An Approximate Convex Decomposition Protocol for Wireless Sensor Network Localization in Arbitrary-Shaped Fields

Wenping Liu, *Member, IEEE,* Dan Wang, *Member, IEEE,* Hongbo Jiang, *Member, IEEE,* Wenyu Liu, *Member, IEEE* and Chonggang Wang, *Senior Member, IEEE*

Abstract—Accurate localization in wireless sensor networks is the foundation for many applications, such as geographic routing and position-aware data processing. In this paper, we develop a new localization protocol based on approximate convex decomposition (ACDL), with reliance on network connectivity information only. ACDL can calculate the node virtual locations for a large-scale sensor network with a complex shape. We first examine one representative localization algorithm and study the influential factors on the localization accuracy, including the sharpness of the angle at the concave point and the depth of the concave valley. We show that after decomposition, the depth of the concave valley becomes irrelevant. We thus define the *concavity* according to the angle at a concave point, which reflects the localization error. We then propose ACDL protocol for network localization. It consists of four main steps. First, convex and concave nodes are recognized and network boundaries are segmented. As the sensor network is discrete, we show that it is acceptable to approximately identify the concave nodes to control the localization error. Second, an approximate convex decomposition is conducted. Our convex decomposition requires only local information and we show that it has low message overhead. Third, for each convex subsection of the network, an improved MDS algorithm is proposed to compute a relative location map. Fourth, a fast and low complexity merging algorithm is developed to construct the global location map. Besides, by slight modification on the third step, we propose a variant of ACDL, denoted by ACDL-Tri, which is fully distributed and scalable while the localization accuracy is still comparable. We finally show the efficiency of ACDL by extensive simulations.

Index Terms—Localization, approximate convex decomposition, wireless sensor networks

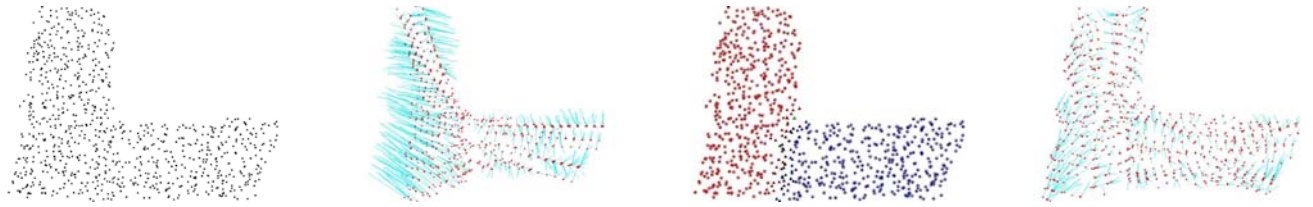
1 Introduction

Location-based service in wireless sensor networks is a key technology for many applications; and localization has attracted academic interest for a long time. The most straightforward method is to use the global positioning system (GPS). Nevertheless, having each node GPS-equipped is extremely expensive for wireless sensor networks. Many algorithms have been proposed to estimate the sensor locations using local information only; a survey on location, localization and localizability can be found in [20].

Recently there has been growing interest in localization protocols that use the connectivity information only. This aims to produce a relative coordinate system for a network without reliance on extra hardware supplements. These schemes can accurately recover the original network topology, up to scaling and rotation. Among the many studies, Multi-Dimensional Scaling (MDS) based localization techniques have been proved to compute locations of high accuracy and request low node density. A state-of-the-art MDS based algorithm, MDS-MAP [24], [25], takes an inter-node hop distance matrix as input, and generates a set of relative coordinates for each node. Nevertheless, the accuracy of MDS-MAP heavily depends on the assumption that the hop-count distance between two nodes correlates well with their Euclidean distance. Such assumption is valid only when the network is in a convex field. In real world, however, this is hardly true. In *anisotropic* networks with concave regions, the shortest path may be significantly bent [14]. As a result, the hop-count distance between nodes would deviate from the Euclidean distance.

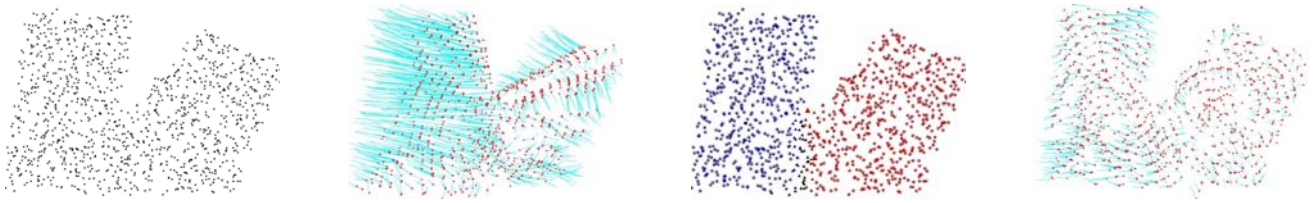
To avoid using hop-count distance between far-away nodes (or to avoid mistakenly using the deviated shortest path), some studies [3], [11], [12], [31] first locate landmark network. The landmark network is composed of nodes that are uniformly sampled from the original network and the density of the landmark network is usually set by a system parameter. In [11], [12], [31], a triangular mesh structure (the landmark network) is constructed. Each non-landmark node then trilaterates its own location according to the distances to its closest three landmarks. MDS-MAP(P) [23], an improved MDS-based localization scheme, proposed to build relative coordinate system for each node, and then merge these relative coordinate systems to form a global coordinate system. Unfortunately, its performance relies heavily on the properly chosen core map, and the refinement used

- *W. Liu, H. Jiang (corresponding author), W. Liu are with Department of Electronics and Information Engineering, Wuhan National Laboratory for Optoelectronics, Huazhong University of Science and Technology, China. W. Liu is also with Hubei University of Economics, China. E-mail: hongbojiang2004@gmail.com*
- *D. Wang is with Department of Computing, The Hong Kong Polytechnic University, Hong Kong.*
- *C. Wang is with InterDigital Communications, U.S.A.*



(a) (b) (c) (d)

Fig. 1: Localization of an L-shape network with 851 nodes, avg. deg. 11.29. (a) Original map; (b) MDS-MAP; (c) ACD; (d) ACDL.



(a) (b) (c) (d)

Fig. 2: A sharper L-shape network has 1191 nodes with avg. deg. 12.47. (a) Original map; (b) MDS-MAP; (c) ACD; (d) ACDL.

in MDS-MAP(P) for a better result is time-consuming which is not feasible for large-scale sensor networks. Respecting the fact that in anisotropic sensor networks with holes, the hop count distance between nodes may do not correlate well with the Euclidean distance, Li [14] proposed REP, a rendered path protocol which can estimate the pairwise distance between nodes by rendering the shortest path and constructing the virtual holes, and thus deliver a localization result with high accuracy. However, the performance of REP relies on the position of beacon nodes, and it is a nontrivial task to construct a proper virtual hole in discrete sensor networks. CATL [27] is a recent state-of-the-art localization algorithm. The key idea of CATL is to identify notch nodes where the hop-count of the shortest path between nodes deviates the true Euclidean distance. CATL then uses an iterative notch-avoiding multilateration scheme to localize the network. The performance of CATL heavily depends on proper deployment of beacon nodes. In addition, due to the iterative procedure, CATL suffers from error propagation.

In this paper we develop a new localization protocol based on approximate convex decomposition (ACDL). ACDL decomposes the network into several convex subsections. In each subsection, the hop-count distance between nodes can provide a good approximation of the Euclidean distance. ACDL finally unifies the locations of all subsections. ACDL works well not only for arbitrary network shape but also for low density networks since it does not rely on the quality of the extracted triangular mesh like [11], [12], [31]. It avoids localization error propagation introduced by iterative procedure like [27]. More importantly, ACDL is not exclusive. It can easily incorporate the state-of-the-art localization algorithms such as [11], [12], [27], [31]. For instance, for the case when individual nodes are extremely resource-constrained and not capable of performing the improved MDS which we use in each convex subsection to compute the relative location map, CATL [27] or a trilateration scheme can be applied where a node use distance measurements to at least three reference points within its convex subsection.

Convex decomposition, with or without Steiner point, has attracted interests from computer graphics community for a long time [1], [4], [7], [15]. For instance, Chazelle *et al.* [4] proposed to decompose a non-convex polygon into minimal number of convex sub-polygons by introducing so-called X-patterns, where an X_k-pattern, composed of k segments with one common end point, is a particular interconnection of k notch (or reflex) vertices to remove the k notches without incurring any new notches; Lien *et al.* [15] have studied the problem of approximate convex decomposition of a polygon. It first computes concavity of each vertex based on so-called *bridge* and *pocket* for vertices on external and hole boundaries separately; it then cursively resolves concave points in order of decreasing significance until each point has a concavity larger than the given threshold, and eventually, the polygon is decomposed into approximate convex pieces. However, these algorithms often require coordinate information and work in a centralized fashion, which is not applicable for large-scale sensor networks.

As such, for approximate convex decomposition based localization in sensor networks, there are many difficulties to be addressed. *First*, there is a lack of understanding of localization error and concavity. *Second*, since the sensor network is discrete, due to boundary noise, it is not easy to clearly specify the nodes that are concave of the network. *Third*, for the available convex decomposition schemes from the computer graphics community, they target at continuous shapes and use centralized solutions. Due to the nature of the random deployment of sensor networks, it is impractical for a man to manually identify convex/concave regions dur-

ing deployment or just extract a graph of the network. In addition, in our situation, the network is discrete and the sensor nodes can only obtain local information by message exchange. A low communication complexity scheme is necessary considering the limited resource of each sensor node. *Fourth*, after convex decomposition, it is not straightforward to restore a global map with the relative coordinates. In addition, the restoration algorithm should be efficient considering that it is possible that the network is decomposed into a great number of convex subsections.

In this paper, we provide a systematic study on the aforementioned problems. Our key contributions are summarized below:

- We illustrate the intrinsic problems of localization algorithm on concave-shaped networks. We show that the location accuracy is closely related to the angle at the concave points and it is also related to the depth of the concave "valley". After convex decomposition, the depth of the valley becomes irrelevant, however. With these observations, we make a formal definition of network *concavity* according to the angle at the concave point.
- We develop a distributed approximate convex decomposition algorithm, based on the boundary branches. Our algorithm has low communication complexity.
- An improved MDS is applied in each convex subsection to compute the relative location map. The computation of MDS is $O(N^3)$ but our improved MDS has a complexity of $O(N^2)$.
- We have shown that our proposed ACDL protocol is by no means exclusive. Instead, it can be easily extended to incorporate other algorithms. For instance, a fully distributed procedure for the local relative map establishment using trilateration can be developed. This variant of ACDL, denoted by ACDL-Tri, only causes $O(N)$ message cost and $O(1)$ computational cost. See the supplementary file.

The rest of this paper is organized as follows. Section 2 presents the motivation of our work and the background of our convex decomposition scheme. Section 3 is devoted to our approximate convex decomposition based localization algorithm. We evaluate the performance of ACDL in Section 4, and section 5 concludes the paper.

2 MDS Localization and Network Concavity

Our localization protocol relies upon the convex decomposition. To this end, one essential step is to identify concave node(s). Our idea is quite simple: a boundary node identifies itself a concave node when its curvature is greater than a given threshold (Please refer to [5] for the definition of boundary). We will define the node curvature later. First we illustrate the intrinsic problems of traditional algorithm on concave-shape networks.

TABLE 1: List of notations

Notation	Definition
V	The set of sensor nodes.
N	The number of sensor nodes.
D	The pairwise distance matrix of the network.
B	The inner product matrix of D.
$C_k(V)$	The concavity of the network.
$N_k(p)$	k neighborhood of p, which is the set of nodes which are at most k hops from node p.
$\partial N_k(p)$	k-hop neighborhood of p, which is the set of nodes exactly k hops away from node p.
$D_k^p(q_1, q_2)$	The perimeter from q_1 to q_2 within $\partial N_k(p)$.
$\|D_k^p(q_1, q_2)\|$	The perimeter distance from q_1 to q_2 within $\partial N_k(p)$.
$c_k(p)$	The k-hop curvature of node p.
Sec_i	The i-th approximate convex subsection.
n_i	The number of sensors in Sec_i.
D_i	The pairwise distance matrix in Sec_i.
B_i	The inner product matrix of D_i.

Let N be the number of sensors in the network. (A list of notations can be found in Table 1). Multi-dimensional scaling [2] first computes an $N \times N$ distance matrix D, which represents the pairwise distance between two nodes, thereby calculating the inner product matrix B of the pairwise distance matrix D. MDS then applies spectrum decomposition on matrix B to extract all eigenvalues and their corresponding eigenvectors of matrix B. Finally, MDS computes locations based on the first 2 largest eigenvalues and eigenvectors. MDS-MAP [24], [25] is an MDS-based localization algorithm. It uses the hop-count of the shortest path between two nodes as the pairwise distance in the matrix. MDS-MAP works well for the sensor field with a simple shape such as a square or a disk. This is because the hop-count of the shortest path between two nodes is a good approximation of the Euclidean distance. For complex networks, this condition could be no longer valid.

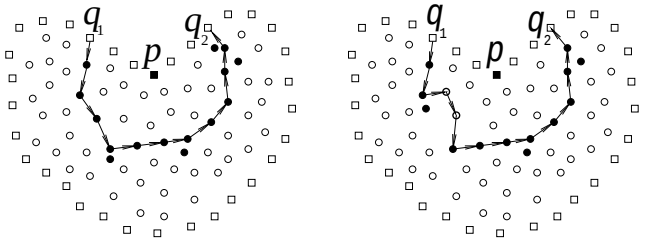
As a concrete example, in Fig. 1, we apply MDS-MAP directly to an L-shape network (Fig. 1 (a)). In Fig. 1 (b), the line associated with each node is the deviation between the real location and computed location by MDS-MAP. Not surprisingly, the accuracy of MDS-MAP is low. Especially, the nodes at the end of the two arms of the L-shape network generally have greater errors. To further understand the impact of concavity, we evaluate another L-shape network in Fig. 2. Note that the angle at the concave point in Fig. 2(a) is sharper as compared to that in Fig. 1 (a). From Fig. 2(b), we can see the location accuracy is even worse due to the larger deviation between hop count distance and Euclidean distance caused by larger angle at concave point. Based on these experiments, it is easy to observe 1) the angle of the concave point is of crucial importance; the sharper the angle is, the worse the performance of MDS-MAP; and 2) the sensors at the two arms of the L-shape network suffer greater errors. This indicates that the depth of a concave valley can also be an important factor. A natural idea is to decompose the network into convex regions. To evaluate the effectiveness of such idea, we manually decompose the L-shape network into two convex subsections, see Fig. 1 (c). We then conduct

the MDS localization algorithm, and the result is shown in Fig. 1 (d). We see that the localization errors are greatly reduced. In addition, the depth of the concave valley becomes irrelevant.

With these observations, next we turn to the concave node definition. We define the k *neighborhood* of node p, represented by $N_k(p)$, as the set of the nodes at most k hops away to p. Let k-*hop neighborhood* of p, denoted as $\partial N_k(p)$, be the nodes exactly k hops away from p. Intuitively, $\partial N_k(p)$ can be treated as a (or a part of) circle centered at node p.

For a boundary node p and two boundary nodes $q_1, q_2 \in \partial N_k(p)$, denote by $D_k^p(q_1, q_2)$ and $|D_k^p(q_1, q_2)|$ the *perimeter* and *perimeter distance* from q_1 to q_2, respectively. To estimate the perimeter distance, there are two cases:

1) If $\partial N_k(p)$ is connected, $D_k^p(q_1, q_2)$ is the set of nodes on the shortest path from q_1 to q_2(including q_1 and q_2) using the nodes in $\partial N_k(p)$, and $|D_k^p(q_1, q_2)|$ is the number of sensors in $D_k^p(q_1, q_2)$ minus one, as shown in Fig. 3(a).



(a) (b)

Fig. 3: Perimeter distance of boundary node p (shown in solid rectangular). Boundary nodes are shown in empty rectangular. The nodes in $\partial N_3(p)$ are shown in solid circle. Boundary nodes q_1 and q_2 are $p's$ 3-hop neighborhood, and the hop-count is indicated by the arrows. (a) $\partial N_3(p)$ is connected. The perimeter distance $|D_3^p(q_1, q_2)| = 13 - 1 = 12, c_3(p) = \frac{12}{3\pi} = 1.27$. If $\delta_1 < 0.27$, p will be a concave node; (b) $\partial N_3(p)$ is disconnected. The auxiliary nodes to make $\partial N_3(p)$ a connected component are shown in solid diamond, and $\triangle_1 = 1$. The perimeter distance $|D_3^p(q_1, q_2)| = (14-1)-1 = 12, c_3(p) = \frac{12}{3\pi} = 1.27$. If $\delta_1 < 0.27$, p will be a concave node.

2) If $\partial N_k(p)$ is disconnected, we use some auxiliary nodes to estimate the perimeter and perimeter distance in a greedy manner. More specifically, the boundary node q_1(or q_2) initiates a flooding. When an intermediate node q receives the flooding message, say, from $q\prime$, it executes the following rules:

- if q has not received the message before, then q keeps record of the parent node $q\prime$ and its hop count distance to p, and forwards the message to its neighbors;
- else if the hop count distance of $q\prime$ to p is no greater than k and larger than that of the parent of q to p, then q updates its parent node as $q\prime$, and forwards the message to its neighbors;
- otherwise, q discards the message.

As such, the greedy path from q_1 to q_2 can be built, please see Fig. 3(b). We denote by $\overline{D_k^p(q_1, q_2)}$ the nodes on the greedy path. Note that the perimeter might be broken into components owing to the presence of small holes, and thus the auxiliary nodes will form connected components. As such, the perimeter distance can be estimated by $(|\overline{D_k^p(q_1, q_2)}| - \sum_i \triangle_i)$, where $\triangle_i$ is the difference between k and the minimal hop count distance of the nodes in the i-th auxiliary connected component on the greedy path. Obviously, this process is applicable for the case when $\partial N_k(p)$ is connected where $\sum_i \triangle_i = 0$.

With the obtained perimeter distance, we thus define the *concavity* by:

Definition 1. *Assume $q_1, q_2 \in \partial N_k(p)$ are two boundary nodes which are on the same boundary with p. The k-*hop concavity *(or so-called curvature) of p, $c_k(p)$, is given by:*

$$c_k(p) = \frac{|D_k^p(q_1, q_2)|}{\pi \times k} \tag{1}$$

Intuitively, if $c_k(p)$ is equal to 1, p is not a concave/convex point. Due to the discrete nature of wireless sensor networks, the presence of boundary noise will incur many boundary nodes have a curvature which is larger than 1. As such, to control such boundary noise, we introduce two thresholds and our definitions on concave/convex nodes are as follows.

Definition 2. *Given $\delta_1 > 0$ and $\delta_2 > 0$, a boundary node p is a* concave node *if $c_k(p) > 1 + \delta_1$, or a* convex node *if $c_k(p) < 1 - \delta_2$.*

Clearly, the larger the curvature of the concave node, the larger the deviation between the real location and computed location. Note that for a sensor network, there can be many concave nodes. The concave node q with the maximal concavity has the most significant influence on the localization accuracy of the network. The *network concavity*, represented by $C_k(V)$, is defined by the maximum concavity of all concave nodes, $C_k(V) = max_{s \in V_c} c_k(s)$, where V_c is the set of the concave nodes.

Based upon the definition of the concave/convex nodes, we will design a distributed algorithm in next section, which can find the concave/convex nodes, decompose the network and compute the locations with high accuracy. Note that the definitions of concavity and concave/convex node depend on parameters k, which corresponds to the depth of the valley, as well as δ_1 and δ_2, which correspond to the sharpness of the angle. We will conduct extensive simulations to study the impact of the parameter settings. Intuitively, a smaller value of δ_1 implies that more concave nodes will be identified, and the network will be partitioned into more subsections, each of which has a smaller concavity. Consequently, within each subsection, the localization errors (defined in Section 4) will be smaller. However, as the number of subsections increases, the accumulated errors in the process of merging local maps increase accordingly, and the message overhead will also increase. On the other side, if δ_1 is larger, less concave nodes will be identified, and the number of convex subsections (therefore the accumulated errors) will also be smaller. However, the localization errors of each subsection will become larger.

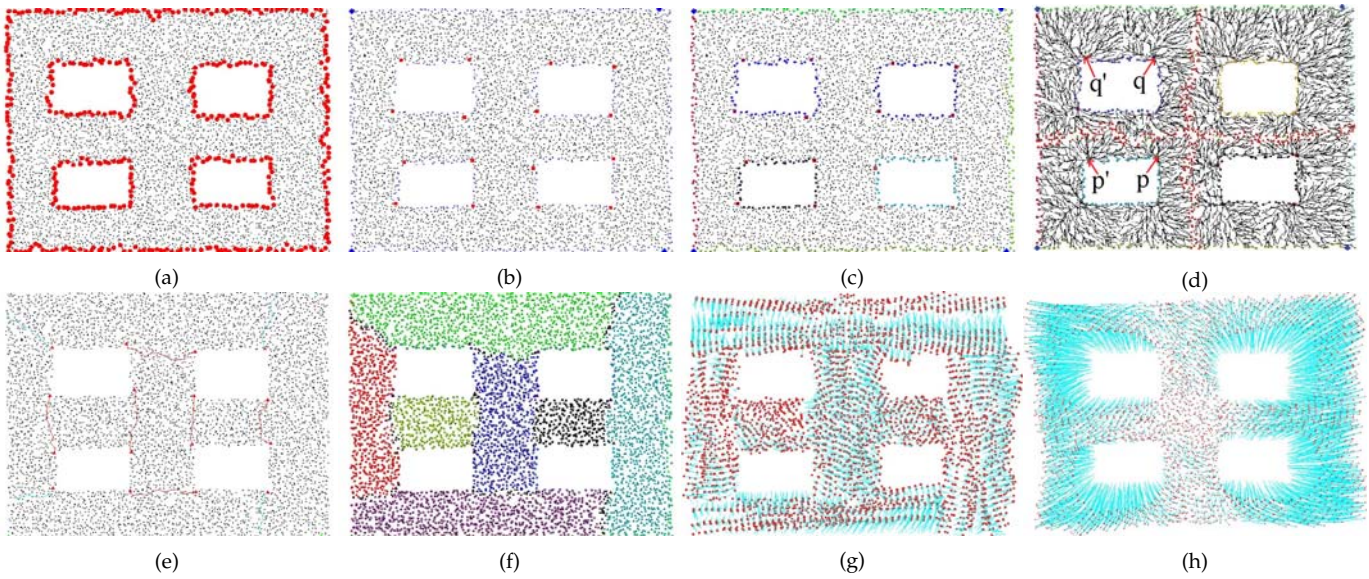


Fig. 4: Localization for window-shape network with 5184 nodes and average degree 12.77. (a) Original map; (b) Concave/convex nodes, $k = 5, \delta_1 = 0.5, \delta_2 = 0.2$. The red rectangular-shaped nodes are concave nodes and the blue diamond-shaped nodes are convex nodes; (c) Boundary branches. Boundary nodes on different branches are marked different colors; (d) Cut trees. Cut nodes are marked red; (e) Segment lines; (f) Approximate convex decomposition; (g) Localization result by our method. Average localization error is 0.43; (h) Localization result by MDS-MAP. Average localization error is 2.18.

We conduct extensive simulations in Section 4 to study the impact of the parameter settings.

3 Approximate Convex Decomposition-based Localization (ACDL)

3.1 An Overview of ACDL

We capitalize that since the sensor network is discrete, it is impractical to decompose the wireless sensor network into strictly convex subsections due to the boundary noise. MDS-based localization tolerates localization error gracefully due to its over-determined nature [23]. As such, we only have to decompose the network into approximate convex subsections. For each approximate convex subsection, the maximal concavity is less than the given threshold value $1 + \delta_1$.

Our Approximate Convex Decomposition-based Localization algorithm consists of the following four steps:

(1) Concave/convex Node Recognition and Boundary Segmentation: Assume we have the boundaries of the network (boundary identification is out of the scope of our paper and there are many existing works [5], [6], [22], [30] and we use [30] in our simulation), we first identify the concave and convex nodes. We then segment the boundaries into several boundary branches using these convex nodes. These branches will serve as the basis for our decomposition.

(2) Approximate Convex Decomposition (ACD): The key problem is to find the "lines" that can decompose the network into convex subsections. These lines may end at concave nodes (and boundary nodes). We develop a distributed algorithm to identify these lines.

(3) Local Relative Map Establishment: We proposed an improved MDS technique to build the relative map for each convex subsection. Our algorithm has a computational complexity of $O(N^2)$ while the conventional MDS is $O(N^3)$,

(4) Global Map Establishment: Finally, we will combine the coordinates of all the subsections into a global map. Note that the combination process will need to inform each sensor in the subsection of the new coordinates. If the combination process is conducted one subsection at a time, such combination will be slow. Thus, we develop an algorithm which can balance the time of combination process and the message overhead.

3.2 Concave/Convex Node Recognition and Boundary Segmentation

Given the definition for concave/convex node, we present how each node identifies itself distributedly .

As discussed, each node first uses the technique in [30] to identify whether it is a boundary node. For each boundary node p, it first floods the network for k hops to obtain its k-hop neighborhood $\partial N_k(p)$. Two nodes who belong to $\partial N_k(p)$ and are also boundary nodes identify themselves. Let these two nodes be q_1, q_2. q_1 and q_2 then flood to derive the perimeter distances $|D_k^p(q_1, q_2)|$ and $|D_k^p(q_2, q_1)|$; and send this information to p. Obviously, $|D_k^p(q_1, q_2)| = |D_k^p(q_2, q_1)|$. With the perimeter distance $|D_k^p(q_1, q_2)|$ estimated, node p computes its curvature and identifies whether it is a concave/convex node, as shown in Fig. 4(b). We show this process in Algorithm 1.

Using the convex nodes, the boundaries are automatically segmented into several *boundary branches*, i.e., a sub-boundary including all nodes between two adjacent convex nodes; see Fig. 4(c) for an example. Each convex node floods in the boundary nodes to segment the

Algorithm 1 Concave/convex Node Recognition

1: p obtains its k-hop neighborhood $\partial N_k(p)$, two of which are boundary nodes, denoted by q_1, q_2.
2: p derives the perimeter distance $|D_k^p(q_1, q_2)|$.
3: p computes its concavity $c_k(p)$ using Equ. (1).
4: **if** $c_k(p) > 1 + \delta_1$ **then**
5: Node p identifies itself as a concave node.
6: **else if** $c_k(p) < 1 - \delta_2$ **then**
7: Node p identifies itself as a convex node.
8: **end if**

boundary into several branches. When a boundary node p receives a packet from a convex node, it sends this packet to its neighboring boundary node if p is not a convex node, otherwise it discards this packet. As such, each boundary nodes will keep track of two convex nodes which determine a unique boundary branch. Note that it is possible that there are no convex nodes identified for some boundaries (e.g., the boundaries of the inner holes in Fig. 4). To deal with this, any node on each of these boundaries floods within the same boundary; and the node with smallest node ID will determine a unique boundary branch. Based on these boundary branches, we can find segment lines and decompose the network into convex subsections.

As there can be several concave nodes that are on the same boundary branch and within a small distance (e.g., k hops), resulting in a heavily fragmented network, we choose the concave node with the largest concavity and disregard the others. Notice that after this concave/convex recognition, every node s in $\partial N_k(p)$ obtains its distances to q_1 and q_2. We denote these as $h_k(s, q_1), h_k(s, q_2)$, and let $H_k(s) = \max\{h_k(s, q_1), h_k(s, q_2)\}$. We will see that $H_k(s)$ is very important for convex decomposition in Section 3.3.

3.3 Approximate Convex Decomposition (ACD)

In this subsection, we present a distributed algorithm for ACD. The key is to find a line from each concave node which reduces the concavity down to below the given threshold $1 + \delta_1$.

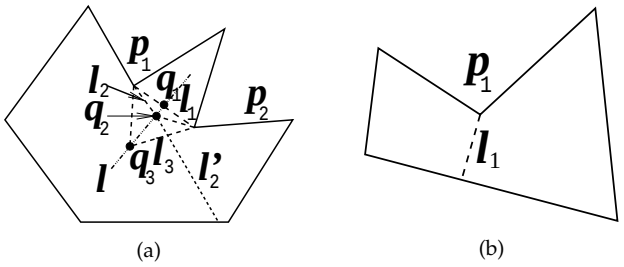


Fig. 5: Segment line. (a) Two nearby concave nodes; (b) One concave node.

3.3.1 Principle

Note that our goal is to decompose the network into approximate convex components with less number of components for achieving less accumulated localization error. To this end, similar with [4], we try to connect two nearby concave nodes if possible. Please see Fig. 5. In Fig. 5(a), p_1 and p_2 are nearby concave nodes. Drawing an angle bisector from each concave node will derive three components. If p_1 and p_2 can be connected, the number of convex components is only two. The connection between p_1 and p_2 can be done in two ways: 1) Connecting p_1 and p_2 by a straight line l_1. However, the concavities of p_1 and p_2 can not be guaranteed to be reduced down to below $1 + \delta_1$; 2) Connecting p_1 and p_2 by a curved line (e.g., l_2, l_3) which can reduces their concavities down to below $1+\delta_1$, with the possibility that new concave node might be generated on the curved line. Clearly, the curved line(e.g., l_2) with the smallest length is the best line since the possibility new concave node generating on this line is the smallest. In Fig. 5(a), the concavity of q_2 is smaller than q_3, so q_2 is more unlikely to be identified as a concave node. Note that here q_1, q_2 and q_3 are all on the perpendicular bisector of the straight line $\overline{p_1 p_2}$. On the downside, q_2 has a concavity larger than the threshold and thus identifies itself a concave node. For this case, an extended line is needed for reducing the concavity of q_2, e.g., l_2'. When there is a concave node that has no nearby concave nodes, the concave node simply find a line, which can reduce its concavity down to below $1+\delta_1$, to the nearest boundary, as shown in Fig. 5(b). As such, to conduct approximate convex decomposition, an important step is to find lines from concave nodes such that the number of convex components is as small as possible; we call such a line (e.g., l_2 in Fig. 5(a), and l_1 in Fig. 5(b)) from a concave node as a *segment line*, which reduces the concavity of a concave node down to below $1 + \delta_1$.

3.3.2 Implementation

As discussed, the key of ACD is to find a segment line. To that end, we propose a distributed algorithm of ACD as follows. First, each concave node p initiates a flooding within the network to figure out whether there is a nearby concave node. When an intermediate node s receives the flooding message from a concave node, say p, there are two cases: 1) if s has not received a message from any concave node, s will join the tree and broadcast the message; or, 2) if s has already received a message before, s will discard the newly arrived message. Eventually, a tree rooted at concave node p will be constructed and we call this tree *concave tree*, denoted by $CT(p)$. Note that in Sec.3.2, each node s who is k hops from a concave node computes $H_k(s)$. If node s has a concavity less than $1 + \delta_1$, we call s a *candidate segment node*, which means by connecting the concave node and s, the concavity of the concave node can be reduced down to below $1 + \delta_1$.

For each concave tree $CT(p)$, there are two cases:

1) There is no adjacent concave tree rooted on different boundary branches with p, which corresponds to the case of p_1 in Fig. 5 (b). For this case, we find a *best* boundary node $q \in CT(p)$ which satisfies that the shortest path from p to q will cross at least one candidate segment

node, and the concavity of q is the largest among all boundary nodes on $CT(p)$. This way, the shortest path from p to q (i.e., the segment line) can reduce the concavity of p down to below $1+\delta_1$.

2) There is one adjacent concave tree, say $CT(p')$, rooted on different boundary branches with p, corresponding to the case of p_1(or p_2) in Fig. 5 (a), which implies that these two concave nodes might be connected. As such, we detect *cut*, denoted by $\mathcal{C}(p,p')$, which are defined as the nodes where two trees $CT(p)$ and $CT(p')$ meet (see Fig. 4 (d)), corresponding to the perpendicular line l in Fig. 5 (a); and the nodes that lie on the cut are called *cut nodes*. Further, we define a cut-pair (v,v') as two neighboring cut nodes $v,v' \in \mathcal{C}(p,p')$ on different cut trees, where p and p' are the roots of v and v', respectively. If a cut-pair $(v,v') \in \mathcal{C}(p,p')$ satisfies that the two shortest paths from v to p and v' to p' both pass one candidate segment node, these two paths together with the path from v to v' form a line (referred to as *candidate segment line*) which can reduce the concavities of p and p', and decompose the network into two subsections S_1, S_2. Clearly, the nodes on a candidate segment line are possibly boundary nodes of S_1, S_2. However, there can be more than one *candidate segment line*, and the nodes (specifically, the cut-pair (v,v')) on such line may be concave for one subsection, say, S_1. If we treat a cut-pair (v,v') as a *dummy* node v'', then we only consider the concavity at v'' in S_1.

Lemma 1. *For two cut-pairs $(v,v'),(s,s') \in \mathcal{C}(p,p')$, if $d(v,p)+d(v',p') < d(s,p)+d(s',p')$, then we have $c_k(p'') < c_k(s'')$ where s'' is the dummy node by* merging *s and s'.*

As such, we can obtain a segment line which crosses a cut-pair (v,v') and satisfies $d(v,p) + d(v',p') = \min_{\forall(s,s')\in\mathcal{C}(p,p')}\{d(s,p)+d(s',p')\}$. If $c_k(p'') > 1+\delta_1$, we extend the shortest path from v to p (or v' to p') until the path *meets* a boundary node.

Corollary 2. *If the segment line between two concave nodes is the shortest path between them, then all nodes on the segment line are not concave.*

Theorem 3. *All the subsections generated in above way are approximately convex.*

3.4 Local Relative Map Establishment

So far we have decomposed the network into several convex subsections. We assume here that, there exists a localization *coordinator* (or simply coordinator which can be an arbitrary node) within each subsection, and the coordinator is in charge of the process of local relative map establishment. Let the number of sensors in Sec_i be n_i. Within one subsection, each node sends its neighbor list to the coordinator (this process has a message complexity of $O(n_i \log_2 n_i)$), which is subject to solving the all-pairs shortest paths problem in undirected graph with integer weights [26]. Next take the distance matrix as an input, the coordinator applies an improved MDS algorithm to establish a local relative map.

Given an $n_i \times n_i$ pairwise distance matrix D_i of Sec_i, MDS first constructs the inner product matrix $B_i = -\frac{1}{2}HD_iH$, where $H = I - \frac{1}{n_i}e^Te$, I is an n_i orders unit matrix and e is an n_i-dimensional vector of all ones. MDS then conducts spectral decomposition on matrix $B_i = Q\Lambda Q$, where Λ is the eigenvalues diagonal matrix and Q is the eigenvectors matrix of B_i. The complexity of spectral decomposition on matrix B_i is $O(n_i^3)$.

Note that in MDS-based localization scheme, only the first m ($m = 2$ for 2-dimensional networks and $m = 3$ for 3-dimensional networks) largest eigenvalues are used. Thus we can use the power method of a matrix only m times to obtain these m eigenvalues and eigenvectors, instead of all eigenvalues and eigenvectors. The power method (also known as the power iteration) of a matrix B is designed for extracting the dominant eigenvalue (i.e., the first eigenvalue with the largest magnitude) and the corresponding eigenvector. Repeating power method m times can derive the first m largest eigenvalues and eigenvectors. In this paper, for each subsection Sec_i, it will execute Algorithm 2 to localize those nodes in Sec_i.

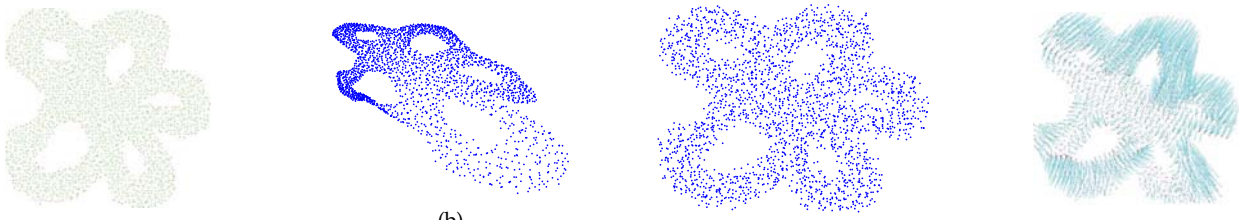
Algorithm 2 Local Map Establishment

1: **for all** convex subsection Sec_i **do**
2: Coordinator p of Sec_i computes pairwise distance matrix D_i in Sec_i and the inner product matrix of D_i, denoted by B_i .
3: Initialize:$\lambda_0 = 0, q_0 = e$
4: **for** $k = 1$ to m **do**
5: p uses power method on $B_i - \sum_0^{k-1}\lambda_l q_l q_l^T$ to extract the largest eigenvalue λ_k and eigenvector q_k.
6: **end for**
7: The locations of node s_j in Sec_i are given as: $X_{ij1} = \sqrt{\lambda_1}q_{1j}, \ldots, X_{ijm} = \sqrt{\lambda_m}q_{mj}$
8: **end for**

3.5 Global Map Establishment

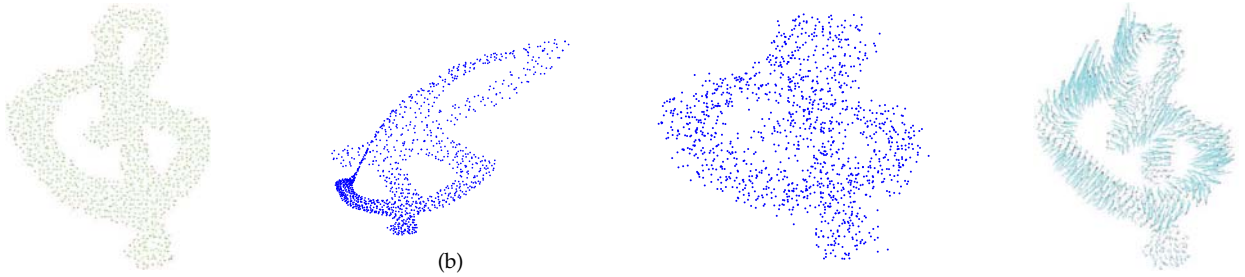
After assigning virtual coordinates within each convex subsection, we now combine them to form a global map. For every two adjacent subsections, there are some nodes on their common segment line, which are assigned two virtual coordinates accordingly. We find a linear transformation for these common nodes using their two virtual coordinates. Based on the linear transformation,we combine these two subsections into a bigger one. Note that the fact of non-overlapping partitions determines a unique way of putting adjacent partitions together. This way we can recover the global layout of the network.

In this subsection, we introduce a time round scheme to minimize the time cost for global map establishment. To clarify our statement, we use ACG (Adjacent Constraint Graph) of a network [10]. Here in ACG, each vertex i represents a subsection Sec_i; and two vertexes i,j are neighborhood if Sec_i and Sec_j are adjacent. Let d_i denote the degree of vertex i which indicates how many subsections adjacent to subsection Sec_i. Obviously, d_i equals the number of segment lines in Sec_i. In addition, each vertex i is assigned a weight w_i. w_i equals the



(a) (b) (c) (d)

Fig. 6: Flower, 2422 nodes, average degree 12.31. (a) Original map; (b) MDS-MAP(P); (c) REP; (d) ACDL.



(a) (b) (c) (d)

Fig. 7: Music, 1301 nodes, average degree 11.67. (a) Original map; (b) MDS-MAP(P); (c) REP; (d) ACDL.

node number in Sec_i and will be used in our merging process. For two neighboring vertex i and j, if $d_i > d_j$ or $d_i = d_j, w_i > w_j$, vertex i merges vertex j which means the coordinates of nodes in Sec_j will be transformed in terms of the coordinate system in Sec_i, and these two vertexes is treated as one larger vertex with degree $d_i = d_i + d_j - 1$ and weight $w_i = w_i + w_j - n_{ij}$. Here n_{ij} is the number of nodes on their common segment line.

Our algorithm starts with finding the linear transformation between two neighboring vertexes i and j, and then conducts the merging process round by round. In each round, two neighboring vertexes will merge together; and the number of vertexes will be reduced by fifty percent. The merging process will end when all vertexes are merged as one vertex; and the global map establishment is completed (see Fig. 4(g)). The algorithm 3 shows the process of global map establishment.

Algorithm 3 Global Map Establishment

```
while The number of subsections is larger than one do
    For every two adjacent subsections Sec_j and Sec_k,
    if d_j < d_k or (d_j = d_k and w_j < w_k) then
        Sec_k = Sec_k ∪ Sec_j.
        d_k = d_k + d_j − 1, w_k = w_k + w_j − n_kj.
    end if
end while
```

4 Performance Evaluation

4.1 Simulation Setup

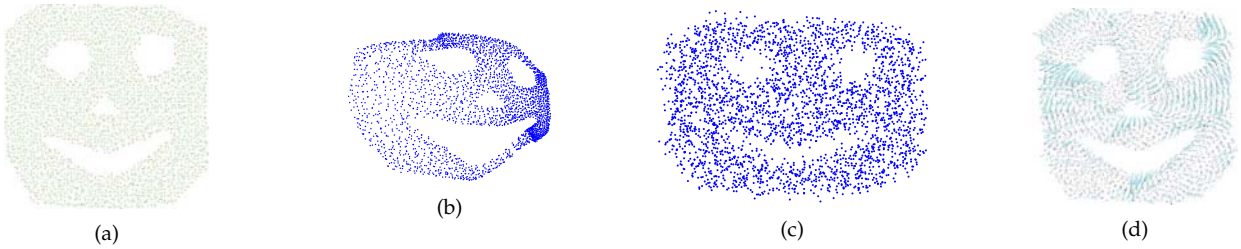
We evaluate our algorithm on several network topologies, namely, Flower (Fig. 6), Music (Fig. 7), Smile (Fig. 8) and Snake (Fig. 9). In these networks, nodes are uniformly distributed and have the same communication range. Two nodes are connected if and only if the Euclidean distance between them is no greater than a given communication radio range R.

We compare our algorithm with MDS-MAP(P) [23] and REP [14]. To evaluate the performance, two metrics are used in this paper: *localization error (LE)* and *average localization error (ALE)*. *LE* of node p is defined as the Euclidean distance between the estimated location of p and its location. Besides, we refer to *ALE* as the ratio of the mean localization error of each node to the communication range R. To have absolute locations, we randomly deploy three beacon nodes equipped with GPS. The default parameters are $k = 4, \delta_1 = 0.3, \delta_2 = 0.4$. More results can be found in the supplementary file.

4.2 Performance under Different Scenarios

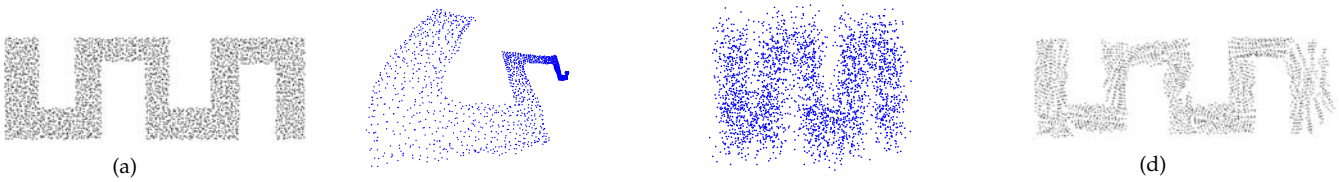
We present our results in two different forms. First, we show in Fig. 6, 7, 8, and 9 the localization error of each individual node for six different network shapes. Second, we summarize comprehensively, the statistical localization error information of the four networks. We show in Table 2 five kinds of localization errors, the *ALE*, 5-percentile, 50-percentile, 95-percentile and the maximum of the localization errors.

Fig. 6 shows the localization results using MDS-MAP(P) (see Fig. 6(b)), REP (see Fig. 6(c)), and ACDL (see Fig. 6(d)) on Flower-shape network. We can see that in general REP and ACDL are all with reasonable localization accuracies, where MDS-MAP(P) is much worse. Looking into Table 2, we found that the ALE of MDS-MAP(P) is 4.14, REP is 0.95 and ACDL is 0.69. This is not surprising as in MDS-MAP(P), localization errors will be accumulated during the process of merging local maps, especially when the starting local map is not in a



(a) (b) (c) (d)

Fig. 8: Smile, 2924 nodes, average degree 12.58. (a) Original map; (b) MDS-MAP(P); (c) REP; (d) ACDL.



(a) (b) (c) (d)

Fig. 9: Snake, 2759 nodes, average degree 8.17. (a) Original map; (b) MDS-MAP(P); (c) REP; (d) ACDL.

Model	Scheme	ALE	5-prtl Err.	50-prtl Err.	95-prtl Err.	Max Err.
Flower	MDS-MAP(P)	4.14	1.19	3.91	8.26	10.55
	REP	0.95	0.31	0.97	1.75	2.82
	ACDL	0.62	0.10	0.69	1.36	1.56
Music	MDS-MAP(P)	7.98	0.34	3.94	29.62	36.88
	REP	1.89	0.26	1.82	4.47	6.72
	ACDL	0.94	0.08	0.76	3.45	4.21
Smile	MDS-MAP(P)	3.95	0.67	3.76	8.01	10.95
	REP	1.06	0.46	1.02	1.33	1.63
	ACDL	0.76	0.39	0.73	1.16	1.3
Snake	MDS-MAP(P)	5.02	0.72	3.38	17.72	23.19
	REP	1.31	0.54	1.34	2.66	4.29
	ACDL	0.55	0.09	0.53	0.69	1.71

TABLE 2: Comparison of localization errors.

good position. ACDL is better than REP in ALE, 5% Err, 50% Err, 95% Err, and Max Err.

We next examine the performance of MDS-MAP(P), REP and ACDL on Music-shape network (see Fig. 7). As different from the Flower-shape network, there is a concave hole in the topology. We see similar trend for all three schemes as in the Flower-shape network except that MDS-MAP(P) is extremely worse. Further, looking into Table 2, we observe that the ALE and Max Err of MDS-MAP(P) and REP are more than two times larger than ACDL. The reason is that MDS-MAP(P) localizes the network in an iterative fashion. After each round of local map merging, newly localized map will serve as a reference coordinate system to merge the neighboring local map. As such, the localization errors after each round will be accumulated. Obviously, the node farther from the starting local map tends to have larger error. REP estimates the distance between nodes by constructing virtual hole, and it has dependency on the right position of beacon nodes. ACDL is an MDS-based algorithm; and ACDL does not suffer from the accumulated errors and improper chosen beacon nodes. Therefore the ALE and Max Err of ACDL will be smaller than MDS-MAP(P) and REP.

We next examine the performance of MDS-MAP(P), REP and ACDL on Smile-shape network (see Fig. 8). Note that the accuracy of MDS-MAP(P) is extremely poor. The localization errors of MDS-MAP(P) are four times larger than ACDL, and twice than REP. When looking into the logs of the simulation, we found that the starting local map is not in a good position. Intuitively, the choice of the starting local map is crucial for MDS-MAP(P). When the starting local map locates in the "middle" of the network, the localization will be relatively small, otherwise, errors will be accumulated, and the localization accuracy is low. We can see that ACDL and REP recover the network layout with small localization errors while ACDL provides higher accuracy than REP.

We finally study the performance of MDS-MAP(P), REP and ACDL on Snake-shape network (see Fig. 9). As different from the Flower-shape network, there is a long and narrow neck in the topology. We see that MDS-MAP(P) can not faithfully recover the network layout while the results by REP and ACDL are both desirable. Looking into Table 2, we observe, furthermore, that the ALE and Max Err of MDS-MAP(P) are more than five times larger than ACDL, while the ALE and Max Err of REP are more than twice lager than ACDL. The reason is that MDS-MAP(P) localizes the network in an iterative fashion, and the localization errors after each round will be accumulated while the performance of REP relies on the properly chosen beacon nodes. ACDL is an MDS-based algorithm; and ACDL does not suffer from the accumulated errors. Therefore the ALE and Max Err of ACDL will be smaller than REP and MDS-MAP(P).

Overall, MDS-MAP(P) suffers from accumulated errors, and the performance of REP depends on the choice of beacon nodes and properly constructed virtual holes. As opposed to previous two methods, ACDL decomposes the network into convex subsections and uses MDS to localize nodes in each subsection. As a result, ACDL

can faithfully recovers the network layouts.

5 Conclusion

We have proposed a novel connectivity based algorithm, Approximate Convex Decomposition based Localization (ACDL), for localization of wireless sensor networks in irregular shape networks. We overcame a series of difficulties in concave/convex node identification, network decomposition, MDS computation and global map reconstruction. All these were achieved in a way that is discrete, distributed, and low message and computation overhead. The extensive simulations show the efficiency of ACDL.

We believe that ACDL has room for improvement. Besides, ACDL may incorporate with other localization algorithms to further improve localization accuracy. In future, we plan to explore the possibility of using localization results to facilitate data processing [8] and collection [9], [29], coverage [21], [28], load-balanced routing [13], diagnosis [19], and skeleton extraction [16], [17] in sensor networks.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China and Microsoft Research Asia under Grant 60933012; by the National Natural Science Foundation of China under Grant 61073147, Grant 61173120, Grant 61202460 and Grant 61271226; by the National Natural Science Foundation of Hubei Province under Grant 2011CDB044; by the Scientific Research Foundation for the Returned Overseas Chinese Scholars (State Education Ministry); and by the Program for New Century Excellent Talents in University under Grant NCET-10-408 (State Education Ministry). Dan Wang's work was supported by Hong Kong PolyU/A-PJ19, A-PK95, A-PL23, and RGC/GRF PolyU 5305/08E. An earlier version of this work appeared as [18]. The corresponding author of this paper is Hongbo Jiang.

References

[1] P. K. Agarwal, E. Flato, and D. Halperin. Polygon decomposition for efficient construction of minkowski sums. *Computational geometry: theory and applications*, 21(1-2):39–61, 2002.

[2] I. Borg and P. Groenen. *Modern Multidimensional Scaling, Theory and Applications*. New York, Springer-Verlag, 2005.

[3] N. Bulusu, J. Heidemann, and D. Estrin. Gps-less low cost outdoor localization for very small devices. *IEEE Personal Communication Magazine, Special Issue on Smart Spaces and Environments*, 7(5):28–34, 2000.

[4] B. Chazelle and D. P. Dobkin. Optimal convex decompositions. *Computational Geometry*, pages 63–133, 1985.

[5] D. Dong, Y. Liu, and X. Liao. Fine-grained boundary recognition in wireless ad hoc and sensor networks by topological methods. In *Proceedings of ACM MOBIHOC*, 2009.

[6] S. P. Fekete, A. Kroller, D. Pfisterer, S. Fischer, and C. Buschmann. Neighborhood-based topology recognition in sensor networks. In *Proceedings of the Int. Workshop on Algorithmic Aspects of Wireless Sensor Networks*, 2004.

[7] M. Ghosh, N. M. Amato, Y. Lu, and J.-M. Lien. Fast approximate convex decomposition using relative concavity. In *Proceedings of ACM Solid and Physical Modeling Symposium (SPM)*, 2012.

[8] H. Jiang, S. Jin, and C. Wang. Parameter-based data aggregation for statistical information extraction in wireless sensor networks. *IEEE Transactions on Vehicular Technology*, 59(8):3992–4001, 2010.

[9] H. Jiang, S. Jin, and C. Wang. Prediction or not? an energy-efficient framework for clustering-based data collection in wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 22(6):1064–1071, 2011.

[10] H. Jiang, W. Liu, D. Wang, C. Tian, X. Bai, X. Liu, Y. Wu, and W. Liu. Connectivity-based skeleton extraction in wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 21(5):710–721, 2010.

[11] M. Jin, S. Xia, H. Wu, and X. Gu. Scalable and fully distributed localization with mere connectivity. In *Proceedings of IEEE INFOCOM*, 2011.

[12] S. Lederer, Y. Wang, and J. Gao. Connectivity-based localization of large scale sensor networks with complex shape. *ACM Transactions on Sensor Networks*, 5(4):1–32, 2009.

[13] F. Li, J. Gao, and Y. Wang. Distributed load balancing mechanism for detouring schemes of geographic routing in wireless sensor networks. *International Journal of Parallel, Emergent and Distributed Systems*, 28(2):184–197, 2013.

[14] M. Li and Y. Liu. Rendered path: Range-free localization in anisotropic sensor networks with holes. *IEEE/ACM Transactions on Networking*, 18(1):320–332, 2010.

[15] J.-M. Lien and N. M. Amato. Approximate convex decomposition of polygons. *Computational geometry: theory and applications*, 35(1-2):100–123, 2006.

[16] W. Liu, H. Jiang, X. Bai, G. Tan, C. Wang, W. Liu, and K. Cai. Skeleton extraction from incomplete boundaries in sensor networks based on distance transform. In *Proc. of IEEE ICDCS*, 2012.

[17] W. Liu, H. Jiang, C. Wang, C. Liu, Y. Yang, W. Liu, and B. Li. Connectivity-based and boundary-free skeleton extraction in sensor networks. In *Proc. of ICDCS*, 2012.

[18] W. Liu, D. Wang, H. Jiang, W. Liu, and C. Wang. Approximate convex decomposition based localization in wireless sensor networks. In *Proceedings of IEEE INFOCOM*, 2012.

[19] Y. Liu, K. Liu, and M. Li. Passive diagnosis for wireless sensor networks. *IEEE/ACM Transactions on Networking*, 18(4):1132–1144, 2010.

[20] Y. Liu, Z. Yang, X. Wang, and L. Jian. Location, localization, and localizability. *Jounral of Computer Science and Technology*, 25(2):274–297, 2010.

[21] M. J. Nene, R. S. Deodhar, and L. M. Patnaik. Urea: an algorithm for maximisation of coverage in stochastic deployment of wireless sensor networks. *International Journal of Parallel, Emergent and Distributed Systems*, 27(3):249–274, 2012.

[22] O. Saukh, R. Sauter, M. Gauger, P. J. Marron, and K. Rothernel. On boundary recognition without location information in wireless sensor networks. In *Proceedings of ACM/IEEE IPSN*, 2008.

[23] Y. Shang and W. Ruml. Improved mds-based localization. In *Proceedings of IEEE INFOCOM*, 2004.

[24] Y. Shang, W. Ruml, M. P. J. Fromherz, and Y. Zhang. Localization from connectivity in sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 15(11):961–974, 2004.

[25] Y. Shang, W. Ruml, Y. Zhang, and M. P. J. Fromherz. Localization from mere connectivity information. In *Proceedings of ACM MOBIHOC*, 2003.

[26] A. Shoshan and U. Zwick. All pairs shortest paths in undirected graphs with integer weights. In *Proceedings of IEEE FOCS*, 1999.

[27] G. Tan, H. Jiang, S. Zhang, and A. Kermarrec. Connectivity-based and anchor-free localization in large-scale 2d/3d sensor networks. In *Proceedings of ACM MOBIHOC*, 2010.

[28] D. Tao, S. Tang, H. Zhang, X. Mao, X. Li, and H. Ma. Strong barrier coverage detection and mending algorithm for directional sensor networks. *Ad Hoc & Sensor Wireless Networks*, 18(1-2):17–33, 2013.

[29] C. Wang and H. Ma. Data collection in wireless sensor networks by utilizing multiple mobile nodes. *Ad Hoc & Sensor Wireless Networks*, 18(1-2):65–85, 2013.

[30] Y. Wang, J. Gao, and J. S. B. Mitchell. Boundary recognition in sensor networks by topological methods. In *Proceedings of ACM MOBICOM*, 2006.

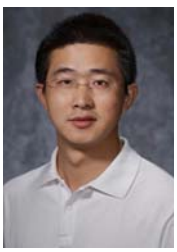
[31] Y. Wang, S. Lederer, and J. Gao. Connectivity-based sensor network localization with incremental delaunay refinement method. In *Proceedings of IEEE INFOCOM*, 2009.

Wenping Liu received the B.S., M.S. and Ph.D. degrees from Huazhong University of Science and Technology, China. Now he is a faculty member of Hubei University of Economics, and a post-doc fellow at Huazhong University of Science and Technology. His research interests include statistical modeling and wireless sensor networks. He is a member of IEEE.

Dan Wang (S'05-M'07) received the B. Sc degree from Peking University, Beijing, China, in 2000, the M. Sc degree from Case Western Reserve University, Cleveland, Ohio, USA, in 2004, and the Ph. D. degree from Simon Fraser University, Burnaby, B.C., Canada, in 2007; all in computer science. He is currently an assistant professor at the Department of Computing, The Hong Kong Polytechnic University. His research interests include wireless sensor networks, Internet routing, and peer-to-peer networks. He is a member of IEEE.

Hongbo Jiang received the B.S. and M.S. degrees from Huazhong University of Science and Technology, China. He received his Ph.D. from Case Western Reserve University in 2008. After that he joined the faculty of Huazhong University of Science and Technology as an associate professor. His research concerns computer networking, especially algorithms and architectures for high-performance networks and wireless networks. He is a member of IEEE.

Wenyu Liu received the B.S. degree in Computer Science from Tsinghua University, Beijing, China, in 1986, and the M.S. and Ph.D. degrees, both in Electronics and Information Engineering, from Huazhong University of Science and Technology (HUST), Wuhan, China, in 1991 and 2001, respectively. He is now a professor and associate dean of the Department of Electronics and Information Engineering, HUST. His current research areas include computer networking, multimedia information processing. He is a member of the IEEE.



Chonggang Wang received his PhD degree in computer science from Beijing University of Posts and Telecommunications. He has conducted research with NEC Laboratories America, AT&T Labs Research, and University of Arkansas, and Hong Kong University of Science and Technology. His research interests include future Internet, machine-to-machine (M2M) communications, and cognitive and wireless networks. He is a senior member of the IEEE.