This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE INFOCOM 2010 proceedings
This paper was presented as part of the main Technical Program at IEEE INFOCOM 2010.

Distance Estimation by Constructing The Virtual Ruler in Anisotropic Sensor Networks

Yun Wang Kai Li
School of Computer Science & Engineering, Southeast University
Key Lab of Computer Network and Information Integration, MOE
Nanjing, China, 210096
Email: {yunwang,newlikai}@seu.edu.cn

Jie Wu
Department of Computer and Information Sciences
Temple University
Philadelphia, PA 119122
Email: jiewu@temple.edu

***Abstract*—Distance estimation is fundamental for many functionalities of wireless sensor networks and has been studied intensively in recent years. A critical challenge in distance estimation is handling anisotropic problems in sensor networks. Compared with isotropic networks, anisotropic networks are more intractable in that their properties vary according to the directions of measurement. Anisotropic properties result from various factors, such as geographic shapes, irregular radio patterns, node densities, and impacts from obstacles. In this paper, we study the problem of measuring irregularity of sensor networks and evaluating its impact on distance estimation. In particular, we establish a new metric to measure irregularity along a path in sensor networks, and identify turning nodes where a considered path is inflected. Furthermore, we develop an approach to construct a virtual ruler for distance estimation between any pair of sensor nodes. The construction of a virtual ruler is carried out according to distance measurements among beacon nodes. However, it does not require beacon nodes to be deployed uniformly throughout sensor networks. Compared with existing methods, our approach neither assumes global knowledge of boundary recognition nor relies on uniform distribution of beacon nodes. Therefore, this approach is robust and applicable in practical environments. Simulation results show that our approach outperforms some previous methods, such as DV-Distance and PDM.**

***Index Terms*—Distance Estimation, Wireless Sensor Networks, Virtual Ruler**

I. Introduction

Wireless sensor networks (WSNs) have become increasingly available for various fields, such as industrial sensing and scheduling, critical construction protection, environmental monitoring, and scouting in battlefields. In WSN applications, abundances of sensor nodes are deployed randomly in dangerous or polluted regions to watch for specified phenomena. Sensor nodes are equipped with limited communication capabilities which they can use to organize and reorganize themselves as ad hoc networks in response to geographical conditions or triggers emanating from the environment. Unlike in traditional networks, a datum in WSNs is identified by its location. Thus, locations of sensor nodes are important for operations in WSNs. Localization in WSNs has been intensively studied in recent years, with most of these studies relying on the condition that only a small proportion of sensor nodes, called beacon nodes, know their exact positions through GPS devices or manual configuration. Other sensor nodes estimate their distances to beacon nodes and calculate positions with multilateration techniques. These methods enable us to provide high accuracy with a small proportion of beacon nodes in WSNs. However, their performance heavily depends on the precision of distance estimation. Besides, accurate distance estimation is essential for many other functionalities of WSNs. For example, data storage and retrieval.

Because the communication range of a sensor node is very limited, distance estimation has to be carried out on a hop-by-hop basis. For any pair of sensor nodes, the shortest path between them can be set up. The distance between pairwise sensor nodes is then obtained by accumulating the measured distance of each hop along the shortest path [1]. In a WSN, isotropic properties refer to the fact that measurement properties are identical in all directions, and measurement errors can be forecasted by some statistical models and parameters, e.g., Gauss distribution. Compared to isotropic networks, anisotropic networks are more intractable in that their properties vary according to the directions of measurement. Anisotropic properties result from various factors, such as geographic shapes, irregular radio patterns, node densities, and impacts from obstacles.

There are two challenging problems in distance estimation in such a manner: (1) environmental noises fluctuate tremendously, even in a local area, which leads to uncontrollable measurement errors [2]; and (2) the shortest path is inflected when it bypasses holes where there are no sensor nodes deployed. The inflections make the measured distance remarkably longer than its corresponding Euclidean distance [3]. Moreover, the characteristics of the above mentioned factors are unknown when a WSN is deployed. Ignoring these critical problems may bring about huge errors. For example, the most proposed localization methods make the assumption that a deployed WSN is isotropic. Unfortunately, this assumption does not hold true in practice. Whitehouse et al [4-5] have empirically proven that the most proposed localization methods perform weakly in anisotropic WSNs.

In this paper, we study the problem of making distance estimations between any pair of sensor nodes in WSNs. Roughly speaking, we enable a sensor node to judge the inflection of a path passing through it. The judgement is carried out on each sensor node by investigating the correlation

978-1-4244-5837-0/10/$26.00 ©2010 IEEE

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE INFOCOM 2010 proceedings
This paper was presented as part of the main Technical Program at IEEE INFOCOM 2010.

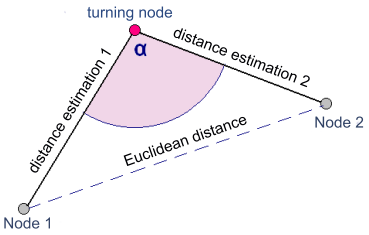


Fig. 1. Distance estimation along a path with turning nodes.

with its neighboring nodes along the path. Some special nodes, which are called turning nodes, are differentiated from other sensor nodes according to their lower correlation values. As has been observed in [6], the inflected path in the WSN is segmented into sub-paths, which are almost straight with slight zigzags, and it is the angle between each pair of sub-paths that determines the inflection of the path and the errors in distance estimation along the path as well. This gives us some clues about making nonlinear optimizations on distance estimation according to the length of each sub-path together with the angle of the path at the corresponding turning node.

Fig. 1 illustrates an example of distance estimation in anisotropic networks. The path from $node1$ to $node2$ is inflected and segmented into two sub-paths. Distance estimations along sub-paths are accurate. We then identify the $turningnode$ where the path is segmented and calculate the angle between two sub-paths. Subsequently, the cosines law is applied according to distance estimation along sub-paths and the angle to obtain the true distance between $node1$ and $node2$. In order to obtain the angle corresponding to each turning node along a path, we further propose an approach that leverages the accurate position information from beacon nodes. The shortest paths between beacon nodes are regarded as a virtual ruler for a WSN. The virtual ruler is then used to evaluate the inflections of paths in the WSN. The procedure of constructing a virtual ruler and obtaining distance estimations are performed according to local features around sensor nodes.

The contributions of this paper are summarized as follows:

1) A new metric is proposed to describe local characteristics along paths. With this metric, sensor nodes can make judgements on whether a path passing through is inflected or not.
2) A new approach is proposed to construct a virtual ruler, which is applied to conduct distance estimations in a WSN. The approach performs local optimization along paths. Thus, it can be implemented in a distributed manner.
3) The proposed approach in this paper does not rely on isotropic environment assumptions, e.g., uniform or dense deployment of beacon nodes, or a priori recognition of boundaries of holes or other global knowledge. Also, we do not assume the superior capability of beacon nodes.

The rest of the paper is organized as follows: In Section 2, we discuss related work and analyze the pros and cons of the proposed methods. In Section 3, we describe the model, and then present methods to construct a virtual ruler for a WSN. Section 4 presents an approach for distance estimation. Applicability and overhead are investigated in Section 5. We evaluate the performance of our method, and compare our method with PDM and classical DV-Distance in Section 6. The conclusion is given in Section 7.

II. Related Work

To alleviate the large error introduced by DV-Distance [1] in anisotropic networks, many approaches have been proposed. According to the fashion of adjustments on the firsthand distance measurement, existing approaches fall into two categories, i.e., methods with adjustments by global and local features. In this section, we give a brief overview of their principles and pros and cons.

A. Methods with Adjustments by Global Features

Global optimization methods usually make use of a priori knowledge or accurate information, which is obtained by powerful devices, e.g., location information from a GPS. Lim and Hou proposed a distributed localization method PDM [7], in which the relationship between the Euclidean distance and the measured distance was investigated. The same distance estimation strategy obtained by distance estimations among beacon nodes was applied to all the sensors in a WSN. PDM worked well if the deployment of beacon nodes was uniform; otherwise, its performance was quite poor. Savvides et al. proposed a collaborative multi-lateration method [10] involving three stages: sub-tree formation, initial estimation, and refinement. Nodes whose position could be determined according to rigid graphs were first collected into a sub-tree. Then, based on simple geometric relationships among measured distances and known beacon node locations, the nodes obtained a set of initial location estimates. Finally, iterative least-squares were used to obtain final location estimates. This method relied on dense and uniform deployment of beacon nodes in WSNs. Otherwise, the method could not accomplish the first stage.

Pan et al. proposed a multidimensional vector regression method [9] to build a mapping function by addressing the problems of both uncertainty and nonlinearity, directly. The method performed a kernel-based transformation from the signal to the physical location space to capture their nonlinear relationship. Furthermore, kernel canonical correlation analysis was executed for feature extraction. This allowed the pairwise similarity of samples in both spaces to be maximally correlated. This method also required uniform deployment of beacon nodes. Otherwise, the canonical vectors obtained in the training phase could not reflect the feature of a region lacking beacon nodes. Li and Kunz proposed a method applying an efficient neural network nonlinear projection method called curvilinear component analysis [8]. Compared with other multi-dimensional scaling methods, the method was more efficient and accurate in the networks with lower connectivity.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE INFOCOM 2010 proceedings
This paper was presented as part of the main Technical Program at IEEE INFOCOM 2010.

However, the unit-disk model is required by the method to guarantee the correctness of the deduced topology of the networks.

B. Methods with Adjustments by Local Features

Li et al. proposed *rendered paths* (REP) [11]. It rendered the shortest path between any two sensors with different colors, where each color represented a hole. Then, it estimated the angle between two segments of a path that intersected with the hole by creating a virtual hole. Thereafter, the distance between the two ends of the path was estimated according to the law of cosines with two known edges and one corresponding internal angle. REP could discover enough characteristics of areas near the hole with the help of virtual holes to make local optimizations. Wang and Xiao studied the problem of sensor localization in concave environments [6]. They proposed an improved multi-hop algorithm that can recognize and filter out the erroneous distance estimation. The method could choose appropriate beacon nodes to make accurate distance estimations under the condition that there were enough beacon nodes deployed uniformly in the networks.

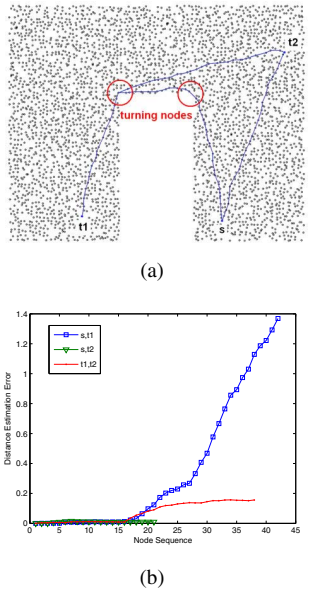
Wang et al. also proposed a method [12] by selecting a subset of sensor nodes as landmarks and partitioning a WSN into Voronoi cells, where all the sensor nodes closest to the same landmark were grouped into the same cell. The combinatorial Delaunary complexes, which were globally rigid, were then generated. Subsequently, each sensor node determined its position according to distance estimations from landmarks, which could reflect the topological characteristics of areas near the hole. However, both methods relied on a rather strict assumption that the remaining part of the WSN, excluding the holes, was isotropic and that the hole boundaries were recognized beforehand. Moreover, it was difficult to create regular virtual holes in a practical environment. In a recent work, the same author improved the method of [13] by removing dependance on boundary detection. In this method, the landmarks were selected incrementally without knowledge of network boundary. However, distance estimation in single tile was not adjusted, which led to error propagation in a large scale network.

III. Constructing The Virtual Ruler

A. The Model and Assumptions

Suppose that a large number of sensor nodes are deployed randomly in a region. Let $G(N, E)$ denote an undirected graph. N is a set of vertices representing the nodes. Some of them, which have the knowledge of accurate positions, are called beacon nodes. The others are called sensor nodes. There are some holes inside the region. E is a set of edges representing the links among the nodes.

$P(s,t)$ denotes the shortest path between any two vertices $s, t \in N$, represented by a sequence of vertices along the path. There are three types of distance values related to $P(s,t)$: measured distances, Euclidean distances, and estimated distances. They are denoted by $M(s,t)$, $D(s,t)$, and $ED(s,t)$, respectively. $M(s,t)$ is obtained by accumulating the measured distances of all the edges along $P(s,t)$. $D(s,t)$ is obtained by calculating the coordinates of both s and t. $ED(s,t)$ is an estimated distance between s and t obtained with the distance estimation method. Obviously, $M(s,t) \geq D(s,t)$ holds in most cases due to inflections of $P(s,t)$. The aim of our method is to establish relationships between measured distances and Euclidean distances of the paths among beacon nodes. Then, the relationships are applied to distance estimation of any path $P(s,t)$ in the WSN in order to make the estimated distance $ED(s,t)$ as close to its corresponding $D(s,t)$ as possible.



(a)

(b)

Fig. 2. An anisotropic WSN and its impact. (a) An anisotropic sensor network. (b) The distance estimation errors along different paths.

We assume that there are a large number of sensor nodes in a WSN so that any node is able to obtain enough neighboring nodes with a limited flooding operation. We do not require isotropic deployment, which is required by most existing methods. We also assume that each pair of neighboring nodes can communicate with each other, whereas the unit-disk model and accurate distance measurement are not introduced in our method. In order to simplify the discussion, we are not concerned with the issues of energy consumption and robustness of sensor nodes. We believe that these missing issues do not invalidate the correctness of the proposed method.

B. Overview

In an anisotropic WSN, huge errors may be introduced into distance estimation because of irregular deployment. As shown in Fig. 2(a), a network is deployed in the shape of an inverted 'U'. Moreover, regarding any path affected by holes, measurement errors may fluctuate tremendously. As shown in Fig. 2(b), the measured distance between s and $t1$ is much greater than its Euclidian distance because the path is heavily curved to bypass the hole, whereas, the measured distance from s to $t2$ is quite close to its Euclidian distance. On the other

hand, the measurement error reveals valuable information of the underlying WSN. The noticeable measurement errors of $P(s,t1)$ and $P(t1,t2)$ indicate irregularity of the network around two paths. $P(s,t1)$ is divided into three sub-paths, all of which are almost straight, at two turning nodes. It is obvious that the relationship between the measurement error along $P(s,t1)$ and the corresponding measured distance is nonlinear. Moreover, the measurement error is absolutely determined by two inflections corresponding to adjacent sub-paths. In fact, other paths passing through the local region around the turning node may also be inflected, and bring measurement errors.

The basic idea of our method is then vividly portrayed. We leverage the accurate location information from beacon nodes, and regard paths between beacon nodes as a virtual ruler that can measure the irregularity of the WSN. By comparing the measured distances along the shortest paths between pairs of beacon nodes and the corresponding Euclidean distances obtained from their coordinates, we are able to measure the degree of anisotropy and evaluate its impact on distance estimations of paths. Small differences between measured distances and Euclidean distances mean that paths are quite straight. Otherwise, paths are curved. In the latter case, turning nodes divide a path into several nearly straight sub-paths, and determine the inflection characteristics of the path. To differentiate the turning nodes from other sensor nodes along the path, we propose a new metric, dominating degree, which represents the correlations between nodes along the path. With dominating degrees of nodes along a path are calculated, turning nodes are distinguishable because of their lower dominating degrees. We further propose a distributed algorithm to calculate the bending angles of paths, i.e., the scale of the virtual ruler, at turning nodes according to the law of cosines. Subsequently, we can make appropriate adjustments on measured distances between any pair of sensor nodes with the virtual ruler.

C. Calculation of Dominating Degree

In order to represent the correlations among sensor nodes along a path, a new metric, dominating degree, is proposed as follows:

$$DD(n1,m,n2) = \frac{AvgDegree(DN(n1,m,n2))*M(n1,n2)}{NodeNum(DN(n1,m,n2))} \quad (1)$$

The principle of the dominating degree is described in Fig. 3(a). Regarding any node m on the path, together with its correlative nodes, e.g., $n1$ and $n2$, $DD(n1,m,n2)$ is the dominating degree of m. The set of dominating nodes $DN(n1,m,n2)$ is a set of nodes that are correlated with $n1$, m and $n2$. We obtain a set of surrounding nodes by executing a limited flooding operation from m. To eliminate the effects from uncertain shapes of holes near the path, we further divide these nodes into two sub-sets with the algorithm proposed in [14], and select the larger set. For an arbitrary node n, in the selected set of m, we compare its measured distance to m, $n1$ and $n2$, respectively, and then indicate it as an element of $DN(n1,m,n2)$ if $M(n,m) < M(n,n1)$ and $M(n,m) < M(n,n2)$ hold,

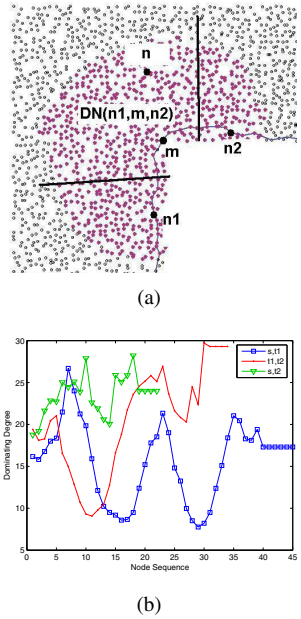


Fig. 3. Compute dominating degrees of sensor nodes. (a) A set of dominating nodes of a sensor node on a path. (b) Dominating degrees of sensor nodes along different paths in Fig. 2(a).

simultaneously. $AvgDegree(DN(n1,m,n2))$ is the average degree of all the nodes in $DN(n1,m,n2)$. By collecting all the messages of degree information from all the nodes in the $DN(n1,m,n2)$, m calculates the average degree value. $NodeNum(DN(n1,m,n2))$ is the size of the $DN(n1,m,n2)$. It is obvious that if the path is inflected, there are more nodes involved in the set of dominating nodes. Thus, the dominating degrees of turning nodes are relatively small when we investigate sensor nodes along inflected paths. Experiment results illustrate the validity of dominating degrees. As shown in Fig. 3(b), dominating degrees of nodes along different paths in Fig. 2(a) are depicted based on the order of the node sequence, where turning nodes are highlighted by their obviously lower dominating degrees.

D. Identifying Turning Nodes

Turning nodes along a path are identified by executing a 2-round calculation. In the first round, both the average value and the standard deviation of dominating degrees of all nodes along the path are computed. Thus, a threshold, which equal the average minus the standard deviation, is set up. In the second round, nodes along the path compare their dominating degrees with the threshold. If the dominating degree is below the threshold, the corresponding node sets itself as a turning node. This procedure can be launched by either end point of a path, where dominating degrees of all sensor nodes along the path have been collected.

We further set up a threshold $Min\_Interval$ to restrict the minimum interval between any adjacent turning nodes. The value of $Min\_Interval$ depends on the scale of the deployed network. It means if the distance between adjacent turning nodes is less than $Min\_Interval$, these adjacent

Algorithm 1 Get adjustment factor for beacon path $P(s,t)$

Require: dominating degrees along $P(s,t)$ have been obtained
Input: $M(s,t)$, $D(s,t)$
Output: adjustment factor of $P(s,t)$

1: $Link\_error \leftarrow \frac{M(s,t)-D(s,t)}{D(s,t)}$;
2: $Valid\_dist \leftarrow 0$;
3: **for** (each node m on $P(s,t)$) **do**
4: **if** (m is Turning Node) **then**
5: $Valid\_dist \leftarrow M(s,m) * 2$;
6: $Scale \leftarrow Scale * \frac{Basic\_Scale * Avg\_DD}{DD(m)}$;
7: **end if**
8: $HopDist \leftarrow M(s,m) - M(s,m-1)$;
9: **if** ($M(s,m) < Valid\_dist$) **then**
10: $Mod \leftarrow Mod + HopDist * Scale$;
11: **else if** ($M(s,m) < Valid\_dist * 2$) **then**
12: $Mod \leftarrow Mod - HopDist * Scale$;
13: **end if**
14: **end for**
15: $Factor \leftarrow \frac{Link\_error}{Mod}$;
16: **return** $Factor$;

turning nodes are considered to be continuous nodes. The set of the continuous turning nodes along a sub-section of the path is then represented by a single turning node, which is identified by calculating the average of the accumulating distances from continuous turning nodes to one end point of the path. Identified turning nodes compose the skeleton of a path, and partition the path into several sub-paths, which are nearly straight. Thus, the shape of a path is determined by two factors: the relative positions of turning nodes and the bending angles of a path at turning nodes. It is clear that the whole path is nearly straight if no turning node is identified. Otherwise, we should deduce the shape of a path according to these two factors. The relative positions of turning nodes, i.e., the lengths of the corresponding sub-paths, can be acquired by the same means as those used in the DV-Hop or the DV-Distance, whereas the bending angles cannot be acquired directly.

E. Constructing The Virtual Ruler

In this paper, the shortest paths between beacon nodes serve as a *virtual ruler* (VR) to evaluate the uncertainties and irregularities of a WSN. We leverage the accurate position information provided by the beacon nodes to establish the relationships between the dominating degree and the bending angle at each turning node, which is regarded as the scale of the VR. In particular, the method employed to mine the global characteristics consists of two stages: 1) initialization and 2) scale setting. We first adjust the distance estimations from each node on the VR to the corresponding beacon nodes. Then, we calculate the bending angles of the VR at each turning node. Finally, we can establish the relationships between the bending angles and the dominating degrees of the turning nodes.

1) Initialization Stage: As mentioned above, a curved path is separated by turning nodes into several straight segments, and the error of the measured distance increases nonlinearly according to the bending angle at a turning node. In the initialization stage, we try to ascertain the true distance from each turning node to the end-point of the path, which is used to calculate the bending angle of the path at each turning node. For an indicated beacon path, $P(s,t)$, this stage consists of two steps. The first step is a coarse-grained calculation. An adjustment factor of $P(s,t)$ is computed based on the $M(s,t)$, $D(s,t)$ and dominating degrees of turning nodes as described by Algorithm 1. The second step is a fine-grained adjustment along the beacon path, and can be regarded as a reverse procedure of the first step. Based on the adjustment factor of the beacon path and the dominating degree of each turning node, the scale of increase in measurement errors corresponding to each sub-path is then calculated. Subsequently, we can obtain adjustments on distance estimations from each turning node to both of the beacon nodes of the path.

Algorithm 1 is executed by beacon nodes. First, we calculate the error of the measured distance of the hole path (line 1), which is regarded as the result of errors accumulated along the beacon path. The error accumulating is denoted by Mod. After each node, Mod is incremented according to the measured distance of the hop, which is denoted by $HopDist$, and an incremental scale, which is denoted by $Scale$. After each turning node, we enlarge $Scale$ according to the dominating degree of the turning node, $DD(m)$(lines 4-7). The parameter $Basic\_Scale$ is a predefined parameter indicating the increase in $Scale$ after a turning node if the dominating degree is equal to the average value. If the measured distance between a sensor node and the turning node is below $Valid\_dist$, the measurement error increases when passing through this sensor node (line 10). Otherwise, it means that the sub-path is long enough to lead to a decrease in measurement errors (line 12). Finally, the adjustment factor of $P(s,t)$ is obtained (line 15).

In the fine-grained adjustment step, we calculate the measurement error for each node along the path based on the adjustment factor, and make appropriate adjustments to obtain the true distance from each turning node to the beacon node. The adjustment procedure is similar to that in Algorithm 1. For each node, we calculate the measurement error according to its relative distance to the preceding turning node as well as the adjustment factor, and make proper adjustments on the measured distance. Particularly, if the node is a turning node, we enlarge the incremental scale in measurement errors corresponding to according to its dominating degree.

2) Scale Setting Stage: In this stage, we establish the relationship between the bending angle and the dominating degree of each turning node, i.e., set the scale of the VR. In fact, the angle of an inflection on the beacon path can be calculated according to the law of cosine, with the true distance from the turning node to the beacon node, which has been calculated in the initialization stage. As shown in Fig. 4, $\angle(t1, TN1, TN2)$ and $\angle(TN1, TN2, t1)$ can be obtained according to $ED(t1, TN1)$, $ED(TN1, TN2)$, and $ED(TN2, t1)$, which have been obtained in the initialization stage. In the same way, $\angle(t1, TN2, s)$ can be obtained

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE INFOCOM 2010 proceedings
This paper was presented as part of the main Technical Program at IEEE INFOCOM 2010.

Algorithm 2 Calculate bending angle of inflections at each turning node along the beacon path $P(s,t)$

Require: algorithm 1 has been executed upon $P(s,t)$
Input: adjusted distance to s and t of all turning nodes
Output: bending angle of $P(s,t)$ at each turning node

1: collect all turning nodes into $Array[]$;
2: $LastNode \leftarrow s$; $LastAngle \leftarrow 0$; $index \leftarrow 1$;
3: **for** (each node $m \leftarrow Array[index > 0]$) **do**
4: $DA \leftarrow ED(m,t)$;
5: $m' \leftarrow Array[index+1]$; $DB \leftarrow ED(m,m')$;
6: $DC \leftarrow ED(m',t)$;
7: $CurrentAngle \leftarrow arcos(\frac{DA^2+DB^2-DC^2}{2*DA*DB})$;
8: bending angle at $m \leftarrow LastAngle + CurrentAngle$;
9: $LastAngle \leftarrow arcos(\frac{DB^2+DC^2-DA^2}{2*DB*DC})$;
10: $LastNode \leftarrow Array[index]$;
11: $index \leftarrow index + 1$;
12: **end for**
13: **return**

according to $ED(t1,TN2)$, $ED(TN2,s)$, and $ED(s,t1)$. Therefore, we have:

$$\angle(TN1,TN2,s) = \angle(TN1,TN2,t1) + \angle(t1,TN2,s)$$

For a path with many inflections, the procedure of angle calculation is depicted by Algorithm 2. We start from a beacon node, e.g., $t1$ in Fig. 4, to compute the angle of each turning node one by one along the path. Each angle is composed of two sub-angles which are represented by two parameters $LastAngle$ and $CurrentAngle$ in Algorithm 2. For each turning node, a triangle can be identified with two other nodes, i.e., the next turning node along the path and the starting beacon node (lines 4-6). Thus, we can compute $CurrentAngle$ by the law of cosine (line 7), and obtain the angle of the considered turning node (line 8). Afterwards, a new value of $LastAngle$ is calculated by the law of cosine (line 9) to serve for the computation of the next turning node.

For each pair of beacon nodes, Algorithm 2 can be executed twice from both of the end points of a path. We may obtain different values for the same angle corresponding to a turning node and take the average of them as the final result. After Algorithm 2 has been executed upon the paths among all of the beacon nodes, the bending angles corresponding to all the turning nodes are obtained. For each turning node, we then establish the relationship between the bending angle and its dominating degree by setting up a linear equation as follows:

$$Avg\_DD\_Value + DD\_Scale * Angle = DD\_Value$$

Avg_DD_Value is the average of the dominating degrees of the nodes on all the beacon paths in a WSN, except for all the turning nodes. As mentioned, a path is straight if there is no turning node identified. Thus, we can regard Avg_DD_Value as the characteristics of the straight paths. $Angle$ is the bending angle of the beacon path that passes through that turning node; DD_Value is the dominating degree of that

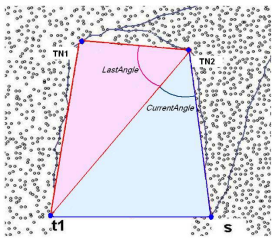


Fig. 4. Compute angles of turning nodes along the path $p(S,T)$. $TN1$ and $TN2$ are two identified turning nodes.

turning node. Supposing that there are N_{TN} turning nodes that have been identified on all the beacon paths, we can sum up all the equations. Then, the relationships between the dominating degrees and the bending angles in a WSN can be represented as follows:

$$DD\_Scale = \frac{\sum_{i=1}^{N_{TN}} DD\_Value_i \ - \ N_{TN} * Avg\_DD\_Value}{\sum_{i=1}^{N_{TN}} Angle_i}$$

Once the relationships between the dominating degrees and the bending angles are established, we can evaluate the inflections of paths from any sensor node to all the beacon nodes in the WSN. For each path, we first identify the turning nodes; then we can compute the bending angle of each inflection along the path with the following equation:

$$Angle \ = \ \frac{DD\_value - Avg\_DD\_value}{DD\_Scale} \qquad (2)$$

DD_value is the dominating degree of the identified turning node. $Angle$ is the bending angle of the path.

IV. Distance Estimation

After we have identified all the turning nodes along a path and have obtained the corresponding bending angles, we can then make adjustments to the measured distance of the path. This procedure consists of two phases: the measurement phase, followed by the computation phase. Suppose there exists a path $P(s,t)$ from an ordinary node s to a beacon node t. In the measurement phase, we measure $P(s,t)$ with the VR to identify all the turning nodes along this path. Meanwhile, we can obtain the measured distances between each pair of adjacent turning nodes and the dominating degrees of these turning nodes. In the computation phase, we compute the bending angles according to the dominating degrees of the turning nodes with equation (2). Finally, we can compute $ED(s,t)$ with the law of cosine.

A. A Basic Scenario

A basic scenario of distance estimation by a VR is shown in Fig. 5 (a), where $P(s,t)$ is curved by a hole at node n. The task is to make adjustments to the measured distance $M(s,t)$, i.e., the sum of $ED(s,n)$ and $ED(n,t)$. We assume that segments $P(s,n)$ and $P(n,t)$ are nearly straight, i.e., with fixed errors for zigzag effects. For the turning node n, the estimated distance to s and t can be calculated as follows:

$$ED(s,n) \ = \ \frac{M(n,s)}{(1 + Min\_Path\_Error)}$$

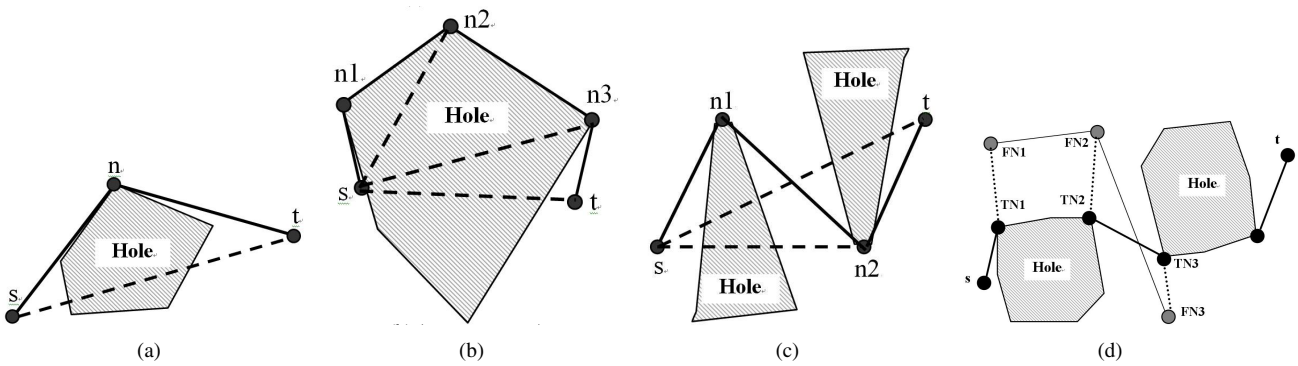


Fig. 5. Distance estimation in different scenarios. (a) A basic scenario. (b) A convex scenario. (c) A concave scenario. (d) $TN1$, $TN2$ and $TN3$ are identified turning nodes along $P(s,t)$. $FN1$, $FN2$ and $FN3$ are the farthest nodes corresponding to $TN1$, $TN2$ and $TN3$, respectively.

$$ED(n,t) \;=\; \frac{M(t,s) - M(n,s)}{(1 + Min\_Path\_Error)}$$

where Min_Path_Error is a predefined parameter denoting fixed error for zigzag effects. The angle of $\angle(s,n,t)$ can be calculated by equation (2) according to the DD value of n and DD_Scale of the WSN. Consequently, the estimated distance between t and s can be calculated according to the law of cosine, as follows:

$$ED(s,t) \;=\; \sqrt{d_1^2 + d_2^2 - 2 * d_1 * d_2 * cos(\angle(s,n,t))} \quad (3)$$

where $d_1 = ED(n,s); d_2 = ED(t,n)$.

In practical environments, a path may intersect with many holes or may intersect with a big hole at many nodes, and forms a convex path. A convex path is curved in the same direction at all turning nodes, as shown in Fig. 5(b). In this case, a sequence of turning nodes is obtained as $\{n1, n2, n3\}$. For each turning node, the angle is composed of two sub-angles as follows:

$$\angle(n1,n2,n3) \;=\; \angle(n1,n2,s) + \angle(s,n2,n3)$$
$$\angle(n2,n3,t) \;=\; \angle(n2,n3,s) + \angle(s,n3,t)$$

Obviously, each sub-angle can be calculated as in the corresponding sub-triangle. For example, the distance from $n2$ to s and the angle of $\angle(n1,n2,s)$ can be calculated in the triangle $\triangle(s,n1,n2)$. Then, we can obtain the angle of $\angle(s,n2,n3)$ as the difference between the angle of $n2$ and the angle of $\angle(n1,n2,s)$. The distance from $n3$ to s and the angle of $\angle(n2,n3,s)$ can be calculated in the same way. Finally, the distance between t and s can be achieved by the law of cosines.

B. Concave Path

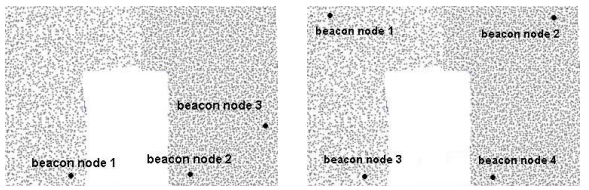
A concave path is different from a convex path in that the path is curved in different directions, as shown in Fig. 5(c). Similar to the method in the case of a convex path, a sequence of turning nodes is obtained as $\{n1, n2\}$. However, the angle corresponding to each turning node is part of a larger angle. For example, the distance from $n2$ to s and the angle of $\angle(n1,n2,s)$ can be calculated in the triangle $\triangle(s,n1,n2)$. The angle of $\angle(s,n2,t)$ is the sum of $\angle(n1,n2,s)$ and the bending angle corresponding to $n2$. Then, the estimated distance between t and s can be calculated by the law of cosines. Compared with the case of a convex path, we can find that if the path is concave at a turning node, the angle used in the next step of the computation is the sum of the preceding angle and the angle of this turning node. In the case of a convex path, the needed angle is the difference between the angle of this turning node and the angle of the preceding turning node. Therefore, a convex path is just a special case of a concave path.

C. Distinguishing Convex Paths from Concave Paths

In this paper, we distinguish concave paths from convex paths with a simple approach, which is similar to the method proposed in [11]. Regarding each turning node, we designate the node that has the largest distance from the path in the set of dominating nodes as the farthest node of the turning node. Considering a convex path, the farthest nodes of all turning nodes lie on the same side of the path. As shown in Fig. 5(d), the farthest nodes of $TN1$ and $TN2$ are $FN1$ and $FN2$, respectively, which are on the same side of $P(s,t)$. Whereas in the case of concave paths, the farthest nodes of turning nodes, which bend in different directions, lie on the opposite sides of the path, for example, $FN3$ is the farthest node of $TN3$, which is on the opposite side of $FN2$. Therefore, the difference helps us to tell one kind from the other. Let's connect the farthest nodes of all the adjacent turning nodes. If the connection crosses the path, the path is concave at that turning node. Otherwise, the path is convex.

V. Applicability And Overhead

In this section, we summarize our method regarding aspects of applicability and overhead. Our method can supply relatively accurate distance estimations for localization in an anisotropic WSN with as few as 3 beacon nodes. Benefiting from its distributed manner, no super node, cluster header, or sink node is needed to accomplish computation and adjustment. Except for beacon nodes, all flooding operations for sensor nodes are limited within a threshold (< 10). Therefore, the communication overhead is bounded by $O(n^2)$, where n is the number of sensor nodes in a WSN. Beacon nodes bear most of the computation burden. For each path, the beacon



(a) (b)

Fig. 6. The WSN deployments used in the simulation. (a) Three beacon nodes are deployed at the bottom. (b) Four beacon nodes are deployed uniformly.

node needs to perform $O(L)$ computations to accomplish characteristic mining, where L is the average path length in the WSN. Thus, for each beacon node, the computation overhead is $O(nL)$.

We compare our method with three proposed approaches: DV-Distance, PDM, and REP. DV-Distance has the same communication cost as our method, but it has a lower computation cost, which is equal to $O(n)$. PDM can handle anisotropic deployment, and has the same communication cost as our method; however, it relies on uniform deployment of beacon nodes and requires a higher computation cost of $O(n^3)$ because of its global manner of optimization. REP has the same principle as our method in making adjustments on measured distances by making local optimizations and adjustments, so that it can deal with anisotropic deployment of beacon nodes. It has the same communication cost as our method. REP assumes that the boundaries of the holes in the WSN have been identified, so it has a lower computation cost, which is equal to $O(n)$. However, identification of boundaries is a difficult problem that requires both high communication costs and high computation costs [14].

VI. Performance Evaluation

A. Simulation Setup

To evaluate the ranging ability of our scheme, especially in anisotropic WSNs where beacon nodes are not deployed uniformly, a series of simulations are conducted. The simulation system is implemented by MS Visual C++, and runs on a PC with $1G$ memory. DV-Distance, PDM and our scheme are implemented, respectively. Two scenarios are shown in Fig. 6. The unit-disk model is abandoned in this paper. 1-hop neighbors are connected by bidirectional links, which are established by probability, as shown in Table 1. For each of the different types of networks, we perform 10 rounds of running the simulation, during which the quantity of the deployed sensor nodes maintains unchanged. However, the topology of the network varies because we establish connectivity between pairs of sensor nodes, randomly. We focus on investigating distance estimation errors from sensor nodes to beacon nodes.

B. Comparative Study

In the first scenario, we evaluate the impact of asymmetrical deployment of beacon nodes and the anisotropic properties of a WSN including deployment densities and irregular obstacles. The deployment of sensor nodes and beacon nodes is shown in Fig. 6 (a). 5548 nodes are randomly deployed with an average node degree of 6.2. Three beacon nodes are deployed, all of which are laid on the bottom side of the WSN. There is a hole on the bottom side. In addition, the node density of the right-side is 1.4 times that of the left-side.

TABLE I
Probabilities to establish links at different distances.

Distance ratio	100%	90%	80%	70%	60%
Probability	0.5	0.5	0.5	0.6	0.7
Distance ratio	50%	40%	30%	20%	10%
Probability	0.8	0.8	0.8	0.9	0.9

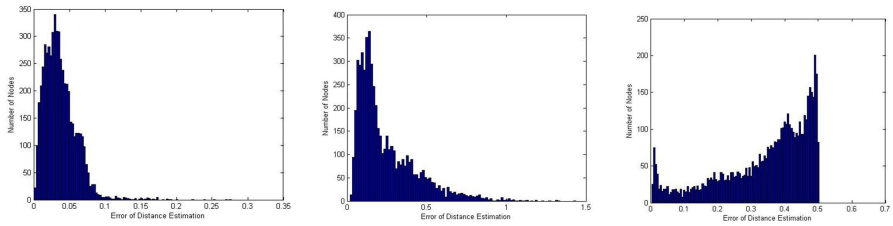
Distributions of average estimation errors by three implemented methods are shown in Fig. 7. Our method yields the smallest amount of errors, where the estimation errors of most nodes are below 10%. DV-Distance gives a moderate performance where the estimation errors of most nodes lie from 20% to 50%. PDM's performance is poor in this scenario, where most sensor nodes obtain a high error rate from 30% to 50%. However, PDM gives an upper limit of estimation errors because of the global optimization on each sensor node.

Fig. 8 gives insight into the relationships between distance estimation errors and the positions of beacon nodes. DV-Distance yields higher estimation errors when the path becomes more inflected, i.e., the path is close to the hole. It also yields a higher error rate in the region with lower deployment density, i.e., on the left-side. Compared with DV-Distance, our method can overcome anisotropy of both irregular holes and varied node densities. PDM performs well near beacon nodes. However, it gives serious errors for nodes far away from beacon nodes because it ignores the local characteristics of nodes in different positions. Owing to localized optimization, our method performs better against all of these anisotropic properties.

In the second scenario, the simulation is re-executed in the same node density setting, where four beacon nodes are deployed more uniformly, as shown in Fig. 6(b). Distributions of average estimation errors by the three implemented methods are shown in Fig. 9. All three methods receive benefits from uniform deployment of beacon nodes. Particularly, estimation errors by PDM clearly drop from about 50% to about 6%. In this scenario, the results of the three methods are quite similar to one another.

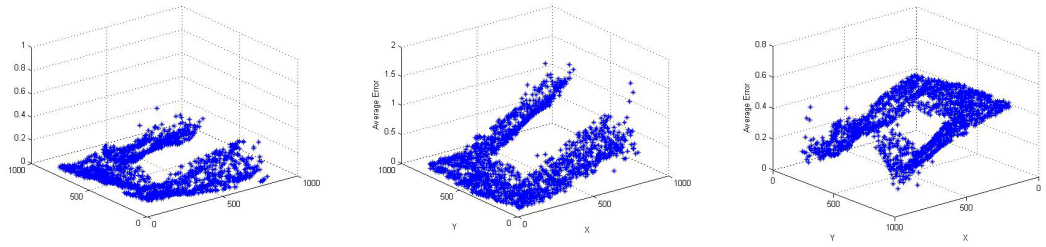
VII. Conclusion

In this paper, we design and evaluate a new distance estimation scheme for anisotropic WSNs. Different from most existing work, our scheme extracts unique characteristics of local regions through a metric, dominating degree, and chooses local and nonlinear, rather than global and linear, optimization to adjust measured distances for beacon node paths. Therefore, the method performs better in a network where beacon nodes are not deployed uniformly. Simulation results and analysis show that our scheme conducts more accurate distance estimation in anisotropic WSNs, especially when beacon node



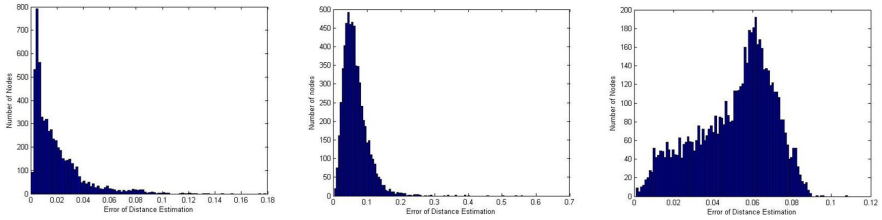
(a) (b) (c)

Fig. 7. The distribution of distance estimation errors in the first scenario. (a) Our method. (b) DV-Distance. (c) PDM.



(a) (b) (c)

Fig. 8. The errors of distance estimation at different positions. (a) Our method. (b) DV-Distance. (c) PDM.



(a) (b) (c)

Fig. 9. The distribution of distance estimation errors in the second scenario. (a) Our method. (b) DV-Distance. (c) PDM.

deployment is non-uniform. Future research will focus on improving the effectiveness of the method in scenarios where it is hard to construct such a virtual ruler. For example, holes are in very special shapes, such as circles, and are far from all beacon nodes.

ACKNOWLEDGMENT

This research work is partially supported by the Natural Science Foundation of China under grant No:60973122, the 973 Program under grant No: 2009CB320705 in China and in part by US NSF grants CNS 0422762, CNS 0626240, CCF 0830289, and CNS 0948184. Thanks to all the anonymous reviewers for their valuable comments.

REFERENCES

[1] D. Niculescu and B. Nath, Ad hoc positioning system (APS), In Proc. IEEE GLOBECOM, 2926-2931 (2001)
[2] K. Whitehouse, C. Karlof and D. Culler, A Practical Evaluation of Radio Signal Strength for Ranging-based Localization, ACM Mobile Computing and Communications Review, Special Issue on Localization Technologies and Algorithms. Vol.11, No.1, 41-52 (2007)
[3] D. Niculescu and B. Nath, Error characteristics of ad hoc positioning systems (APS), In Proc. ACM international symposium on mobile ad hoc networking and computing, 20-30 (2004)
[4] K. Whitehouse, C. Karlof, A. Woo, and et al, The Effects of Ranging Noise on Multihop Localization: An Empirical Study, In Proc. the 4th International Symposium on Information Processing in Sensor Networks, 73-80 (2005)
[5] K. Whitehouse and D. Culler. A Robustness Analysis of Multi-hop Ranging-based Localization Approximations, In Proc. the 5th International Conference on Information Processing in Sensor Networks, 317-325 (2006)
[6] C. Wang and L. Xiao, Sensor Localization in Concave Environments, In In ACM Transactions on Sensor Networks, Vol. 4, No. 1, Article 3 (2008)
[7] H. Lim and J.C. Hou, Distributed Localization for anisotropic sensor networks, In ACM Transactions on Sensor Networks, Vol. 5, No.2, Article 11 (2009)
[8] L. Li and T. Kunz, Cooperative Node Localization Using Nonlinear Data Projection, In ACM Transactions on Sensor Networks, Vol. 5, No.1, Article 1 (2009)
[9] J.J. Pan, J.T. Kwok and Q. Yang, Multidimensional Vector Regression for Accurate and Low-Cost Location Estimation in Pervasive Computing, In IEEE Transactions on Knowledge and Data Engineering Vol.18, No.9, 1181-1194 (2005)
[10] A. Savvides, W. Garber, R. Moses, and et al. An Analysis of Error Inducing Parameters in Multihop Sensor Node Localization, In IEEE Transactions on. Mobile Computing. Vol.4, No.6, 567-577 (2005)
[11] M. Li and Y. Liu, Rendered Path: Range-Free Localization In Anisotropic Sensor Networks with Holes, In Proc. ACM MOBICOM, 51-62 (2007)
[12] S. Lederer, Y. Wang and J. Gao, Connectivity-based Localization of Large Scale Sensor Networks with Complex Shape, In Proc. IEEE INFOCOM, 789-797 (2008)
[13] Y. Wang, S. Lederer and J. Gao, Connectivity-Based Sensor Network Localization with Incremental Delaunay Refinement Method, In Proc. IEEE INFOCOM, 2401-2409 (2009)
[14] Y. Wang, J. Gao and J. Mitchell, Boundary Recognition in Sensor Networks by Topology Methods, In Proc. ACM MOBICOM, 122-133 (2006)