Medial Axis Construction and Applications in 3D Wireless Sensor Networks

Su Xia, Ning Ding, Miao Jin, Hongyi Wu, and Yang Yang

***Abstract*—The medial axis of a shape provides a compact abstraction of its global topology and a proximity of its geometry. The construction of medial axis in two-dimensional (2D) sensor networks has been discussed in the literature, in support of several applications including routing and navigation. In this work, we first reveal the challenges of constructing medial axis in a three-dimensional (3D) sensor network. With more complicated geometric features and complex topology shapes, previous methods proposed for 2D settings cannot be extended easily to 3D networks. Then we propose a distributed algorithm with linear time complexity and communication cost to build a well-structured medial axis of a 3D sensor network without knowing its global shape or global position information. Furthermore we apply the computed medial axis for safe navigation and distributed information storage and retrieval in 3D sensor networks. Simulations are carried out to demonstrate the efficiency of the proposed medial axis-based applications in various 3D sensor networks.**

I. Introduction

The medial axis of a shape is the set of all points that have more than one closest point on the boundary of the shape. For a given shape, its medial axis provides a compact abstraction of its global topology and a proximity of its geometry. The medial axis of a two-dimensional (2D) sensor network has been discussed and applied for applications including routing and navigation [1], [2]. In [1], an approximated medial axis of a 2D sensor network is constructed and represented compactly by a graph with size proportional to the number of the geometric features of the network field. A greedy routing scheme with guaranteed packet delivery and load balance is then proposed with local decisions guided by the computed medial axis. In [2], the medial axis of a distributed 2D sensor network is dynamically maintained to provide guidance for users to escape from dangerous areas.

While most earlier studies assume sensor networks on a plane, there has been increasing interest in deploying wireless sensors in three-dimensional (3D) space for applications such as underwater reconnaissance, environmental monitoring and exploration [3]–[15]. With significantly higher complexity in geometric features and topology shapes, a 3D sensor network needs more urgently the guidance provided by a medial axis for numerous applications including but not limited to 3D routing, 3D navigation, and data storage and retrieval. Although the construction of medial axis has been discussed for 2D sensor networks, it is anti-intuitively nontrivial to extend it to 3D settings. Next we show the fundamental challenges for constructing medial axis in 3D sensor networks.

This work is partially supported by National Science Foundation under Award Number CNS-1018306. The authors are with the Center for Advanced Computer Studies, University of Louisiana at Lafayette, Lafayette, LA.

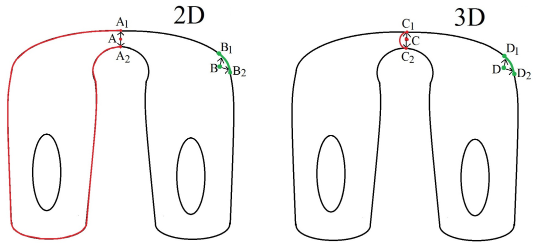


Fig. 1. The true medial axis points are mistakenly considered as noises when simply extending a 2D medial axis algorithm to 3D. (a) A 2D sensor network. (b) The cross-section of a 3D network.

A. Challenges in 3D Networks

An algorithm for constructing medial axis in 2D sensor networks has been proposed in [1]. In a nutshell, each boundary node floods a control packet over the network. Upon receiving such flooding packets, an internal node can discover its shortest distance (in hops) to the boundary and the corresponding closest boundary node. According to the definition of medial axis, an internal node is identified as part of the medial axis if it has two or more closest boundary nodes. However, noises exist in discrete sensor networks, due to the lack of accurate distance information. For example, Node B in Fig. 1(a) can be misinterpreted as a medial axis node because it has equal hops to two closest boundary nodes (i.e., B_1 and B_2). To filter out such noise, the algorithm considers the hop distance between B_1 and B_2 along the boundary (i.e., the distance of the shortest path between B_1 and B_2 along the boundary, which is marked as green color). Node B is deemed a non-medial axis node if the distance is less than a threshold. On the other hand, a true medial axis node (e.g., Node A) will not be filtered out, because it has at least two closest boundary nodes (e.g., A_1 and A_2) that are well separated along the boundary (i.e., the shortest path along the boundary is long, which is marked as red color). The filtering process is critical for the success of the medial axis algorithm.

At the first glance, it appears straightforward to extend this approach to 3D networks. However, it is surprisingly hard for such an approach to distinguish true medial axis nodes and noises based on hop distance only in 3D. Fig. 1(b) illustrates a cross-section of a 3D sensor network. When the same filtering strategy discussed above is applied here, Node C will be considered as noise, because the shortest path between Node

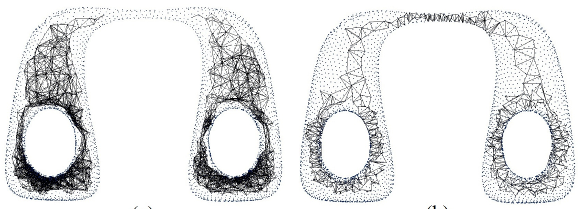


Fig. 2. (a) A disconnected and very noisy medial axis computed by an extension of the 2D hop count-based method [1] on a 3D sensor network. (b) A connected medial axis computed with the proposed UTC mesh structure-based method on the same network. Edges represent communication links between computed medial axis nodes.

C's two closest boundary nodes (i.e., C_1 and C_2) along the boundary goes around a thin pipe and thus is very short. Clearly, it becomes impossible to distinguish Node C (a true medial axis node) and Node D (which is noise) based on hop distance only in a 3D domain. Fig. 2(a) shows the result based on a simple extension of the 2D method. The medial axis is disconnected even under a small threshold which is in fact too small to filter out noises close to the boundary surface.

A deeper investigation reveals the intrinsic difficulties in constructing medial axis of 3D shapes. The structure of the medial axis of a 2D planar shape is well-understood. It consists of curve segments bounded by either an end-point corresponding to a curvature extremum on the curve, or by a junction point where three branches meet [16]. On the contrary, the structure of the medial axis of a 3D shape involves greater complexity. It consists of sheets, seams, and junctions [17]. A sheet is the bisector of two boundary elements, represented as a trimmed quadric surface; a seam is an algebraic space curve defined by the intersection of two or more sheets; and a junction point is the intersection of three or more sheets, or a sheet and a seam. Fig. 3 illustrates the medial axis of a simple 3D shape. The medial axis of a 3D shape is not stable as a structure since a small perturbation in a 3D shape can cause a relatively large change of its medial axis [17], [18]. It is difficult to accurately compute the medial axis due to numerical instability associated with the computations [19], [20].

B. Constructing Medial Axis in 3D Sensor Networks

As discussed above, it is intrinsically difficult to construct medial axis for a 3D shape. The randomness of sensors, at the same time, greatly exacerbates the problem of instability and noise. A filtering process will unavoidably yield isolated medial axis nodes or clusters, which are extremely hard for further processing. We propose a medial axis construction algorithm based on the unit tetrahedron cell (UTC) mesh structure of 3D sensor networks [21]. The extracted UTC mesh structure approximates well the geographic structure of the sensor network. Based on the UTC mesh structure, a distributed algorithm starts from boundary surfaces with iterations. In each iteration, the outer boundary surface shrinks, while inner boundary surfaces grow at the same pace, to "peel" off one layer of the UTC mesh structure (as illustrated in

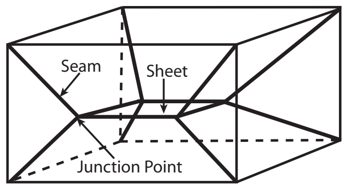


Fig. 3. An example of the medial axis of a simple hexahedron. It consists of thirteen sheets, twelve seams and four junction points.

Fig. 4(b)). When those surfaces meet and there is no space for further shrinking and growing, the algorithm yields a set of fully connected faces (see Fig. 4(c)). It can well approximate the accurate medial axis because each node has equal distance or differs by at most one hop to two closest boundary charts. Fig. 2(b) shows the computed medial axis based on our proposed algorithm, which is connected and well-structured.

II. Medial Axis Construction in 3D Networks

We represent a 3D wireless sensor network by a graph $G(V,E)$, where V denotes the sensor nodes and E indicates the communication links in the network. We assume the network has no global position information available.

Definition 1: A unit tetrahedron cell (UTC) is a tetrahedron formed by four network nodes, which does not intersect with any other tetrahedrons.

Definition 2: Two UTCs are neighbors if and only if they share a face.

Definition 3: A face is a boundary face if it is shared by one UTC only.

Definition 4: A UTC is a boundary UTC if it contains at least one boundary face. A non-boundary UTC is called an internal UTC.

Definition 5: A boundary surface of a network is a closed surface that consists of connected boundary faces. Each boundary surface is assigned a unique identifier.

We first construct a unit tetrahedron cell (UTC) structure of a given 3D sensor network [21]. A UTC mesh structure approximates well the global geometric structure and topology shape of the 3D sensor network. Once the UTC mesh is established, the boundary faces and boundary surfaces can be easily identified according to the definition. As discussed in Sec. I, the detected boundary surfaces grow at the same pace to "peel" off one layer of the UTC mesh structure in each iteration. The iteration stops until there is no more space for further growing. The main challenges to implement the algorithm are how to grow boundary surfaces based on the UTC mesh structure and how to refine the updated surfaces after each iteration by removing noisy branches. The proposed algorithm is distributed and operates on the UTC mesh structure. We explain the three-step algorithm in detail below, including initialization (Sec. II-A), face replacement (Sec. II-B), and branch trimming (Sec. II-C).

A. Initialization

We denote $UTC(A,B,C,D)$ a UTC formed by Nodes A,B,C and D. It has four faces, i.e., $Face(A,B,C)$, $Face\ (D,B,A)$,

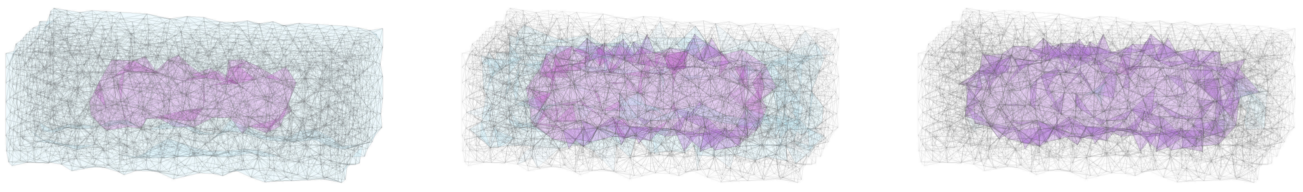

(a) Initial boundary surfaces. (b) After three iteration. (c) Final medial axis (after five iterations).

Fig. 4. The construction of the medial axis in a 3D seabed sensor network with an internal hole. The outer boundary is highlighted in blue, and the inner boundary in purple. In each iteration, the outer boundary shrinks and the inner boundary grows until they meet to yield the medial axis.

$Face(D,A,C)$, and $Face(D,C,B)$. Thus a UTC has at most 4 neighboring $UTCs$. Each face f maintains five parameters: Medial Axis Indicator (m_f) that indicates if Face f is on the medial axis, Active Indicator (a_f) that indicates whether Face f is currently active or not, Associated Boundary Surface (b_f) that keeps the ID of its associated boundary surface, Boundary Distance (d_f) that records its distance to the boundary surface b_f, and Neighboring Face Set (p_f) that consists of the set of three active faces neighboring to Face f.

During initialization, every face is marked as non-medial axis, i.e., $m_f = FALSE$. If Face f is a boundary face (determined based on its local information according to Definition 3), it sets b_f to the ID of the corresponding boundary surface, $d_f = 0$, $a_f = TRUE$, and p_f to be the three boundary faces, each sharing an edge with Face f; otherwise, it sets $b_f = NULL$, $d_f = \infty$, $a_f = FALSE$, and $p_f = NULL$. At the same time, every UTC maintains a flag that is initialized as "unvisited". A UTC will be updated to "visited" when either the outer boundary surface or an inner boundary surface shrinks or grows to it. The purpose is to ensure the algorithm visits each UTC for exactly once.

With the above initialization, it is obvious that a set of active faces sharing the same b_f form a closed surface (which is in fact a boundary surface). It is worth mentioning that the active faces will be updated in each iteration of our algorithm, and the set of active faces with the same b_f always form a closed surface. This is an important attribute that enables the shrink and growth of the boundary surfaces to yield connected medial axis when they meet.

B. Face Replacement

In an iteration, the distributed algorithm is run by each active Face f with $a_f = TRUE$. With local information only, it neither differentiates outer or inner boundaries, nor considers directions of shrink or growth of the boundary surfaces.

If $d_f = 0$, Face f is clearly a boundary face and involved in one UTC only. Assume the UTC is formed by Nodes A, B, C and D, and Face f corresponds to $Face(A,B,C)$. If $UTC(A,B,C,D)$ is "unvisited", Face f is then replaced by three other faces in the UTC, i.e., $Face(D,B,A)$, $Face(D,A,C)$, and $Face(D,C,B)$. More specifically, Face f is deactivated (i.e., $a_f = FALSE$), while the three new faces are activated (or $a_{Face(D,B,A)} = a_{Face(D,A,C)} = a_{Face(D,C,B)} = TRUE$). The three new faces are assigned the same boundary identifier as Face f's, i.e., $b_{Face(D,B,A)} = b_{Face(D,A,C)} = b_{Face(D,C,B)} = b_f$. Their distance to the boundary surface is set to $d_{Face(D,B,A)} = d_{Face(D,A,C)} = d_{Face(D,C,B)} = d_f + 1$. Meanwhile, their *neighboring face sets* are updated based on the set of new active faces. Finally, the UTC (i.e., $UTC(A,B,C,D)$) is set to "visited".

If Face f has $d_f \neq 0$, it must be a non-boundary face shared by two UTCs. Since Face f is active, at least one of the two UTCs must have been marked as "visited". If both of them are "visited", the algorithm goes directly to the trimming process to be discussed next. Otherwise, assume the UTC marked as "visited" is denoted as $UTC(A,B,C,D)$, and the "unvisited" UTC is $UTC(A,C,B,E)$. Then Face f (i.e., $Face(A,B,C)$) is replaced by three faces in the unvisited UTC: $Face(E,A,B)$, $Face(E,C,A)$, and $Face(E,B,C)$. Similar to the earlier discussion, Face f is deactivated, while the three new faces are activated with their *associated boundary surface*, *boundary distance*, and *neighboring face sets* updated. Then, $UTC(A,C,B,E)$ is marked as "visited".

Under such face replacement, all active faces with the same b_f still form a closed surface that can be traced by their associated p_f, as if the original boundary surface has been shrank or grown (see Fig. 4 for example). Moreover, since each active face performs only one replacement in an iteration, all boundary surfaces shrink or grow at the same pace.

C. Branch Trimming

The face replacement discussed above may yield some small branches due to the noise on the boundaries. They should be trimmed. The idea of trimming has been discussed in [22] for 2D and [23]–[25] for 3D image processing. In this work, we propose a distributed algorithm based on the UTC mesh for effective trimming in 3D sensor networks.

More specifically, after the face replacement in each iteration, an active non-boundary face, e.g., Face f, with both of its shared UTCs marked with "visited" will have two sets of parameters (i.e., m_f, a_f, b_f, d_f and p_f), because either two different boundary surfaces or two parts of one boundary surface have shrunk or grown to Face f from two different directions. Since all boundary surfaces shrink or grow at the same pace, the face should have two equal d_f to two closest boundary surface charts or they differ by at most one hop (due to the discrete setting). If its two b_f values are different, the face must be a medial axis face because it has equal distances to two different closest boundary surfaces. Thus the algorithm sets $m_f = TRUE$. Otherwise, the face has equal distances to two charts of the same boundary surface, and consequently it could either be a medial axis face if the two charts are widely separated or a noise that should be trimmed if the two charts are very close. This can be easily tested because Face f has two sets of p_f, and each of such neighboring faces has its own *neighboring face set* too. The algorithm traces either one layer or two layers of the local charts with Face f centered at each chart, and checks whether the two charts share any faces. If they do not share any face or the removal of Face f will break the closed surface formed by the active faces with the same b_f, the algorithm sets $m_f = TRUE$; otherwise $m_f = FALSE$ and Face f should be trimmed as discussed below. The choice of using one or two layer charts depends on how many small branches the algorithm wants to keep in the final medial axis.

To trim Face f, the algorithm first checks its two sets of p_f. If both of them include Face f itself, it is deactivated (i.e., $a_f = FALSE$). Moreover it is removed from the *neighboring face set* of each face in its two sets of p_f. After Face f is removed, its original neighboring faces need to update their *neighboring face sets* locally according to the new set of active faces. Otherwise, the algorithm waits to the next iterations until the above condition is satisfied after other branches are trimmed. Again, under such trimming strategy, all active faces with the same b_f still form a closed surface.

The iteration stops when all UTCs have been "visited". The face replacement and branch trimming processes virtually let the outer boundary surface shrink and inner boundary surfaces grow at the same pace. Since the active faces associated with the same boundary surface (i.e., with the same b_f) always form a close surface in each iteration, the surfaces shrunk or grown from their corresponding boundary surfaces are always closed and consequently yield a connected medial axis when they meet as shown in Fig. 4.

III. Applications of Medial Axis in 3D Sensor Nets

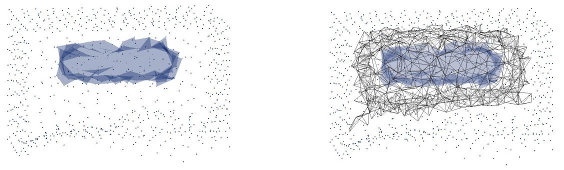
We introduce two medial axis-based applications in 3D sensor networks: medial axis-based safe navigation and medial axis-based distributed information storage and retrieval.

A. Medial Axis-Based Safe Navigation

Safe navigation aims to provide efficient guidance for human beings, robots and/or vehicles working in a 3D domain or traveling through it to move to a safe exit while keeping the farthest distance away from the dangerous areas during their movement. Assume each sensor in a 3D sensor network acquires environmental data, and then performs local computation to determine a *risk factor*. A sensor inside a dangerous area either has been destroyed or reports an extremely high risk factor. Such areas are identified as unaccessible voids (or "holes"). The area outside the network field is also treated as a special hole because of potentially high risks. A computed medial axis provides a locally safest route inside a 3D network.

B. Medial Axis-Based Distributed Information Storage and Retrieval

A key issue in distributed information storage and retrieval of a network is to appropriately choose the storage nodes


(a) Network model 1. (b) Computed medial axis.

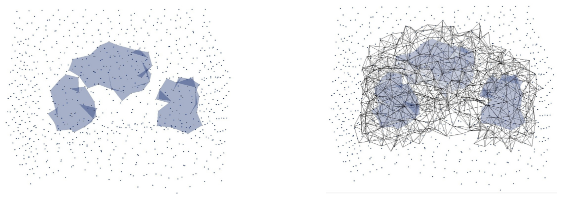

(c) Network model 2. (d) Computed medial axis.

Fig. 5. Selected 3D network models and their computed medial axes: Blue dots are sensors; blue regions are inner holes; black triangular surfaces are computed medial axis.

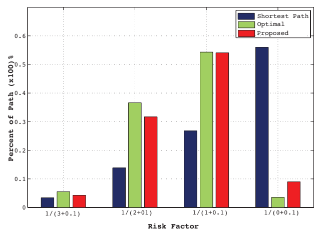
in order to reduce the overall communication cost. In most such systems, a datum is stored once but may be queried for many times. Thus the total communication cost is dominated by retrieval. In addition, to reduce the cost in retrieval, a datum is often replicated among part or all of the storage nodes. Intuitively, the best approach is to scatter the data storage nodes "*uniformly*" in the network, which gives the minimal average data query cost. However, it is practically infeasible due to the lack of global geometric information of the network. To this end, we propose to employ nodes on medial axis for data storage. Since a medial axis residents at the "center" and expands along the shape of a network, it is able to provide a great balance between query performance and algorithm complexity. A subset of medial axis nodes are uniformly chosen as storage nodes. A datum is stored at one (or all) storage nodes on the medial axis of the network. For data retrieval, a query travels to the nearest storage node to collect data.

IV. Performance Evaluation

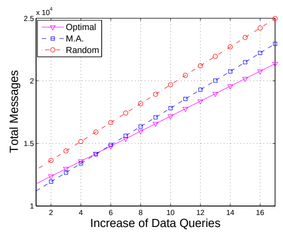
We implement our proposed computing medial axis algorithm and evaluate in various 3D sensor network models with different shapes and sizes. Sensor nodes are randomly distributed in a 3D space with average nodal degrees ranging between 12 to 20. Figs. 2 and 5 show several selected network models and the computed medial axes.

A. Medial Axis-based Safe Navigation

We randomly choose 1000 internal locations of a network to safely route to a few selected safe exits. For comparison, we have also implemented two other approaches. The first approach, denoted by "Shortest Path", is a simple shortest path algorithm that finds the shortest (in terms of hops) navigation path from any point to its closest exit. The second approach, dubbed "Optimal", is a brute force search to identify the optimal safe route which minimizes the maximal risk factor along the route from a location to an exit, which can be



(a)



(b)

Fig. 6. (a) Distribution of risk factor of routes. (b) Total cost vs. ratio of query/insertion.

considered as ground truth. Denote b the minimum distance along a safe navigation route to the dangerous areas. The risk factor of this route is defined as $r = 1/(b+\delta)$, where δ is a small constant to ensure non-zero denominator. δ is set to 0.1 in our simulations. Fig. 6(a) shows the distribution of risk factors of navigation routes. Since b varies from 0 to 3 in our simulated networks, the risk factor spans between 0.32 to 10. As can be seen, our proposed scheme performs close to the optimal solution. On the other hand, the shortest path based scheme has more than 50% routes with a risk factor of 10. This is because many shortest paths go through the boundaries of dangerous areas.

B. Medial Axis-based In-network Data Storage and Retrieval

We compare the proposed medial axis-based method (i.e., M.A.) with the "Optimal" scheme and a naive "Random" approach. The Optimal scheme uniformly selects storage nodes, while the Random one randomly selects data storage nodes. For all three methods, we assign the same number of nodes (which is 8) to store data. The simulation is based on the network model shown in Fig. 5(c) with a total of 1827 nodes. Each node in the network has equal probability to generate data or send query. In our simulation, 100 randomly generated data packets are sent to all 8 data storage nodes and $\alpha \times 100$ nodes are randomly selected to generate queries, where α denotes the ratio between query and insertion frequency. We assume each node keep records of the shortest path to the data storage nodes. In addition, we repeat the simulation under the Random scheme 20 times to obtain the average total cost. Fig. 6(b) shows the cost of both data storage and query, i.e., the number of total communication messages, with the increase of α from 1 to 17. The performance of the M.A. one is very close to the Optimal one. Both of them outperform the Random scheme as expected.

V. Conclusion

In this paper we proposed a distributed algorithm with linear time complexity and communication cost that builds a well-structured medial axis of a 3D sensor network without knowing its global shape or global position information. We have further applied the computed medial axis for two applications, i.e., safe navigation and distributed information storage and retrieval in 3D sensor networks and demonstrated their efficiency via simulations.

References

[1] J. Bruck, J. Gao, and A. Jiang, "MAP: Medial axis based geometric routing in sensor networks," in *Proc. of MobiCom*, pp. 88–102, 2005.
[2] M. Li, J. Wang, Z. Yang, and J. Dai, "Sensor Network Navigation without Locations," in *Proc. of INFOCOM*, pp. 2419 – 2427, 2009.
[3] X. Bai, C. Zhang, D. Xuan, J. Teng, and W. Jia, "Low-Connectivity and Full-Coverage Three Dimensional Networks," in *Proc. of MobiHOC*, pp. 145–154, 2009.
[4] X. Bai, C. Zhang, D. Xuan, and W. Jia, "Full-Coverage and K-Connectivity (K=14, 6) Three Dimensional Networks," in *Proc. of INFOCOM*, pp. 388–396, 2009.
[5] C. Liu and J. Wu, "Efficient Geometric Routing in Three Dimensional Ad Hoc Networks," in *Proc. of INFOCOM*, pp. 2751–2755, 2009.
[6] T. F. G. Kao and J. Opatmy, "Position-Based Routing on 3D Geometric Graphs in Mobile Ad Hoc Networks," in *Proc. of The 17th Canadian Conference on Computational Geometry*, pp. 88–91, 2005.
[7] J. Opatrny, A. Abdallah, and T. Fevens, "Randomized 3D Position-based Routing Algorithms for Ad-hoc Networks," in *Proc. of Third Annual International Conference on Mobile and Ubiquitous Systems: Networking & Services*, pp. 1–8, 2006.
[8] R. Flury and R. Wattenhofer, "Randomized 3D Geographic Routing," in *Proc. of INFOCOM*, pp. 834–842, 2008.
[9] F. Li, S. Chen, Y. Wang, and J. Chen, "Load Balancing Routing in Three Dimensional Wireless Networks," in *Proc. of ICC*, pp. 3073–3077, 2008.
[10] J. Zhou, Y. Chen, B. Leong, and P. S. Sundaramoorthy, "Practical 3D Geographic Routing for Wireless Sensor Networks," in *Proc. of SenSys*, 2010.
[11] H. Zhou, S. Xia, M. Jin, and H. Wu, "Localized Algorithm for Precise Boundary Detection in 3D Wireless Networks," in *Proc. of ICDCS*, pp. 744–753, 2010.
[12] D. Pompili, T. Melodia, and I. F. Akyildiz, "Routing Algorithms for Delay-insensitive and Delay-sensitive Applications in Underwater Sensor Networks," in *Proc. of MobiCom*, pp. 298–309, 2006.
[13] W. Cheng, A. Y. Teymorian, L. Ma, X. Cheng, X. Lu, and Z. Lu, "Underwater localization in sparse 3d acoustic sensor networks," in *Proc. of INFOCOM*, pp. 798–806, 2008.
[14] J. Allred, A. B. Hasan, S. Panichsakul, W. Pisano, P. Gray, J. Huang, R. Han, D. Lawrence, and K. Mohseni, "SensorFlock: An Airborne Wireless Sensor Network of Micro-Air Vehicles," in *Proc. of SenSys*, pp. 117–129, 2007.
[15] J.-H. Cui, J. Kong, M. Gerla, and S. Zhou, "Challenges: Building Scalable Mobile Underwater Wireless Sensor Networks for Aquatic Applications," *IEEE Network, Special Issue on Wireless Sensor Networking*, vol. 20, no. 3, pp. 12–18, 2006.
[16] P. J. Giblin and B. B. Kimia, "On the Local Form and Transitions of Symmetry Sets, and Medial Axes, and Shocks in 2D," Technical Report LEMS-170, Brown University, 1998.
[17] P. Giblin and B. B. Kimia, "A Formal Classification of 3D Medial Axis Points and Their Local Geometry," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 2, pp. 238–251, 2004.
[18] F.-E. Wolter, "Cut Locus & Medial Axis in Global Shape Interrogation & Representation," in *MIT Design Laboratory Memorandum*, 1992.
[19] T. Culver, J. Keyser, and D. Manocha, "Accurate Computation of the Medial Axis of a Polyhedron," in *Proc. of ACM Symposium on Solid Modeling Applications*, pp. 179 – 190, 1999.
[20] C. Hoffman, "How to Construct the Skeleton of CSG Objects," in *The Mathematics of Surfaces IV, Oxford Univ. Press*, 1990.
[21] S. Xia, X. Yin, H. Wu, M. Jin, and X. Gu, "Deterministic Greedy Routing with Guaranteed Delivery in 3D Wireless Sensor Networks," in *Proc. of MobiHoc*, pp. 1–10, 2011.
[22] R. Ogniewicz and M. Ilg, "Voronoi Skeletons: Theory and Applications," in *IEEE Computer Vision and Pattern Recognition*, pp. 63–69, 1992.
[23] M. Naef, O. Kuebler, G. Szekely, R. Kikinis, and M. Shenton, "Characterization and Recognition of 3D Organ Shape in Medical Image Analysis Using Skeletonization," in *Proc. of the Workshop on Mathematical Methods in Biomedical Image Analysis*, pp. 139–150, 1996.
[24] G. S. d. B. Attali, D. and E. Thiel, "Skeleton Simplification Through Non Significant Branch Removal," *Image Processing and Communication*, vol. 3, no. 3, pp. 63–72, 1997.
[25] M. Styner, G. Gerig, S. Joshi, and S. Pizer, "Automatic and Robust Computation of 3D Medial Models Incorporating Object Variability," *International Journal of Computer Vision*, vol. 55, pp. 107–122, 2003.