Connectivity-Based Segmentation in Large-Scale 2-D/3-D Sensor Networks: Algorithm and Applications

Hongbo Jiang, *Member, IEEE*, Tianlong Yu, *Student Member, IEEE*, Chen Tian, *Member, IEEE*, Guang Tan , *Member, IEEE, ACM*, and Chonggang Wang, *Senior Member, IEEE*

***Abstract*—Efficient sensor network design requires a full understanding of the geometric environment in which sensor nodes are deployed. In practice, a large-scale sensor network often has a complex and irregular topology, possibly containing obstacles/holes. Convex network partitioning, also known as *convex segmentation*, is a technique to divide a network into convex regions in which traditional algorithms designed for a simple network geometry can be applied. Existing segmentation algorithms heavily depend on concave node detection, or sink extraction from the median axis/skeleton, resulting in sensitivity of performance to network boundary noise. Furthermore, since they rely on the network's 2-D geometric properties, they do not work for 3-D cases. This paper presents a novel segmentation approach based on *Morse function*, bringing together the notions of convex components and the *Reeb graph* of a network. The segmentation is realized by a distributed and scalable algorithm, named *CONSEL*, for CONnectivity-based SEgmentation in Large-scale 2-D/3-D sensor networks. In CONSEL, several boundary nodes first flood the network to construct the Reeb graph. The ordinary nodes then compute mutex pairs locally, generating a coarse segmentation. Next, neighboring regions that are not mutex pairs are merged together. Finally, by ignoring mutex pairs that lead to small concavity, we provide an approximate convex decomposition. CONSEL has a number of advantages over previous solutions: 1) it works for both 2-D and 3-D sensor networks; 2) it uses merely network connectivity information; 3) it guarantees a bound for the generated regions' deviation from convexity. We further propose to integrate network segmentation with existing applications that are oriented to simple network geometry. Extensive simulations show the efficacy of CONSEL in segmenting networks and in improving the performance of two applications: geographic routing and connectivity-based localization.**

***Index Terms*—Localization, routing, segmentation, wireless sensor networks.**

Manuscript received December 11, 2011; revised December 06, 2012 and October 29, 2013; accepted November 01, 2013; approved by IEEE/ACM TRANSACTIONS ON NETWORKING D. Manjunath. Date of publication November 20, 2013; date of current version February 12, 2015. This work was supported in part by the National Natural Science Foundation of China under Grants 61073147, 61173120, 61271226, and 61272410; the National Natural Science Foundation of China and Microsoft Research Asia under Grant 60933012; the Fundamental Research Funds for the Central Universities under Grant 2012QN078; the National Natural Science Foundation of Hubei Province under Grant 2011CDB044; the CHUTIAN Scholar Project of Hubei Province; the Youth Chenguang Project of Wuhan City under Grant 201050231080; the Scientific Research Foundation for the Returned Overseas Chinese Scholars (State Education Ministry); and the Program for New Century Excellent Talents in University under Grant NCET-10-408 (State Education Ministry). The work of G. Tan was supported by the National Natural Science Foundation of China under Grant 61103243, the Youth Innovation Promotion Association, the Chinese Academy of Sciences, the Ministry of Science and Technology under 863 Key Project No. 2011AA010500, and the Shenzhen Overseas High-level Talents Innovation and Entrepreneurship Funds under KQC201109050097A. An early version of this work appeared in the Proceedings of the IEEE International Conference on Computer Communications (INFOCOM), Orlando, FL, USA, March 25–30, 2012

H. Jiang, T. Yu, and C. Tian are with the Department of Electronics and Information Engineering, Wuhan National Laboratory for Optoelectronics, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: hongbojiang2004@gmail.com).

G. Tan is with SIAT, Chinese Academy of Sciences, Shenzhen 518055, China.

C. Wang is with InterDigital Communications, King of Prussia, PA 19406 USA.

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

Digital Object Identifier 10.1109/TNET.2013.2289912

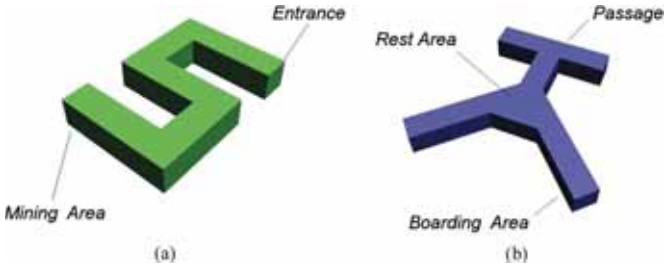

Fig. 1. The topologies of an S-shaped coal mine tunnel and the Chicago airport terminal building.

I. Introduction

RECENT years have witnessed a wide usage of 3-D wireless sensor networks (WSNs) in emerging applications where nodes are typically deployed in 3-D settings, such as safety monitoring of coal mine tunnels [1] [Fig. 1(a)] and fire detection in the corridors of buildings [15] [Fig. 1(b)]. Efficient sensor network design requires a full understanding of the geometric environment in which sensor nodes are deployed. In practice, the global topology of a large-scale sensor network is rarely in a simple or regular shape, such as a square or a disk, as assumed in many previous studies (e.g., [6] and [25]). For example, geographic routing [14] is a routing scheme for sensor networks, where a node makes routing decisions greedily based on a local coordinates. Specifically, a node routes a message to its neighbor closest to the destination. Despite its success in regular sensor fields, this protocol fails (or performs poorly) in irregular and concave areas [16], [17], [27], [30], [32]. Also, irregular network shapes may lead to inaccurate localization results, as many existing localization algorithms assume straight-line shortest paths between nodes. This does not

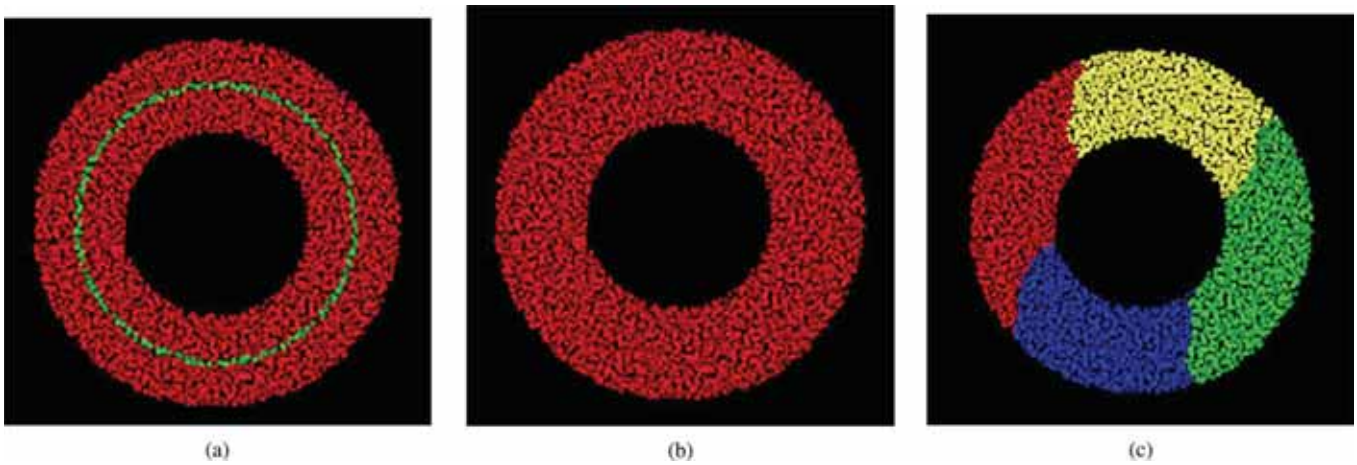

Fig. 2. Network of 3844 nodes. The average node degree is 16.4. (a) Skeleton result by [32] where nodes in skeleton are shown in dark green. (b) Segmentation result by [32] or [27] where no obvious geometric features on the boundary exist. Existing algorithms generate a single region. (c) Segmentation result by CONSEL where four (approximately) convex regions are generated.

hold when the network topology is highly irregular, resulting in distorted path length estimation [18], [28] and thus inaccurate localization results.

Numerous methods [16]–[18] have been proposed to improve the traditional algorithms' performance by adapting them to irregular sensor field. These efforts are mostly application-specific, increasing the design complexity significantly. Alternatively, to tame the challenges brought by irregular shapes, there has been an increasing interest on a convex network partition, also known as *segmentation*/*convex partition* [27], [32] (in this paper, we consider shape segmentation instead of data segmentation [23] or signal field segmentation [33]), which is to divide a network into convex regions or subnetworks, so that traditional algorithms designed for a simple geometric region can be applied with good performance. By doing so, without heavily modifying particular algorithms, traditional algorithms are able to perform well in each convex region.

A. Related Work

A pioneer piece of work on sensor network segmentation is [32]. It first extracts the skeleton and constructs the distance field of a network. Next, based on flow complex [3], a node on a flow is assigned a flow direction and identifies itself as a sink if there is no flow direction. According to the flow direction to the sink, the ordinary nodes are grouped into regions. Inspired by polygonal partition, the solution in [27] partitions the network via concave node identification on the boundary. The main idea of their algorithm is to perform bisector-induced convex partitioning. These segmentation algorithms, unfortunately, depend on the existence of nodes on distinguished concave boundaries, and thus they may fail for networks where no such nodes (or sinks on the skeleton) exist, as shown in Fig. 2. The reason is that their algorithms do not consider the global topology of the network. Besides, this dependency makes their performance quite sensitive to boundary noise, e.g., incorrect estimation of boundaries. Lastly, since they rely on the network's 2-D geometric properties, they cannot work for 3-D sensor networks. In a 3-D space, we are aware of one partial solution proposed in [31] based on bottleneck identification. A parameter named *injectivity radius* is calculated by each boundary node. The purpose of this parameter is to measure the narrowness of the nearby boundary area, so as to identify the undesired bottlenecks in a 3-D sensor network. These bottlenecks are then used to partition the network boundary, and the nonboundary nodes are grouped finally. Thus, this algorithm does not work for networks without bottlenecks (see the networks in Fig. 10). In addition, it does not work for 2-D networks.

2-D/3-D segmentation algorithms have been studied in the computer vision and graphics areas [19], which target continuous shapes and use centralized solutions. Despite the resemblance to those works, the problem we strive to address is more challenging because of the nature of a distributed sensor network. First, the random deployment of sensor nodes makes the geometric objects (e.g., holes) not necessarily follow the properties of their counterparts in a continuous space. The nature of the random deployment also makes it impractical to manually identify convex/concave regions during deployment or extract a graph of the network. Second, in practice, sensor nodes may have no knowledge of location information. As a result, internode distance is often estimated by shortest-path hop count, whose measurement accuracy is adversely affected by network concavities. Third, since the sensor network is discrete, it is difficult or impossible to decompose a sensor network into strictly convex regions due to the boundary irregularity. It is more meaningful to decompose the network into approximately convex regions [27], [32]. However, how to provide a bound of convexity deviation is not straightforward.

B. Our Contributions

This paper proposes a novel segmentation scheme using Morse function [5], linking the notions of convex regions and the Reeb graph of a network. We propose CONSEL for CONNnectivity-based SEgmentation in Large-scale 2-D/3-D sensor networks. In CONSEL, several boundary nodes first perform flooding to construct a *Reeb graph*. The ordinary nodes then compute *mutex pairs* locally, generating a coarse segmentation layout. Next, the neighbor regions that are not mutex pairs are merged. Finally, by ignoring mutex pairs that lead to small concavities, we provide a configurable bound for the

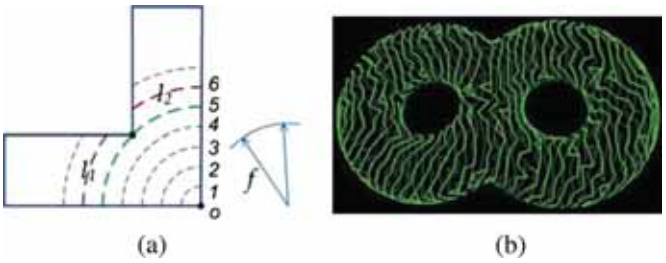

Fig. 3. (a) Morse function and level sets. l_1 and l_2 are two L-components of $f^{-1}(6)$. (b) Level sets of a 3-D network.

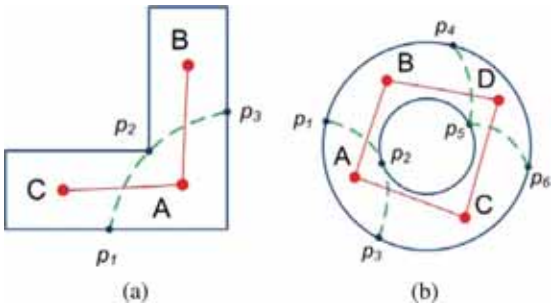

Fig. 4. Reeb graphs (red nodes and lines) of two network topologies. Green dashed arcs represent cuts.

subnetworks' deviation from convexity. CONSEL is desirable compared to previous solutions: 1) it works for both 2-D and 3-D sensor networks; 2) it uses merely network connectivity information; 3) it provides a bound on a region's deviation from convexity. Additionally, CONSEL works for a variety of radio models, including the quasi-UDG model, log-normal model, and the probabilistic connectivity model. CONSEL is fully distributed and is scalable as its time and message complexities are both linear with network size. Extensive simulations show that CONSEL works well in the presence of holes and shape variation, always yielding appropriate segmentation results. Furthermore, we propose to integrate network segmentation with existing applications such as geographic routing and connectivity-based localization and evaluate their performance in Section VI.

The remainder of the paper proceeds as follows. Section II gives the background of Morse functions and Reeb graph. Section III is devoted to a description of CONSEL algorithm. Section IV discusses the complexity of CONSEL. We evaluate our CONSEL in Section V. Section VI proposes two interesting protocols that integrate shape segmentation with existing applications. Finally, Section VII concludes the paper.

II. Preliminaries

A. Morse Functions and Reeb Graph

1) Morse Functions: For a manifold M, a Morse function [5] is a mapping $f : M \to R$, where R is the set of real numbers. The mapping can be considered as a projection from high-dimensional to one-dimensional manifold. In a discrete network, the Morse function f is constructed as follows. First, we select an *origin node* o, then every other node p obtains its hop-count distance, denoted by $f(p)$, to node o. This process establishes a mapping $f : M \to Z$, where Z is the set of nonnegative integers. In Fig. 3, the nodes on the same arc have an equal value of the Morse function. It is noted that the Morse function is more general that the level of nodes on the shortest path tree. Moreover, it is useful to proceed on our description of Reeb graph where we have to make use of the inverse function of Morse function. Let $f^{-1}(r_0)$ be the set of nodes, called the r_0th *level set*, whose Morse function values are r_0. For example, all the nodes on the green arc in Fig. 3(a) share the same Morse function value 5. The zeroth level set, $f^{-1}(0)$, contains the origin node only.

2) Reeb Graph: The Reeb graph [5] is constructed based on the Morse function of a network. Given a Morse function f, each of its level sets consists of a number of connected components, called *level components* (L-components for short); see Fig. 3(a) for an example. In an L-component $\mathcal{C}_r^i$ at level $r \geq 0$, each node has a number of neighbors in the $(r+1)$th level. $\mathcal{C}_r^i$ is referred to nodes at level r, and the origin node is the node i. That is, all nodes in $\mathcal{C}_r^i$ are r hops away from the origin node. The set of such above-mentioned neighbors is called $\mathcal{C}_r^i$'s *children set*, denoted by $h(\mathcal{C}_r^i)$. As a result, the L-components expand as the level number grows. There are three basic types of events during the transition from one L-component to its children set (for simplicity of exposition, we omit composite events).

1) *Extend event* $h(\mathcal{C}_r^i)$ is a maximal L-component in the $(r+1)$th level.
2) *Split event* $h(\mathcal{C}_r^i)$ contains two or more L-components in the $(r+1)$th level.
3) *Merge event* $h(\mathcal{C}_r^i)$ is a subset of an L-component in the $(r+1)$th level, which means that multiple L-components in the rth level share the same connected children set.

A series of extend events may take place in succession [say the L-components from level 1 to level 5 in Fig. 3(a)], and the involved L-components will form a connected, multilevel component of the network, called a *Reeb component*. A vertex in the final Reeb graph, or a *Reeb vertex*, represents a maximal Reeb component, and an edge in the Reeb graph, or a *Reeb edge*, reflects the split or merge events. Initially, the Reeb graph is empty. The zeroth-level set $f^{-1}(0)$, which contains only the origin node and thus a single L-component, triggers the generation of the first Reeb vertex q. This vertex continues to represent the descendant L-components until a split or merge event happens. For a split event [say the L-components from level 5 to level 6 in Fig. 3(a)], it means that an L-component $\mathcal{C}_r^i$ has expanded into $j > 1$ L-components, in which case j new Reeb vertices will be generated as children of q in the Reeb graph. When a merge event happens, it means that multiple L-components share the same connected children set, so a new Reeb vertex will be generated to represent this children set, serving as the common child of those L-components' corresponding Reeb vertices.

Fig. 4(a) shows an example of the Reeb graph (red nodes and edges) corresponding to the network in Fig. 3(a). In Fig. 3(a), from $f^{-1}(5)$ to $f^{-1}(6)$, there exists a split event. As such, two new Reeb vertices B and C are generated under the original vertex A, which represents the union of $f^{-1}(0)$ through $f^{-1}(5)$.

Fig. 4(b) shows another example of Reeb graph corresponding to a network deployed in a 3-D ring. In this figure, two Reeb vertices B and C are merged into a new Reeb vertex D.

Generally, a split event happens when a *concave node* emerges [5] during the expanding process of L-components. Here, a concave node [see p_2 in Fig. 4(a)] is a node where the inward angle (i.e., the angle spanning across the sensing area) is greater than π [27]. A merge event happens when the network topology forms a loop. By capturing the splitting and looping behavior, the Reeb graph creates a useful abstraction of the original network topology.

In Fig. 4, the lines $l(p_1, p_2)$ and $l(p_2, p_3)$ are called *cuts*, defined as the new L-components arising in the split event, or the new children set generated by the merge event. The cuts are meant to separate the regions. In Fig. 4(a), for example, the two cuts $l(p_1, p_2)$ and $l(p_2, p_3)$ partition the network into three regions.

B. Approximate Convexity

For most applications, a strictly convex segmentation is unnecessary [32], [27]. Therefore, we aim at partitioning the network into *approximately convex* regions. To that end, we define an ϵ-*straight line* as follows.

Definition 1 (ϵ-Straight Path): A path between two nodes p and q is called an ϵ-*straight path* if and only if any node on the path is at most $\epsilon \cdot R_0$ away from the Euclidean straight line between p and q, where R_0 is the radio range of sensor nodes.

It is noted that the term "the radio range" is used in Definition 1 since here two nodes are assumed to be able to communicate within the radio range. The concept of ϵ-convex regions thus follows.

Definition 2 (ϵ-Convex Region): A network region is called an ϵ-*convex region* if within that region, for any two nodes p and q, there always exists an ϵ-straight path connecting them, and all the nodes on that path belong to the region.

When the network is segmented into several regions, each region is approximately convex, or ϵ-convex. For example, in Fig. 4, let $\mathbf{dist}(p_1, p_3) = d$ and $\mathbf{dist}(o, p_1) = r$. The distance function $\mathbf{dist}(\cdot, \cdot)$ is a mapping from two nodes to the real number set. To satisfy the condition of ϵ-convexity, we should have $r - \sqrt{r^2 - d^2/4} < \epsilon$. That is, $d^2 + 4\epsilon^2 < 8r\epsilon$. When $d \ll r$, this condition always holds.

C. Mutex Pairs

In a continuous space, if an area is convex, then it contains the line segment between every pair of its inner points. This fact motivates us to consider the relation between inner points of a network in the light of Morse functions. Here, we use the term "mutex" to indicate that two nodes/parts can not be in one convex region.

Definition 3 (Morse Path): Given a network and a Morse function, a *Morse path* between two nodes p_1 and p_2 is a path on which all nodes have the same Morse function value.

Definition 4 (Mutex Pair): Given a network and a Morse function, two nodes p_1 and p_2 are called a *mutex pair*, denoted by $p_1 \sim p_2$, if there exists no Morse path between them.

In Fig. 4, it is easy to find a node in area B and a node in area C that form a mutex pair. Intuitively, after segmentation, they should not be in the same region. For two disjoint regions A and B, if there is a mutex pair formed by a node from A and a node from B, then A and B are called a mutex pair as well, denoted by $A \sim B$. Note that a node can be regarded as a singleton set, so a mutex pair of regions in fact includes a mutex pair of nodes.

Given the Reeb graph constructed from f, the Reeb components corresponding to any two vertices are a mutex pair. A mutex pair formed by two adjacent Reeb components is separated by the cuts between the two Reeb components.

Next, we extend the definitions of Morse path and mutex pair by introducing approximation.

Definition 5 (δ-Morse Path): Given a network and a Morse function f, two nodes p_1 and p_2 with equal Morse function values have a δ-*Morse path* between them if the path contains two intermediate nodes t_1 and t_2 such that: 1) $f(t_1) = f(t_2)$; 2) there exists a Morse path between t_1 and t_2; and 3) $\mathbf{dist}(p_1, t_1) < \delta$ and $\mathbf{dist}(t_2, p_2) < \delta$.

Definition 6 (δ-Mutex Pair): Given a network and a Morse function f, two nodes p_1 and p_2 with the same Morse function values are called a δ-*mutex pair*, denoted by $p_1 \sim_\delta p_2$, if there exists no δ-Morse path between them.

Definition 6 allows a small part of nonconvex subregions to be merged finally. Overall, ϵ and δ can be considered as two parameters allowing the network operator to control the level of convexity of generated network segments. As such, our algorithm is to generate a set of $(\epsilon + \delta)$-convex regions. That is, CONSEL ensures a bound for the regions' deviation from convexity.

III. CONSEL Algorithm

In this section, we present the implementation details of the segmentation algorithm. It is noted that even in a 2-D domain, computing a minimum number of convex components for a polygon with holes is NP-hard [20]. We therefore do not pursue an optimal convex segmentation. Instead, our goal is to provide a simple and practical algorithm that can be performed in a distributed way.

A. Establishing the Morse Function

In the first step, we randomly choose I nodes roughly on the outer boundary of the network. We use a technique similar to the one in [17]: An arbitrary node p floods the network to find the farthest node o_1 to p. Thereafter, o_1 floods the network to find the farthest o_2 to itself. Then, o_3 is the node that has the maximum sum of the square roots of the hop counts from the nodes o_1 and o_2. This process continues until I nodes $(o_i, 1 \le i \le I)$ are obtained on the outer boundary. Note that this selection phase works well for both 2-D/3-D networks.

Next, each of the I nodes on the boundary floods the network. The goal of the flooding operations is twofold. First, after a flooded message from o_i reaches a node p, p records the parent from which it receives the message, as well as the hop count to the node o_i. By doing so, the node p has the knowledge of the Morse function value $f_i(p), 1 \le i \le I$ corresponding to $o_i, 1 \le i \le I$. Second, the flooding allows us to construct the Reeb graph in a distributed way, as described in Section III-B.

B. Constructing the Reeb Graph

The key to constructing a Reeb graph is to identify the Reeb components and select a *Reeb landmark node* (or Reeb landmark) for each of them, representing a Reeb vertex. To this end, each node will be assigned a *Reeb component ID*, which is equal to the ID of its landmark node. (When no confusion occurs, we use the ID to refer to a node.) In addition, each Reeb landmark also maintains a set of Reeb edges, connecting to the landmarks of neighboring Reeb components.

In the construction of the Reeb graph, we will often use two primitive routines.

L-Component Landmark Election: The nodes in each L-component elect a leader, called the L-component's *landmark*. To do so, each node in the L-component declares itself to be a landmark with a predefined probability and broadcast to its neighbors. If a node does not select itself to be a landmark, and does not receive any broadcast message within a predefined period of time, it declares itself to be a landmark with the same probability again. This process continues until each node receives a broadcast message. When multiple nodes in the same component compete to be a landmark, the node with a larger ID will prevail. In the end, each L-component will have elected a landmark, whose ID is recorded by each node in this L-component.

Neighboring L-Components Discovery: Recall that given an L-component $\mathcal{C}_r^i$, its children in the $(r+1)$th level constitute a *children set* of $\mathcal{C}_r^i$. Likewise, the parent nodes of an L-component's $\mathcal{C}_r^i$ in the $(r-1)$th level constitute $\mathcal{C}_r^i$'s *parent set*. An L-component landmark q floods its component, asking all nodes to contact their neighbors in a specified neighboring level. Those neighbors return their own L-component IDs to the requesting nodes, which in turn report the collected L-component IDs to q.

In the beginning, an L-component in the first level set constitutes a (temporary) Reeb component. Suppose at the current stage, a Reeb component has elected a Reeb landmark, and one of its L-components at the maximum level r, $\mathcal{C}_r^i$, has a children set $h(\mathcal{C}_r^i)$. Through the neighboring L-components discovery routine, $\mathcal{C}_r^i$ and $h(\mathcal{C}_r^i)$ (more precisely their landmark nodes) may find themselves in one of the following situations.

1) (Extend event) $\mathcal{C}_r^i$ has only one neighboring L-component in $h(\mathcal{C}_r^i)$, and $h(\mathcal{C}_r^i)$ has only one neighboring L-component in its parent set.
2) (Split event) $\mathcal{C}_r^i$ has more than one neighboring L-components in $h(\mathcal{C}_r^i)$, which all have only one parent set.
3) (Merge event) $\mathcal{C}_r^i$ has only one neighboring L-component in $h(\mathcal{C}_r^i)$, and $h(\mathcal{C}_r^i)$ has more than one neighboring L-components in its parent set.

Fig. 5(a) shows an example of the first case, where there are two L-components, whose nodes have Reeb component ID q_1 and q_2. The two L-components each have a single neighboring L-component, led by landmarks q_3 and q_4, respectively, in the next level. Thus, the nodes in the L-components led by q_3 and q_4 will set their Reeb component ID to that of their parents (i.e., either q_1 or q_2).

Fig. 5(b) shows an example of the second case, where the L-component led by landmark q_2 has two neighboring L-components in the next level, led by landmark nodes q_4 and q_5, respectively. Thus, two new Reeb components, represented by q_4 and q_5, respectively, are generated. Also, the edges (q_2, q_4) and (q_2, q_5) are generated and recorded by the three end nodes q_2, q_4, and q_5.

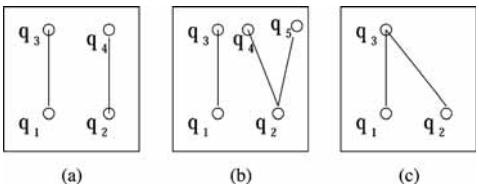

Fig. 5. Local Reeb graph. (a) Extend event. (b) Split event. (c) Merge event.

Fig. 5(c) shows an example of the third case, where the L-components led by landmarks q_1 and q_2 have the same neighboring L-component in the next level. Thus, a new Reeb component, represented by q_3 is generated. Also, the edges (q_1, q_3) and (q_2, q_3) are generated and recorded by the three end nodes.

To guarantee a bound on an L-component's deviation from convexity, the condition of $d^2 + 4\epsilon^2 < 8r\epsilon$ should be satisfied, as we discussed in Section II-B. Otherwise, an L-component should be partitioned. Specifically, we identify two nodes, say p' and p'', that have the longest path length in it, and the nodes closer to p' than to p'' form a region, and the remaining nodes form another region. The condition will be examined again for each region until the convexity condition is satisfied.

Finally, all Reeb landmarks send their local topologies back to the origin o_i, following o_i's flooding tree. After this phase, the origin o_i has the full picture of the Reeb graph. Fig. 6(a) and (b) shows the result of the Reeb graph after the flooding for a Morse function. In Fig. 6(a), the nodes marked with the same color are in the same region, corresponding to a vertex in Reeb graph.

It is noted that we have randomly choosen I nodes as origin nodes. Therefore, besides the Reeb graph shown in Fig. 6(b), there are $I-1$ additional Reeb graphs obtained in a similar way.

C. Identifying Mutex Pairs and Coarse Segmentation

Although a Reeb graph is defined as a undirected graph, each of its edges can actually be assigned a direction following the generation order of Reeb vertices. Therefore, we can say that a Reeb vertex is the ancestor of another Reeb vertex if and only if there is a directed path between them in the Reeb graph.

Mutex pairs are identified with the following observation: If two Reeb vertices are not in an ancestor–descendant relationship, then their corresponding Reeb components (and Reeb landmarks) form a mutex pair. This is because if two Reeb components are not in such a relationship, then any path between them must pass a common ancestor Reeb component, which makes the path not ϵ-straight. For example, in Fig. 2, q_2 and q_3 are a mutex pair, but q_1 and q_4 are not. At the same time, the cuts between the adjacent Reeb vertices can separate these mutex pairs.

As introduced earlier, all nodes in the same Reeb component should have recorded their Reeb landmark's ID. Since there are I origin nodes, there are I Reeb graphs constructed. As a result, each node records a set of I Reeb landmark IDs. In Fig. 6(a), $I = 8$, and there are four Reeb vertices in each Reeb graph, corresponding to 32 Reeb landmarks.

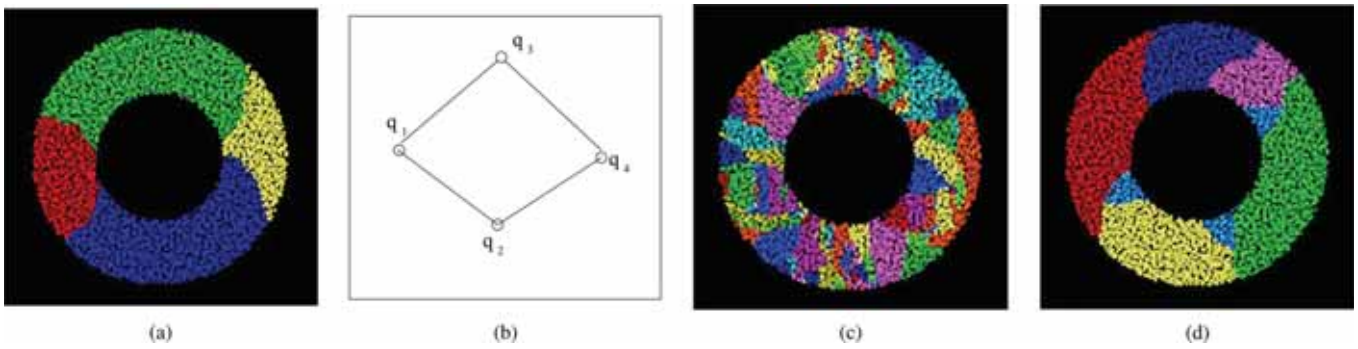

Fig. 6. Steps of the CONSEL algorithm. (a), (b) Reeb graph construction. (c) Coarse segmentation layout. (d) Merging result. The final refined result is in Fig. 2(c).

Here, each node p maintains a set of eight Reeb landmark IDs $\mathrm{LID}_1(p), \mathrm{LID}_2(p), \ldots, \mathrm{LID}_8(p)$.

The basic idea of coarse segmentation is that any pair of nodes with a different set of Reeb landmark IDs should be in different regions. Taking the network in Fig. 6 as an example, assume the Reeb landmarks are $q_1, q_2, \ldots q_{32}$. Each of them performs flooding with its own ID to represent a region, denoted by $\mathrm{SID}(q_k) = \mathrm{ID}(q_k)$, as well as its set of eight Reeb landmark IDs. Hereinafter, q_k is called a *representative* of a region if $\mathrm{ID}(q_k) = \mathrm{SID}(q_k)$. When a node p receives a flooded message, it compares the received set of eight Reeb landmark IDs to its own set of Reeb landmark IDs. When they are different, or when p has already had a region ID $\mathrm{SID}(p) < \mathrm{ID}(q_k)$, p discards the message. Otherwise, p updates its region ID as $\mathrm{SID}(p) \leftarrow \mathrm{ID}(q_k)$ and then forwards the message to all its neighbors.

Not all the 32 Reeb landmarks will become regional representatives because some of them may have the same set of landmark IDs. In addition, some regions may find no representative since they may not contain a Reeb landmark. To address this problem, each of the remaining nodes, say node p', that still have not been assigned a region ID will asynchronously declare itself to be a representative, and then broadcast to all its neighbors that have the same set of landmark IDs. When a node q receives this message, it again compares $\mathrm{ID}(p')$ and $\mathrm{SID}(q)$. Only when $\mathrm{ID}(p') < \mathrm{SID}(q)$, the node p updates its region ID as $\mathrm{SID}(q) \leftarrow \mathrm{ID}(p')$ and forwards this message containing $\mathrm{SID}(p)$.

In the end, every node has a region ID, and each region has a representative, which maintains the information of the I Reeb landmark IDs as well as the I Reeb graphs. All regions are the result of the *coarse segmentation* of the sensor network. Fig. 6(c) shows a result of coarse segmentation of the network where 65 subregions are generated.

D. Merging Regions

The cuts induced from the Morse function will often generate too many regions. For instance, as many as 65 regions are generated in Fig. 6(c). For the purpose of convex segmentation, we only need to select some cuts to satisfy all mutex pairs.

The process of cuts selection is called *merging regions*. Each representative node in the region maintains a landmark ID list. The representative node, say q_1, sends a message, containing this landmark list $\mathrm{LID}_1(q_1), \ldots, \mathrm{LID}_I(q_1)$, to another representative node, say q_2, of one neighboring region. If q_2 does not find any mutex pair in the two landmark list, it replies a positive message to q_1 to allow the merging of $\mathrm{SID}(q_1)$ and $\mathrm{SID}(q_2)$. The merging process is simple: q_2 broadcasts a message to ask all nodes in its own region to update their region IDs, that is, $\mathrm{SID}() \leftarrow \mathrm{ID}(q_1)$. In addition, q_1 will update its landmark list to include q_2's landmark list. This process will continue until no more neighboring regions can be merged. The result of this step is shown in Fig. 6(d), where eight regions are finally generated.

E. Refinement

Generally, the set of a network's mutex pairs can be used to generate strictly convex regions. As we target approximate convex segmentation, we can generate significantly fewer partitions by ignoring mutex pairs that lead to small concavities.

To do so, we exploit the definition of δ-mutex pair, introduced in Section II-C. Based on the coarse segmentation result [Fig. 6(d)], two regions that are not a δ-mutex pair can be merged.

This process works as follows. Each representative node q first finds all mutex pairs with its neighboring region. Without loss of generality, we assume $\mathrm{LID}_1(q)$ and $\mathrm{LID}_1(q')$ are a mutex pair. If there exist two cuts, say l_1 and l_2, in $L(f_1)$ such that $l_1 \cap l_2 \neq \emptyset$, $l_1 \cap \mathrm{SID}(q) \neq \emptyset$, and $l_2 \cap \mathrm{SID}(q') \neq \emptyset$, the regions $\mathrm{SID}(q)$ and $\mathrm{SID}(q')$ determine whether a δ-mutex pair exists. Thus, the two cuts l_1 and l_2 perform flooding within the regions $\mathrm{SID}(q)$ and $\mathrm{SID}(q')$, so that the remaining nodes have the information of its hop distance to the cuts.

Each node with a landmark ID $\mathrm{LID}_1(q)$ ($\mathrm{LID}_1(q')$) then checks its hop distance to the cut l_1 (l_2). A negative message is then replied to the representative node $\mathrm{SID}(q)$ when its hop distance is greater than δ. If both q and q' do not receive any negative reply, no δ-mutex pair exists. In this case, the node q notifies the node q' that these two regions can be merged. That is, all nodes in the region of $\mathrm{SID}(q')$ will update their region ID to $\mathrm{SID}(q)$. This process will continue until no more neighboring regions can be merged. Fig. 2(c) shows the refined result. Note that each of the obtained subnetworks is an $(\epsilon + \delta)$-convex region.

IV. Discussion

A. Time Complexity and Message Complexity

Time and message complexity are important factors for an efficient segmentation algorithm. Let N be the total number of nodes in the network, and C the number of regions generated by the CONSEL algorithm. For simplicity, we assume that all

the nodes are roughly uniformly distributed over the sensing area. We consider two cases in 2-D (for 3-D, the proof is quite similar): If the sensor area is like a circle or square, the diameter is assumed to be $O(\sqrt{N})$, otherwise $O(N)$.

Theorem 1: The time complexity of CONSEL is $O(N)$, where N is the network size.

Proof: We only prove this theorem in a 2-D space for simplicity; for the 3-D case, the proof is very similar. First, the I origin nodes on the network boundary are found via global flooding. This step introduces a time complexity of $O(I \cdot \sqrt{N})$ (or $O(I \cdot N)$ for a network with a $O(N)$ diameter). Second, to construct the Reeb graph, the set of nodes on $f_i^{-1}(r_0)$ perform landmark selection. Assuming a $O(\sqrt{N})$ diameter, the number of nodes on $f_i^{-1}(r_0)$ is $O(N/\sqrt{N}) = O(\sqrt{N})$ (for a $O(N)$ diameter, the number is $O(1)$). In this case, this process on results in a time complexity of $O(\sqrt{N})$ (or $O(1)$ for a $O(N)$ diameter) since the nodes only forward the message containing the largest node ID. The total time complexity of constructing the Reeb graph is thus $O(\sqrt{N} \cdot \sqrt{N} + \sqrt{N}) = O(N)$ (for a network with a $O(N)$ diameter, the time complexity is $O(N) \cdot O(1) + O(N) = O(N)$), after all landmark nodes inform their local simplified topology graphs to the origin nodes.

The step of coarse segmentation can be done in a distributed way—the flooding is limited within regions. We consider the worst case where the number of nodes in each region is comparable to the number of sensor nodes in the whole network. That is, there are $O(N)$ nodes in each regions. Since this process is similar to the landmark selection, its time complexity is $O(\sqrt{N})$ [22]. Next, for the procedure of merging regions, since the neighboring representative nodes will communicate with each other, the time complexity is the diameter of regions, that is $O(\sqrt{N})$ (or $O(N)$ for a diameter $O(N)$).

Finally, in the step of refinement, the cuts perform flooding within the regions, which has a time complexity of $O(\sqrt{N})$ (or $O(N)$ for the worst case). Then, the reply messages traveling from nodes to the representative node lead to a time complexity of $O(\sqrt{N})$ (or $O(N)$ for the worst case). Overall, the time complexity of CONSEL is $O(N)$. ■

Theorem 2: CONSEL has a message complexity of $O(N)$.

Proof: For brevity, we focus on the 2-D space again. First, the global flooding operations are performed to find the I origin nodes, introducing a message complexity of $O(I \cdot N)$. Second, constructing the Reeb graph requires a process similar to the landmark selection. Recall that on $f_i^{-1}(r_0)$, there exist $\sqrt{N}$ nodes (or $O(N)$ for the worst case), each of which randomly declares itself to be a landmark with a given probability. For simplicity of analysis, this probability is set to $O(\sqrt{N}/\alpha)$ where α is a constant. As such, only $O(1)$ nodes perform flooding within $f_i^{-1}(r_0)$, which incurs a few scoped flooding operations. Accordingly, the cost of local flooding on $f_i^{-1}(r_0)$ is at most $\sqrt{N}$ (or $O(N)$ for the worst case). Since the diameter is $\sqrt{N}$ (i.e., $1 \leq r_0 \leq \sqrt{N}$) (or $O(N)$), the message cost is $O(N)$. It is noted that for a network with a diameter $O(N)$, there only exist $O(1)$ nodes on $f_i^{-1}(r_0)$. In this case, the message cost is still $O(N)$. Third, in the step of coarse segmentation, the flooding is limited within regions, not over the whole network, resulting in a message complexity of $O(N)$. Finally, during the refinement, the cuts perform flooding within the regions, which also introduces a message complexity of $O(N)$. Overall, the message complexity of CONSEL is $O(N)$. ■

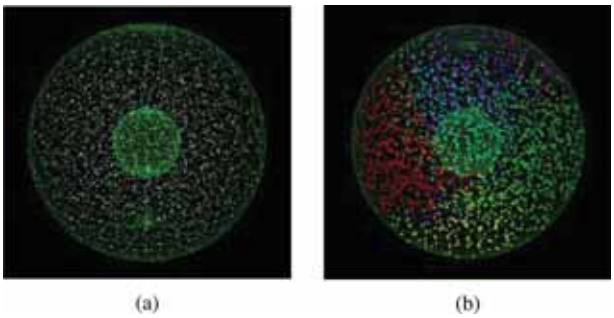

Fig. 7. Example 3-D network with an internal hole with 2094 nodes and average degree is 36.17. CONSEL generates 11 convex regions where the poor visualization is due to the display limitation of 3-D objects. (a) Original network. (b) Segmentation result.

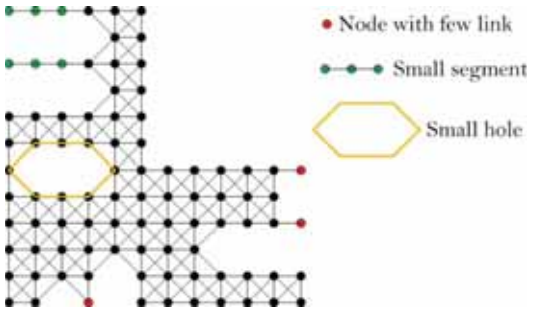

Fig. 8. Very sparse sensor network deployed in an L-shape area.

Theorem 2 implies that while our algorithm always uses flooding operations to collect necessary information in each step, the communication cost will not increase too much since we limit the flooding within local areas as much as possible.

B. Special Cases for Segmentation

While the segmentation algorithm requires no boundary information in most cases, in some 3-D cases, rough network boundaries may have to be known when internal holes exist. Fig. 7 shows an example where there exist two boundary surfaces (with an internal hole) in a 3-D network.

It can be seen that when the flooded message from the origin node reaches a new boundary surface, a new vertex should be generated in the Reeb graph. To achieve rough boundary identification, existing solutions such as the one in [13] can be used. As a result, each boundary surface is assigned a surface ID. To construct the Reeb graph, new vertices are inserted when a new boundary surface is met during the origins' flooding process. The rest of the steps are similar, and thus coarse segmentation can still be generated, followed by the merging and refinement steps. The result is shown in Fig. 7(b).

C. Node Density Requirement of CONSEL

CONSEL only requires a network to be connected, so in principle it works correctly for both dense and sparse networks. Generally, the denser the network nodes, the better the segmentation performance. For a very sparse network, however, the algorithm may produce many small and less meaningful segments. This is because in this case, the connectivity of the network nodes cannot well represent the geometric shape of the network. Fig. 8 shows an example of a sensor network deployed

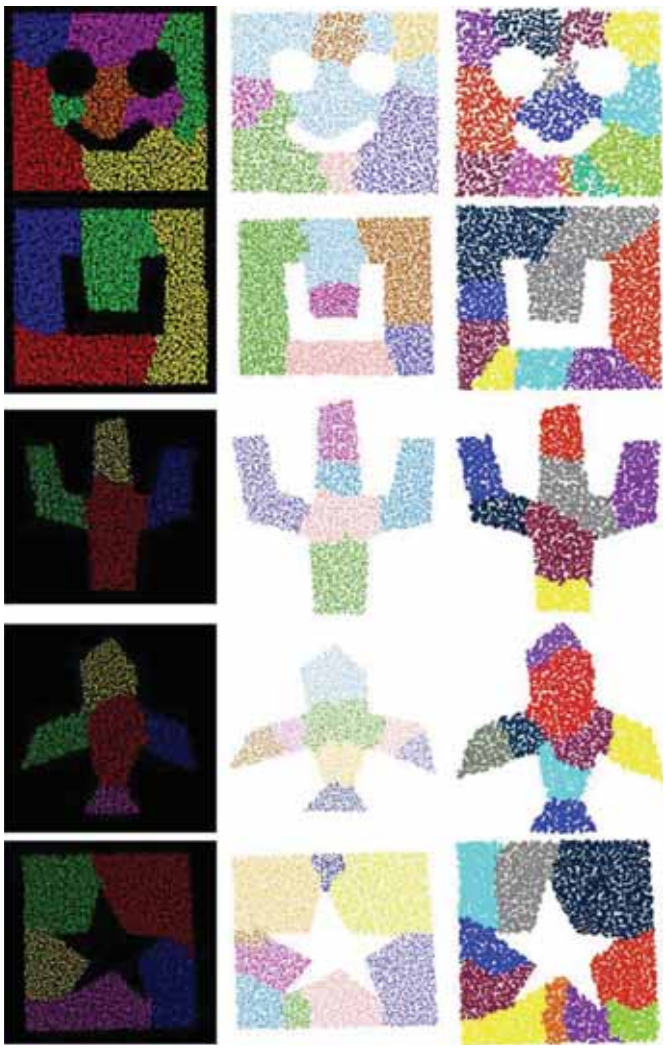

Fig. 9. Comparisons on 2-D networks. Columns (from left to right): results using CONSEL, results of [32], and results of the CONVEX algorithm [27]. Rows: 1) Smile shape, 3045 nodes, avg. deg. 10.09. 2) single-hole shape, 3700 nodes, avg. deg .13.1. 3) Cactus shape, 2172 nodes, avg. deg. 8.76. 4) Airplane shape, 1878, avg. deg. 8.65. 5) Star shape, 3893, avg. deg. 8.90.

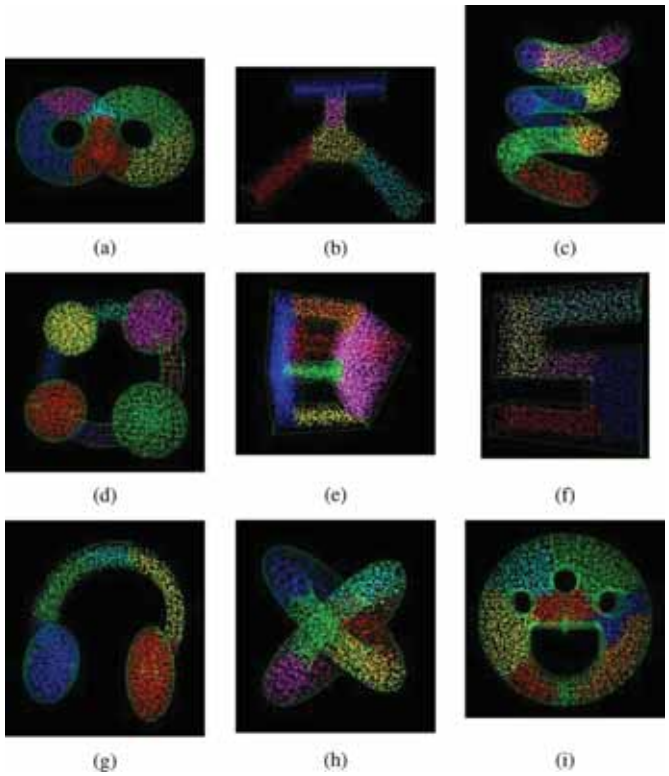

Fig. 10. 3-D performance of CONSEL. (a) 3-D 8 shape, 3486 nodes, avg. deg. 37.16. (b) Chicago airport terminal-2 shape, 2502 nodes, avg. deg. 31.09. (c) 3-D spiral shape, 3464 nodes, avg. deg. 31.07. (d) 3-D ring-ball shape, 3790 nodes, avg. deg. 33.85. (e) 3-D single-hole shape, 5681 nodes, avg. deg. 32.28. (f) 3-D S shape, 2244 nodes, avg. deg. 31.38. (g) 3-D headset shape, 3260, avg. deg. 31.5. (h) 3-D cross-ring shape, 2416, avg. deg. 25.4. (i) 3-D footprint shape, 3996, avg. deg. 33.8.

in an L-shaped area. Due to the sparse nodes and links, the network does not quite exhibit an L shape. The main difficulty for CONSEL arises from the fact that the network's connections may not determine a deterministic shape of the network. In other words, given very limited connection information, CONSEL cannot precisely infer the shape of the network, due to the lack of sufficient constraints on the nodes' positions. The reason behind this is explained by the theory of rigid graphs [4], which studies the conditions for a graph to have a unique embedding onto the plane. Because of this theoretical restriction, there exists no segmentation algorithm that works well for cases like this. Therefore, the consequence of oversegmentation must be taken into account before one chooses any connectivity-based algorithms, including CONSEL, for a very sparse network whose shape is hard to define geometrically.

V. Simulations

We have conducted a series of simulations experiments on various network topologies to compare CONSEL with the algorithms in [27] and [32]. We evaluate how the performance of CONSEL is affected by factors such as node density, radio model, node distribution, etc. The default settings are as follows. Sensors are deployed with a perturbed grid model where the grid has a width around 4.5 and the radio range is around 6.8. The networks have an average node degree of around 10 for 2-D, and 30 for 3-D. We set the parameter δ to be 5 hops and ϵ to be 2 hops.

A. Evaluation on 2-D Networks

We first compare CONSEL to two state-of-the-art algorithms in [27] and [32] designed for 2-D networks. Fig. 9 shows the results. Since these algorithms rely on the knowledge of network boundaries, it can be seen that they produce significantly more regions. Also, the results by [27] and [32] are quite sensitive to boundary noise (or boundary deformation), as we explained in Section I. In contrast, CONSEL provides a bound of convexity deviation of regions, resulting in appropriate segmentation results (smaller number of segmentation regions).

B. Evaluation on 3-D Networks

Fig. 10 shows the segmentation results of CONSEL for several 3-D networks. We do not include a comparison to the only previous algorithm with limited 3-D segmentation capability in [31] because its reliance on bottleneck identification prevents it from working for most networks in Fig. 10, except for Fig. 10(d). We can see that CONSEL works well for 3-D

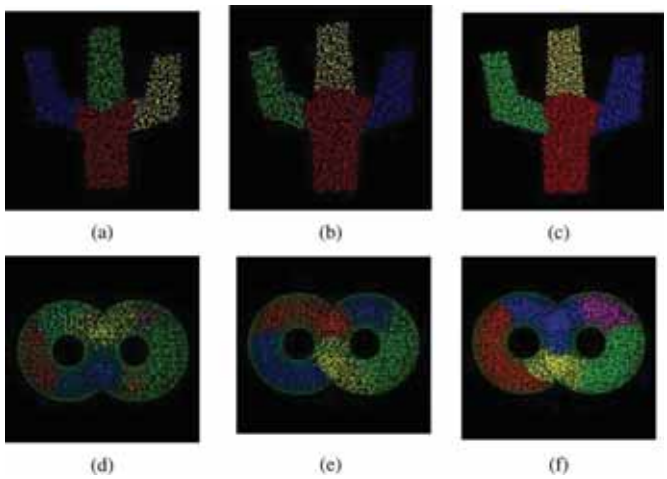

Fig. 11. Segmentation results with varying node densities. (a) 1663 nodes, avg. degg 6.86. (b) 1723 nodes, avg. deg. 7.23. (c) 3 059 nodes, avg. deg. 12.77. (d) 1774 nodes, avg. deg. 18.4. (e) 2612 nodes, avg. deg. 26.7. (f) 4707 nodes, avg. deg. 50.0.

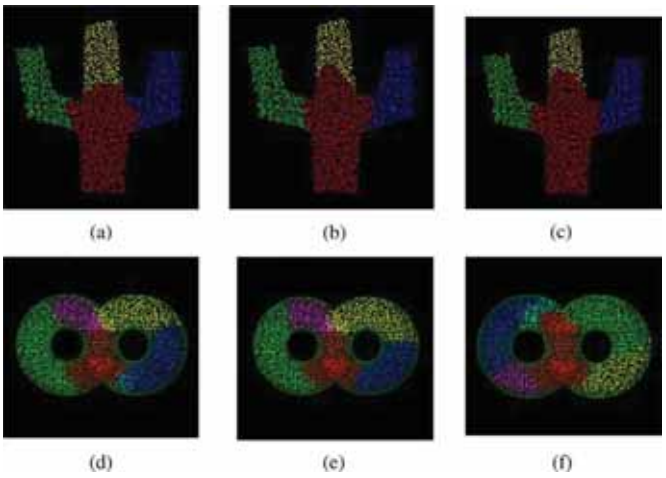

Fig. 12. Segmentation results under the Quasi-UDG/UBG radio model. (a) $\alpha = 0.2$. (b) $\alpha = 0.3$. (c) $\alpha = 0.4$. (d) $\alpha = 0.2$. (e) $\alpha = 0.3$. (f) $\alpha = 0.4$.

networks, always yielding proper segmentation results. For example, in the 8-shape network, six convex regions are generated, reflecting the impact of the two holes in the network topology.

C. Sensitivity to Node Density

Fig. 11 shows the results of networks with a variety of node densities. Theoretically, the higher the node density, the closer the network to a continuous domain. This means that hop count serves as a good approximation of Euclidian distance. First, the edge between two regions is smoother (closer to a straight line) when the node density is higher. Second, we observe in Fig. 11(d) an increased number of regions. The reason is that when the average degree is too low, smaller regions of few sensor nodes can be produced due to the variation of node connectivity.

D. Sensitivity to Radio Model

We run simulations under a different radio model: Quasi-UDG (Quasi-UBG for 3-D networks) radio model, with a parameter α. There exists a link between two nodes if the Euclidian distance is less than $(1-\alpha)R$ where R is the communication range. If the distance is between R and $(1+\alpha)R$, the link exists with a probability of $0 < p < 1$. No link exists when the distance is greater than $(1+\alpha)R$. We vary the α value while adjusting p to make the average node degree in the networks nearly the same.

We show the segmentation results under the Quasi-UDB/UBG model by varying the α values in Fig. 12. We can observe degraded performance of cuts (shared edges/surfaces between two regions) under this model. Nonetheless, the network is still partitioned into several approximately convex regions by CONSEL.

A second radio model that we examine is the log-normal model. For two nodes p and q, the probability of a link existing between them is based on a log-normal shadowing radio model [7]

$$\Pr(\hat{r}) = \frac{1}{2}\left[1 - \mathrm{erf}\left(\alpha \frac{\log \hat{r}}{\xi}\right)\right], \qquad \xi \triangleq \sigma/\eta$$

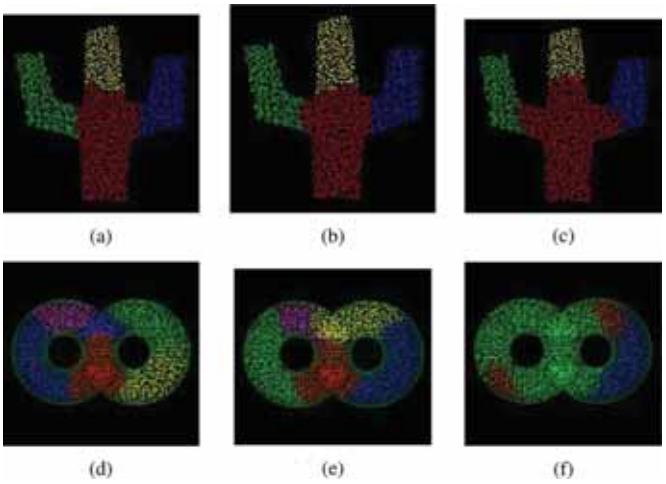

Fig. 13. Segmentation results under the log-normal radio model. (a) $\xi = 0$. (b) $\xi = 3$. (c) $\xi = 6$. (d) $\xi = 0$. (e) $\xi = 2$. (f) $\xi = 4$.

where $\hat{r}$ is the normalized distance between nodes p and q, $\alpha = 10/(\sqrt{2}\log 10)$ is a constant, σ is the standard deviation of shadowing, and η is the pathless exponent. Empirically, ξ may vary between 0 and 6 [7]. An important feature of the log-normal radio model is that the link between two nodes whose normalized distance is less than 1 may not exist, and the link between nodes whose normalized distance is larger than 1 exists with a nonzero probability.

Fig. 13 shows the segmentation results for various ξ values. We observe degraded performance with a larger ξ. The reason is similar: A node may find a neighbor far away from itself. More severely, when ξ is large, say $\xi = 4$ in Fig. 13(f), only four regions are generated finally. We conclude that CONSEL algorithm works best for a moderate value of ξ (say, $\xi < 3$) in the log-normal radio model.

The third radio model used in our simulation is the probabilistic connectivity model, where we start with the unit disk graph model and remove each edge with probability $(1-\beta)$. As can be seen from Fig. 14, CONSEL can generate appropriate segmentation results, reflecting its robustness to this radio model.

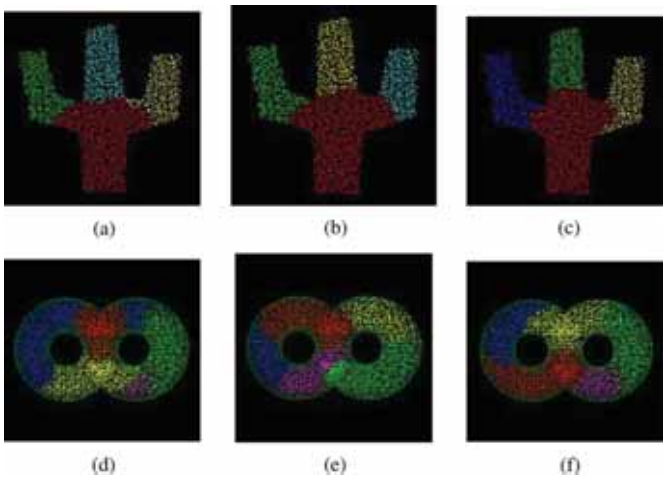

Fig. 14. Segmentation results under the probabilistic connectivity model. (a) $\beta = 0.6$. (b) $\beta = 0.7$. (c) $\beta = 0.8$. (d) $\beta = 0.6$. (e) $\beta = 0.7$. (f) $\beta = 0.8$.

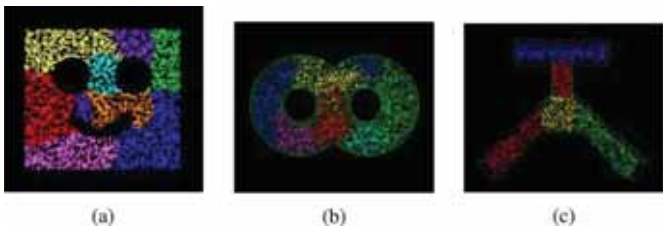

Fig. 15. (a) 3388 nodes, avg. deg. 11.5. (b) 2859 nodes, avg. deg. 29.6. (c) 2585 nodes, avg. deg. 31.0.

E. Sensitivity to Node Distribution

We next consider another sensor placement scheme: uniform random distribution. This distribution, compared to the perturbed grid model, generates more randomness in node deployment. We show the results in Fig. 15. With the increased randomness, the segmentation results show no significant difference compared to their counterparts in Figs. 9 and 10. Despite the higher variability of the node distribution, CONSEL partitions the network according to the Reeb graph and mutex pairs, which makes itself robust to the variation of node connectivity.

We next examine CONSEL under a nonuniform node distribution. Fig. 16 shows the results of CONSEL where 70% of all nodes are distributed in the right half of the sensor area. This nonuniform distribution leads to more variable node connectivity. Nonetheless, there is no significant difference compared to the results from the previous experiments. The reason is that the Morse function used in CONSEL utilizes no neighborhood density information.

F. Sensitivity to the Parameter Δ

Recall that ϵ and δ can be considered as two parameters allowing the algorithm to control the level of convexity of generated regions. Since they have similar influence on the convexity deviation, we study the algorithm's sensitivity to the parameter δ. Fig. 17 depicts the segmentation results with varying δ values. It is found that, with an increased δ value, more and more approximately convex regions are grouped together. Recall that each of the obtained regions is an $(\epsilon + \delta)$-convex region. This parameter also provides a flexible tool to balance the number of generated regions and their convexity.

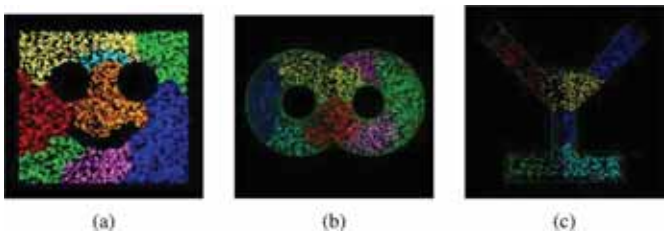

Fig. 16. (a) 4284 nodes, avg. deg. 15.85. (b) 3070 nodes, avg. deg. 17.7. (c) 1314 nodes, avg. deg. 16.2.

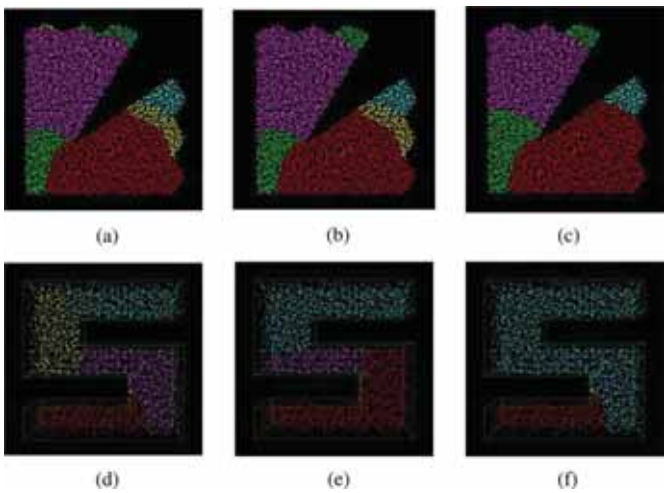

Fig. 17. Segmentation results with varying δ values. The first row shows a network with many small boundary deformations. It contains 2878 sensor nodes and its average degree is 10.52. (a) $\delta = 0$. (b) $\delta = 2$. (c) $\delta = 3$. (d) $\delta = 8$. (e) $\delta = 10$. (f) $\delta = 15$.

TABLE I
MESSAGE COST OF CONSEL FOR VARIOUS NETWORKS

Network	Network size	Step 1	Step 2	Overall
Fig. 11(a)	1,663	39,888	4,680	44,568
Fig. 11(b)	1,723	41,310	5,383	46,693
Fig. 9, Row 3	2,172	52,070	6,102	58,172
Fig. 11(c)	3,059	73,411	11,323	84,734
Fig. 11(d)	1,774	42,485	6,122	48,607
Fig. 11(e)	2,612	62,693	11,326	73,919
Fig. 10(a)	3,486	83,578	10,561	94,139
Fig. 11(f)	4,707	112,918	24,056	136,974
Fig. 2(c)	3,844	92,237	11,804	104,041
Fig. 7(b)	2,094	50,229	5,638	55,867
Fig. 9, Row 1	3,045	73,066	8,371	81,437
Fig. 9, Row 2	3,700	88,713	11,261	99,974
Fig. 10(b)	2,502	59,985	7,216	67,201
Fig. 10(c)	3,464	83,100	9,969	93,069

G. Message Cost

We next evaluate the message cost of CONSEL for different networks, measured by the total number of transmitted messages; see Table I. In this table, the procedures of establishing

the Morse functions and constructing the Reeb Graph are labeled "Step 1," whose message cost arises from the global flooding from the I origin nodes. Also, the remaining procedures of CONSEL are labeled "Step 2," whose message cost is caused by the operations of grouping sensor nodes. We make several observations. First, the cost of Step 1 dominates the cost. That is, the flooding operations initiated by the I origin nodes account for the majority of traffic by CONSEL. The reason is that this step not only involves I global network-wide flooding operations, but also requires many local flooding operations that have to be done for every set of $f^{-1}(r)$. These are needed to find dominating landmark nodes in order to figure out the local simplified network topology and to bound the level of concavity. Second, CONSEL is scalable in terms of message cost, confirming the theoretical result in Theorem 2. Take rows 1–4 in Table I as an example. With increased network size, the traffic cost is increased, roughly in proportion to the network size. This linear trend makes CONSEL potentially applicable to a wide range of scenarios in practice. Third, we cannot see significant difference of cost between 2-D and 3-D networks. We conclude that the message complexity of CONSEL is not correlated to the dimension, but rather depends on the network size.

VI. Applications

Network segmentation enables traditional algorithms oriented to simple network geometry to run efficiently for complex and irregular environments. In this section, we study how it benefits two important applications: geographic routing and connectivity-based localization.

A. Geographic Routing

Geographic routing is a routing paradigm in which a sensor node greedily forwards packets to its neighbor closest to the destination. It works well when the shape of a sensor field is approximately convex, where the sought route should approximately follow a straight line. This assumption, however, does not hold for geometrically complex environments. In the presence of holes or concave corners, the straight-line course of greedy forwarding can be seriously disrupted, leading to very poor performance [27], [32].

With network segmentation, we can circumvent the above problem by running geographic routing within each region. After the segmentation, each node p obtains a unique regional ID and knows the ID of its Reeb component landmark node, or its Reeb landmark. In addition, in each region, using existing coordinate assignment algorithms such as [17] and [24], we can assign a set of virtual coordinates to each node. At the same time, the Reeb component landmark nodes can help with interregion routing. To that end, each Reeb landmark floods the global network to establish a shortest path tree, allowing every other node in the network to reach it via shortest path. The segmentation-assisted geographic routing is simple. When a packet is requested to route from a source node s to a destination node t, s is first greedily routed to s's Reeb landmark using the virtual coordinates. Then, the packet is routed to t's Reeb landmark via shortest path. Finally, the packet is routed to t, again using greedy forwarding based on the virtual coordinates.

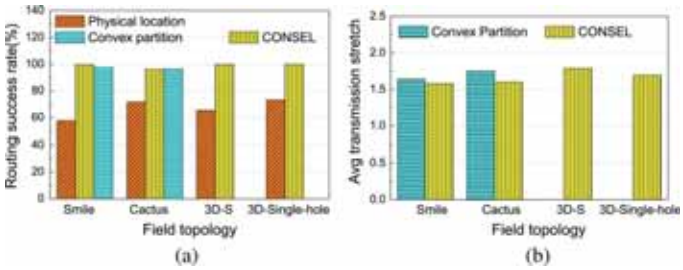

Fig. 18. (a) Routing successful rate and (b) transmission stretch.

We carry out simulations on various networks and the results of routing successful rate and transmission stretch are shown in Fig. 18. We also implement the algorithm in [27], and its segmentation results are labeled "Convex partition" (or CONVEX for short). First, in most cases, direct greedy routing without segmentation, labeled "Physical location," has a routing successful rate below 70%. With segmentation, the successful rate is significantly higher, coming close to 100%. Second, Fig. 18(b) shows the results of transmission stretch, defined as the ratio of the routing path length to the shortest-path hop count between the nodes. The CONSEL protocol outperforms CONVEX, resulting in a 7%–12% improvement. The reason is that for 2-D networks, CONSEL generates fewer regions compared to CONVEX, potentially leading to a shorter routing path. Third, for 3-D networks, clearly the results with CONSEL segmentation is much better than that without segmentation, with a 50% performance improvement in successful ratio. Finally, for 2-D/3-D networks, the stretch of greedy routing with CONSEL segmentation is only around 1.7. That is, the traditional algorithm—greedy routing designed for a simple network shape—is successfully adapted to complex 2-D/3-D environments with the help of CONSEL.

B. Connectivity-Based Localization

Localization is crucial for many applications, such as position-aware data dissemination and processing. Connectivity-based localization aims to produce a relative coordinate system for a network without using ranging techniques (for measuring physical distance between nodes). In this approach, the physical distance between nodes is estimated with the hop count of their shortest path, relying on the assumption that the hop distance correlates well with Euclidean distance. However, when the network topology is irregular, the shortest path may be significantly bent. As a result, the hop distance between two nodes may deviate vastly from the Euclidean distance.

With shape segmentation, we only need to apply a traditional localization algorithm to individual regions, thereby avoiding the above problem. The segmentation-assisted localization algorithm is a simple extension of existing algorithms. First, the network is segmented into a set of regions. Second, within each region, a traditional multilateraion algorithm (such as the atomic trilateration method in [26]) is applied. The algorithm computes a node's coordinates using distance measurements to three/four reference points. Third, we combine the coordinates of all the regions into a global map. In this step, for every pair of adjacent regions, there are some nodes located on their shared segment lines, and these nodes are assigned two virtual coordinates. We

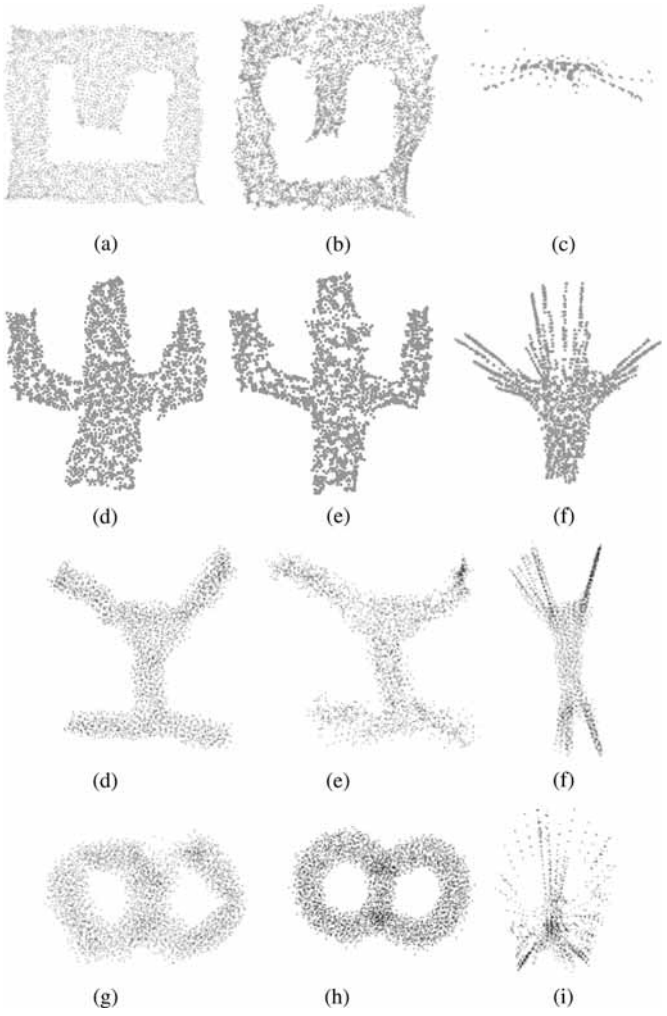

Fig. 19. Localization results. Columns (from left to right): 1) 2-D/3-D localization results with CONSEL; 2) 2-D localization results with the CONVEX [27] algorithm; 3-D localization results using CATL [28]; 3) 2-D/3-D localization results using pure multilateraion without segmentation. Rows: 1) Single-hole shape; 2) Cactus shape; 3) Chicago airport terminal-2 shape; 4) 3-D-8 shape. (a) LE is 0.578. (b) LE is 1.01. (c) LE is 2.5. (d) LE is 0.58. (e) LE is 0.7. (f) LE is 1.78. (d) LE is 0.84. (e) LE is 2.93. (f) LE is 3.56. (g) LE is 1.15. (h) LE is 2.01. (i) LE is 6.66.

perform a linear transformation on these virtual coordinates. Finally, the use of spring-mass algorithm [17] is applied to refine nodes' coordinates with respect to their neighbor connections.

We run simulations using segmentation results from CONSEL and CONVEX [27]. For comparison, we implement two additional algorithms. First, a pure multilateraion without segmentation is evaluated. Second, CATL [28], a state-of-the-art localization algorithm with 3-D capability, is also evaluated. The key idea of CATL is to identify notch nodes where the hop count of the shortest path between two nodes deviates from the true Euclidean distance. CATL then uses an iterative notch-avoiding multilateration scheme to localize the network. In addition to shape recovery, we measure the localization error (LE), defined as the ratio of the Euclidean distance between the estimated location and its true location to the communication range.

Fig. 19 shows the localization results. First, for the 2-D network, we see a 40% improvement in localization accuracy with the CONSEL segmentation, compared to the CONVEX scheme, because CONSEL generates fewer convex regions, thereby avoiding the error propagation in CONVEX. Second, the accuracy of pure multilateraion is too poor to be acceptable for complex networks containing holes or concave regions. Third, for the 3-D network, an up to 50% improvement in localization accuracy can be attained using our scheme, compared with CATL [28]. The reason is that CATL suffers from the location estimation from a long distance when the node is far away from the anchors. More specifically, after each iteration, newly localized nodes will serve as beacon nodes and flood their locations through the network. The localization errors after each round will be accumulated. In addition, the performance of CATL highly depends on the choice of beacon nodes. In contrast, the traditional localization algorithm applied within each region avoids this problem. Overall, the segmentation-assisted localization can recover the network layout with small errors.

VII. Conclusion

We have presented CONSEL, a distributed and scalable algorithm for segmenting 2-D/3-D sensor networks. Using connectivity information only, this algorithm is the first solution with both 2-D and 3-D segmentation capability. In addition, the convexity deviation of network regions after segmentation can be bounded. We have demonstrated the effectiveness of CONSEL through extensive simulations. Moreover, we show how segmentation benefits existing applications such as geographic routing and connectivity-based localization.

In the future, in addition to routing and localization, we will study its uses in other applications such as data processing [8], [9], [11], skeleton extraction [2], [10], and especially for 3-D sensor networks, which have attracted recent attention of many researchers. We will also seek more efficient tools such as amorphous computing [21] and spatial computing [29] to conduct simulations of our algorithm. Finally, we plan to implement our algorithm on sensor network testbeds to study its efficiency in more realistic environments.

References

[1] I. F. Akyildiz and E. P. Stuntebeck, "Wireless underground sensor networks: Research challenges," *Ad Hoc Netw. J.*, vol. 4, no. 6, pp. 669–686, 2006.

[2] J. Bruck, J. Gao, and A. A. Jiang, "Map: Medial axis based geometric routing in sensor networks," *Wireless Netw.*, vol. 13, no. 6, pp. 835–853, 2007.

[3] T. K. Dey, J. Giesen, and S. Goswami, "Shape segmentation and matching with flow discretization," in *Proc. Workshop Algor. Data Struct.*, 2003, pp. 25–36.

[4] T. Eren, D. K. Goldenberg, W. Whiteley, Y. R. Yang, A. S. Morse, B. D. O. Anderson, and P. N. Belhumeur, "Rigidity, computation, and randomization in network localization," in *Proc. IEEE INFOCOM*, 2004, pp. 2673–2684.

[5] A. Fomenko and T. Kunii, *Topological Modeling for Visualization*. New York, NY, USA: Springer-Verlag Telos, 1997.

[6] R. Ghrist and A. Muhammad, "Coverage and hole-detection in sensor networks via homology," in *Proc. IEEE IPSN*, 2004, pp. 254–260.

[7] R. Hekmat and P. V. Mieghem, "Connectivity in wireless ad-hoc networks with a log-normal radio model," *Mobile Netw. Appl.*, vol. 11, no. 3, pp. 351–360, 2006.

[8] H. Jiang, S. Jin, and C. Wang, "Parameter-based data aggregation for statistical information extraction in wireless sensor networks," *IEEE Trans. Veh. Technol.*, vol. 59, no. 8, pp. 3992–4001, Oct. 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 Trans. Parallel Distrib. Syst.*, vol. 22, no. 6, pp. 1064–1071, Jun. 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 Trans. Parallel Distrib. Syst.*, vol. 21, no. 5, pp. 710–721, May 2010.

[11] H. Jiang, W. Liu, D. Wang, C. Tian, X. Bai, X. Liu, Y. Wu, and W. Liu, "A general framework for efficient continuous multidimensional top-k query processing in sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 9, pp. 1668–1680, Sep. 2012.

[12] H. Jiang, T. Yu, C. Tian, G. Tan, and C. Wang, "CONSEL: Connectivity-based segmentation in large-scale 2D/3D sensor networks," in *Proc. IEEE INFOCOM*, 2012, pp. 2086–2094.

[13] H. Jiang, S. Zhang, G. Tan, and C. Wang, "CABET: Connectivity-based boundary extraction of large-scale 3D sensor networks," in *Proc. IEEE INFOCOM*, 2011, pp. 784–792.

[14] B. Karp and H. T. Kung, "GPSR: Greedy perimeter stateless routing for wireless networks," in *Proc. ACM MOBICOM*, 2000, pp. 243–254.

[15] M. Klann, *Mobile Response. Chapter Tactical Navigation Support for Firefighters: The LifeNet Ad-Hoc Sensor-Network and Wearable System*. New York, NY, USA: Springer-Verlag, 2009, pp. 41–56.

[16] F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger, "Geometric ad-hoc routing: Of theory and practice," in *Proc. ACM PODC*, 2003, pp. 63–72.

[17] B. Leong, B. Liskov, and R. Morris, "Greedy virtual coordinates for geographic routing," in *Proc. IEEE ICNP*, 2007, pp. 71–80.

[18] M. Li and Y. Liu, "Rendered path: Range-free localization in anisotropic sensor networks with holes," in *Proc. ACM MobiCom*, 2007, pp. 51–62.

[19] J.-M. Lien and N. M. Amato, "Approximate convex decomposition of polyhedra," in *Proc. ACM Symp. Solid Phys. Model.*, 2007, pp. 121–131.

[20] A. Ling, "The power of non-rectilinear holes," in *Proc. 9th Int. Colloq. Automata Lang. Program*, 1982, pp. 369–383.

[21] MIT, Cambridge, MA, USA, "Amorphous computing," 2002 [Online]. Available: http://groups.csail.mit.edu/mac/projects/amorphous/

[22] A. Nguyen, N. Milosavljevic, Q. Fang, J. Gao, and L. J. Guibas, "Landmark selection and greedy landmark-descent routing for sensor networks," in *Proc. IEEE INFOCOM*, 2007, pp. 661–669.

[23] A. Panangadan and G. S. Sukhatme, "Data segmentation for region detection in a sensor network," in *Proc. DCOSS*, 2005, pp. 1–12.

[24] A. Rao, S. Ratnasamy, C. Papadimitriou, S. Shenker, and I. Stoica, "Geographic routing without location information," in *Proc. ACM MobiCom*, 2003, pp. 96–108.

[25] O. Saukh, R. Sauter, M. Gauger, P. J. Marron, and K. Rothernel, "On boundary recognition without location information in wireless sensor networks," in *Proc. IPSN*, 2008, pp. 207–218.

[26] A. Savvides, C.-C. Han, and M. B. Strivastava, "Dynamic fine-grained localization in ad hoc networks of sensors," in *Proc. ACM MobiCom*, 2001, pp. 166–179.

[27] G. Tan, M. Bertier, and A.-M. Kermarrec, "Convex partition of sensor networks and its use in virtual coordinate geographic routing," in *Proc. IEEE INFOCOM*, 2009, pp. 1746–1754.

[28] G. Tan, H. Jiang, S. Zhang, and A.-M. Kermarrec, "Connectivity-based and anchor-free localization in large-scale 2D/3D sensor networks," in *Proc. ACM MobiHoc*, 2010, pp. 191–200.

[29] B. Technologies, "Spatial computing," 2012 [Online]. Available: http://proto.bbn.com/commons/

[30] S. Xia, X. Yin, H. Wu, M. Jin, and X. D. Gu, "Deterministic greedy routing with guaranteed delivery in 3D wireless sensor networks," in *Proc. ACM MobiHoc*, 2011, Art. no. 1.

[31] H. Zhou, N. Ding, M. Jin, S. Xia, and H. Wu, "Distributed algorithms for bottleneck identification and segmentation in 3D wireless sensor networks," in *Proc. IEEE SECON*, 2011, pp. 494–502.

[32] X. Zhu, R. Sarkar, and J. Gao, "Shape segmentation and applications in sensor networks," in *Proc. IEEE INFOCOM*, 2007, pp. 1838–1846.

[33] X. Zhu, R. Sarkar, and J. Gao, "Topological data processing for distributed sensor networks with Morse-Smale decomposition," in *Proc. IEEE INFOCOM*, 2009, pp. 2911–2915.

Hongbo Jiang (M'08) received the B.S. and M.S. degrees in applied math from Huazhong University of Science and Technology, Wuhan, China, in 1999 and 2012, respectively, and the Ph.D. degree from Case Western Reserve University, Cleveland, OH, USA, 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.

Tianlong Yu (S'11) received the B.S. degree in computer science from Huazhong University of Science and Technology HUST, Wuhan, China, in 2011 and is currently pursuing the M.S. degree in electronics and information engineering at HUST.

His current research areas include wireless and sensor networks.

Chen Tian (M'09) received the B.S., M.S., and Ph.D. degrees in lectronics and information engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2000, 2003, and 2008, respectively.

He joined the faculty as a Lecturer with the Department of Electronics and Information Engineering, Huazhong University of Science and Technology. His research interests include distributed networks and wireless networks.

Guang Tan (M'12) received the Ph.D. degree in computer science from the University of Warwick, Coventry, U.K., in 2007.

He is currently an Associate Professor with the Shenzhen Institute of Advanced Technology (SIAT), Chinese Academy of Sciences, Shenzhen, China, where he works in the area of distributed systems and networks. From 2007 to 2010, he was a Postdoctoral Researcher with INRIA-Rennes, Rennes, France.

Dr. Tan is a member of the Association for Computing Machinery (ACM).

Chonggang Wang (M'04–SM'09) received the Ph.D. degree in computer science from Beijing University of Posts and Telecommunications, Beijing, China, in 2002.

He has conducted research with NEC Laboratories America, Princeton, NJ, USA; AT&T Labs Research, Middletown, NJ, USA; the University of Arkansas, Fayetteville, AR, USA; and Hong Kong University of Science and Technology, Hong Kong. He has published more than 80 journal/conference articles and book chapters. His research interests include future Internet, machine-to-machine (M2M) communications, and cognitive and wireless networks.