On the Utility of Concave Nodes in Geometric Processing of Large-Scale Sensor Networks

Shengkai Zhang, Guang Tan, *Member, IEEE,* Hongbo Jiang, *Member, IEEE,* Bo Li, *Fellow, IEEE,* and Chonggang Wang, *Senior Member, IEEE*

***Abstract*—As a sensor network grows large, it may become increasingly complex in topology due to its close ties to the surrounding environment. Previous work has shown that proper geometric processing of the network (e.g., boundary detection and localization) can provide very helpful information for applications to optimize their performance. To that end, numerous algorithms have been developed, providing a variety of inspiring solutions, yet exhibiting an ad hoc style in principle and implementation. In this paper we show that the crux of solving many of the problems caused by complex topology is to identify the *concave nodes*, nodes that are located at *concave network corners*, where the boundary has an inner angle greater than π. The knowledge of such nodes makes several important tasks, namely geometric embedding, full localization, convex segmentation, and boundary detection, relatively easier or perform significantly better, as confirmed by simulations. These findings suggest that concave nodes can serve as a basic supporting structure for general geometric processing tasks and geometry-related applications in sensor networks.**

***Index Terms*—Wireless sensor networks, geometric processing, concave nodes.**

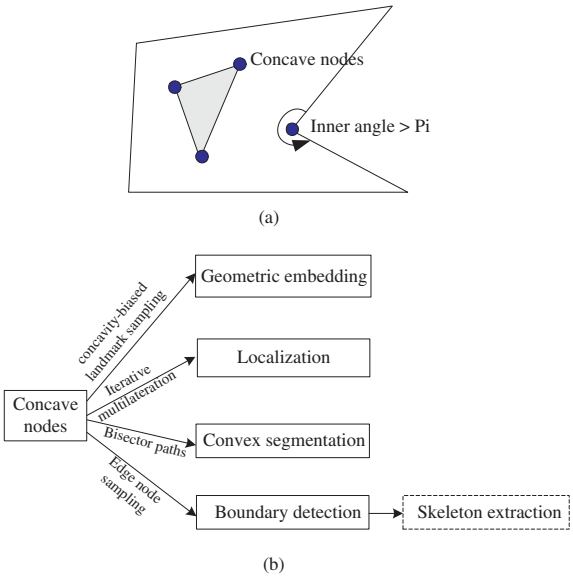


Fig. 1. (a) Concave network corners (where the boundary's inner angle is greater than π) and concave nodes, represented by filled blue circles. (b) The knowledge of concave nodes facilitates several geometric processing tasks of sensor networks. Problems in solid-line boxes are our focus in this paper.

I. Introduction

WHEN a sensor network grows large, it tends to cover an increasingly wide area, which may exhibit a complex layout. This naturally produces a complex network topology that poses significant challenges to some applications. To tackle this problem, researchers have found the information of network geometry very helpful for applications to accomplish their tasks with improved efficiency. For example, knowing the boundaries of a network [6], [9] helps an application to detect the entering and leaving of a monitored object; segmenting a network into nicely shaped pieces improves the performance of applications such as facility placement and random sampling [32]. The most thorough geometric processing of a network is full localization [15], [17], [23]–[25], [30], whereby the position of every node needs to be recovered. The location information proves to be extremely useful for network routing and data aggregation.

To make few assumptions about network configuration and thus obtain maximum applicability, it is preferable to assume only connectivity information for geometric processing [6], [9], [10], [15], [17], [24], [25], [29], [30], [32]. That is, one has to infer as much geometric information of the network as possible from merely the neighboring relations between nodes. Existing work in this line has been extensive. Connectivity-based network localization, for example, finds dozens of solutions alone. The various approaches and methods provide us with many inspiring ideas, yet these ideas are somewhat ad hoc, in that they arise from different intuitions and follow different principles. In this paper we attempt to pinpoint the common reason for the challenges faced by the various geometric processing tasks. We argue that identifying the *concave nodes*, nodes that are located around the concave holes/boundaries (see blue points in Figure 1(a)), is the crux of solving the complex topology problem.

The concave nodes identification is a key challenge of many applications and research work and the knowledge of concave nodes facilitates a number of nontrivial geometric processing tasks (see an illustration in Figure 1(b)), including:

1) Geometric embedding [3], [5], [7], [16], [22], [26], [31],

Manuscript received November 30, 2012; revised June 5 and October 28, 2013; accepted November 4, 2013. The associate editor coordinating the review of this paper and approving it for publication was L. Cai.

S. Zhang and G. Tan are with SIAT, Chinese Academy of Sciences (e-mail: {shengkai.zhang,guangtan}@gmail.com). S. Zhang is also with Huazhong University of Science and Technology, as well as Hong Kong University of Science and Technology.

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

B. Li is with Hong Kong University of Science and Technology, Hong Kong (e-mail: bli@cse.ust.hk).

C. Wang is with InterDigital Communications, U.S.A. (e-mail: cg-wang@ieee.org).

Digital Object Identifier 10.1109/TWC.2013.120313.121898

which aims to embed a network into a geometric space, that is, to assign each node a set of coordinates and define a distance function based on those coordinates. The distance between two nodes in the geometric space should ideally conform to their distance in the original network. This technique is useful for designing efficient routing algorithms.

2) Full localization [15], [17], [23]–[25], [30], which aims to recover the position of each node. It has been shown in [25] that the concave nodes (*notch nodes* in their terminology) are the key to a concavity-avoiding iterative localization algorithm that enables accurate 2D/3D localization using connectivity information alone.
3) Convex segmentation [26], which aims to divide the network into convex pieces that allow many applications to work to their full advantages [32]. Concave nodes provide the exact information as to where the dividing lines should start, as a bisector line from a concave corner always eliminates that concavity in the newly generated regions.
4) Boundary detection [6], [9], [13], [29], which attempts to find nodes that are located at the network boundaries, and to connect them into cycles corresponding to the boundaries of the sensing field. Knowing concave nodes helps to find a complete, though coarse, boundary of the network, based on which a restricted flooding process is performed to find a set of connected and tight boundary nodes.

Concave nodes are useful because they signify one of the most critical geometric features of a network: the network concavities. These form a major obstacle for many algorithms to stay simple or perform optimally. Compared to other assisting structures for geometric processing such as boundaries, concave nodes are cheaper to identify while providing fairly strong support for inferring a network's geometry. In this paper, we show how to detect the concave nodes in a localized way, assuming a general radio model and using only connectivity information.

To the best of our knowledge, the only previous work that stresses the function of concave nodes is [25], where concave nodes are used to help with localization. The detection of concave nodes relies on network-wide flooding from $O(n_b)$ nodes, where n_b is the number of boundary nodes, leading to a global message cost of $O(n \times n_b)$. In comparison, all the communication involved in our detection method is localized, covering a neighborhood of at most few hops. This amounts to a total message cost of $O(\text{poly}(\overline{d}) \times n_b)$, where "poly" denotes a polynomial function and $\overline{d}$ is the average node degree. Since $\overline{d}$ normally does not grow with network size in real systems, our method achieves a significant reduction of message cost.

The remainder of the paper is organized as follows. Section II describes the method to identify concave nodes; Sections III through VI show how the knowledge of concave nodes helps with four important geometric processing tasks; Section VII concludes the paper.

II. Detecting Concave Nodes

This section discusses how to detect concave nodes. Our method first identifies *edge nodes*, the nodes near the network boundaries, and then from these nodes, finds a set of nodes at the concave corners. Before presenting the design, we first explore several alternative approaches that might be tempting to a designer, and explain why they do not work. In this paper, node density is measured by average node degree. Nodes are by default deployed in a perturbed grid pattern [15], [17], [25], [30]: a node is placed at a grid point with a perturbation governed by a Gaussian distribution. A node v is said to be an edge node if its degree is smaller than $\delta \times \overline{N}$, where $0 < \delta < 1$ is a parameter and $\overline{N}$ is the average node degree.

A. Definition of Concave Nodes

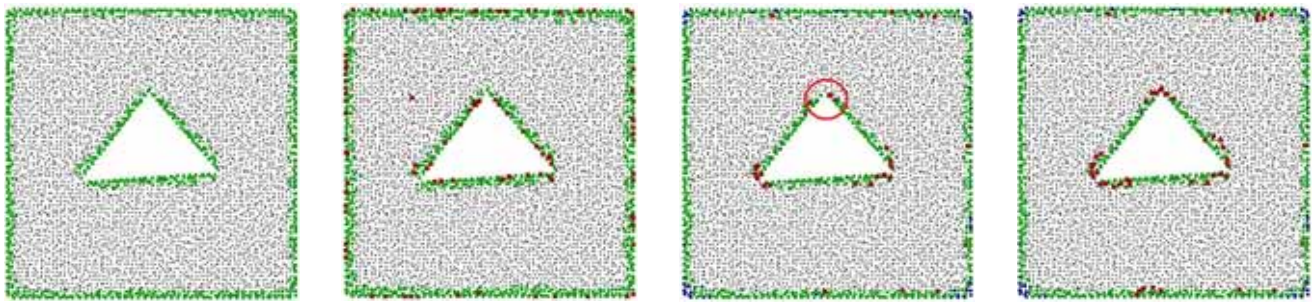
We define concave nodes of a discrete network based on the definition of their counterparts in the continuous domain. In practice, a site survey on the sensing field should be conducted before the sensor nodes are deployed. It is thus reasonable to assume that the shape and size of the field's boundaries are known. We use a set of polygons to (approximately) represent these boundaries, called the field's *profile polygons*. For each polygonal vertex whose inner angle is greater than π, draw a circle of radius $k_c\mathcal{R}$, where k_c is a parameter (e.g., 3) and $\mathcal{R}$ is the communication range. Such a circular area is called a *concave corner* of the sensing field. The definition of concave nodes thus follows.

Definition 1: (**Concave Node**) Suppose a sensor network is deployed in a sensing field with known profile polygons. A node is called a *concave node* if it falls in at least one of the field's concave corners.

When an algorithm successfully detects a concave node, we say that the algorithm detects the corresponding concave corner. The percentage of concave corners detected is called the algorithm's *detection rate* (DR). An algorithm's *false detection rate* (FDR) is defined as the percentage of detected nodes that do not fall in any concave corner. DR represents the effectiveness of our proposed detection method.

B. The Failure of Several Simple Methods

1) Using Node Degree: It is easy to see that boundary nodes on average have a smaller degree (i.e., number of neighbors) than nodes in the network interior. This characteristic seems to suggest a way to find nodes with distinct geometric features. For example, concave nodes tend to have a smaller degree than that of interior nodes, and a larger one than that of average boundary nodes. However, a close examination immediately shows that this approach is not very helpful, since a proper difference margin unlikely exists for distinguishing nodes: when a concave network corner is very sharp, the nodes around the corner may well have a degree very close to average interior nodes. To capture the concave nodes, a small differentiating margin would be needed, which will generate excessive false positives across the network. Raising the margin can help reduce noise, but inevitably at the cost of missing some real concave nodes, as illustrated by Figure 2(a), where the green belt areas comprise the edge nodes. The belts are not entirely continuous – they break at some hole corners (e.g., the left corner of the triangular hole) where concave nodes should reside.



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

Fig. 2. An example network with 5175 nodes. (a) Edge nodes (in green). (b) Concave nodes (in red) detected by the edge degree based method (Attempt 3). (c) Extended concave nodes detected by the betweenness centrality measure. (d) All concave nodes (including extended ones) detected by our method.

2) Using Edge Degree: Though unable to give us concave nodes, the determined edge nodes seem to form a basis for our further pursuit. A heuristic, for example, is to find the nodes at the ends of those broken belt areas. Those nodes are normally close enough to, or simply are, the concave nodes. Consider a measure *edge degree* that counts the number of a node's neighbors from the edge node set. Intuitively those nodes should have fewer edge neighbors than other edge nodes, since they are at the ends of the belt segments. Some simple experiments show that this criterion helps little since the edge belt is not regular, and degree information is too susceptible to noise.

3) Aggregating Edge Degree: Recall that a node is marked as an edge node if its degree is smaller than δ times the average node degree, where $0 < \delta < 1$ is a tunable parameter. Since a concave node's degree is normally close to that of an interior node (thus average node degree), it will be detected only when δ is close to 1. On the other hand, a concave node's edge degree is relatively smaller than that of a regular edge node, because the edge belts tend to break at the concave corners. Thus, it is expected that the concave nodes would be detected by a large δ with a relatively small edge degree.

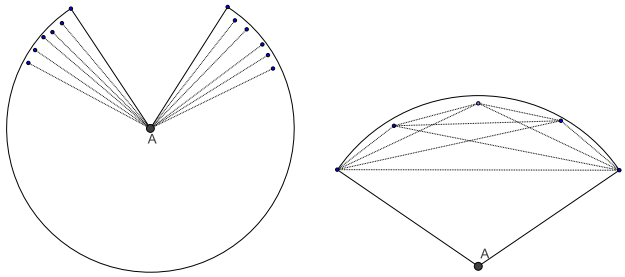
We thus varied δ between 0.5 and 0.9 with an increment 0.1, and added up the edge degree of each edge node during each iteration. We marked the bottom 5% of nodes with smallest aggregate degrees as concave nodes. This effort again turned out to be unsatisfactory, because the accumulation helps little in reducing noise. Figure 2(b) gives an example result. It can be seen that many nodes are taken as concave nodes when they are really not (see red nodes on the square boundary). On the other hand, some concave nodes (around the corner of the triangular hole) are still missing from the supposed concave node set.

C. Our Method

In this section we present a more effective method to detect concave nodes. Before detecting those nodes, we identify a set of *extended concave nodes*, defined roughly as nodes in the vicinity of concave nodes.

We first describe an implementation of the edge nodes detection mentioned earlier, which will be a starting point of our core design.

1) Detecting Edge Nodes: Detecting edge nodes is straightforward. These nodes are meant to be only an assisting structure, so do not need to be tight or complete with respect to the real boundaries. Through simple local communications, every node obtains the number of neighbors in two-hop distance, called its *two-hop degree*. Nodes exchange this degree information in a gossip manner, so finally every node gets a roughly uniform random sample of nodes' two-hop degrees in the network. From these samples a node calculates the average two-hop *interior degree*, $\overline{d}$, corresponding to the degree of nodes found in the interior of the network. To avoid bias toward boundary areas, the calculation is based on a fraction (e.g., 1/2) of top samples.



(a) Concave node (b) Convex node

Fig. 3. Illustration of betweenness centrality for node A.

With $\overline{d}$, a node p compares its own two-hop degree d_p with this average value. If $d_p < (1-\epsilon) \times \overline{d}$, where $0 < \epsilon < 1$ is a parameter (by default 0.2 in our experiments), then p marks itself as an *edge node*. It may happen that some interior nodes are falsely taken as edge nodes. To reduce such errors, nodes whose two-hop neighborhoods do not contain any other edge nodes are removed from the edge node set.

2) Identifying Extended Concave Nodes: We shall use a measure called *Betweenness Centrality* [8] (centrality for short) to identify the extended concave nodes.

Definition 2: (**Betweenness Centrality**) The betweenness centrality of a node p, $C(p)$, is given by

$$C(p) = \sum_{s \neq t} \frac{\sigma_{st}(p)}{\sigma_{st}},$$

where σ_{st} denotes the number of shortest paths from s to t, $\sigma_{st}(p)$ is the number of shortest paths from s to t that pass through p.

For ease of distributed implementation, we modify this measure to restrict the set of nodes from which s and t are drawn. In our method this set consists of p's rth hop neighbors, called p's *rth level*. Figure 3 illustrates this modified concept with two nodes, one assumed to be at a concave network corner (hence a concave node) and the other a convex network

corner (convex node). For the concave node, there are a number of shortest paths passing through it, and the more concave the corner, the more shortest paths it sees. In contrast, for a convex node, there is no shortest path passing through it. This distinction is the basis for our design.

To compute shortest paths, a node p collects a size-limited local subgraph of the network. Throughout the experiments, we set this limit size to be 50 nodes or a full two-hop neighborhood size, whichever is larger. In our experiments, this typically covers a full two-hop neighborhood plus some nodes on the third-level. The nodes on the outmost level are required to be pairwise non-adjacent. All links between the selected nodes are included in the subgraph.

Only edge nodes build their local subgraphs and compute centrality. The computed centrality values are gossiped [12] among the edge nodes, with the restriction that a node's centrality is not propagated beyond three hops away. This gossip protocol can be understood as aggregated and scoped broadcast, which is message efficient. If a node finds that it's centrality value is greater than zero and is among the largest (top 10% in our case), it marks itself as a concave node.

Algorithm 1 gives the pseudocode of identifying extended concave nodes at an edge node p and Figure 2(c) is the result of extended concave nodes.

Algorithm 1 Extended concave nodes determination (at edge node p)

```
Build p's local subgraph G_p(of 50 nodes maximum);
for each q ∈ G_p do
    for each q' ∈ G_p do
        if q' ≠ q then
            count all possible shortest paths from q to q' to get σ_qq';
            count all paths from q to q' that pass through p to get σ_qq'(p);
        end if
        C(p) ← C(p) + σ_qq'(p) / σ_qq';
    end for
end for
Collect centrality values of edge nodes in three-hop neighborhood by gossip;
if C(p) > 0 and C(p) is among top 10% of the neighbors' centralities then
    p marks itself as an extended concave node;
end if
```

3) Identifying Concave Nodes: Since concave nodes may not be included in the edge nodes, they will not be detected by the process described so far. We thus need an extra step to find these nodes. The basic observation is that these nodes, if undiscovered, are right between two groups of extended concave nodes. In Figure 2(c), several extended concave nodes within the circled area will be used to find the concave nodes.

For every extended concave node p, it performs a scoped flooding within its r-hop neighborhood ($r = 3$ in our case).The flooded message will carry information about the node ID sequence it has traversed. The node ID sequence is effectively a shortest path which will be collected by p's r-hop neighbor, say q. q will receive multiple messages from its neighbors. These messages contain multiple node sequences between p and q. If q is an edge node and finds that these shortest paths contain no edge nodes except p and q, then all those non-edge nodes on these paths mark themselves as concave nodes.

Figure 2(d) gives the final output of the whole concave detection process. Compared to Figure 2(c), this result includes more concave nodes. On the other hand, the supplementation mechanism incurs some additional false positives, as shown on the outer network boundary. Our experience suggests that completeness is more important than tightness on detecting concave nodes, since missing concave nodes is more adversely influential than generating false concave nodes. The former could result in much distorted view of the network topology, while the latter only makes the algorithm behave more conservatively than necessary.

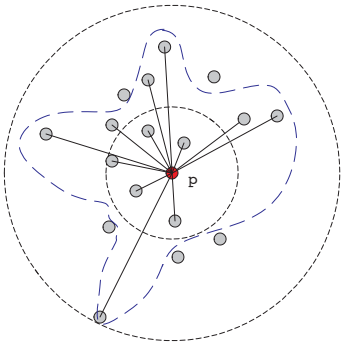


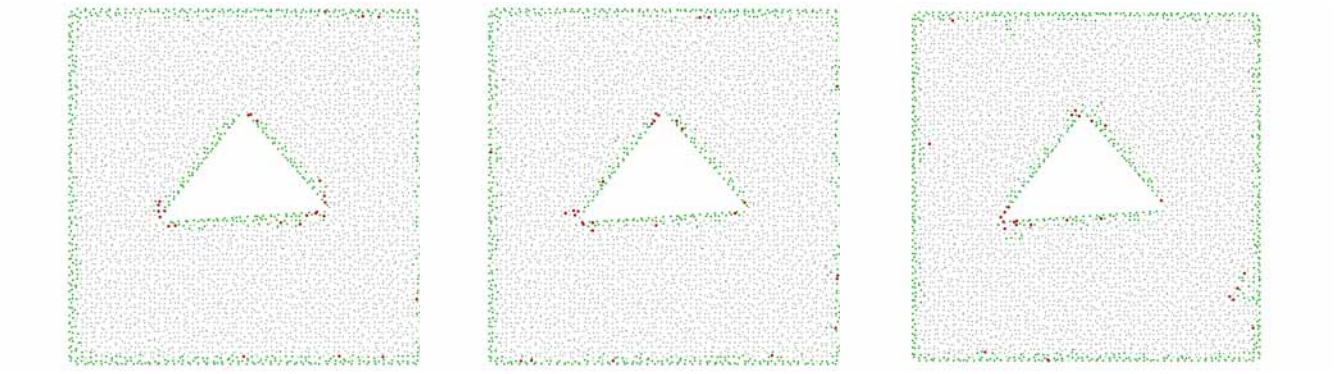
Fig. 4. Default Quasi-UDG radio model ($\alpha = 0.4$) used in the simulations. The annular area between the two dashed circles is the uncertain region, in which the existence of a link is governed by a probability. Within the inner circle a link always exists between p and a neighbor beyond the outer circle no link from p is allowed.

D. Performance Evaluation

In this section we use simulations to examine how successfully the proposed method detects the concave nodes, and how robust it is to different network settings. For all settings, the network remains connected. The radio range follows a Quasi-UDG model with a tunable parameter $0 < \alpha < 1$ controlling how irregular the radio range can be. In the UDG (Unit Disk Graph) model, we simply set a communication range $\mathcal{R}$ as the condition of a link between two nodes u and v. If and only if $|uv| \leq \mathcal{R}$, the link uv exists. In the Quasi-UDG model, $\mathcal{R}$ becomes a baseline range. A link between two nodes u and v exists with probability 1 when $|uv| \leq (1-\alpha)\mathcal{R}$; with probability $0 < \eta < 1$ when $(1-\alpha)\mathcal{R} < |uv| \leq (1+\alpha)\mathcal{R}$; the link does not exist when $|uv| > (1+\alpha)\mathcal{R}$. The larger the α, the more uncertain the link relationship between two nodes, imitating the irregular radio patterns in reality. Throughout our experiments, $\alpha = 0.4$ by default. The corresponding model is illustrated in Figure 4, where the blue dashed line approximately represents p's radio range.

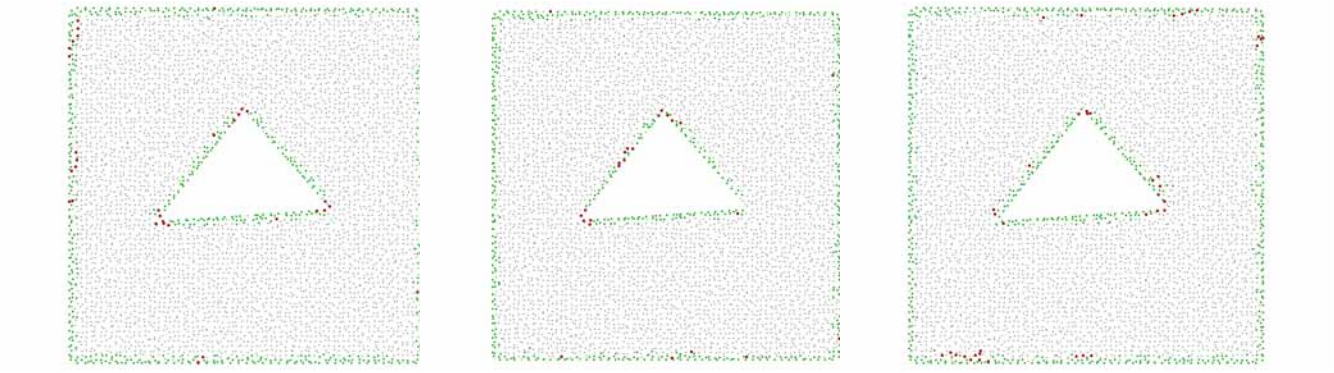
The network configuration can vary in many dimensions. We consider a few important ones: topology, density, and radio irregularity (α).

1) Effect of Node Density: Node density is adjusted by changing the radio range. The lowest average node degree is 8, below which the network starts to become disconnected. Figure 5 shows the performance of concave nodes detection under different node densities. We can see that despite the trend of degraded performance and increased noise, the method is able to find the most critical concave nodes. Normally, a lower density will be more challenging for geometric processing tasks [17], [25], [29]. However the calculation of centrality is based on a fixed upper limit of neighborhood size, rather



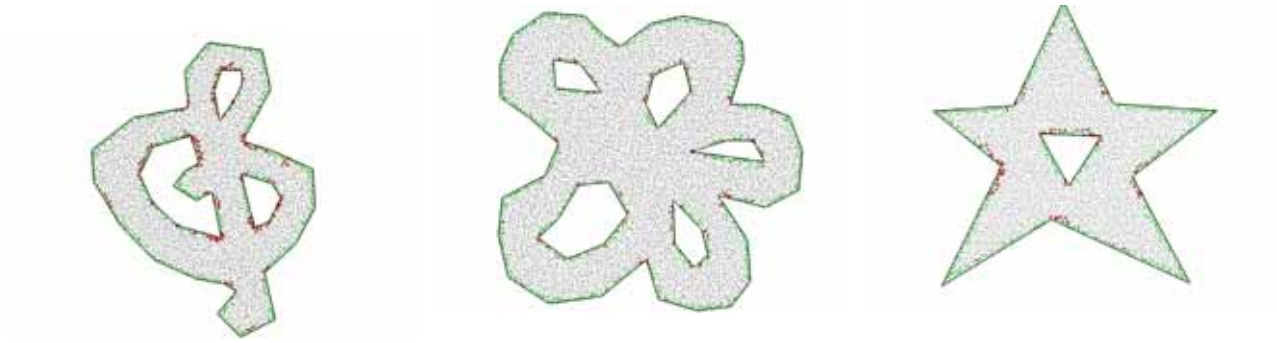
(a) avg node degree 12, FDR = 42.3%, DR = 100%. (b) avg node degree 10, FDR = 48.3%, DR = 100%. (c) avg node degree 8, FDR = 51.8%, DR = 100%.

Fig. 5. Effect of different node densities on concave node detection. The network has 5175 nodes, $\alpha = 0.4$.



(a) $\alpha = 0$, FDR = 62.2%, DR = 100%. (b) $\alpha = 0.2$, FDR = 56.0%, DR = 100%. (c) $\alpha = 0.3$, FDR = 71.8%, DR = 100%.

Fig. 6. Effect of radio irregularity on concave node detection. The network has 5175 nodes, with average node degree 8.



(a) Music topology (b) Flower topology (c) Star topology

Fig. 7. Several complex networks reproduced from the topology examples in [29], [30]. The red points represent the concave nodes detected. The violet points represent the false negative vertices. The music topology (a) has 7,058 nodes, with average node degree 12.2. The flower topology (b) has 12,777 nodes, with average node degree 12.3. The star topology (c) has 5,375 nodes, with average node degree 11.9.

than on a fixed hop count. This to some degree alleviates the impact of low node density. One can see that the false detection rate increases with decreasing node density, because a lower density leads to more noises in node distribution.

2) Effect of Radio Model: Figure 6 presents the effect of different values of parameter α. It can be seen that our method produces reasonably good results for fairly large α values. Although there are more noise (false positives) for a larger α, the detection method still captures all the critical concave corners.

3) Effect of δ: Finally, we examine how sensitive the detection performance is to δ, the degree differentiating margin used in edge node determination. A too large δ (e.g., close to 1) will lead to excessive noise across the network which makes the subsequent task impossible, and a small δ (e.g., $\delta < 0.5$) generates only a few and isolated edge nodes, which are also undesirable. We therefore focused on $\delta \in [0.5, 0.8]$. Varying the value did not produce significant and consistent differences in the results. The reason is that the edge nodes merely serve as a redundant set for centrality checking, which itself is independent of being on the network edge. The output networks are very similar to the results above, so are not presented here.

4) Effect of network topology: Figure 7 presents several complex network topologies, where the first two are repro-

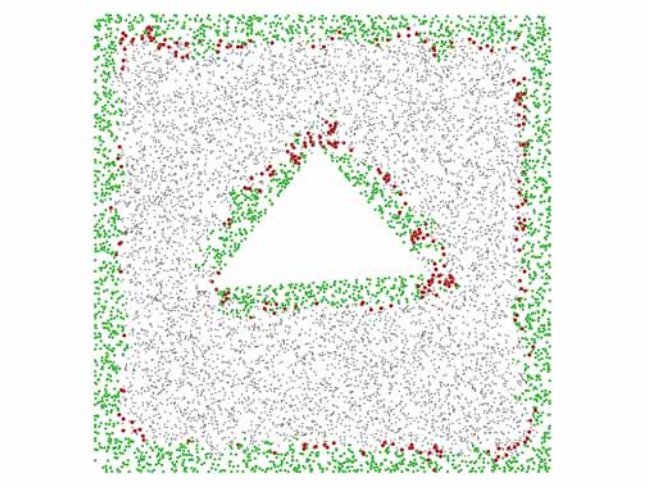


Fig. 8. The performance under uniform random distribution. The average degree is 22.4. FDR=73.8%, DR=100%.

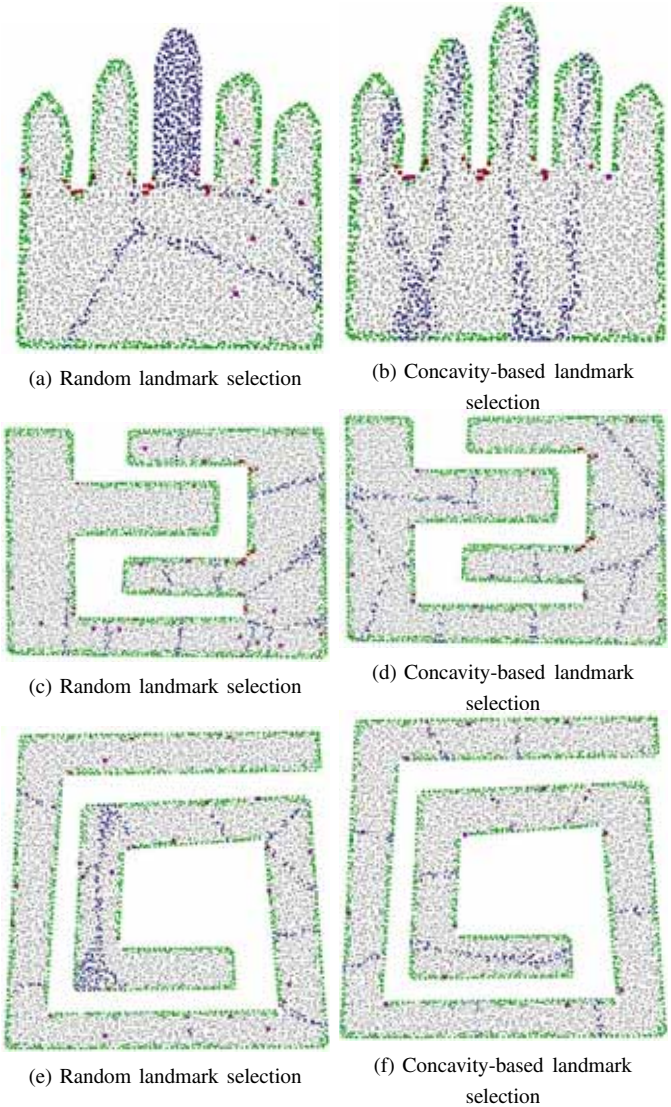
duced from the examples in [29], [30]. The concave nodes are marked red in the picture. It should be noted that several concave corners are missing in the music and flower topologies, mainly because the corners are too 'flat'. These present a major challenge to connectivity-based algorithms. Fortunately, these flat corners do not cause as much disturbance to applications as those sharp corners, due to their diminishing difference from a straight line boundary. Table I provides the specific detection rates for the different topologies.

5) Uniform random distribution: Figure 8 shows the concave node detecting result for uniform random distribution of network nodes. This distribution exhibits more randomness in nodes' positions, and thus generates more false edge nodes, which lead to a higher false detection rate of concave nodes. Nevertheless, the algorithm still successfully detects all the concave corners. These results should be taken as a reference for comparison purposes. Practical sensor networks rarely exhibit such a level of randomness (which is very suboptimal in terms of coverage and connectivity guarantees), since a sensor network is always deployed after proper survey of the sensor field. It is thus not advisable to scatter around the nodes in a truly 'random' manner, without regard to its potential cost, when a moderate amount of human control may bring significant performance benefits.

6) Message cost: In the process of concave nodes detection, all the communication is local. Assuming no packet retransmission due to link failures, we count the total number of messages transmitted by the nodes in the topologies in Figure 7. The message costs of the three topologies are 48559, 77739, and 36130, respectively. On average, the per-node message costs are merely 6.88, 6.08, and 6.72, respectively. Taking into account link failures may increase the per-node by a small factor, but the message cost will still remain very low.

We also calculated the message cost of the flooding based concave node detection method in [25]. The per-node message costs of the three topologies in Figure 7 are 93, 131, and 71, respectively. Clearly, the method in [25] is more expensive.

The algorithm in [25] claims a per-node message cost of $O(n_e)$, where n_e denotes the number of edge nodes.



(a) Random landmark selection

(b) Concavity-based landmark selection

(c) Random landmark selection

(d) Concavity-based landmark selection

(e) Random landmark selection

(f) Concavity-based landmark selection

Fig. 9. Several network topologies with two landmark selection approaches. The red, pink, and blue points represent concave nodes, cell leaders (landmarks), and cell boundary nodes, respectively. The Fingers shape topology ((a), (b)) has 4,862 nodes with average degree 11.96. The Gears shape topology ((c), (d)) has 6,844 nodes with average degree 11.97. The Spiral shape topology ((e), (f)) has 6,643 nodes with average degree 11.83.

Our algorithm's message cost consists of three parts: edge nodes identification, extended concave node identification and concave node identification. The overall message cost is $O(n)$, which translates to a per-node message cost of $O(1)$.

III. Application I: Geometric Embedding for Routing

Geometric embedding aims to place a network into a geometric space so that every node is equipped with a set of coordinates that can be used by applications. In sensor networks, one of the main uses of network embedding is geometric routing (or geographic routing), whereby data forwarding is carried out by decreasing distance to the destination. What matters in such a paradigm is only the *relative* positions of nodes, so geometric embedding tries to assign *virtual coordinates* to nodes instead of finding the true positions of nodes. The generated network may therefore have a different shape from the ground truth.

TABLE I
DETECTION RATE AND FALSE DETECTION RATE OF CONCAVE NODES

Topology	Real concave corners	Detection rate	False detection rate
Music	23	87%	19.7%
Flower	30	93.3%	17.5%
Star	8	100%	46.6%

A. Previous Solutions

There are two typical approaches to assigning virtual coordinates to nodes: landmark based approach and iterative approach.

In landmark based embedding, a number of landmark nodes are selected to serve as references, and then every other node calculates its coordinates by measuring distances to these references. Fonseca et al.'s Beacon Vector Routing (BVR) [7] protocol can be seen as a representative of this approach. In BVR, a node's coordinates are simply equal to a distance vector $\langle d_0, d_1, \ldots, d_m \rangle$, where d_i is its distance in hops to the ith landmark node. Some other similar techniques were proposed in [3], [4], [31].

Rao et al. [22] proposed perhaps the earliest algorithm for iterative coordinate assignment in sensor networks. In the algorithm, boundary nodes first determine their coordinates using a triangulation algorithm; other nodes then determine their virtual coordinates through an iterative relaxation procedure. This scheme also has difficulty adapting to complex networks [26]. Arad and Shavitt [1] proposed a Node Elevation Ad-hoc Routing (NEAR) protocol to address the concave voids problem. The aim is to make network holes more convex. In a similar spirit, GSpring [16] uses a modified spring relaxation algorithm to incrementally increase the convexity of voids in the network.

B. Our Solution

Both landmark-based and iterative embedding approaches can work well with concave topologies. However, their ability of dealing with topological complexity is limited. It was found [26] that when the network contains complex holes or boundaries, their performance will quickly drop to an unacceptable level. A natural solution to this problem is to divide the network into smaller pieces [5] [26], in the hope that each piece is more regular (e.g., containing fewer holes or concave corners), so is more friendly to the embedding algorithms. To that end, a Voronoi cell based technique [5] can be used: a number of landmark nodes are selected, each creating and leading a subnetwork called a *Voronoi cell*, defined as the set of nodes that find the leader closest (in hops) among all the landmarks. In each Voronoi cell, some method is used to assign *local coordinates* to each nodes. Routing can be done following a typical inter-cell and intra-cell paradigm. Each landmark will establish a global shortest path tree by which every node outside its cell C can route to C following a shortest path. The intra-tile routing will use greedy forwarding plus an expanding ring search routine [22] to deal with local minima.

Two questions need to be answered when realizing the landmark based coordinates assignment: Which nodes should be selected as the landmarks and how many of them do we need? For the first question, a common approach is to select the landmarks randomly among all network nodes [5] [7] [3], while for the second question, the answer is less clear in the literature: a trial and error approach seems to be the common choice in order to meet a certain performance requirement. In BVR, for example, it is suggested that a fixed proportion (e.g., 2%) of landmarks suffice to produce good routing performance; this however turns out to be not true for topologically more complex networks, as shown in [26].

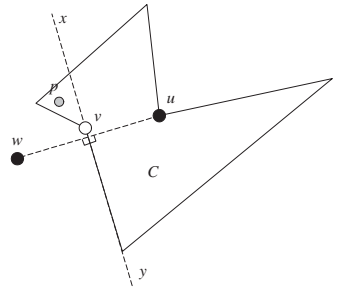


Fig. 10. A Voronoi cell cannot contain more than one concave node.

We design a new scheme that selects landmarks in a simple way, and provides a clearer answer to the second question. In principle, the scheme lets each landmark represent a concave network corner. Since the area around such a corner normally contains multiple concave nodes, our scheme chooses nodes with locally maximal (in two-hop neighborhood) betweenness centrality as landmarks. As such, the total number of landmarks selected will be close to the number of concave network corners.

To see the effect of the new scheme on routing performance, we first look at its effect on the geometry of the Voronoi cells. Consider a continuous model for the network layout in which network boundaries and holes are represented by simple polygons, the following theorem tells us that in a Voronoi cell, there is only a single concave node.

Theorem 1: Given a polygonal area that contains at least one concave node. If we set its concave nodes to be landmarks and create Voronoi cells with these landmarks, then each Voronoi cell contains exactly one concave node.

Proof: Assume a Voronoi cell C contains more than one concave nodes, see the polygon in Figure 10. Let node u be its host landmark (thus a concave node) and v be a second concave node. Let w be the landmark of C's neighboring Voronoi cell. Since v is a concave node, the line xy will split its inner angle and leave a point in C, say p, on the left hand side of xy. According to the property of Voronoi diagram, the line xy must be a perpendicular bisector of the line segment $\overline{uw}$, so p must be closer to w than to u, which contradicts with

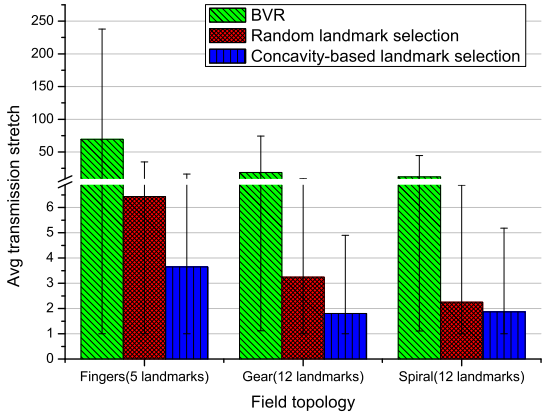


Fig. 11. Average transmission stretch (with 5th and 95th percentiles).

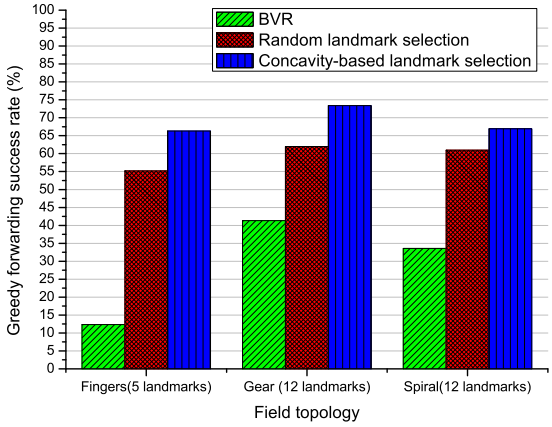


Fig. 12. Greedy forwarding success rate.

the definition of a Voronoi cell. Thus v cannot be a concave node of C. This proves the theorem. ■

The fact that each Voronoi cell contains only a single concave node means that every cell is only "mildly" complex in geometry. Previous studies and our experiments show that such a level of complexity can be well accommodated by embedding algorithms such as NoGeo, thus every Voronoi cell can be embedded with low distortion. In contrast, with random landmark selection, some Voronoi cells may end up being quite large, containing many hole corners, while others may be unnecessarily small and regular (see examples in the left column of Figure 9). Those highly complex cells are likely to cause great distortion in geometric embedding, thus making greedy forwarding difficult and bringing down the overall routing performance. In effect, our landmark selection scheme improves the embedding quality by balancing the geometric complexity of each cell and exploiting the existing algorithms' (limited) ability of dealing with network concavity.

C. Performance evaluation

We create several topologically complex networks as shown in Figure 9. The left column shows a typical Vonoroi division with random landmark selection, while the right column shows the outcome of concave node based landmark selection. For each network, the number of landmarks selected is the same for two schemes.

In each Voronoi cell C, we establish a local virtual coordinate system using the NoGeo method [22]. Let p be C's landmark node and let C's radius, R_C, be the maximum hop distance from all nodes in C to p. Define p's *disk set* as the set of nodes within R_C hops of p, then p will assign virtual coordinates to its disk set using NoGeo. [1]

A main metric used in the evaluation is *transmission stretch* [7], defined for a given node pair as the ratio of the total number of transmitted messages involved in the routing process to the shortest path hop count between the two nodes. This metric takes into account the cost of expanding ring search for dealing with local minima. Another metric is greedy forwarding success rate (GFSR) that measures how frequently the geometric routing can be accomplished without invoking the expanding ring search routine. For the random landmark selection scheme, the experimental results are the average of multiple runs for each topology case.

Figure 11 shows the average transmission stretch of the two landmark selection schemes. It can be seen that our scheme significantly reduces stretch compared to the random landmark selection scheme. For the Fingers and Gears topologies, for example, the stretch reduction is up to 44%. The figures also present the stretch results of BVR, showing extremely poor performance. This reflects how difficult it is for representative embedding algorithms to deal with the complex network topology as a whole, and that it is necessary to divide the networks into small pieces.

Figure 12 shows the GFSR of the two schemes for the various topologies. The GFSR of our scheme outperforms the random landmark selection scheme by up to 11%. This explains the stretch reduction results in Figure 11.

D. Summary

Landmark selection has been a very important issue for the performance of landmark-based applications, most notably routing [3], [5], [7], [20], [26]. By enabling *geometry-aware* landmark selection, concave nodes can effectively help improve routing performance. Utilizing the concave nodes incurs minimal effort from the application's viewpoint, thus concave nodes promise to be a useful structure for future routing design in sensor networks.

IV. Application II: Full Localization

A. Previous Solutions

Connectivity-based localization of sensor networks has been a research topic of enduring interest in the field [15], [17], [18], [21], [23]–[25], [30]. The recent work [25] shows that it is possible to localize the network based on concave nodes rather than more complicated structures such as boundaries or Delaunay triangles [15], [17]. Compared with other solutions, the proposed scheme, named *CATL*, proves to be more adaptive to various network conditions while showing equally good or superior performance. The key idea is to determine whether a path is straight or curved by checking whether it contains concave node. The straight paths are used to estimate

[1] The disk set is used instead of the cell itself because its boundary nodes are easier to determine which are needed in NoGeo. Since this disk set is usually a superset of C, some nodes may have to join the relaxation processes of multiple cells, and end up being assigned multiple virtual coordinates. We let each node use only the virtual coordinates associated with its own cell leader for routing.

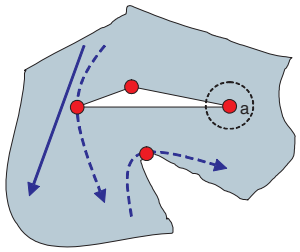


Fig. 13. Concave nodes (red circles) and shortest paths. Paths bending around the concave nodes (dashed lines) are not used for estimating Euclidean distance, while other paths (solid lines) are straight so the Euclidean distance between their endpoints can be approximated with minimum hop count.

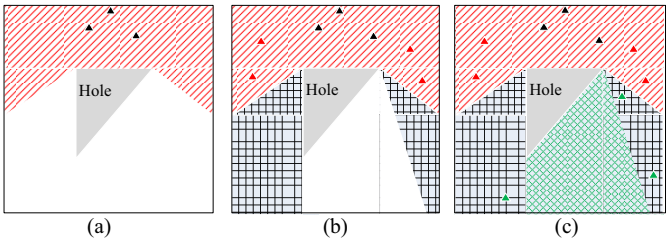


Fig. 14. Iterative localization in CATL. Concave nodes play a key role in the accurate estimation of inter-node distance and guiding the iterative multilateration process.

distance between the paths' endpoints, while the curved ones are ignored or used with a low priority (see Figure 13). The estimated distance will then be used by an iterative multilateration algorithm to localize nodes. An illustration of CATL's iterative localization is given in Figure 14, where the network is deployed in a square field with a triangular hole in the center. In the first round (Figure 14(a)), three seed nodes (small black triangles) allow the nodes in the red area to be localized; in the second round, several localized nodes will be selected as new seeds (small red triangles) to localized the grid-covered area; in the third round, some more seed nodes are chosen that allow the rest of the network to be localized.

B. Our Solution

CATL heavily relies on an expensive method to discover the concave nodes. We show that our localized method for concave nodes detection can provide a new basis for the CATL algorithm with a much reduced message cost. Our implementation basically follows CATL [25], with a number of simplifications. To start with, we manually select three non-collinear seed nodes with known positions. They are used to bootstrap the localization process. In the beginning of multilateration, only those nodes whose shortest paths to the seeds do not contain concave nodes (so they are straight) are localized; some of these newly localized nodes will then serve as new seeds and broadcast in the network, so more un-localized nodes can obtain their locations. This process continues until all the nodes are localized.

Figures 16(b), (e), (h) present the localization results of our algorithm for different topologies. The results indicate that our method can successfully support CATL's recovery of the network shape. Table II shows the location error, defined as the ratio of the distance between localized and true positions to

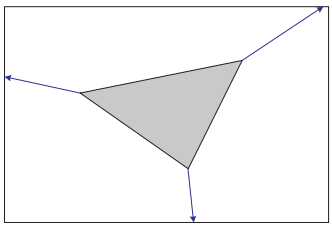


Fig. 15. Bisectors remove concave nodes in a topology, generating three convex partitions.

radio range, and average per-node message cost in comparison with CATL [25] and the DV-Hop [21] algorithms. Compared with CATL, our solution achieves comparable location accuracy, with a significantly lower message cost to find the concave nodes: the message savings for the three topologies are 48.1%, 45.9%, and 54.8%, respectively.

V. Application III: Convex Segmentation

A convex network segment means a network component whose high-level layout is convex; that is, the network topology does not contain large holes or boundary concavities. Such network segments possess a property that allows some algorithms to work to their advantage: the shortest path between any node pair is approximately straight. As can be seen from Sections IV and III, lack of this property is precisely what makes localization and geometric routing difficult in a topologically complex network. The CONVEX protocol [26] first considers the benefit of convex segmentation to routing; the work in [27] further shows that this technique can also help with global network shape recognition.

The idea of CONVEX protocol is to find *bisector paths* from concave nodes. In a continuous domain, a bisector path is a bisector line from a polygonal vertex. Since such a line can always remove a concave vertex (whose polygonal angle is greater than π), this approach can ensure that all resulting sub-areas of a given polygon are convex; see Figure 15 for an example.

CONVEX needs the knowledge of network boundaries to do the segmentation. Here we show that it is possible to accomplish the task directly from the concave nodes, without assuming the knowledge of boundaries. For each concave node p, if it has a locally maximal centrality in its two-hop neighborhood, it regards itself as representative of a network corner, and starts to establish a bisector path from itself. The protocols of establishing such paths and of partition recognition follow [26]. Figures 16 (c), (f), (i) present some results of the protocol based on our concave nodes detection. It can be seen that the obtained network segments are all approximately convex.

VI. Application IV: Boundary Detection

The goal of boundary detection is to find nodes on the network boundaries and connect them into meaningful cycles. Fekete et.al [6] proposed a simple method of boundary construction from edge nodes as we defined. It samples edge nodes in a certain density to construct a curve. Each edge node sample initiates a local flooding in search of two closest samples as its sample-neighborhood. Finally, the algorithm

TABLE II
COMPARISON OF LOCATION ERRORS AND PER-NODE MESSAGE COST.

Topology	Scheme	Avg Err	5-prtl Err	95-prtl Err	Per-node #msg
Music	Our method	1.61	0.75	2.51	**99.2**
	CATL	0.96	0.39	1.67	**191.0**
	DV-Hop	9.22	0.35	24.6	**3.0**
Flower	Our method	0.93	0.31	1.99	**153.2**
	CATL	0.88	0.32	1.65	**283.0**
	DV-Hop	2.90	0.30	9.30	**3.0**
Star	Our method	1.01	0.29	2.56	**73.2**
	CATL	1.04	0.43	2.11	**162.0**
	DV-Hop	2.75	0.19	8.36	**3.0**

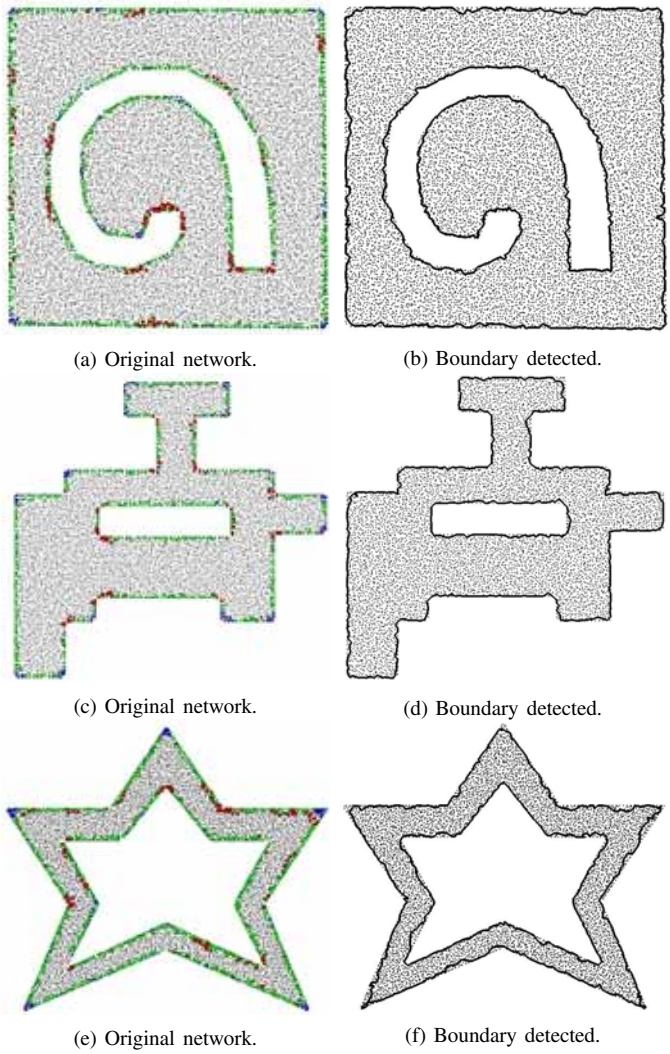
connects them by shortest paths along edge nodes to form a boundary. The scheme assumes an unrealistically high node density. Wang, Gao, and Michell [29] proposed an algorithm that uses an interesting concept *cuts* to detect boundaries. We will use their topologies to evaluate our method.

Our boundary detection algorithm is coordinated by a *coordinator*, which can be an arbitrary node. We assume this node is already identified (for example, by the smallest ID). The coordinator floods the network to establish a global shortest path tree so that it can conveniently collect information from the network. Note that the coordinator does not maintain state information for every node in the network, so it will not become a bottleneck.

Define *Critical Nodes* as a subset of nodes that have locally maximum or minimum centrality values, where "local" means a two-hop neighborhood. The critical nodes with local maximum centrality are concave nodes, while the nodes with local minimum centrality are *convex nodes*. A tight network boundary should contain a sequence of critical nodes representing the turning points of the boundary cycle. Thus connecting these critical nodes provides a reasonable way of obtaining a connected and tight network boundary.

All the critical nodes report to the coordinator, which then selects an arbitrary critical node to start a *directional edge forwarding* process. To ensure that the forwarding proceeds in a particular direction, an arbitrary edge node that is two hops away from the root is selected as the *stop node*. Only the edge nodes that are more than two hops away from the stop node can forward messages. The forwarded message travels along shortest paths between critical nodes, and the shortest paths together form a connected boundary. When the directional edge forwarding reaches the root's two-hop neighborhood, the forwarding finishes, and the root can then make up the last two hops to create a boundary cycle. After a boundary cycle has been created, all critical nodes on that boundary will report to the coordinator, which then starts to select another root node (if any exists) to detect other boundaries.

The critical nodes are mainly used to help the algorithm obtain a tight boundary cycle. Without them, the edge forwarding process may still be able to produce a boundary cycle, but the cycle might be too loose to be useful (e.g., when the edge belt is thick). Among the two types of critical nodes, the convex nodes are relatively easier to identify – they can actually be distinguished using the fact that they have locally minimum degrees. Therefore, the identification of concave nodes becomes specially important in the boundary detection algorithm.

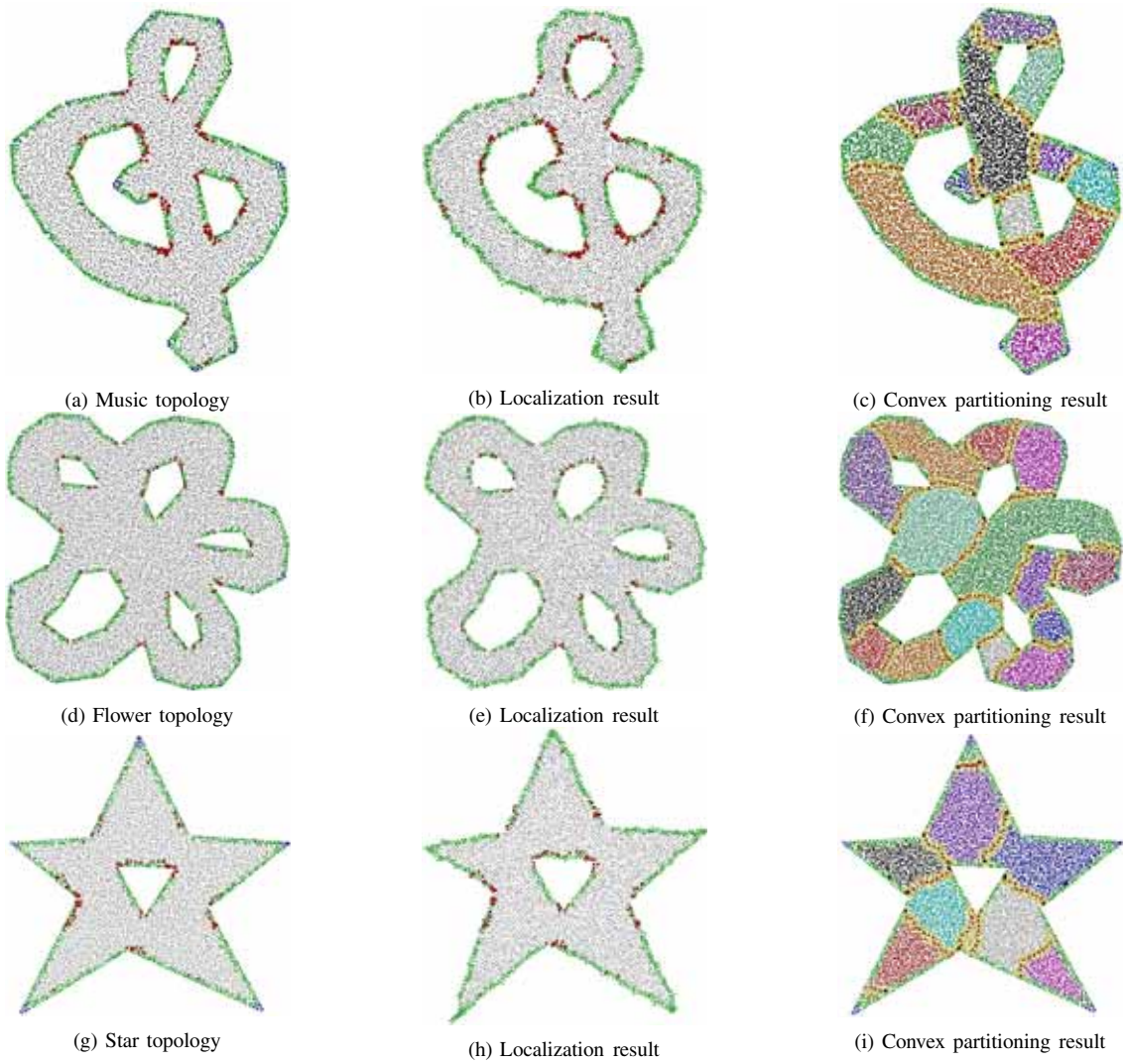


(a) Original network. (b) Boundary detected.

(c) Original network. (d) Boundary detected.

(e) Original network. (f) Boundary detected.

Fig. 17. Boundary detection results for several topologies from [29]. The concave nodes are marked with red points. The spiral shape network has 10,899 nodes with average degree is 12.3, The gun shape network has 7,559 nodes with average degree is 12.2, the double-star shape network has 6,186 nodes with average degree 12.5.



(a) Music topology (b) Localization result (c) Convex partitioning result

(d) Flower topology (e) Localization result (f) Convex partitioning result

(g) Star topology (h) Localization result (i) Convex partitioning result

Fig. 16. Several complex networks reproduced from the topology examples in [29], [30]. The red points represent the concave nodes detected. The music topology ((a), (b), (c)) has 7,058 nodes, with average node degree 12.2. The flower topology ((d), (e), (f)) has 12,777 nodes, with average node degree 12.3. The star topology ((g), (h), (i)) has 5,375 nodes, with average node degree 11.9.

Figure 17 presents the results of our method for several topologies taken from [29]. The concave and convex nodes are respectively marked in red and blue. It can be seen that our method performs very well, successfully producing a set of continuous and tight boundaries.

VII. Conclusion

In this paper we have highlighted the utility of concave nodes in geometric processing of sensor networks. Concave nodes reflect some distinguishing features of a network's geometry, and thus can often provide crucial information for more advanced geometric processing tasks or applications to optimize their performance. We have presented a method to detect concave nodes, and show through four examples the utility of such a structure. We expect that the knowledge of concave nodes lends itself to other applications of general multihop wireless networks, especially those with a large scale and complex topology.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China under Grant 61073147, Grant 61173120, Grand 61271226, and Grand 61272410; by the National Natural Science Foundation of China and Microsoft Research Asia under Grant 60933012; by a grant from RGC under the contract 615613, by a grant from NSFC/RGC under the contract N_HKUST610/11, and by a grant from ChinaCache Int. Corp. under the contract CCNT12EG01, by the Fundamental Research Funds for the Central Universities under Grant 2011QN014 and Grant 2012QN078; by the National Natural Science Foundation of Hubei Province under Grant 2011CDB044; by the Fok Ying Tung Education Foundation under Grant 132036; and by the Program for New Century Excellent Talents in University under Grant NCET-10-408 (State Education Ministry). Dr. Guang Tan's work was supported by the NSFC under Grant 61103243, Grant 61379135, Youth Innovation Promotion Association, Chinese Academy of Sciences, the Ministry of Science and Technology 863 Key Project No. 2011AA010500, and Shenzhen Overseas High-level Talents Innovation and Entrepreneurship Funds KQC201109050097A. The corresponding author is Hongbo Jiang.

REFERENCES

[1] N. Arad and Y. Shavitt, "Minimizing recovery state in geographic ad-hoc routing," in *Proc. 2006 ACM MobiHoc.*

[2] J. Bruck, J. Gao, and A. Jiang, "MAP: medial axis based geometric routing in sensor networks," in *Proc. 2005 ACM MobiCom.*

[3] Q. Cao and T. Abdelzaher, "A scalable logical coordinates framework for routing in wireless sensor networks," in *Proc. 2004 IEEE Real-Time Systems Symposium.*

[4] A. Caruso, A. Urpi, S. Chessa, and S. De, "GPS free coordinate assignment and routing in wireless sensor networks," in *Proc. 2005 IEEE INFOCOM.*

[5] Q. Fang, J. Gao, L. Guibas, V. de Silva, and L. Zhang, "GLIDER: gradient landmark-based distributed routing for sensor networks," in *Proc. 2005 IEEE INFOCOM.*

[6] S. P. Fekete, A. Kroeller, D. Pfisterer, S. Fischer, and C. Buschmann, "Neighborhood-based topology recognition in sensor networks," in *Proc. 2004 Algorithmic Aspects of Wireless Sensor Networks: First International Workshop.*

[7] R. Fonseca, S. Ratnasamy, J. Zhao, C. T. Ee, D. Culler, S. Shenker, and I. Stoica, "Beacon vector routing: scalable point-to-point routing in wireless sensornets," in *Proc. 2005 NSDI.*

[8] L. C. Freeman, "A set of measures of centrality based on betweenness," *Sociometry*, vol. 40, pp. 35–41, 1977.

[9] S. Funke, "Topological hole detection in wireless sensor networks and its applications," in *Proc. 2005 Joint Workshop on Foundations of Mobile Computing.*

[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 and Distrib. Syst.*, vol. 21, no. 5, pp. 710–721, 2010.

[11] B. Karp and H. T. Kung, "GPSR: greedy perimeter stateless routing for wireless networks," in *Proc. 2000 ACM MobiCom.*

[12] D. Kempe, J. Kleinberg, and A. Demers, "Spatial gossip and resource location protocols," *J. ACM*, vol. 51, no. 6, Nov. 2004.

[13] A. Kroller, S. Fekete, D. Pfisterer, and S. Fischer, "Deterministic boundary recognition and topology extraction for large sensor networks," in *Proc. 2006 ACM-SIAM SODA.*

[14] F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger, "Geometric ad-hoc routing: of theory and practice," in *Proc. 2003 ACM International Symposium on the Principles of Distributed Computing.*

[15] S. Lederer, Y. Wang, and J. Gao, "Connectivity-based localization of large scale sensor networks with complex shape," in *Proc. 2008 INFOCOM.*

[16] B. Leong, B. Liskov, and R. Morris, "Greedy virtual coordinates for geographic routing," in *Proc. 2007 IEEE ICNP.*

[17] M. Li and Y. Liu, "Rendered path: range-free localization in anisotropic sensor networks with holes," in *Proc. 2007 MobiCom.*

[18] H. Lim and J. C. Hou, "Localization for anisotropic sensor networks," in *Proc. 2005 IEEE INFOCOM.*

[19] J. Newsome and D. Song, "Gem: graph embedding for routing and data-centric storage in sensor networks without geographic information," in *Proc. 2003 ACM SenSys.*

[20] A. Nguyeny, N. Milosavljevic, Q Fang, J. Gao, and L. J. Guibas, "Landmark selection and greedy landmark-descent routing for sensor networks," in *Proc. 2008 IEEE INFOCOM.*

[21] D. Niculescu and B. Nath, "DV based positioning in ad hoc networks," *J. Telecom. Systems*, 2003.

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

[23] Y. Shang and W. Ruml, "Improved MDS-based localization," in *Proc. 2004 IEEE INFOCOM.*

[24] Y. Shang, W. Ruml, Y. Zhang, and M. P. J. Fromherz, "Localization from mere connectivity," in *Proc. 2003 ACM MobiHoc.*

[25] 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. 2010 ACM MobiHoc.*

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

[27] G. Tan, H. Jiang, J. Liu, and A.-M. Kermarrec, "Convex partition of large-scale sensor networks: algorithms and applications," technical report SIAT-CAS 2011-3-27.

[28] M. Tsai, H. Yang, and W. Huang, "Axis-based virtual coordinate assignment protocol and delivery-guaranteed routing protocol in wireless sensor networks," in *Proc. 2008 IEEE INFOCOM.*

[29] Y. Wang, J. Gao, and J. S. B. Mitchell, "Boundary recognition in sensor networks by topological methods," in *Proc. 2006 ACM MobiCom.*

[30] Y. Wang, S. Lederer, and J. Gao, "Connectivity-based sensor network localization with incremental Delaunay refinement method," in *Proc. 2009 INFOCOM.*

[31] Y. Zhao, B. Li, Q. Zhang, Y. Chen, and W. Zhu, "Hop ID based routing in mobile ad hoc networks," in *Proc. 2005 IEEE ICNP.*

[32] X. Zhu, R. Sarkar, and J. Gao, "Shape segmentation and applications in sensor networks," *ACM Trans. Sensor Netw.*, vol. 5, no. 2, pp. 1–32, 2009.

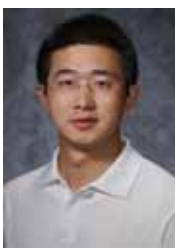
[33] M. Duckham, L. Kulik, M. Worboys, and A. Galton, "Efficient generation of simple polygons for characterizing the shape of a set of points in the plane," *Pattern Recognition*, 2008.

Shengkai Zhang received the M.S. degree from Huazhong University of Science and Technology in 2012. For now His research concerns wireless networking, especially algorithms and architectures for sensor networks.

Guang Tan is currently an Associate Researcher at Shenzhen Institute of Advanced Technology (SIAT), Chinese Academy of Sciences, China, where he works in the area of distributed systems and networks. He received his BS degree from the Chongqing University of Posts and Telecommunications, China, in 1999, MS degree from the Huazhong University of Science and Technology, China, in 2002, and Ph.D. degree in computer science from the University of Warwick, U.K., in 2007. From 2007 to 2010 he was a postdoctoral researcher at INRIA-Rennes, France.

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.

Bo Li is a Professor in the Department of Computer Science and Engineering, Hong Kong University of Science and Technology. His recent research interests include: large-scale content distribution in the Internet, Peer-to-Peer media streaming, the Internet topology, cloud computing, green computing and communications. He has been an editor or guest editor for 17 IEEE/ACM journals and magazines, and he was involved in organizing 50 conferences. He was the Co-TPC chair for IEEE Infocom'04. He was a distinguished lecturer in IEEE Communications Society (2006-2007). He is a Fellow of IEEE for "contribution to content distributions via the Internet".



Chonggang Wang received his Ph.D. 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 has published more than 80 journal/conference articles and book chapters. He is a senior member of the IEEE.