CASE: Connectivity-based Skeleton Extraction in Wireless Sensor Networks

Hongbo Jiang[1]   Wenping Liu[1]   Dan Wang[2]   Chen Tian[1]   Xiang Bai[1]   Xue Liu[3]   Ying Wu[4]   Wenyu Liu[1]

[1]Department of EIE, Huazhong University of Science and Technology, Hubei, China
[2]Department of Computing, The Hong Kong Polytechnic University, Hong Kong
[3]School of Computer Science, McGill University, Quebec, Canada
[4]EECS department, Northwestern University, Evanston, IL 60208
[1]{hongbojiang2004,openheart611,alexandretian,xiang.bai}@gmail.com,liuwy@mail.hust.edu.cn
[2]csdwang@comp.polyu.edu.hk,[3]xueliu@cs.mcgill.ca,[4]yingwu@ece.northwestern.edu

Abstract—**Many sensor network applications are tightly coupled with the geometric environment where the sensor nodes are deployed. The topological skeleton extraction has shown great impact on the performance of such services as location, routing, and path planning in sensor networks. Nonetheless, current studies focus on using skeleton extraction for various applications in sensor networks. How to achieve a better skeleton extraction has not been thoroughly investigated.**

There are studies on skeleton extraction from the computer vision community; their centralized algorithms for continuous space, however, is not immediately applicable for the discrete and distributed sensor networks. In this paper we present CASE: a novel Connectivity-bAsed Skeleton Extraction algorithm to compute skeleton graph that is robust to noise, and accurate in preservation of the original topology. In addition, no centralized operation is required. The skeleton graph is extracted by partitioning the boundary of the sensor network to identify the skeleton points, then generating the skeleton arcs, connecting these arcs, and finally refining the coarse skeleton graph. Our evaluation shows that CASE is able to extract a well-connected skeleton graph in the presence of significant noise and shape variations, and outperforms state-of-the-art algorithms.

I. INTRODUCTION

Sensor networks today are widely used as they are able to capture the phenomena of the physical world that were originally difficult or impossible to obtain by traditional techniques. Existing examples include disaster relief [5], habitat monitoring [17], battlefield surveillance [9], etc. For these applications the sensor networks usually cannot be deployed well-planned in advance as conventional networks. The geographical locations and deployment methods may vary greatly, and the topology of the sensor networks is affected by such factors as obstacles, deployment randomness, holes, etc. The shape of the sensor network hardly conforms to simple shapes such as a square or a disk; which are mainly used in research studies [10], [15]. As a concrete example, a butterfly-shape sensor network (e.g., Fig. 1 (a)) is representative for airport terminals or train maps shown in Fig. 2.

The performance of the sensor network, e.g., the efficiency of data routing, localization, and path planning, heavily depends on the distribution of the sensors and the overall network topology. Some previous studies have shown that utilizing the topological *skeleton* (or so-called *medial axis*) information of the sensor network can greatly improve routing performance [2] over other routing scheme such as GPSR. Followups have used skeleton information for landmark selection to provide location service [13], segmentation [19], and navigation algorithms [4]. We argue that geometric skeleton information can be useful far more than routing, location, segmentation, and path designs. For instance, for a mobility-assisted sensor network for field coverage, the skeleton can be used for path planning since the nodes in the skeleton represent a medial axis and thus cover most area where all nodes are observing. Thus, to understand the characteristics of the sensor topology and extract some important geometric information are essential to the sensor networks.

The previous works focus on applying the skeleton information for various purposes in the sensor networks. The performance of the *skeleton extraction*, however, is not the concentration of these studies. In this paper, we present CASE: a novel distributed Connectivity-bAsed Skeleton Extraction algorithm which can better represent the topological shape of the sensor network than previous studies while incurring comparable communication cost. We illustrate the comparison by a simple example in Fig. 1. We see that Fig. 1(c) shows the medial axis graph computed by [2], [3] and Fig. 1(d) is the skeleton graph computed by CASE. Clearly, CASE grasps better geometric information, and avoids possible loops.

Skeleton or medial axis extraction schemes have been studied in computer vision [6] and graphics [16] communities. While our proposed algorithm carries some of their advantages, there are some intrinsic differences due to the unique characteristics of the wireless sensor networks; for example, the sensor networks are discrete and the skeleton have to be computed with local connectivity information. We face numerous challenges in designing an efficient skeleton extraction algorithm. *First*, sensor nodes randomly deployed often have no location information or distance information. This makes topology recognition difficult as various techniques widely used in computer vision fields for skeleton extraction using centralized computation are not suitable for a sensor network. *Second*, in a discrete network, the definition of the
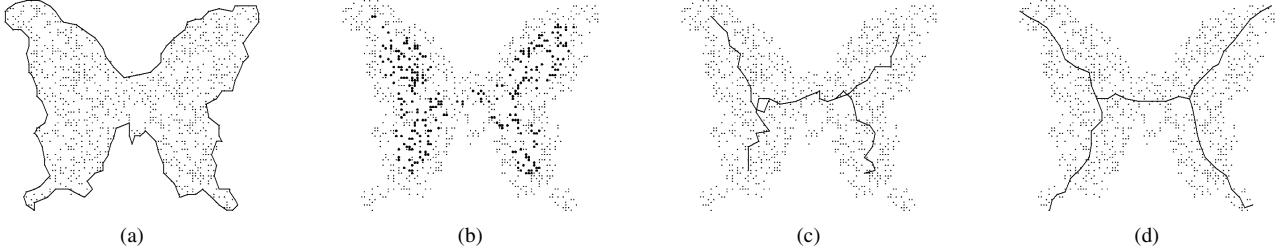
Fig. 1.   An example network. (a)Boundary by the approach in [18]; (b)Medial axis points according to the definition in [2], [3]; (c)Medial axis graph computed by MAP in [2], [3]; (d)Skeleton graph computed by CASE.

distance between skeleton nodes has to use hop-count distance instead of Euclidean distance. This will result in noise, e.g., incorrect estimation of boundaries. It has been shown that a little noise or deformation at the boundary often seriously disturb the topology of the skeleton graph [2]. As a result, many well-known algorithms [1], [6], [11] applied in other fields to extract skeleton cannot be used directly. Effective elimination of the unstable segments in skeleton to keep the genuine geometric features is challenging. *Third*, even after the skeleton nodes are identified, connecting all nodes in a proper way is not as straightforward as that in the continuous case. In many cases, there are not enough skeleton nodes that can be extracted to construct a connected skeleton. In addition, the paths generated to connect the skeleton nodes may either not be the shortest paths or introduce cycles as shown in Fig 1(c).
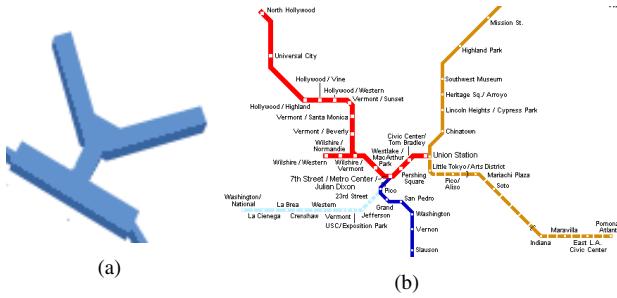

Fig. 2.   (a)Chicago Airport Terminal 2; (b)A part of Los Angeles train map.

Towards taming these challenges, in this paper, we propose a novel skeleton extraction approach–CASE, and make the following contributions:

- We present the design and implementation of a practical and distributed skeleton extraction algorithm which requests only connectivity information and preserves the topology well.
- Based on boundary partition techniques, our proposed algorithm is robust against noise, and thus capable of keeping the genuine geometric features of the sensor network.
- Our work has implications for topology recognition in general. The results suggest that the user should keep track of the importations points such as skeleton joints and corner points in order to make better understanding of the topology.

The remaining part of this paper proceeds as follows: Section II is devoted to the skeleton extraction algorithm. We show the preliminary evaluation result in Section III. Section IV presents related work and finally, Section V summarizes the paper and future plans.

II. SKELETON EXTRACTION ALGORITHM

Our skeleton extraction algorithm is derived from the boundary, but boundary construction is out of the scope of this work. Numerous recent studies, [15], [18], have provided boundary detection and recognition algorithms. We bear this in mind and state that many of these algorithms can be used in conjunction with our approach. That is, we assume the boundary points are given as an input for our algorithm. Overall, CASE includes five main building blocks:

1) Find corner points on the boundary such that the whole boundary can be decomposed into a finite number of boundary segments.
2) Identify the skeleton points such that they form a connected component.
3) Generate a set of skeleton arcs by connecting the skeleton points.
4) Generate a coarse skeleton result by connecting skeleton arcs and corner points.
5) Refine the coarse result to obtain the final skeleton.

A. Boundary Segments

The first step of CASE is to decompose the boundary ∂P into a finite number of open boundary segments. To that end, we first identify the corner points on the boundary.

While the corner point can be defined in terms of the curvature of the local shape in a continuous case, it is not straightforward to use it in discrete networks. We formalize here the discrete curvature as a simplification criterion, which is able to represent geometric shape well. For any boundary point p, we refer to its h-hop neighbors as $N_h(p)$. Let the maximal hop-count between two nodes in $N_h(p)$ be denoted by $M_h(p)$. It is noted that, in this paper, we measure the distance between two points by the hop-count. Intuitively, for an inner point or some boundary points, $M_h(p)$ is often $2h$. Thus, we define the discrete curvature by $\rho_p = \max_{h=1,\cdots,H} M_h(p)/2h$ (in experiments, we found that $H = 3$ or 4 provides good results in most cases). For a point $p \in \partial P$, if ρ_p is less than a given threshold δ_ρ, say 0.5 in our implementation results), the point p is defined to be a *corner point*. With a set of cornet points, marked with thick black circles in Fig. 3(a), the

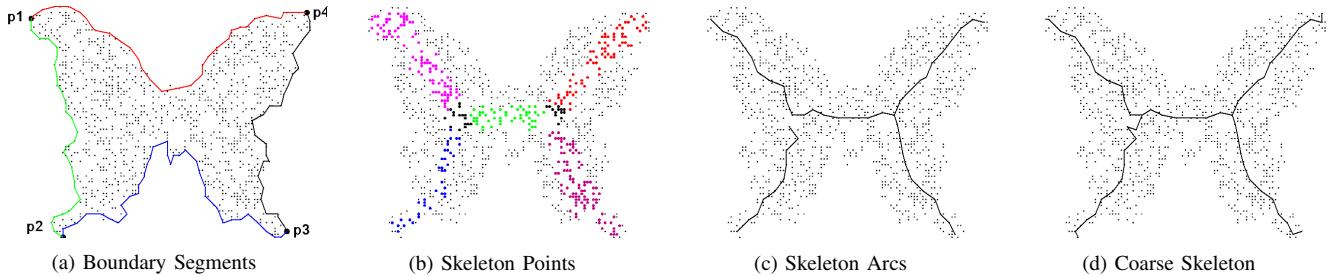| (a) Boundary Segments | (b) Skeleton Points | (c) Skeleton Arcs | (d) Coarse Skeleton |

Fig. 3. Skeleton extraction algorithm of an example sensor network with the average degree of 20.9. The final refined skeleton is shown in Fig. 1(d).

boundary ∂P can be decomposed into four open *boundary segments* $p1p2$, $p2p3$, $p3p4$, and $p1p4$ shown in Fig. 3(a).

B. Skeleton Points Identification

As mentioned in [12], skeleton S can be decomposed into skeleton arcs which are composed of skeleton points. To that end, we turn to identifying the skeleton points.

The definition that the nodes who have exact equal distances to at least two boundary segments are skeleton points [3] is not suitable in many applications, especially for a sparse sensor network. We observed that the skeleton points based on this definition will be quite few in sensor networks with a moderate average degree. It is noted that in a discrete network, there is no guarantee that all skeleton points constitute a connected component. Thus, we refer to *skeleton points* associated with two neighbor boundary segments, say C_i and C_j, as those nodes such that: 1) the difference of the absolute distances to two nearest boundary segments is less than a given threshold, denoted by δ_p; 2) the component composed of those nodes, denoted by $S(C_i, C_j)$, is a connected component. Via this definition, parts of skeleton points can be well connected into skeleton arcs described in next section. Our process to identify skeleton points works as follows. First all the nodes who have equal distances to C_i and C_j definitely belong to S. Then any node $p \in S$ initiates a flooding to check whether S is a connected component or not. If not, we iteratively increase δ_p value by one to include more nodes into S until $S(C_i, C_j)$ is a connected component.

As a result, the set of skeleton points after the procedure by resetting δ_p is obtained, denoted by $S(P)$ and shown in Fig. 3(b) where the skeleton points associated with different boundary segments are marked with different color thick circles. This result shown in Fig. 3(b), obviously, is much better than that using the method in [3], shown in Fig. 1(b). Here the better results are obtained due to: 1) the set of our skeleton points are connected; 2) by identifying skeleton points via being coupled with two neighbor boundary segments, CASE is able to remove many undesirable nodes introduced by MAP [3]. Unlike the continuous case, we refer to the *joint points* as the nodes who belong to at least two component $S(C_i, C_j)$ and $S(C_i, C_k)$ in a discrete network, that is, $S(C_i, C_j) \cap S(C_i, C_k)$. The joint points are marked with thick black circle in the Fig. 3(b).

C. Skeleton Arcs Generation

Since skeleton S can be decomposed into a set of skeleton arcs, an important step in the construction of the skeleton is to generate skeleton arcs by connecting the skeleton points $S(P)$ in a correct way. Unlike previous work [3], we do not claim to include all skeleton points into our skeleton arcs. Instead, we construct the skeleton arcs based on the fact that parts of skeleton points are enough to construct a good approximation of the theoretical skeleton in a discrete network.

We select such two nodes x' and y' that they have longest length path in the component corresponding to two segments C_i and C_j, that is, $(x', y') = argmax_{x,y \in S(C_i, C_j)} \hat{dist}(x, y)$. We call the *diametral path* [6] from x' to y' the longest path in $S(C_i, C_j)$. Here we shed light on the means to calculate the distance here $\hat{dist}(\cdot, \cdot)$. Each node $p \in S(C_i, C_j)$ initiates a flooding with its own ID and a counter to indicate how many hops the message has traveled, as well as each node ID across this path. This flooding process is limited within the connected component $S(C_i, C_j)$. For p's all neighbors, those nodes who have exactly equal distances to C_i and C_j begin with forwarding the message and increment the counter. For those nodes whose distance to C_i and C_j is unequal, they then forward the flooding message. By doing so, we allow to calculate the distance between two nodes. Meanwhile, the path between two nodes are recorded and this path is almost traveling those nodes who have equal distances to C_i and C_j. Usually, only a joint point $p' \in S(C_i, C_j) \cap S(C_i, C_k)$ need to initiate a flooding to find the other end-point with longest distance between them. That is, when there exists joint point in the component, we reduce the traffic cost greatly. Unlike the continuous case [11], it is noted that the joint points $S(C_i, C_j) \cap S(C_i, C_k)$ may contain many nodes as shown in Fig. 3(b) where joint points are marked with thick black circles. In this case, our idea is to selected the node closest to the center of $S(C_i, C_j) \cap S(C_i, C_k)$. Hence, the joint point with the largest degree in the sub-graph $S(C_i, C_j) \cap S(C_i, C_k)$ is selected to be the end-point of the skeleton arc.

So far we have obtained two end-points of the longest path in the sub-graph $S(C_i, C_j)$. One characteristic of this path is that all nodes on the path are skeleton points. We call this path to be *skeleton arc*, denoted by $A(C_i, C_j)$. For every pair of boundary segments C_i and C_j, one skeleton arc can be extracted accordingly, shown in Fig.3(c). Next we discuss how to connect all skeleton arc in a right way such that a coarse

skeleton graph is obtained.

D. Coarse Skeleton: Connecting Skeleton Arcs and The Corner Points

While in a continuous case the skeleton S can be seamlessly decomposed into a set of skeleton arcs which are intrinsically connected, that is not always the case in a discrete sensor network. Hence, an additional step of CASE is to connect all the skeleton arcs and corner points, in order to generate a *coarse* skeleton, as shown in Fig. 3(d).

For two skeleton arcs A_1 and A_2 we refer to $\hat{dist}(A_1, A_2) = \min \hat{dist}_{x \in A_1, y \in A_2}(x, y)$ as the distance of these two arcs. In a discrete network, it is possible that two close skeleton arcs, A_1 and A_2, are not neighbor, that is, $\hat{dist}(A_1, A_2) > 1$. It is known that each skeleton corresponds to boundary segments. When these two skeleton arcs correspond to the same boundary segment, these two arcs are supposed to be connected. In this case, we find two nearest points x' and y' where $\hat{dist}(x', y') = \hat{dist}(A_1, A_2)$. To find a path from x' to y', the approach to flooding is similar to that in Section II-C. Again, during flooding, the path is based on the best-effort style to travel those nodes who have similar distance to different boundary segments.

Another step is to connect the corners and skeleton arcs to extract better skeleton used for topology recognition. To that end, each corner point initiates a flooding to find a path to connect its closest arc's end-point.

E. Coarse Skeleton Refinement

A careful investigation of above connecting method reveals that this coarse skeleton often cannot be very useful for topology recognition in practice. First, based on the flooding process in Section II-C and Section II-D, the path from one end-point to another skeleton arc is not necessarily the shortest path. Fig. 4(a) shows that when $p3$ is connected to $p4$ despite the fact that $p3$ is a neighbor of $p5$. Second, unlike the continuous case, the cycle could be obtained when we connect two skeleton arcs, as shown in Fig. 4(b). These two problems were also observed in [3].
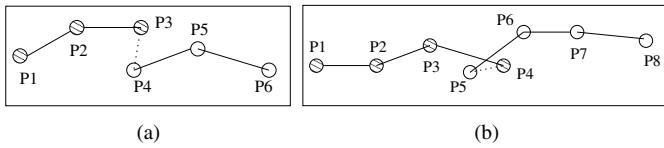

Fig. 4. Some situations where refinement is required.

To overcome these disadvantages, we randomly select one corner point to initiate a final flooding along the coarse skeleton graph. We argue that it does not introduce much traffic since the flooding is limited within the coarse skeleton. Each node on the coarse skeleton only forwards the message once. For example, when node $p3$ in Fig. 4(a) receives the message from $p2$, it forwards to $p4$ and $p5$. In this case, when $p4$ tries to forward the message to $p5$, node $p5$ will discard the message but only record its parent to be $p3$. As a result, node $p4$ is removed from the skeleton and the new path is $p1p2p3p5p6$.

As for the case shown in Fig. 4(b), node $p3$ forwards the message to $p5$ and $p4$, but only the first message, say from $p4$, will be sent to $p6$ and $p6$ records its parent to be $p4$ as well. Thus, the new path is $p1p2p3p4p6p7p8$ (noted that the new path could be $p1p2p3p4p5p7p8$ as well due to the random fashion of the flooding). Finally, the refined skeleton is obtained after the last refinement, as shown in Fig. 1(d).

III. SIMULATIONS

We have implemented a simulator to conduct a series of simulations on various simulated communication graph topologies. We compared CASE with state-of-the-art algorithms in [2], [3]. Due to space limitation, we only present some representative results in this section.

Fig. 5 shows results on a network with many small deformations on the boundary. This network contains 1,714 sensor nodes and the average degree is 5.68. To identify medial axis nodes using definition in [2], [3], we set the threshold to be four hops, in order to disregard unstable nodes whose two nearest boundary nodes are too close. Fig. 5(a) depicts the identified medial axis nodes. Obviously, due to deformations on the boundary, many unstable nodes are regarded but have no contribution for skeleton extraction. That is, the deformations on the boundary incur significant noise on medial axis nodes. Besides, the component composed of all medial axis nodes is not connected and thus connecting these nodes becomes challenging. Fig. 5(b) shows the identified skeleton points computed by CASE where $\delta_\rho = 0.5$. These skeleton points are connected and exhibit the significant pattern of the shape as we identify them via boundary segments (one segment is marked with red line and the other is marked with green line). The skeleton graph is accordingly well extracted as shown in Fig. 5(c).

IV. RELATED WORK

Topology recognition is crucial to many sensor network applications. Boundary detection and skeleton extraction are widely used for topology recognition in sensor networks. The first category, boundary detection, focuses on detecting nodes on the inner or outer boundaries. Fang *et al.* [7] proposed a simple algorithm built on the idea that a packet can only get stuck at a node on hold boundaries, while assuming the nodes know their geographical location information. Fekete *et al.* [8] proposed an algorithm based on the observation that the boundary nodes often have much lower degree than inner nodes. Without the assumption of location geographical information, Ghrist *et al.* [10] proposed a centralized algorithm to detect holes via homology.

For the second category, little has been done on skeleton extraction even though it is considered to be an important step for many other sensor applications. Lin *et al.* [14] proposed a dynamic medial axis model that represents shapes and changes of shapes in a geometric space. One work that is close to our work is MAP [2], [3]. They considered to define skeleton points using hop count such that nodes with equal distances to two closest boundary nodes can be identified to skeleton

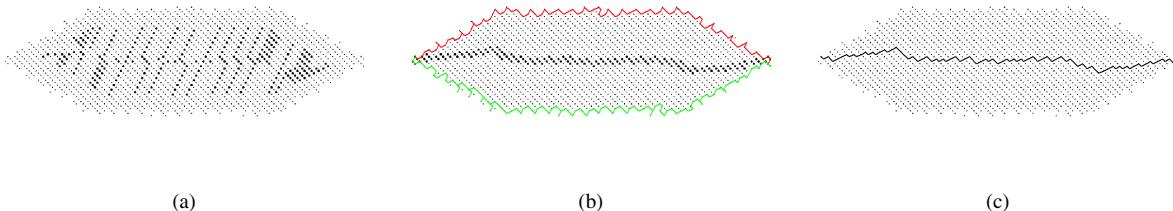(a)   (b)   (c)

Fig. 5.   Comparison on an example network with many small deformations on the boundary. (a)Medial nodes by the method in [2], [3]; (b)Skeleton points and segments by CASE; (c)Skeleton graph by CASE.

points. In addition, connecting these skeleton points provides a construction of the skeleton graph. The differences of our work from theirs are as follows. *First*, and most importantly, we partition the boundary according to the corner points. Thus global information, in stead of local significance of the shape, is maintained. This makes our algorithm more stable under small deformations. This is partially motivated by the process of Discrete Curve Evolution [1]. *Second*, MAP does not guarantee that the skeleton points it identifies consist a connected component, and thus leads to the problem where the shortest path between two nodes may not represents the shape of the geometric environment. Instead, our algorithm preserves the shape via constructing a connected component using skeleton points and thus the path in this component is able to represent the exact shape of the environment. *Third*, we refine the coarse skeleton graph using local flooding on it and achieve better results.

V. Conclusion

We have presented CASE: a novel distributed algorithm for skeleton extraction in sensor networks. This algorithm requests the connectivity information only; and it does not need in advance knowledge of location information and distance information, which might not be realistic in practice. We first identify the skeleton points by partitioning the boundary. These skeleton points are then connected to form the skeleton arcs, each of which corresponds to two boundary segments. We connect all the skeleton arcs and conduct a refinement to achieve the final skeleton. We have demonstrated that CASE provides more accurate skeleton results, and is robust against boundary variations. It outperforms the state-of-the-art algorithms under various configurations.

We are interested in several directions in the future. First, we seek more efficient algorithms to reduce the traffic overhead of our extraction technique. In particular, an approximation of skeleton graph could be applicable. Second, we would like to evaluate other boundary detection techniques for our skeleton extraction algorithm. We believe improvement is possible when other approaches are adopted. Third, evaluation of CASE on larger scale networks will be carried out.

Acknowledgment

References

[1] X. Bai, L. J. Latecki, and W. Liu. Skeleton pruning by contour partitioning with discrete curve evolution. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 29(3), 2007.
[2] J. Bruck, J. Gao, and A. A. Jiang. Map: Medial axis based geometric routing in sensor networks. In *Proceedings of ACM MOBICOM*, 2005.
[3] J. Bruck, J. Gao, and A. A. Jiang. Map: Medial axis based geometric routing in sensor networks. *Wireless Networks*, 13(6), 2007.
[4] C. Buragohain, D. Agrawal, and S. Suri. Distributed navigation algorithms for sensor networks. In *Proceedings of IEEE INFOCOM*, 2006.
[5] E. Cayirci and T. Coplu. Sendrom:sensor networks for disaster relief operations management. *Wireless Networks*, 13(3), 2007.
[6] W.-P. Choi, K.-M. Lam, and W.-C. Siu. Extraction of the euclidean skeleton based on a connectivity criterion. *Pattern Recognition*, 36, 2003.
[7] Q. Fang, J. Gao, and L. Guibas. Locating and bypassing routing holes in sensor networks. In *Proceedings of Mobile Networks and Applications*, 2006.
[8] S. P. Fekete, A. Kroller, D. Pfisterer, S. Fischer, and C. Buschmann. Locating and bypassing routing holes in sensor networks. In *Proceedings of International Workshop on Algorithmic Aspects of Wireless Sensor Networks*, 2004.
[9] M. Fennell and R. Wishner. Battlefield awareness via synergistic SAR and MTI exploitation. *IEEE Aerospace and Electronic Systems Magazine*, 13(2), 1998.
[10] R. Ghrist and A. Muhammad. Coverage and hole-detection in sensor networks via homology. In *Proceedings of IEEE IPSN*, 2004.
[11] P. Golland and E. Grimson. Fixed topology skeletons. In *Proceedings of IEEE CVPR*, 2000.
[12] L. J. Latecki and R. Lakamper. Shape similarity measure based on correspondence of visual parts. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(10), 2000.
[13] S. Lederer, Y. Wang, and J. Gao. Connectivity-based localization of large scale sensor networks with complex shape. In *Proceedings of IEEE INFOCOM*, 2008.
[14] L. Lin and H. Lee. A dynamic medial axis model for sensor networks. In *Proceedings of IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*, 2007.
[15] O. Saukh, R. Sauter, M. Gauger, P. J. Marron, and K. Rothernel. On boundary recognition without location information in wireless sensor networks. In *Proceedings of IPSN*, 2008.
[16] S. Schaefer and C. Yuksel. Example-based skeleton extraction. In *Proceedings of Eurographics Symposium on Geometry Processing*, 2007.
[17] R. Szewczyk, E. Osterweil, J. Polastre, M. Hamilton, A. Mainwaring, and D. Estrin. Habitat monitoring with sensor networks. *Communications of the ACM*, 47(6), 2004.
[18] Y. Wang, J. Gao, and J. S. Mitchell. Boundary recognition in sensor networks by topological methods. In *Proceedings of ACM MOBICOM*, 2006.
[19] X. Zhu, R. Sarkar, and J. Gao. Shape segmentation and applications in sensor networks. In *Proceedings of IEEE INFOCOM*, 2007.