segment.

Step 2. Connection point search.
We use C to denote the current node under processing, which belongs to the contour line segment S. Check whether C has any neighbor node in $L_{\rm ucn}$.
Case 1: No. Check if C has any two-hop neighbor node that is in $L_{\rm ucn}$.
A. No. C is the end node of segment S, $L_{\rm ucn} = L_{\rm ucn}/\{C\}$, and go to Step 3.
B. Yes. Look for the intermediate node that connects C and its two-hop neighbor nodes and is nearly perpendicular to the estimated local gradient.
i) No. C is the endpoint of S, $L_{\rm ucn} = L_{\rm ucn}/\{C\}$, and go to Step 3.
ii) Yes. Denote this intermediate node by M, C and M are points for segment S, $L_{\rm ucn} = L_{\rm ucn}/\{C\}$, $C \leftarrow M$, and go to Step 2.
Case 2: If there is one, denote this node by M, and check if the connection direction is adequate.
A. Yes. C is a point in segment S, $L_{\rm ucn} = L_{\rm ucn}/\{C\}$, $C \leftarrow M$, and go to Step 2.
B. No. C is the endpoint of segment S, $L_{\rm ucn} = L_{\rm ucn}/\{C\}$, and go to Step 3.
Case 3: If there are two or more, we look for the node that has a proper connection direction.
A. If there is a proper one denoted by P, C is a point in segment S, $L_{\rm ucn} = L_{\rm ucn}/\{C\}$, $C \leftarrow P$, and go to Step 2.
B. If there is no proper one, C is the endpoint of segment S, $L_{\rm ucn} = L_{\rm ucn}/\{C\}$, and go to Step 3.

Step 3. List checking.
Check whether segment S contains only C. If yes, $L_{\rm ucn} = L_{\rm ucn} \bigcup \{C\}$.

Step 4. Termination.
Check if there is any unprocessed contour point in $L_{\rm ucn}$. If yes, go to Step 1. Otherwise, connect all obtained contour segments, and terminate.

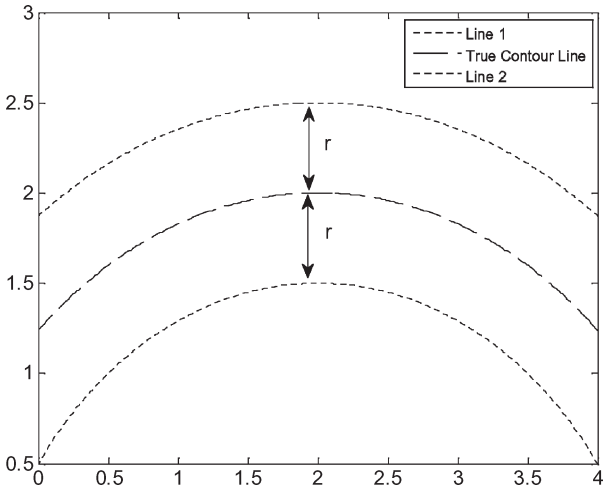

Fig. 8. Tolerance range of the true contour line.

Finally, before we leave this section, it is worthwhile to mention that the proposed algorithm is capable of constructing a contour line that lies within the tolerance range of the true contour line when the information is reliable and its amount is sufficient. The tolerance range is the area between two dotted lines (Lines 1 and 2), as shown in Fig. 8. The width between these two dotted lines is dependent on the tolerance range of a sensor node. If one would like to find a finer contour line, it can be done by choosing a smaller tolerance range of a sensor node r. In addition, the constructed contour line found by the proposed algorithm is deemed to be a good contour line, as long as this constructed contour line is contained in this area.

IV. Design Considerations and Parameter Modeling

In addition to the proposed algorithm for contour-line construction, we will investigate some issues that have an impact on the proposed algorithm in this section, such as how to model key parameters in the sensor system, how the physical (PHY) and medium access control (MAC) layers interact, how to mitigate the distortion due to channel effect, how to forward data, how to achieve power efficiency, and so on. These investigations and parameter selections make the WSN under consideration close to reality. As to some parameters, we will demonstrate their impact on the proposed algorithm through simulation.

A. Parameter Modeling When Considering the MICA2 Mote

In this paper, we consider a real-world sensor system to shed some light on how to model and choose key parameters in sensor systems. From the data sheet of the MICA2 mote [18], the barometric pressure, the humidity, and the temperature sensors have around $\pm 1.5\backslash\%$, $\pm 3.5\backslash\%$, and ± 0.5 °C sensing errors, respectively, at the temperature of 25 °C. Through some manipulations, it is easy to find out that the signal-to-noise ratio (SNR) with respect to sensing noise is greater than 29 dB.

The simplified unit-disk graph model is adopted in the most previous work, where the channel noise effect is neglected. However, the link quality can seriously be affected by channel attenuation in practice. This is particularly true when the link distance is in the transitional region [19]. According to [20], the radio chip of the MICA2 mote Chipcon CC1000 has a noise figure of 13 dB and equivalent bandwidth of 30 kHz. Given the ambient temperature of the receiver, the channel noise power can be calculated by [21]

$$P_n = (F+1)kT_0B$$

where F is the noise figure (in the real number scale), k is Boltzmann's constant, T_0 is the ambient temperature, and B is the equivalent bandwidth. If the ambient temperature is 25 °C, then the channel noise power is -115.87 dBm, whereas the maximal transmission power of the MICA2 mote is 5 dBm [20].

The log-normal shadowing model provides high accuracy in modeling the multipath channel between sensor nodes [22]. It can be written as

$$PL(d) = PL(d_0) + 10n\log_{10}\left(\frac{d}{d_0}\right) + X_\sigma$$

where $PL(d)$ denotes the path loss at distance d, d is the link distance, d_0 is the reference distance, n is the path loss exponent, and X_σ is a zero-mean Gaussian random variable with variance σ^2. Consider the case with $n = 4$, $PL(d_0) = 55$ dB, $d = 25$ m, and $d_0 = 1$ m. Then, the average SNR at the receiver is 9.95 dB. If the link distance d is equal to 20 m, then the SNR increases to 13.83 dB. This shows that the distance and the channel noise have a great impact on the SNR.

Two observations are made here. First, the channel noise and the link distance could lead to significant data degradation if the packet retransmission mechanism and the FEC are not adopted in the MAC and PHY layers, respectively. Second, based on the model in (1), a proper data-fusion algorithm is required to mitigate the effect coming from the data distortion and to enhance the received data quality. These two issues will be handled with care in the following section.

B. PHY and MAC Layers

1) Overall BER of a K-Hop Data Transmission Path: Consider a K-hop data transmission path where the ith hop is a binary symmetric channel with bit error rate (BER) $P_i(E)$, $1 \le i \le K$. The overall BER of a K-hop data transmission path can be calculated as

$$P_{K-\mathrm{hop}}(E) = 1 - \prod_{i=1}^{K}(1 - P_i(E)) \tag{17}$$

where $P_{K-\mathrm{hop}}(C)$ is the probability of correct detection of the entire path. Since each item of the product in the right-hand-side of (17) is less than 1, the overall BER becomes larger as the hop number K increases. In addition, the overall BER is dominated by the worst link. If all single-hop links have the same BER, then the overall BER of this K-hop path becomes

$$P_{K-\mathrm{hop}}(E) = \sum_{i=1}^{K}\binom{K}{i}\cdot(-1)^{i+1}\times P(E)^i. \tag{18}$$

The BER of each hop varies according to the wireless channel condition, the modulation, and the applied FEC schemes. It can be determined by using the E_b/N_0-to-BER mapping, where E_b is the energy per bit, and N_0 is the power density of channel noise. Furthermore, E_b/N_0 can be found by

$$\frac{E_b}{N_0} = \text{SNR} - 10 \times \log_{10}\left(\frac{\text{bit rate}}{\text{bandwidth}}\right) \text{(in decibels)} \quad (19)$$

where SNR is the ratio of the transmitted bit power to the channel noise power.

For layers higher than the physical layer, the packet reception rate (PRR) is of concern. If the error of each bit in a packet is independent, the PRR of a K-hop transmission can be calculated from the BER by [19]

$$\text{PRR}_{K-\text{hop}} = [1 - P_{K-\text{hop}}(E)]^{8f} \quad (20)$$

where f is the packet size (in terms of the byte number). The independent bit error assumption usually holds if a proper interleaving scheme is adopted.

2) Practical Consideration When Considering the MICA2 Mote: We take the noncoherent Binary Frequency Shift Keying (BFSK) modulation, which is used in the MICA2 mote, over the Rayleigh fading channel without FEC as an example. The relationship between BER and E_b/N_0 for noncoherent BFSK modulation is

$$\text{BER} = \frac{1}{2 + E_b/N_0}.$$

The BER is around 10^{-3} at $E_b/N_0 \approx 30$ dB [23]. When the packet size is 100 B,[1] the average attempts of one-hop transmission of a packet to achieve error-free reception is 2.23. As the packet size increases, this number exponentially grows and incurs a greater demand on communication power per hop. Considering now that we have a path consisting of many hops, if the packet size is larger, then we would need an incredibly large amount of power just to have a packet that is correctly received at the intended receiver.

3) Approaches to Reduce Communication Power: There are two methods to lower the communication power consumption. One option is to adopt FEC in the PHY layer, such as the convolutional codes [24] at the expense of higher system complexity, including both computational complexity and memory storage, and overhead. The other possible solution is to develop a robust algorithm to handle the erroneous data in the application layer. That is, if the algorithm is willing to accept distorted data and to sacrifice performance a little bit, then the average number of transmission attempts to have a packet successfully received at the next hop decreases, and less communication power is needed. To incorporate this notion into PRR, let P_{AE} be the acceptable BER of an algorithm, and then, the corresponding acceptable PRR can be expressed as

$$\text{PRR}_{K-\text{hop},A} = \begin{cases} [1-P_{K-\text{hop}}(E)+P_{AE}]^{8f}, & \text{if } P_{K-\text{hop}}(E) > P_{AE} \\ 1, & \text{otherwise.} \end{cases} \quad (21)$$

[1]The packet size ranges from 9 to 127 B in the MAC layer of the IEEE 802.15.4 standard.

When $P_{AE} = 10^{-3}$, and the BER is around 10^{-3}, the average number of transmission attempts reduces to 1, which is much less than 2.23 as previously derived. Both options can reduce the demand on the communication power and have their own pros and cons. In this paper, we adopt the second option to reduce the demand of communication power as well as system complexity. Further, in this paper, we assume that only channel variation can cause the BER without taking the other effect, such as congestion and collision, into account. The assumption is based on the adoption of the spread-spectrum technology in the sensor nodes [25]–[27]. When the spreading code is long enough, the interference between the two signals is low. When the number of spread signals is large, multiaccess interference (MAI) can be modeled as a Gaussian noise, and it only affects the noise level of the received signal. This causes a rise of BER and an increase PRR. Thus, the PRR on which the proposed algorithm is based is reasonable, even when the number of concurrent signals is large.

C. Network Layer

Due to limited communication power, there is a good chance that the direct communication between two sensor nodes is not feasible. When this happens, a routing path should be established prior to the beginning of data exchange. Two issues are of importance when designing routing protocols in WSNs: power efficiency and scalability. There have been several routing protocols with special emphasis on power efficiency [28]. To deal with scalability, the geographical information should be taken advantage of as pointed out in [29] and [30], which can be inferred by sensor localization algorithms [31]–[35] or GPS [36], [37]. How to acquire the location information is beyond the scope of this paper. See [31]–[37] for further information. In this paper, we assume that the location information of a sensor node is always available without error.

The geographical routing protocols make use of the geographical information and promise to be scalable and fault tolerant. Several geographic routing protocols based on different local metrics [14]–[16] were proposed. Karp and Kung [14] presented the Greedy Perimeter Stateless Routing (GPSR) routing protocol that has inspired quite a few geographic routing algorithms in recent years. GPSR uses the hop count as a local metric for route decision. Nevertheless, the hop count alone cannot reflect the true situation that a packet faces during transmission along the chosen route. Hence, Lee *et al.* [15] and Li *et al.* [16] used more realistic local metrics to replace the hop count with functions of PRR and power dissipation. Adopting their metrics, the data are forwarded along the route with the best link quality or the highest power efficiency.

In our system, prior to the dissemination of the data request, sensor nodes within the probing range of a data-requesting node are notified of the probing range and the geographical information of this data requesting node. Upon receiving the data request, these sensor nodes can decide whether they should send their measurements back to the data-requesting node for contour point inference. When they do, the requested data will be forwarded back to the data-requesting node via the route specified by the geographical routing protocols in [15] and [16].

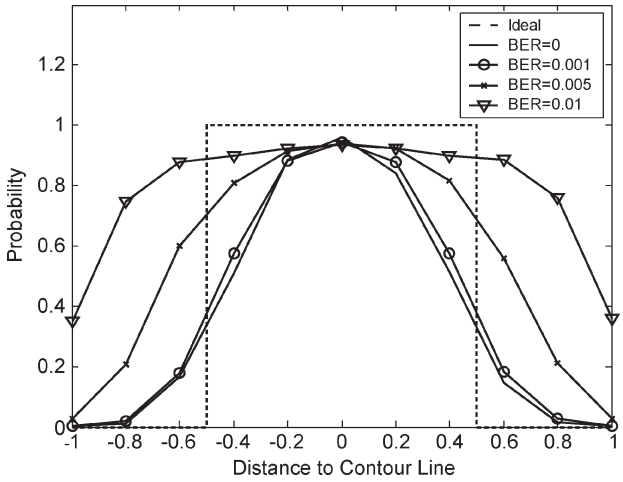

Fig. 9. Performance comparison of contour point inference parameterized by BERs, where the x-axis is the distance from the target sensor node to the contour line of interest, and the y-axis is the probability to be detected as a contour node when the sensor density is ten nodes per unit square.

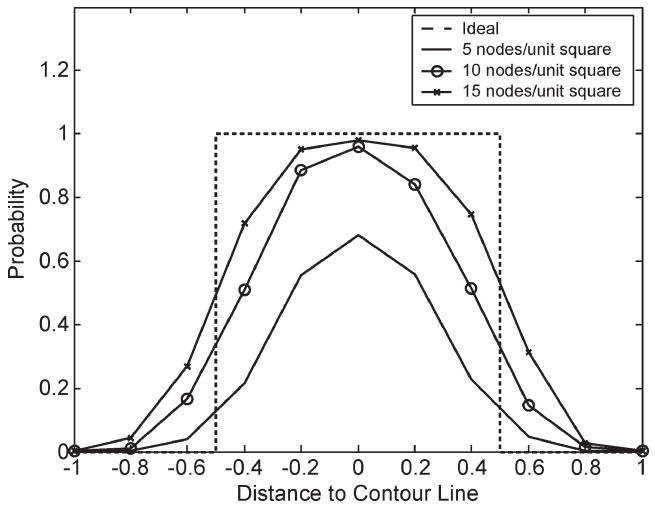

Fig. 10. Performance comparison of contour point inference with different sensor node densities when BER $= 10^{-3}$.

V. Simulation Results

A. Contour Point Inference

The effects of different system parameters on the performance of contour point inference, as studied in Section III-B, are investigated. In our simulation, the sensing noise power σ_n^2 is 30 dB less than the signal power $K = 1$, the tolerance range $r = 0.5$, and the probing range $R = 0.75$. The signal is chosen to be a linear function from $S_{\min} = 0$ to $S_{\max} = 10$ along the x-axis with a constant slope of 1. The number of quantization bits for each measurement is 10, i.e., there are 2^{10} quantization levels. We attempt to detect the contour line with the signal value $\mu = 5$. Note that each BER value has its corresponding P_{AE} according to (21) in the following experiments.

1) Effect of BER: The performance comparison over wireless channels under different BERs is given in Fig. 9, where the sensor density is ten nodes per unit square, the x-axis is the distance from the target sensor node to the contour line of interest, and the y-axis is the probability that a target sensor node is to be detected as a contour point. The negative and positive values mean that the target node is located at the left- and right-hand side of the contour line. The dotted line represents the performance of an ideal case, where all measurements are perfect, and the sensor density is sufficiently large. As shown in the figure, the curves under different BERs all deviate from the ideal case. A higher local BER makes the system to include more sensor nodes, which are relatively less likely to be the contour point, in the candidate list and, thus, induce larger power waste of the network.

The two curves with BER $= 0$ and 10^{-3} almost coincide. That means that the performance becomes saturated when BER $= 10^{-3}$ and smaller BERs will not perform better. This helps us to determine the requirement of the data quality at the MAC layer to be BER $= 10^{-3}$ and $P_{AE} = 10^{-3}$ when the number of quantization bits for the measurement is 10 bits.

2) Effect of Sensor Density: The probabilities that a target sensor node is to be detected as a contour point under different sensor densities are shown in Fig. 10 when BER $= 10^{-3}$ and $P_{AE} = 10^{-3}$. It is obvious that a higher sensor density yields better performance. This is due to the increase in the amount of available sensor data, which can provide more useful information to the target sensor node to mitigate the impact of noise interference on the calculation of contour point probability so as to increase the discriminability of the contour points. The performance gain after the node density of ten nodes per unit square is marginal. Thus, the node density of ten nodes per unit square seems to be a sensible choice.

B. Contour Line Estimation

A sensor field of dimension 4×4 with sensor density of ten nodes per unit square is adopted in the simulation. Unless specified otherwise, we set $r = 0.5$ and $R = 0.75$. A cone-shaped function, which is defined by its slope and base radius over a 2-D plane, is used to model the monitored physical phenomenon. The impact of the sensor density, the local slope of the monitored phenomenon, and the curvature of the contour line, which is the reciprocal of the base radius, will be investigated. We attempt to localize the contour line with signal level $\mu = 5$. Different parameter settings such as BER and quantization levels are compared to evaluate how cross-layer parameters affect the performance in this section. The default number of quantization bits for each measurement is 10, i.e., there are 2^{10} quantization levels. Based on the discussion in Section II, the sensing noise level is 30 dB smaller than the signal level. We are interested in a contour of signal value $\mu = 5$.

1) Effect of Curvature: When monitoring a physical phenomenon, we do not expect that the contour of the monitored event is smooth. Thus, we would like to know how well the proposed contour line estimation algorithm adapts to the change of direction of a curve. To this end, we conduct the experiment of the proposed contour line-estimation algorithm under different curvatures. It is well known that the curvature reflects how smooth a curve is [38]. When the variation of the direction of the unit tangent vector of a point in a curve is small, the curvature is small. However, when that variation is large, the

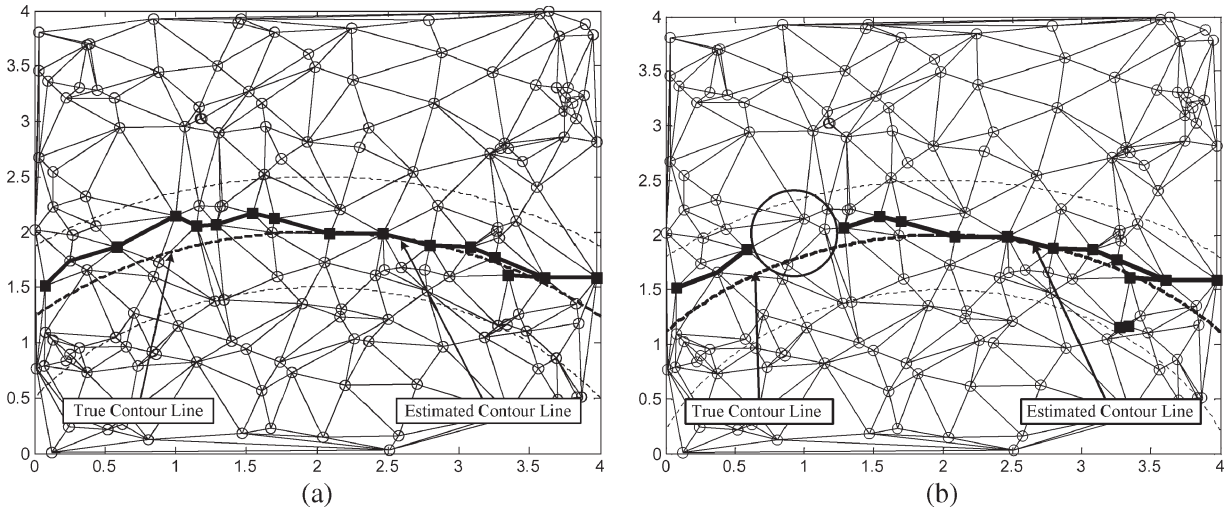

Fig. 11. Simulation results of contour line estimation with (a) local curvature $= 1/3$ and (b) local curvature $= 1/2.7$.

curvature becomes large. Hence, it is a reasonable choice to test the proposed contour line-estimation algorithm under different curvatures and see how the proposed algorithm behaves.

The performance of the proposed contour line estimation algorithm is evaluated under different curvature values in Fig. 11, where BER $= 0$, and the local slope of the contour line is 1. In this figure, circles represent the deployed sensor nodes, squares represent the estimated contour points, and bold solid lines represent the estimated contour lines. Moreover, the true contour line is represented by a bold dashed line, and its tolerance range is bounded by two dotted lines. As observed, when the curvature becomes higher, the performance degrades if the sensor density is kept the same. This can be explained by the fact that the sensor density is not high enough to provide good contour line estimation results when the curvature is too large. The large circle in Fig. 11(b) illustrates the limitation of the proposed contour line connection scheme. Only a gap of one hop can be compensated in the algorithm. The reason for this large circle in Fig. 11(b) is as follows. As stated in the previous paragraph, a large curvature of a point in a curve means that there is a sudden change in direction at that point. Thus, when the deployed sensor nodes around that point are scarce, this rapid change in direction causes the contour line of interest at that point not in the probing range and the tolerance range of sensor nodes around that point and leads to no contour point candidates to be declared.

To give a more specific rule for system design, we can consider the following parameter:

$$R_g = \frac{\sqrt{D_S}}{\kappa} \tag{22}$$

where D_S is a local sensor density, and κ is a local curvature of the contour of interest. This parameter is newly proposed in this paper, and the physical meaning of this parameter is the average number of sensor nodes per unit curvature. If this value is small, then it is hard for the deployed sensor network to estimate the contour lines. When $D_S = 10$ and $\kappa = 1/2.7 = 0.37$, parameter $R_g \approx 8.54$. In this case, the estimated contour points cannot fully be connected by lines, as shown in Fig. 11(b). When $D_S = 10$ and $\kappa = 1/3 = 0.33$, parameter $R_g \approx 9.49$. In this case, all the estimated contour points are connected by lines, as shown in Fig. 11(a). Thus, a rule of thumb is that parameter R_g should be 9 or higher to provide good contour line estimation in the example shown in Fig. 11.

2) Effect of Local Slope: Along the gradient direction of a contour line, we may have different local slopes at different locations. Intuitively, if the local slope is larger, it will be easier to detect the contour line. This is supported by our simulation. The performance of contour line detection with two different local slope values is shown in Fig. 12, where the local curvature is $\kappa = 1/3$, and BER $= 0$. We see that good performance can be achieved when the local slope is 1. However, when the local slope is 0.5, the proposed contour-connection scheme fails. When the slope becomes smaller, it makes the average likelihood estimation less accurate due to the noise effect. Let us recall the spatial average log-likelihood estimate, which is

$$L_{\vec{p}_0}(k) = C_1 - \frac{C_2}{N+1}\sum_{\ell=0}^{N}\left(X_{\chi(\vec{p}_\ell)}(k) - \mu\right)^2$$

where $X_{\chi(\vec{p}_l)}(k)$ is the data sample measured by the ℓth sensor node at time k, $\ell = 0, 1, \ldots, N$, which consists of the signal of desired and the noise. When the local slope is small, the signal of desire perceived by the neighbor sensor nodes of a target is of similar value, and their data samples are then dominated by their respective noises. Thus, when the sensor density is low, this will cause the proposed algorithm to screen out the true contour point candidate and make errors during the step of the construction of the contour line at the remote base station. To improve the performance in the area of a smaller local slope, we have to increase the sensor density to mitigate the effect of noise and increase the accuracy of the proposed algorithm.

3) Effect of BER: As shown in Fig. 9, the proposed algorithm works well where BER $= 0.001$. We show the contour connection results in Fig. 13 at the same BER, where the local curvature is 1/3, and the local slope is 1. It is clear that the proposed algorithm provides a good result under such a

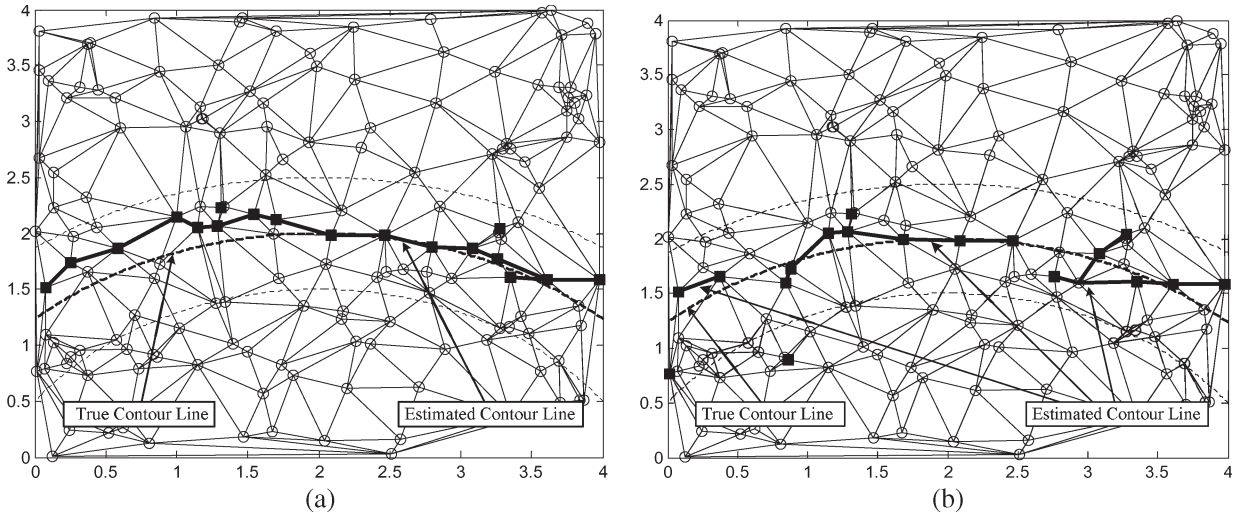

Fig. 12. Simulation results of contour line estimation with (a) local slope $= 1$ and (b) local slope $= 0.5$.

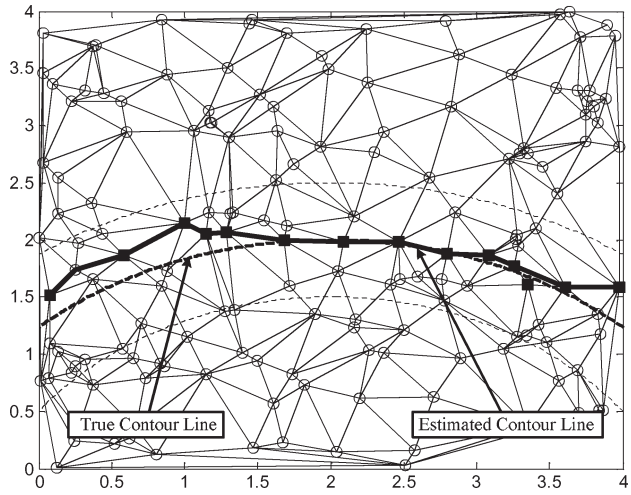

Fig. 13. Simulation results of contour line estimation with BER $= 0.001$ and $P_{AE} = 0.001$.

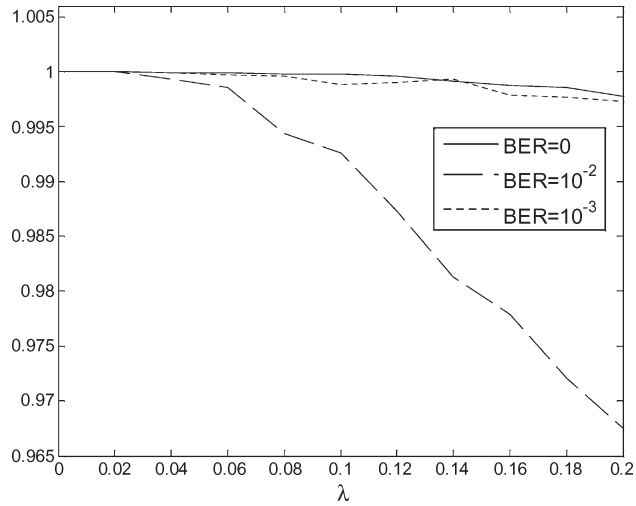

Fig. 14. Probability of the contour point candidate is within the tolerance range of the true contour line.

setting. The tradeoff between different system parameters will be discussed in Section V-C.

4) Effect of Imperfect Location Information: We let $\vec{p}_{i,\text{est}}$ be the estimated location of sensor node i, and let $\vec{p}_i$ be the true location of this sensor node, i.e.,

$$\vec{p}_{i,\text{est}} = \begin{bmatrix} 1+\lambda u_x & 0 \\ 0 & 1+\lambda u_y \end{bmatrix} \vec{p}_i$$

where u_x and u_y are two independent uniform random variables over $[-0.5, 0.5]$, and λ is between 0 and 1. A larger λ means that the location information is less reliable. In this paper, we let the local slop be 1 and the local curvature be 1/3. The imperfect localization causes the proposed algorithm to select a contour point candidate that is outside of the tolerance range of the true contour line, which is the area within the two dotted line in Fig. 8. This wrong selection leads to the wrong contour line estimation. Hence, we would like to look at the probability of the contour point candidate as within the tolerance range of the true contour line. This probability reflects how well the proposed algorithm chooses the contour point candidate when the location information is not perfect. That is, the higher this probability is, the more likely the contour point candidates are contained in the tolerance range of the true contour line. This, in turn, implies that the constructed contour line can be within this area with high confidence. Fig. 14 shows this probability under different values of BER. From Fig. 14, we find that as the BER increases, the impact due to imperfect localization becomes noticeable. When BER $= 0$, the proposed algorithm functions well up to $\lambda = 0.1$. However, when BER $= 10^{-2}$, the maximal tolerable value of λ is 0.02. In addition, from this figure, the results for BER $= 0$ and BER $= 10^{-3}$ are comparable. Thus, as long as we keep BER smaller than 10^{-3}, the proposed algorithm can properly operate. We would like to make a comment here. In many state-of-art localization approaches [34], the probability that the estimation error is smaller than $\lambda = 0.1$ is as high as 0.85 when the noise variance is small and 0.75 when the noise variance is large. Therefore, when BER $= 0$ or 10^{-3}, most of time, the estimated localization information is correct, and the proposed algorithm can still function well. In addition, when $\lambda = 0.2$ and BER $= 0$ or $10^{-}3$, the probability that the

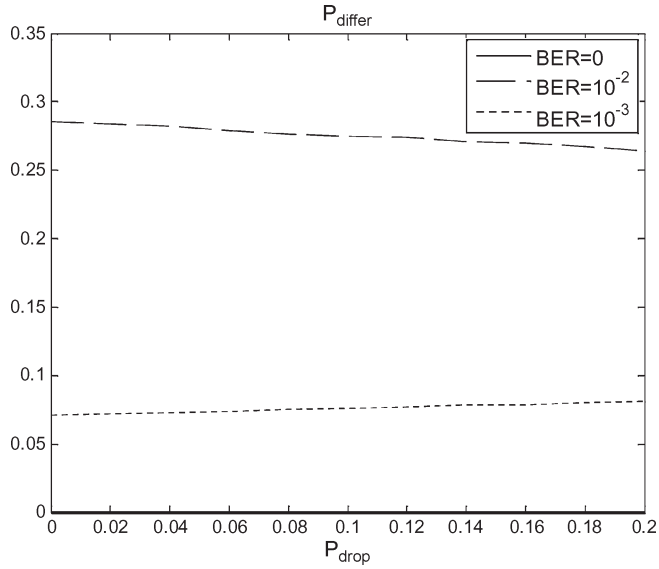

Fig. 15. P_{differ} under different BERs.

contour point candidates are within the tolerance range of the true contour line is around 0.997, as shown in Fig. 14. Since almost all the contour point candidates are inside the tolerant range, the estimated contour line can still predict the tendency of the true contour line. In this case, when adopting [34], the probability that the estimation error is smaller than $\lambda = 0.2$ is close to 0.9, regardless of the noise level. From these, the proposed algorithm can still operate well when the existing localization algorithms are adopted for localization estimation.

5) Effect of Packet Loss/Buffer Overflow: Buffer overflow and packet loss reduce the amount of information that a fusion center can tap. To emulate the effect of packet loss or buffer overflow, we randomly drop the available data packets of a target sensor sent by all the sensor node within its probing range. Let P_{drop} be the probability of a packet being dropped. In this case, the contour point candidates are always within the tolerance range of the true contour line. Thus, we would like to compare the contour point candidates in this case with those found in the perfect case, namely, BER $= 0$ and $P_{\mathrm{drop}} = 0$. The reason we compare the result here against the perfect case is that when the location information is perfect, the optimal contour point candidates are those found in the perfect case. Thus, comparing with the perfect case enables us to realize how good the chosen contour point candidate in this case is. First, we define the following. Let Λ_{drop} be the set of contour point candidates found in this case, and let $\Lambda_{\mathrm{perfect}}$ be the set of contour-point candidates found in the perfect case. Based on this, we define the probability of contour-point candidates that Λ_{drop} and $\Lambda_{\mathrm{perfect}}$ differ as

$$P_{\mathrm{differ}} = 1 - \frac{|\Lambda_{\mathrm{drop}} \bigcap \Lambda_{\mathrm{perfect}}|}{|\Lambda_{\mathrm{perfect}}|}$$

where $|\cdot|$ is the cardinality of the argument set. The comparison is shown in Fig. 15. Again, in this experiment, we let the local slop be 1 and the local curvature be 1/3. From Fig. 15, P_{differ} does not change much as P_{drop} increases for all BERs. This shows that the impact of packet loss or buffer overflow on the performance of the proposed algorithm is mild compared with that of imperfect localization. The reason for this is as follows. When packet loss or buffer overflow occurs at a sensor node, the perception of this sensor node about the deployment of its neighbor nodes correspondingly changes. The deployment could change from dense deployment to sparse deployment. Both possible ways of deployment of sensor nodes are handled in the proposed algorithm (see Algorithm 2 for details). Therefore, the proposed algorithm is less sensitive to packet loss or buffer overflow.

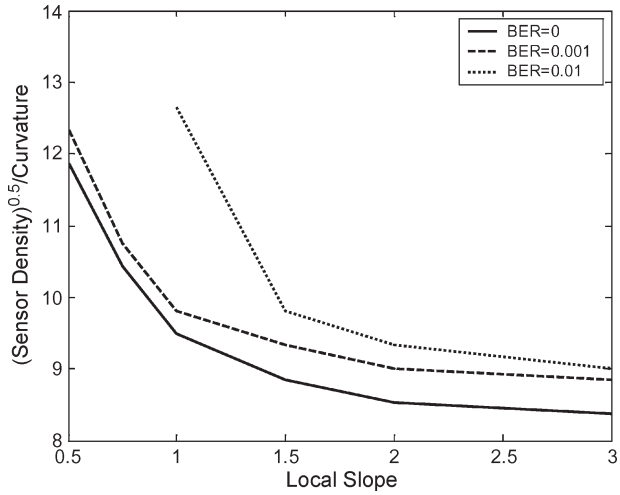

Fig. 16. Parameter R_g versus the local slope under different BERs.

C. Discussion

1) Tradeoff of System Parameters: As described in Section IV, we may adopt the adaptive transmission scheme to increase the power efficiency of communication among sensor nodes, thus extending the network lifetime. To derive a set of simple design rules under various conditions, the relationship among the sensor node density, the local curvature, the local slope, the BER, and the ratio between R and r is investigated here.

First, we show in Fig. 16 the parameter R_g, as defined in (22), as a function of the local slope parameterized by different BER values. The region above the curve is called the effective area, in which the performance of the proposed contour line estimation algorithm is acceptable (at most one missing connection and/or one erroneous branch observed). From Fig. 16, as the BER increases, R_g increases as well for a fixed local slope. Note that the curve with BER $= 0.01$ ends at local slope 1.0 since the BER-induced noise is so large that the proposed algorithm does not work well when the local slope is below 1.0. Similarly, in Fig. 17, we show the parameter R_g as defined in (22) as a function of the local slope parameterized by different ratios of R/r. In Fig. 17, we find that $R/r = 1.5$ outperforms $R/r = 2.0$ and $R/r = 0.5$. Thus, a large R/r does not necessarily improve the performance of the proposed algorithm.

Two conclusions can be drawn from Figs. 16 and 17. First, parameter R_g will eventually converge as the local slope becomes higher, and the converged value is related to other system parameters such as BER and R/r. Second, a larger probing

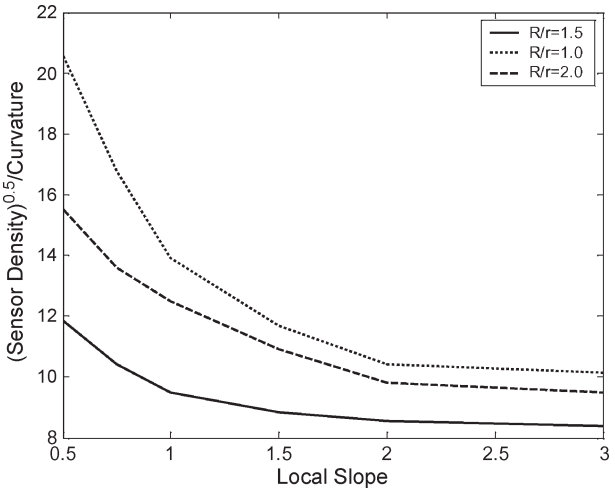

Fig. 17. Parameter R_g versus the local slope under different ratios of R/r and with $r = 0.5$.

range does not guarantee better performance, and there exists an optimal R/r ratio. To explain this, we first recall the spatial average log-likelihood estimate of a target sensor node, which is

$$L_{\vec{p_0}}(k) = C_1 - \frac{C_2}{N+1}\sum_{i=0}^{N}\left(X_{\chi(\vec{p_i})}(k) - \mu\right)^2.$$

This estimate can be used to judge the level of reliability of the information that the sensor nodes within the probing range of a target sensor know about the contour line of interest, as indicated in [5]. The variance of the spatial average log-likelihood estimate can be found to be proportional to

$$\frac{2C_2^2\sigma_n^4}{N+1} + \frac{4C_2^2\sigma_n^2}{(N+1)^2}\sum_{i=0}^{N}\left(E\left\{X_{\chi(\vec{p_i})}(k)\right\} - \mu\right)^2. \quad (23)$$

See [5] for detailed derivations. We see from (23) that a larger fluctuation of the signal strength (the term $\sum_{i=0}^{N}(E\{X_{\chi(\vec{p_i})}(k)\} - \mu)^2$) could induce a larger variance of the spatial average log-likelihood estimate and make the information acquired by the sensor nodes within the probing range of a target sensor less reliable. Consequently, the system performance may degrade as the probing range increases.

2) Quantization Bits: As shown in Fig. 2, there are three sources of data distortion—sensing, quantization, and channel degradation. Although the quantization distortion can be reduced by increasing the number of quantization bits, a larger number of quantization bits will induce larger channel degradation due to the increase of transmission bit rate by (19). Thus, the number of quantization bits has to be chosen to balance the quantization error and the channel error.

As shown in the previous discussion, the quantized measurement of the data sample $X(k)$ at time k can be written as [39]

$$Q_B\left[X(k)\right] = S(k) + n(k) + n_q(k) \quad (24)$$

where $S(k)$ is the desired data, $n(k)$ is the sensing error, and $n_q(k)$ is the quantization error. Assuming that a uniform quantizer with quantization step Δ is adopted, $n_q(k)$ is uniformly distributed over $[0, \Delta]$, and the variance (or power) of $n_q(k)$ is

$$\frac{\Delta^2}{12}.$$

Both $n(k)$ and $n_q(k)$ in (24) contribute to the data distortion. $n(k)$ in (24) is the first distortion source encountered in the whole data-fusion process, whose impact on data distortion cannot be removed or mitigated, even with finer quantization levels and perfect data transmission. To let the sensing error be the dominant contribution to the data distortion, it requires that the power of sensing error should be much larger than that of the quantization error. This makes the sensing error more uncertain than the quantization error and much harder to predict its effect. Meanwhile, this makes the effect coming from the quantization error negligible as well. Based on the above discussion, we can derive the following equation for quantization bit assignment for a uniform quantizer:

$$\frac{\Delta^2}{12} \ll \sigma_n^2 \quad (25)$$

where σ_n^2 is the sensing noise power.

For example, a uniform quantizer is applied to the measured data with values in the range of [0, 10] in the simulation. When the sensing noise power is 0.025, the number of quantization bits should be at least 5.

Compared with 10 bits, a 5-bit quantizer provides an additional 3-dB performance gain for the data transmission, and thus, the communication power can further be saved. In addition, nonlinear quantization schemes could be used to improve the quantization efficiency if the statistical properties of the signal strength are known in advance.

D. Implications on System Design

1) Adaptation of R/r: As we can see in Fig. 17, a larger R/r does not guarantee better performance. However, when both R_g and local slope are not large, the difference in performance is not significant. This observation suggests the adaptation of R/r. When R is large, a sensor node would have to process a large amount of data sent by sensor nodes within its probing range. This would quickly deplete the power of this sensor node. The quick depletion of the power of a sensor node implies the quick change in the topology of a WSN. To let the topology of a WSN gracefully change, R/r should be adapted to the residual power of a sensor node and make the depletion rate of power smoother. Furthermore, the adaptation of R/r also helps extend the lifetime of a WSN.

2) Adaptation of Received Data Quality: From Figs. 9 and 16, the performances of BER $= 0$ and BER $= 10^{-3}$ are similar. This suggests that allowing BER $= 10^{-3}$ or smaller is sensible. As a mentioned previously, when BER $= 10^{-3}$, letting P_{AE} in (21) to be larger than 10^{-3} leads the average number of transmission attempts to be less than 1 in theory. For other values of BER smaller than 10^{-3}, we can find a corresponding P_{AE} so that the average number of transmission attempts can be less than 1. Thus, it is possible to adjust BER and P_{AE} to make the

average number of transmission attempts as close to 1 as possible. This observation motivates the following design strategy. Let the goal of the design be to maintain the average number of transmission attempts as close to 1 as possible, and we assume that the total power per transmission is kept constant. We keep tracking the average number of transmission up to now, which is denoted by NT. When NT increases behind a threshold, the application layer lowers the data quality by increasing the tolerable data distortion due to channel, which is P_{AE} in (21). By adjusting the data quality in this way, the average number of transmission attempts is expected to decrease in theory. This decrease could result in less communication power wasted in retransmission. However, if the maximal tolerable data distortion due to channel is reached and NT is still high, then the tolerable data distortion due to quantization is increased to reduce the transmission bit rate and to obtain an additional reception gain, as indicated in (19). In this case, we have a higher E_b/N_0 and, hence, a lower BER and PRR. Therefore, we can expect a smaller average number of transmission attempts in theory and save communication power. On the contrary, if NT decreases below the threshold, then the tolerable channel distortion can be lowered to improve the data quality.

VI. Conclusion

A statistical approach to contour line estimation with WSNs for environmental monitoring has been investigated in this paper. The proposed solution simultaneously considered sensing noise, channel noise, and quantization noise. These effects are embraced in the calculation of contour point probability of each sensor node. Based on the calculation of contour point probabilities and the average log-likelihood values of sensors, the true contour point candidates are found, and the estimated contour line can accordingly be constructed at the remote base station. Furthermore, we also take the impact of the spare deployment of sensor nodes into account when designing the proposed algorithm. Thus, the proposed algorithm is able to function when the sensor node density is low. In the simulation result, we found that the proposed algorithm properly operates when the sensor density is as low as ten sensors per unit square. In addition, we found that the proposed contour line connection algorithm is robust in the sense that it can overcome several shortcomings (e.g., a one-hop gap between contour points) caused by poor contour point estimation results. Furthermore, since the proposed algorithm takes the dense and sparse deployments into account, it is less sensitive to packet loss or buffer overflow. Several key results and related design issues are outlined below.

1) We found that the ratio of the square root of the sensor node density over the curvature (R_g) serves as a good design parameter, which is first proposed in this paper. From the simulation, a rule of thumb is that R_g should be 9 or higher to provide good contour line estimation.
2) A larger probing range does not guarantee better performance, and there exists an optimal R/r ratio. This fact can be used to adjust R/r according to the residual power of a sensor so that the lifetime of the network can be extended.
3) To have the quantization error negligible, the variance of the quantization error should be far less than that of the sensing error. This relationship can help determine the quantization error once the sensing error is known.
4) To avoid the impact of imperfect localization, the BER should be maintained as close to 10^{-3} as possible.

The proposed algorithm does not fully address the impact of other issues, such as network congestion, packet collision, and buffer overflow. Those factors can be investigated to have an even more robust algorithm. However, dealing with these issues is out of the scope of this paper, and therefore, it will become the future work of the authors.

Acknowledgment

The authors would like to thank the Associate Editor and the anonymous reviewers for their comments.

References

[1] R. Nowak and U. Mitra, "Boundary estimation in sensor networks: Theory and methods," in *Proc. 2nd Int. Workshop Inf. Process. Sensor Netw.*, Apr. 2003, pp. 22–36.
[2] K. K. Chintalapudi and R. Govindan, "Localized edge detection in sensor fields," in *Proc. IEEE Int. Workshop Sens. Netw. Protocols Appl.*, May 2003, pp. 59–70.
[3] P.-K. Liao, M.-K. Chang, and C.-C. J. Kuo, "Statistical edge detection with distributed sensors under the Neynan–Pearson (NP) optimality," in *Proc. IEEE VTC—Spring*, May 2006, vol. 3, pp. 1038–1042.
[4] P.-K. Liao, M.-K. Chang, and C.-C. J. Kuo, "Contour line extraction with wireless sensor networks," in *Proc. IEEE ICC*, May 2005, vol. 5, pp. 3202–3206.
[5] P.-K. Liao, M.-K. Chang, and C.-C. J. Kuo, "Contour line extraction in a multi-modal field with sensor networks," in *Proc. IEEE GLOBECOM*, Nov. 2005, vol. 3, pp. 1309–1313.
[6] D. Satyanarayana and S. V. Rao, "Fault tolerant local Delaunay triangulation for ad hoc sensor networks," in *Proc. IEEE WCSN*, Dec. 2007, pp. 36–40.
[7] C. H. Wu, K. C. Lee, and Y. C. Chung, "A Delaunay triangulation based method for wireless sensor network deployment," in *Proc. IEEE ICPADS*, Jul. 2006, pp. 253–260.
[8] X. Y. Li, P. J. Wan, and O. Frieder, "Coverage in wireless ad hoc sensor networks," *IEEE Trans. Comput.*, vol. 52, pp. 753–763, Jun. 2003.
[9] J. D. Tan, O. Lozano, N. Xi, and W. H. Sheng, "Multiple vehicle systems for sensor network area coverage," in *Proc. IEEE WCICA*, Jun. 2004, vol. 5, pp. 4666–4670.
[10] J. Wang and S. Medidi, "Energy efficient coverage with variable sensing radii in wireless sensor networks," in *Proc. IEEE Int. Conf. Parallel Distrib. Syst.*, Oct. 2007, p. 61.
[11] H. C. Jang and H. C. Lee, "A Voronoi detection range adjustment (VERA) approach for energy saving of wireless sensor networks," in *Proc. IEEE Int. Conf. Parallel Distrib. Syst.*, Dec. 2007, pp. 1–7.
[12] P. Green and R. Sibson, "Computing Dirichlet tessellations in the plane," *Comput. J.*, vol. 21, no. 2, pp. 168–173, 1978.
[13] J.-D. Boissonnat and M. Teillaud, "On the randomized construction of the Delaunay tree," *Theor. Comput. Sci.*, vol. 112, no. 2, pp. 339–354, May 1993.
[14] B. Karp and H. T. Kung, "GPSR: Greedy perimeter stateless routing for wireless networks," in *Proc. ACM MobiCom*, Aug. 2000, pp. 243–254.
[15] S. Lee, B. Bhattacharjee, and S. Banerjee, "Efficient geographic routing in multihop wireless networks," in *Proc. ACM MobiHoc*, May 2005, pp. 230–241.
[16] C.-P. Li, W.-J. Hsu, B. Krishnamachri, and A. Helmy, "A local metric for geographic routing with power control in wireless networks," in *Proc. IEEE SECON*, 2005, pp. 229–239.
[17] X. Yu, S. Mehrotra, and N. Venkatasubramanian, "Sensor scheduling for aggregate monitoring in wireless sensor networks," in *Proc. IEEE SSDBM*, 2007, p. 24.
[18] "MICA2 data sheets," *Crossbow*. [Online]. Available: http://www.xbow.com

[19] M. Zuniga and B. Krishnamachari, "Analyzing the transitional region in low power wireless links," in *Proc. IEEE SECON*, Oct. 2004, pp. 517–526.
[20] Chipcon, *Cc1000 Low Power Radio Transceiver*. [Online]. Available: http://www.chipcon.com
[21] T. S. Rappapport, *Wireless Communications: Principles and Practice*. Englewood Cliffs, NJ: Prentice–Hall, 2001.
[22] J. Zhao and R. Govindan, "Understanding packet delivery performance in dense wireless sensor networks," in *Proc. ACM SENSYS*, Nov. 2003, pp. 1–13.
[23] M. K. Simon and M.-S. Alouini, *Digital Communication Over Fading Channels*. New York: Wiley, 2000.
[24] S. Lin and D. J. Costello, *Error Control Coding: Fundamentals and Applications*. Englewood Cliffs, NJ: Prentice–Hall, 1982.
[25] *Part 15.4: Wireless MAC and PHY Specifications for Low Rate Wireless Personal Area Networks (WPANs)*, IEEE Std. 802.15.4-2006, 2006.
[26] "MICAz 2.4 GHz data sheets," *Crossbow*. [Online]. Available: http://www.xbow.com
[27] Sun Microsystems, *Sun Small Programmable Object Technology (Sun SPOT) Theory of Operation*. [Online]. Available: http://www.sunspotworld.com
[28] J.-H. Chang and L. Tassiulas, "Maximum lifetime routing in wireless sensor networks," *IEEE/ACM Trans. Netw.*, vol. 12, no. 4, pp. 609–619, Aug. 2004.
[29] J. Li, J. Jannotti, D. S. J. Couto, D. R. Karger, and R. Morris, "A scalable location service for geographic ad hoc routing," in *Proc. ACM MobiCom*, 2000, pp. 120–130.
[30] R. Jain, A. Puri, and R. Sengupta, "Geographical routing using partial information for wireless ad hoc networks," *IEEE Pers. Commun.*, vol. 8, no. 1, pp. 48–57, Feb. 2001.
[31] N. B. Priyantha, H. Balakrishnan, E. Demaine, and S. Teller, "Anchor-free distributed localization in sensor networks," Laboratory for Computer Science (LCS), Mass. Inst. Technol., Cambridge, MA, Tech. Rep. TR-892, Apr. 2003.
[32] N. Patwari, A. I. Hero, M. Perkins, N. Correal, and R. O'Dea, "Relative location estimation in wireless sensor networks," *IEEE Trans. Signal Process.*, vol. 51, no. 8, pp. 2137–2148, Aug. 2003.
[33] N. Bulusu, J. Heidemann, D. Estrin, and T. Tran, "Self-configuring localization systems: Design and experimental evaluation," *ACM Trans. Embed. Comput. Syst.*, vol. 3, no. 1, pp. 24–60, Feb. 2004.
[34] A. Ihler, J. I. Fisher, R. Moses, and A. Willsky, "Nonparametric belief propagation for self-localization of sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 4, pp. 809–819, Apr. 2005.
[35] O.-H. Kwon and H.-J. Son, "Localization through map stitching in wireless sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 19, no. 1, pp. 93–105, Jan. 2008.
[36] K.-F. Ssu, C.-H. Ou, and H. C. Jiau, "Localization with mobile anchor points in wireless sensor networks," *IEEE Trans. Veh. Technol.*, vol. 54, no. 3, pp. 1187–1197, May 2005.
[37] K.-F. S. Wong, I. W. Tsang, V. Cheung, S.-H. G. Chan, and J. T. Kwok, "Position estimation for wireless sensor networks," in *Proc. IEEE GLOBECOM*, Nov. 2005, pp. 2772–2776.
[38] Wikipedia, *Curvature*. [Online]. Available: http://en.wikipedia.org/wiki/Curvature
[39] D. Marco and D. L. Neuhoff, "The validity of the additive noise model for uniform scalar quantizers," *IEEE Trans. Inf. Theory*, vol. 51, no. 5, pp. 1739–1755, May 2005.

Pei-Kai Liao (S'03–M'07) received the B.S. and M.S. degrees in electrical engineering from National Tsing Hua University, Hsinchu, Taiwan, in 1997 and 1999, respectively, and the Ph.D. degree in electrical engineering from the University of Southern California, Los Angeles, in 2007.

Since July 2007, he has been a Senior Engineer with MediaTek Inc., Hsinchu, which is a leading fabless semiconductor company for wireless communications and digital multimedia solutions. He is in charge of research and system development for next-generation wireless mobile communication systems. His current research interests are in areas of statistical signal processing, wireless sensor networks, and wireless mobile communications.

Min-Kuan Chang (S'98–M'03) received the B.S. degree in electrical engineering from the National Tsing Hua University, Hsinchu, Taiwan, in 1996 and the M.S. and Ph.D. degrees in electrical engineering from the University of Southern California, Los Angeles, in 1998 and 2003, respectively.

From 2000 to 2003, he was a Research Assistant with the Integrated Media Systems Center, University of Southern California. Since February 2004, he has been with the Department of Electrical Engineering, National Chung Hsing University, Taichung, Taiwan, where he is currently an Assistant Professor. His current research interest includes power control in wireless telecommunication systems, wireless multimedia transmission, cooperative communication systems, and distributed signal processing in a wireless sensor network.

C.-C. Jay Kuo (F'99) received the B.S. degree in electrical engineering from the National Taiwan University, Taipei, Taiwan, in 1980 and the M.S. and Ph.D. degrees in electrical engineering from the Massachusetts Institute of Technology, Cambridge, in 1985 and 1987, respectively.

From October 1987 to December 1988, he was a Computational and Applied Mathematics Research Assistant Professor with the Department of Mathematics, University of California, Los Angeles. Since January 1989, he has been with the University of Southern California, Los Angeles, where he is currently a Professor of electrical engineering, computer science, and mathematics and the Director of the Signal and Image Processing Institute. His research interests are in the areas of digital signal and image processing, multimedia compression, communication, and networking technologies. He has guided about 90 students with their Ph.D. degrees and supervised 20 postdoctoral research fellows. He is the coauthor of about 150 journal papers, 770 conference papers, and nine books.

Dr. Kuo is a Fellow of The International Society for Optical Engineers and a member of the Association for Computing Machinery. He is the Editor-in-Chief of the *Journal of Visual Communication and Image Representation* and the Editor of the *Journal of Information Science and Engineering* and the *EURASIP Journal of Applied Signal Processing*. He was on the Editorial Board of the IEEE SIGNAL PROCESSING MAGAZINE from 2003 to 2004. He served as Associate Editor for the IEEE TRANSACTIONS ON IMAGE PROCESSING from 1995 to 1998, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY from 1995 to 1997, and the IEEE TRANSACTIONS ON SPEECH AND AUDIO PROCESSING from 2001 to 2003. He received the National Science Foundation Young Investigator Award and the Presidential Faculty Fellow Award in 1992 and 1993, respectively.