A Unified Framework for Line-like Skeleton Extraction in 2D/3D Sensor Networks

[1,2]Wenping Liu [1]Hongbo Jiang [1]Yang Yang [1]Zemeng Jin

[1]Huazhong University of Science and Technology, China
[2]School of Statistics, Hubei University of Economics, China
Email:{wenpingliu2009, hongbojiang2004, yangyang8795, jinzemeng}@gmail.com

***Abstract*—In sensor networks, skeleton extraction has emerged as an appealing approach to support many applications such as load-balanced routing and location-free segmentation. While significant advances have been made for 2D cases, so far skeleton extraction for 3D sensor networks has not been thoroughly studied. In this paper, we conduct the first work on the skeleton extraction in 3D sensor networks, and propose a unified framework for *line-like* skeleton extraction in both 2D and 3D sensor networks.**

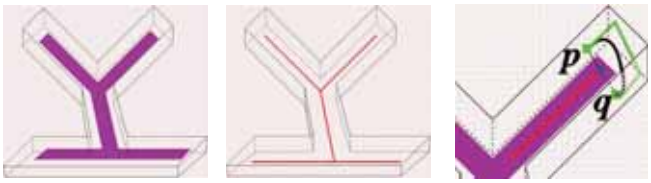
Our algorithm has the following three steps: first, each node identifies itself as a skeleton node if the geodesic shortest paths between its nearest boundary nodes (referred to as *feature nodes*) decompose the boundary of the network into more than one connected component; second, each skeleton node is assigned a monotonically increasing importance measure according to the maximum Lebesgue measure of the connected components of the boundary such that the identified skeleton nodes are self-connected; and finally, the skeleton is pruned based on the proposed metric *branch similarity*. The proposed algorithm is connectivity-based, distributed and of low complexity. Extensive simulations show that it is robust to shape variations and boundary noise.

I. Introduction

Topology feature of sensor networks can greatly affect the design of point-to-point routing, data gathering scheme, and so on. Skeleton (or medial axis) is an important infrastructure of sensor networks as it can accurately capture the topological features of the network. A variety of applications, such as routing [5], [6], navigation [7], [26], localization [24], [25], segmentation and random sampling [39], [40], etc, can benefit from this abstraction of the underlying geometric environment.

Related Work: Recently, a number of techniques have been proposed for skeleton extraction in sensor networks. Assuming that the complete boundary has been identified by boundary recognition algorithms, such as [12], [23], [34], [36], Bruck *et al.* [5], [6] proposed MAP, a Medial Axis-based routing Protocol. MAP identifies medial axis nodes that have at least two nearest boundary nodes, and eliminates unstable medial axis nodes whose nearest boundary nodes are too close. However, MAP is sensitive to boundary noise. To address this problem, Zhu *et al.* [39], [40] proposed to identify skeleton nodes based on the interval of the nearest boundary nodes, trying to alleviate the effect of boundary noise; and Jiang *et al.* [19], [20] proposed an algorithm, called CASE, to identify corner nodes and segment the boundaries into boundary branches, and an interior node is identified as a skeleton node if it has two or more nearest boundary nodes located on different branches; a multi-resolution skeleton can be easily obtained by adjusting the threshold value of the corner node. With an input of incomplete boundaries, e.g., identified by the neighborhood-size based boundary recognition scheme [15], DIST [27] conducts the skeleton extraction by first building the hop count distance transform. Each node then identifies itself by comparing the hop count transform of itself and its neighbors. The merit of Dist is that it can be used in sparse networks where complete boundary information is often difficult to obtain and thus MAP and CASE do not work well. When the network is complex, e.g., with concave nodes, however, the obtained skeleton will bend toward or even cross the boundaries, due to boundary incompleteness. As such, Liu *et al.* [28] proposed a framework for skeleton extraction which, rather than relying on boundary information, is based on purely neighborhood size of each node. For each node, it first constructs an index to quantitatively measure its centredness, and the node with locally maximal index identifies itself as a skeleton node.

978-1-4799-1270-4/13/$31.00 ©2013 IEEE



(a) (b) (c)

Fig. 1: Surface skeleton, line-like skeleton and their applications in routing. (a) The surface skeleton (indicated by the shaded 2-manifolds); (b) The line-like skeleton (red curve); (c) Points p and q have the same nearest skeleton point shown by the solid blue circle. The routing path from p to q generated by the routing protocol on top of the line-like skeleton is indicated by the dark black curve, and the routing path generated by the surface skeleton based routing protocol is indicated by the green curve.

Despite the success on 2D environments, so far there has been no skeleton extraction algorithm designed for 3D sensor networks. Previous solutions mostly assume a 2D space, explicitly or implicitly, and exploit the network's 2D geometric properties. Such properties often take a very different form or are more difficult to utilize in 3D spaces (see Fig. 1).

For example, connecting the skeleton nodes to form a line-like skeleton in a flooding-based manner is the basic idea of the schemes in [6], [20], [27], [28]. It is generally more difficult to connect nodes in 3D space than to connect nodes on the plane. With only connectivity information, in most scenarios, e.g., networks with parallel boundaries as shown in Fig. 2(a), the algorithm using flow complex [10], [9], [39] (referred to as naive algorithm) will fail at local maxima, incurring a self-disconnected skeleton, see Fig. 2(b). Besides, in principle, simply extending these 2D solutions to 3D settings will result in a *surface skeleton* (or medial surface) [1], which is defined as the locus of centers of maximally inscribed ball and composed of a series of 2D manifolds (also called skeletal sheets), and possible 1D curves [4], as shown in Fig. 1(a). This representation of the skeleton is computationally expensive and often demands a great amount of storage space at nodes, and thus is infeasible for resource-constrained sensor networks. More severely, when the surface skeleton is used to facilitate routing (that is, using the skeleton as a reference path, the packet is always forwarded along the direction "parallel" to the skeleton if possible), the routing path can be considerably long. For instance, as shown in Fig. 1(c), points p and q, located on the opposite sides of the surface skeleton, have the same distance h to the nearest surface skeleton point (shown by the solid blue circle). The packet from point p to q is forwarded along the direction indicated by the green curve, and thus suffers a long path to delivery.

In the literature of computer graphics, an alternative skeletal representation of a 3D object is a line-like skeleton [3], [8], [32], as shown in Fig. 1(b), which consists of 1D curves. The line-like skeleton, also referred to as curve-skeleton, centerline, or inverse-kinematic skeleton, is a subset of the surface skeleton. Compared with surface skeleton, line-like skeleton has lower dimensionality, and is easier to compute and occupies less storage space. This kind of line-like skeletons in both 2D and 3D environments satisfies desirable properties as follows: homotopic to the original object, centered, connected, robust, hierarchical, etc [8]. Further, as illustrated in Fig. 1(c), the routing protocol on top of the line-like skeleton is delivery-guaranteed and has low stretch factor. The packet from p to q is forwarded along the circle with the radius h centered at the skeleton point (shown by the solid blue circle). It can be found that the routing path by the line-like skeleton based routing protocol has smaller length than the path by the surface skeleton based routing protocol[1]. However, the existing algorithms like [6], [20], [27], [39] face many difficulties in extracting line-like skeleton in 3D objects since there are too many skeleton nodes and no specific criterion is available yet to connect them to form a line-like representation.

Our Contributions: In this paper, our objective is to extract the line-like skeleton of a 2D/3D sensor network with reliance on mere connectivity information. To that end, we first derive a unified definition of line-like skeleton points in both 2D and 3D settings. Then, in practice, to obtain the skeleton of a sensor network, for each node, we first identify its *feature nodes* and construct a set of connected components, and thus identify it as a skeleton node when the connected components can be connected to form at least one loop (for 3D sensor networks) or curve (for 2D sensor networks) such that the boundary can be decomposed into two or more connected components accordingly. To connect the identified skeleton nodes, we propose importance measure of skeleton nodes, based on which a skeleton tree is constructed. Finally using the proposed branch similarity, we measure and trim the redundant skeleton branches, as shown in Fig. 2(c). To the best of our knowledge, this is the first work on line-like skeleton extraction in 3D networks, and our contribution is a novel skeleton extraction solution feasible for both 2D/3D networks. Our algorithm is desirably distributed, connectivity-based, scalable, and robust to boundary noise, etc.

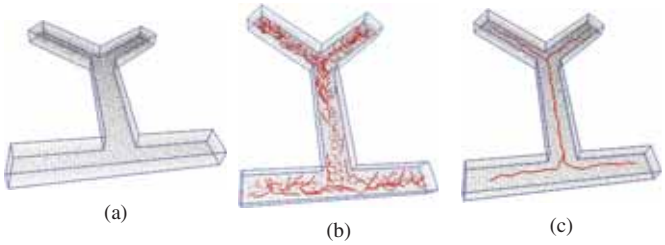


Fig. 2: Skeleton extraction in a Y-shaped 3D sensor network with 12,545 nodes and avg.deg 22.53. (a) The original network. Boundary nodes are marked in blue; (b) The skeleton extracted by naive method; (c) The skeleton extracted by our algorithm.

The remainder of the paper is organized as follows. Section II presents the motivation of our work. Section III is devoted to the skeleton extraction algorithm in 2D/3D sensor networks. We demonstrate the effectiveness of the algorithm in Section IV, and conclude the paper in Section V.

II. Motivation and Theoretical Foundation

In this section, we first derive a unified definition for line-like skeleton points in both 2D and 3D continuous domains. And then we propose a metric to measure the degree to which a point reflects the details of the underlying object. The usage of this metric, as will be shown in Section III, is to serve as a guidance to connect the identified skeleton points.

A. Line-like Skeleton Point Definition

Let $D \subset \mathbb{R}^k$ be a k-dimensional$(k = 2, 3)$ object bounded by a connected $(k-1)$-manifold ∂D. Especially, for a 3D object, we assume its boundary surface to be smooth(C^∞), compact and without boundary. We define the (Euclidean) distance transform $T : D \to \mathbb{R}$ as $T(x \in D) = min_{y \in \partial D} d_E(x, y)$ for $x \notin \partial D$ and 0 otherwise, where $d_E(x, y)$ is the (Euclidean) distance of point x to y; let $F : D \to \mathbb{P}(\partial D)$ be a feature transform assigning to each point $x \in D$ the set of the nearest points (namely, *feature points*) on ∂D to x, where $\mathbb{P}$ is the power set. That is, $F(x) = \{z \in \partial D | d_E(x, z) = T(x)\}$.

We start with the surface skeleton of $D \subset \mathbb{R}^3$, denoted by $S(D)$, followed by our definition for the line-like skeleton

[1]Formally, we can give a brief explanation. First the surface skeleton is a 2-manifold, that is, for any line-like skeleton point v in the 2-manifold, there exists a small value $r > 0$ such that its open r-neighbors (the points having a distance less than r to the point v) are also in the 2-manifold. Then, for any point u which has a distance h to the neighbors of point v, the distance between u and v is $\sqrt{r^2 + h^2} > h$. That is, the circle centered at the line-like skeleton point with radius h is totally included in a region bounded by the routing path generated by the surface skeleton based routing protocol, which suffices to confirm the observation.

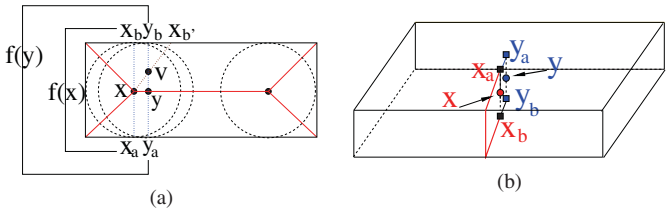


Fig. 3: An illustration for Lemma 1 and Theorem 5.

point. Formally, $S(D) = \{x \in D | \#\{F(x)\} \geq 2\}$. Clearly, $S(D)$ can be decomposed into two mutually exclusive subsets: $S_1(D)$ and $S_2(D)$, where $S_1(D) = \{x \in D | \#\{F(x)\} = 2\}$, and $S_2(D) = \{x \in D | \#\{F(x)\} > 2\}$. We call the point in $S_1(D)$ and $S_2(D)$ as a *generic skeleton point* and *non-generic skeleton point*, respectively. We first study the properties of a generic skeleton point, and then show that a non-generic skeleton point also has the same properties.

For any point $x \in S_1(D)$, let x_a and x_b be its two feature points; let X be the set of the feature points, i.e., $X = \{(x_a, x_b) | x \in S_1(D)\}$. Define the feature distance function $F_d : X \to \mathbb{R}$, and $f = F_d \circ F$, then f is a mapping from $S_1(D)$ to the 1-dimensional space $\mathbb{R}$, i.e., for a point $x \in S_1(D)$, $f(x)$ is the geodesic distance (the distance along the surface rather than through the space) between its feature points x_a and x_b; we call f the *Geodesic Distance Function* (GDF). Specially, for a point with only one feature point, we define its geodesic distance function as 0. Clearly, for each point $x \in S_1(D)$, if there are two geodesic shortest paths between x_a and x_b, they must have equal lengths. As such, f is well defined on $S_1(D)$ and D.

Since we have assumed the boundary to be smooth, it can be shown by using a similar technique as in [2] that $S_1(D)$ is a smooth manifold, and $F : D \to \mathbb{P}(\partial D)$ is differentiable. Define $\Gamma : S_1(D) \to \mathbb{R}^2$ as the local coordinate function. Then Γ is a diffeomorphism, respecting the fact that $S_1(D)$ is a smooth manifold. Further, we define $\Psi : \mathbb{R}^2 \to \mathbb{R}$ such that for each point $x \in D$, we have $\Psi(\Gamma(x)) = f(x)$. As such, from the perspective of differential geometry, f is differentiable at point x if and only if Ψ is differentiable at $\Gamma(x)$.

Next we first introduce a number of results about geodesic distance function and geodesic shortest path, and then we derive a unified definition of the line-like skeleton for both 2D and 3D objects.

Lemma 1: The function f has no local minimum.

Proof: We only prove this for 2D objects, and the proof for 3D cases can be similarly done in a divide-and-conquer fashion. To that end, we only need to prove that for any point y having two feature points x_a, x_b and y_a, y_b, there is a neighboring point x satisfying that the geodesic distance $f(x)$ between its two feature points x_a, x_b is less than $f(y)$. Please see Fig. 3(a). Clearly, we can always find a point x such that one of its feature points, e.g., x_a is on the geodesic shortest path between y_a and y_b. We prove by contradiction that x_b is also on the geodesic shortest path between y_a and y_b. If x_b is not on the geodesic shortest path between y_a and y_b, e.g., the feature point x_b moves to point $x_{b'}$, then the chord $xx_{b'}$ will intersect with the chord yy_b, say at a non-skeleton point v. That is, the point v stays on two chords, contradicting the fact that each non-skeleton point stays on a unique chord, which has already been proved by [5], [6]. Therefore, x_b is also on the geodesic shortest path between y_a and y_b, and the geodesic distance between x_a and x_b is less than that between y_a and y_b. That is, $f(x) < f(y)$ which proves the claim. ■

Based on Lemma 1, we can easily deduce that

Lemma 2: The function f has only one local maximum, which is also a global maximum.

Lemma 2 implies that by letting each point flow to the unique global maximum, the geodesic distance based flow complex will not fail at local maxima, and thus a self-connected skeleton can be guaranteed.

Lemma 3: The geodesic distance function f is not differentiable everywhere.

Proof: We only prove this for $2D$ object D. Assume that x is a skeleton point, x_a and x_b are its two feature points, then we have $f(x) > 0$. For each point y_1, y_2 on the two chords xx_a, xx_b, it has been proven in [5], [6] that x_a (or x_b) is the unique feature point of y_1 (or y_2). That is, $f(y_1) = f(y_2) = 0$. It naturally follows that the geodesic distance function f is not differentiable at the point x. In a similar way, we can prove that the geodesic distance function f is not differentiable at any skeleton point. ■

Lemma 3 says that the geodesic distance function f is not differentiable at skeleton points. Next we will prove that the singularities of f, i.e., the points where f is not differentiable, has a Lebesgue measure zero.

Definition 1: A function $g : \mathbb{R}^k \to \mathbb{R}$ is locally Lipschitz continuous (also referred to as locally K-Lipschitz) if for any point $x \in \mathbb{R}^k$, there exists a real constant $K \geq 0$ and some $\epsilon > 0$ such that, for every point $y \in \mathbb{R}^k$ satisfying $|g(x) - g(y)| < \epsilon$, the following inequality holds:

$$|g(x) - g(y)| \leq K \|x - y\| \tag{1}$$

Obviously, the (Euclidean) distance function F is locally Lipschitz continuous since, for any two points $x, y \in \partial D$, $F(x) \leq F(y) + 1 \cdot \|x - y\|$ always holds. And it has been proven in [11] that Ψ is also locally K-Lipschitz for some $k > 0$. That is,

Lemma 4: Ψ is locally Lipschitz continuous.

With Lemma 3 and Lemma 4, according to Rademacher's theorem [14], the singularities of f has a Legesgue measure zero, implying that the singularities of f consists of 1D curve, namely, the line-like skeleton [11]. In other words, a point v is a line-like skeleton point if the function f is not differentiable at point v.

To compute the singularities of the function f, however, is cumbersome and impractical for sensor networks. Fortunately, we have the following theorem:

Theorem 5: For any non-singular point of f, there is only one geodesic shortest path; and for each singular point, there are two geodesic shortest paths between its feature points, decomposing the boundary into connected components.

Proof: For 2D objects, it is obvious that the geodesic shortest path between any two feature points of a singular point of f segments the boundary into two boundary branches. We now prove that this is the case for 3D objects. We prove by contradiction that for a singular point x, there are more than one geodesic shortest paths between its two feature points x_a and x_b. Suppose that there is only one geodesic shortest path between its feature points, please see Fig. 3 (b). This implies that there exists a neighboring point, say y, such that $y's$ geodesic distance between its feature points y_a and y_b is larger than that of x, namely, $f(y) > f(x)$. According to Lemma 1, there exists a point y' such that $f(y') < f(x)$. When the points y, y' approach to point x, the feature distances $f(y), f(y')$ converge to $f(x)$, that is, the function f is smooth at point x, which contradicts that x is a singular point of f. Since there are two geodesic shortest paths between x_a and x_b, they should have the same length and form at least one loop. Accordingly, the boundary of the object is decomposed into connected components. ■

Theorem 5 provides a simple way for skeleton point identification: if point v is such that the geodesic shortest paths between its feature points form a *feature loop*, decomposing the boundary into components, then point v is a skeleton point; otherwise, it is a non-skeleton point, as shown in Fig. 6(a).

Next we consider the non-generic skeleton in $S_2(D)$, i.e., the points with three or more feature points. It happens at the intersection of $n(\geq 3)$ 2-manifolds with boundaries in $S_1(D)$. It is easy to show that the neighbors of any point $x \in S_2(D)$ do not necessarily belong to $S_2(D)$, that is, $S_2(D)$ is not a 2-manifold. Therefore it either consists of isolated points or forms 1D curve. Clearly, for point $x \in S_2(D)$, it has more than three geodesic shortest paths between its feature points, which form at least one loop and separate the boundary into connected components.

Note that a skeleton point of 2D objects also has the same property, i.e., the shortest paths between its feature points form a curve decomposing the boundary into branches. As such, we now present a unified definition of the line-like skeleton in 2D/3D space as follows:

Definition 2: A point is a *line-like skeleton point* if and only if the set of the geodesic shortest paths between its feature points decomposes the boundary into connected components.

Following this definition, we can identify the skeleton points, without reliance on any system parameters. However, it is noted that these identified skeleton points is not self-connected. A straightforward method is to use flow complex [9], [39], however, there can be many local maxima, resulting a set of disconnected skeleton arcs, as mentioned in Section I. Thus, it faces a non-trivial challenge to connect these skeleton points into a meaningful representation of the underlying 2D/3D object.

B. Importance Measure of The Line-like Skeleton Point

To connect the identified skeleton points, in this subsection, we propose the *importance measure* associated with the line-like skeleton points, based on which a skeleton tree can be easily constructed.

1) Importance Measure in 2D: Let $D \subset \mathbb{R}^2$ has no holes. Its skeleton $S(D)$ can be defined as the singularities of the distance transform $T : D \to \mathbb{R}$. Let ∇T be the gradient field of D. Obviously, ∇T is undefined on $S(D)$. Conventionally, each boundary point only flows to $S(D)$ following the direction of ∇T, so the delivered skeleton is self-disconnected. To address this, we define a flow vector field $\mathbf{V}$ such that after flowing to $S(D)$, the point keeps flowing along the skeleton until it reaches a unique point of D, called *core point* $\mathcal{CP}(\mathcal{D})$, and deliver a self-connected skeleton. Specifically, for non-skeleton point z, similar with [39], we define the driver $d(z)$ as its unique feature point. For a skeleton point, however, we define its driver as the neighboring skeleton point having the smallest GDF, instead of the sink itself in [39], to achieve a self-connected skeleton. For $\mathcal{CP}(\mathcal{D})$, the driver is itself. Accordingly, we define the flow $\mathbf{V}$ as $\mathbf{V}(z) = \frac{z-d(z)}{|z-dz|}$ for $z \neq d(z)$ and 0 otherwise.

Governed by the flow V, each non-skeleton point will first flow to $S(D)$ and then keep moving along $S(D)$ toward the core point $\mathcal{CP}(\mathcal{D})$. Since D has no holes, its skeleton $S(D)$ is a tree-like structure. As such, each skeleton point p can divide it into two sub-trees and accordingly, the flows passing through a skeleton point p will decompose the object into connected components, and the origins of the flows also decompose the boundary ∂D into connected components, say $C_1(p), C_2(p), \cdots, C_l(p)(l \geq 2)$. As such,

Definition 3: We define the ***Importance Measure*** of p, denoted by $\rho(p)$, as follows

$$\rho(p) = 1 - \max_i\{\lambda(C_i(p))\}/\lambda(\partial D) \tag{2}$$

where $\lambda(\cdot) : \Omega \to \mathbb{R}$ is a Lebesgue measure assigning a real value to a subset of n-dimensional space Ω.

Intuitively, the importance measure of a point describes how many *trajectories* flow to this point. Since the core point is a skeleton point where all trajectories "sink" into, it has the largest importance measure.

Lemma 6: Each object D has only one core point $\mathcal{CP}(\mathcal{D})$.

Theorem 7: The importance measure $\rho(p)$ is a monotonic function with the distance from p to $\mathcal{CP}(\mathcal{D})$.

Proof: Since the unique core point $\mathcal{CP}(\mathcal{D})$ is a skeleton point where all trajectories "sink" into, for any skeleton point p with a distance $d(p)(> 0)$ to $\mathcal{CP}(\mathcal{D})$, the amount of trajectories flowing to p is smaller than that of $\mathcal{CP}(\mathcal{D})$. Assume q is a skeleton point at the same side of p to $\mathcal{CP}(\mathcal{D})$. Without loss of generality, assume $d(q) > d(p)$. Clearly, the trajectories flowing to q will surely flow to p but not vice versa. That is, the importance measure of p is larger than that of q, which proves the claim. ■

Theorem 7 implies that the closer a skeleton point is to the core point, the larger its importance measure, and there only exists one local (also global) maximum, i.e., $\mathcal{CP}(\mathcal{D})$, resulting in a self-connected skeleton. As such, it offers a rule to generate a skeleton tree.

2) Importance Measure in 3D: We can easily extend the 2D model to a 3D object D such that a monotonic importance measure is applicable for extracting a robust and self-connected line-like skeleton for the 3D object D.

Similar with 2D model, our goal is to construct a flow vector field $\mathbf{V}$ which can guarantee that each non-skeleton point first flows to the line-like skeleton following the direction of the gradient field ∇T(where T is a distance transform) and then keeps moving along the line-like skeleton toward a unique core point $\mathcal{CP}(\mathcal{D}) \in S(D)$, thereby a self-connected skeleton can be obtained. Compared with 2D model, however, defining $\mathbf{V}$ on a point in a 3D object is more challenging since an improper flow vector field might generate a surface skeleton which contains 2-manifold sheets.

Our solution is a divide-and-conquer strategy. More specifically, we first *cut* the 3D object D into slices, and then apply our 2D model on these slices to define the flow vector field $\mathbf{V}$. Let D_s be a slice with boundary $\partial D_s \subset \partial D$, and $\mathcal{C}P(D_s)$ be its core point, called *local core point*. Within each slice D_s, we can define the flow vector field $\mathbf{V}_s$ using our 2D model. Namely, for each non-skeleton point $z \in D_s$, the driver $d(z)$ is its unique feature point in ∂D_s, and the driver of a skeleton point is the neighboring skeleton point with the smallest GDF. Note that we cannot define the driver of the local core point as itself, otherwise the local core points of these slices are not self-connected. To address this issue, note that each slice divides the boundary into components, we call the local core point of the slice which decomposes the boundary surface into two halves as the *global core point*, and define the driver of each local core point as the neighboring local core point farthest to the global core point. As for the global core point, the driver is itself. As such, the flow vector field V can be constructed accordingly. That is, for each point $z \in D$, $\mathbf{V}(z) = \frac{z-d(z)}{|z-dz|}$ for $z \neq d(z)$ and 0 otherwise.

The slices should be carefully chosen such that its normal is vertical to the gradient field ∇T, and thereby the trajectories originating on each slice boundary remain in the slice. Note that here the slice boundary ∂D_s decomposes the boundary ∂D into components, reminiscent of the geodesic shortest paths between feature points of a line-like skeleton point p that form at least one feature loop and decompose the boundary into connected components, where p corresponds to the local core point. As such, the feature loop provides a good approximation to the slice boundary, and the nodes nearest to the same slice boundary can be regarded as a slice. We thus define the importance measure $\rho(p)$ of the skeleton point in the same way as in the 2D model by Equ. (2). Again, The global core point $\mathcal{CP}(\mathcal{D})$ is where all points sink into and thus has the largest importance measure. In addition, the trajectories flow to the global core point along the direction of monotonically increasing importance measure. The closer a skeleton point (a local core point) is to the global core point, the more details of the object it reflects.

In summary, for both 2D and 3D objects, the line-like skeleton point has a unified definition and has the common property: the (geodesic) shortest paths between its feature points decompose the boundary into connected components $C_1(p), C_2(p), \cdots, C_l(p)(l \geq 2)$, which can be used for computing the importance measure of the skeleton point according to Equ. (2) and thus guiding the skeleton tree construction for achieving a self-connected skeleton in sensor networks, which will be detailed in next section.

III. Algorithm

In this section, we adapt the aforementioned principle of skeleton extraction in continuous domain to the discrete analog, i.e., the sensor networks. Here boundary information is taken as an input; there are many boundary extraction algorithms in 2D/3D sensor networks, e.g., [12], [22], [38], [36]. We first present the outline of our algorithm, followed by the details of each step. Since the 2D cases are quite similar, we limit the details on 3D spaces here.

1) **Skeleton Node Identification:** The node, which satisfies that the geodesic shortest paths (for 3D sensor networks) or shortest paths (for 2D sensor networks) between feature nodes decompose the boundary into 2 or more connected components, marks itself as a skeleton node, as shown in Fig. 4(e) and Fig. 5(d).

2) **Importance Measure Computation and Skeleton Tree Construction:** The *importance measure* of each skeleton node is then derived from the computation of the number of nodes in the connected components. After that, each skeleton node chooses the neighboring skeleton node with the largest importance measure as the parent node, and accordingly, a skeleton tree is constructed following the direction of the monotonically increasing importance measure to derive a self-connected skeleton, as shown in Fig. 4(f) and Fig. 5(e), and thus the cumbersome linking operation to connect the skeleton nodes, which is wildly used in existing algorithms such as [6], [20], [27], [28], can be avoided.

3) **Refinement:** The skeleton tree may contain redundant skeleton branches, owing to the discrete nature of sensor networks. We trim the skeleton tree according to the proposed *branch similarity*. As a result, the final skeleton is obtained, as shown in Fig. 2(c) and Fig. 5(f).

A. Skeleton Node Identification

To identify the feature node(s) of each interior node, the boundary nodes initiate controlled floods inside the network; and each interior node is associated with a boundary tree rooted at one of its feature node and keeps record of the feature nodes, as shown in Fig. 4(a). Note that this flooding process only causes overall $O(N)$ message cost where N is the number of nodes as the individual node just forwards the first flooding message it receives. In our algorithm, if an interior node has a minimum hop count distance k to the boundary, then the boundary nodes that are $k+1$ hops away are also regarded as the feature nodes, respecting the fact that in special scenarios (e.g., a box-shaped network with even width, or sparse networks), some skeleton nodes possibly have only one feature node.

Based on these feature nodes, we mainly address how to identify skeleton nodes in 3D sensor networks (as for 2D cases, the method is quite similar). Note that there is no need for every interior node to determine its identity, which otherwise incurs a large communication cost. Instead, similar with [27], only the leaf nodes of the boundary trees conduct the skeleton node identification process. In continuous domain, as highlighted in Definition 2 (see Section II-A), at each skeleton node, the geodesic shortest paths will form at least one loop, separating the boundary surface into two or more parts, as

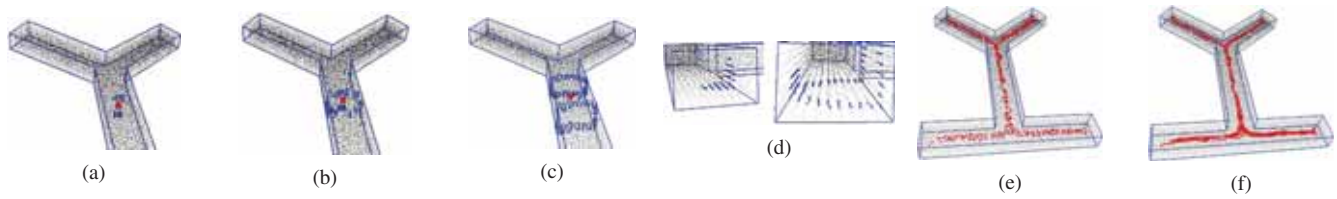


Fig. 4: Line-like skeleton extraction in Fig. 2. (a) An interior node (marked in red) and its feature nodes (marked in blue); (b) Feature loop nodes (marked in blue); (c) The dilated path; (d) The feature nodes (left) and the dilated path (right) of a non-skeleton node; (e) Skeleton nodes; (f) Skeleton tree.

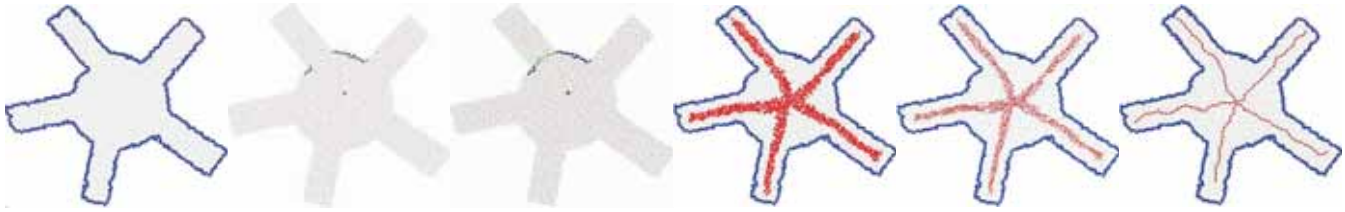


Fig. 5: Skeleton extraction in a sun-shaped 2D sensor network which has 5,217 nodes with average degree 12.27. (a) The original network. Boundary nodes are marked in blue; (b) A skeleton node (marked in red) and its feature nodes (marked in blue); (c) The shortest path (marked in green) between connected components; (d) Skeleton nodes; (e) Skeleton tree; (f) Final skeleton.

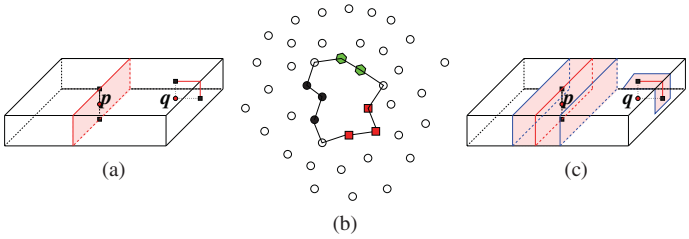


Fig. 6: Feature loop identification. (a) Feature loop curve and non-loop curve; (b) Fake feature loop. The solid nodes are the feature nodes (the feature nodes with the same shape are in a connected component) of an interior node and the empty circles are other boundary nodes. The loop does not decompose the boundary surface into pieces, therefore it is a fake loop; (c) The dilated paths (blue curve) is the set of points having the same distance to the feature loop (red curve). The boundary of the dilated path of p, q have 2 and 1 closed curve respectively, therefore p identifies itself a line-like skeleton point, and q is a non-skeleton point.

shown in Fig. 6 (a), which guides our steps to identify skeleton nodes in 3D sensor networks as follows:

1) **constructing and connecting the feature components.** For any leaf node of a boundary tree, each feature node issues a controlled-flooding to construct a set of connected components, each of which is assigned a unique identifier. Next, in order to connect these components, a controlled hop-by-hop expansion process is conducted for achieving a low communication cost. More specifically, each feature node p broadcasts a message including its identifier, and $p's$ neighboring boundary nodes (or any $p's$ neighboring nodes in 2D networks) which has not been assigned any identifier, are assigned the same identifier. Such process is conducted repeatedly until two boundary nodes (or any nodes in 2D networks) with different identifiers meet. This way, the geodesic shortest paths between feature components can be built in a distributed fashion. Without ambiguity, the feature nodes and the nodes on the shortest paths are henceforth referred to as feature nodes, as shown in Fig. 4(b).

2) **feature loop identification.** In continuous domain, as mentioned in Section II-A, for a line-like skeleton point in 3D objects, the geodesic shortest paths between its feature points form at least one feature loop, while the set of the geodesic shortest paths between the feature points of a non-skeleton point is only a non-loop curve. As such, to identify whether a leaf node is a line-like skeleton node, the presence of the feature loop(s) is a key. One might argue that feature loop identification can be simply done as follows: any feature node floods within the feature nodes to build a shortest path tree, if there are two neighboring feature nodes having no common ancestor but the root, then the feature nodes form a loop. For a network with large node density, this might not be problematic; in a sparse network, however, this method does not work. That is, there might be fake loops. Please see Fig. 6(b) as an example. To deal with this, each feature node first floods within a small scope, i.e., its 2-hop neighbors. Those nodes that are at most 2-hops away from these feature nodes then naturally form a dilated path, as shown in Fig. 6(c). If the boundary of the dilated path is more than one closed curve, i.e., the nodes that are 2 hops away from the feature nodes (as shown in Fig. 4(c)) can not form one connected component, then there is at least one genuine feature loop, and the corresponding interior node p identifies itself as a skeleton node; otherwise, p is not a skeleton node, as shown in Fig. 4(d). Note that this can be easily done in a distributed fashion. Fig. 4(e) shows the skeleton nodes of the 3D sensor network. With the identified skeleton nodes, in next subsection we will propose to compute the importance measure of each skeleton node in a distribute

way such that these skeleton nodes can be self-connected.

B. Importance Measure Computation and Skeleton Tree Construction

In order to connect the identified skeleton nodes to form a meaningful line-like representation of the original network, guided by Definition 3 in Section II-B, each skeleton node is assigned an *importance measure*, a metric measuring how important the skeleton node is to reflect the details of the network. Based on the importance measure, a skeleton tree can be constructed such that a self-connected skeleton is generated.

Note that in 3D sensor networks, the feature loop of a skeleton node decomposes the boundary surface into a set of connected components. In 2D sensor networks without holes, the shortest paths between the feature points of a skeleton node also decompose the boundary into branches. We call the feature loop (in 3D networks) or geodesic path (in 2D networks) as *Feature Component*. For a skeleton node p, denote by $C_1(p), C_2(p), \cdots, C_l(p)$ the connected components of the boundary divided by the feature component of p, satisfying that $|C_1(p)| \leq |C_2(p)| \leq \cdots \leq |C_l(p)|$. Then the importance measure of p, denoted by $\tau(p)$ is given as follows:

$$\tau(p) = 1 - |C_l(p)|/|C| \tag{3}$$

where C is the set of boundary nodes. Generally, the importance measure of p ranges from 0 to 1.

After each skeleton node computes its importance measure, a skeleton tree can be constructed to connect these skeleton nodes. According to Theorem 7, the skeleton tree is constructed in a greedy manner, namely, each skeleton node selects the neighboring node with the largest importance measure as its parent node, which finally forms a skeleton tree rooted at the so-called *core node* (the counterpart of global core point in continuous domain), mimicking the flow in continuous domain as described in Section II-B. Obviously, the core node, where all skeleton nodes "sink" into, has the largest importance measure. Fig. 4(f) and Fig. 5(e) show the result of skeleton tree construction. Note that there are redundant skeleton branches, owing to the discrete nature of sensor networks and thus many nodes being identified as skeleton nodes. As such, most of them are *similar* with each other, i.e., they have some common feature nodes. In the next subsection, we will conduct a pruning process to deliver a line-like skeleton.

C. Refinement

As mentioned, the skeleton tree may have unwanted skeleton branches, and thus refinement is needed for a good skeleton. To this end, we propose a metric named *branch similarity*, based on which we can obtain a simplified skeleton without unwanted skeleton branches.

We first present the definition of branch similarity.

Definition 4: For two skeleton branches B_1, B_2 with a common parent node p, let L_1, L_2 denote the depths of the branches B_1, B_2 respectively. For two nodes p_1^i, p_2^i on the branches B_1, B_2, which are both $i(\leq L = min\{L_1, L_2\})$ hops from node p, define $I(p_1^i, p_2^i)$ to be 1 if p_1^i, p_2^i are neighboring, and 0 otherwise. Then the branch similarities between B_1 and B_2 of B_1, and B_2, denoted by $Sim(B_1|B_2)$, and $Sim(B_2|B_1)$ respectively, are defined as:

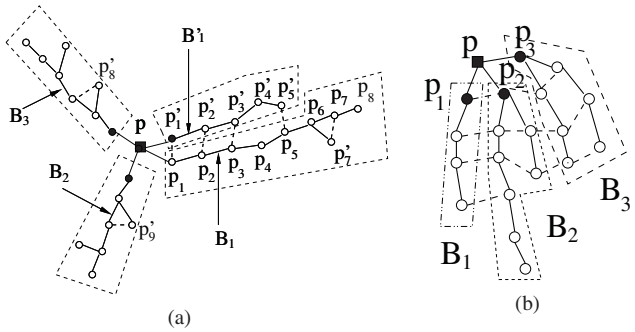


Fig. 7: Skeleton branch similarity. The parent node p(shown by solid rectangle) has several children nodes (shown by solid circles) and thereby skeleton branches (bounded by dashed polygons), and the hollow circle are ordinary skeleton nodes. The solid line between two skeleton nodes indicates they are on the same skeleton branch, and the dashed line indicates they are neighboring nodes. (a) Branch B_1 and B_1' have 4 pairs of neighboring skeleton nodes, and the skeleton branch similarity of B_1 and B_1' are $Sim(B_1'|B_1) = \frac{4}{5} = 0.8$ and $Sim(B_1|B_1') = \frac{4}{9} = 0.44$, respectively; (b) Parent node p has three skeleton branches. $Sim(B_1|B_2) = 1.0$, $Sim(B_2|B_1) = \frac{4}{6} = 0.66, Sim(B_2|B_3) = \frac{2}{6} = 0.33$, $Sim(B_3|B_2) = \frac{3}{6} = 0.5$. Therefore, $Sim(B_1) = 1.0, Sim(B_2) = 0.66, Sim(B_3) = 0.5$.

$$\begin{aligned} Sim(B_1|B_2) &= \sum_{i=1}^{L} I(p_1^i, p_2^i)/L_1 \\ Sim(B_2|B_1) &= \sum_{i=1}^{L} I(p_1^i, p_2^i)/L_2 \end{aligned} \tag{4}$$

If the parent node p has I children nodes that generate I skeleton branches accordingly, then the branch similarity of the branch $B(p)$, including p and the nodes on the I branches, is defined as

$$Sim(B(p)) = max_{i,j \leq I, i \neq j}\{Sim(B_j|B_i)\} \tag{5}$$

Fig. 7 shows the principle of computing branch similarity. With the branch similarities, we now simplify the skeleton tree in an iterative fashion. Initially, we eliminate those skeleton branches with branch similarity of 1.0. If all skeleton branches of a parent node have a branch similarity is no less than the given value, then all branches but the one with the largest length is kept. Next, for any given step size $\delta > 0$, the branches with branch similarity larger than $1.0 - \delta$ will be trimmed, followed by trimming the branches with branch similarity larger than $1.0 - 2 \times \delta$, and so on. The iteration is terminated when no more skeleton branches could be deleted, and the final skeleton is thus generated, please see Fig. 2(c) and Fig. 5(f). Note that our algorithm has no dependency on the step size δ. Interestingly, by setting threshold on the importance measure, we can generate multi-scale skeletons.

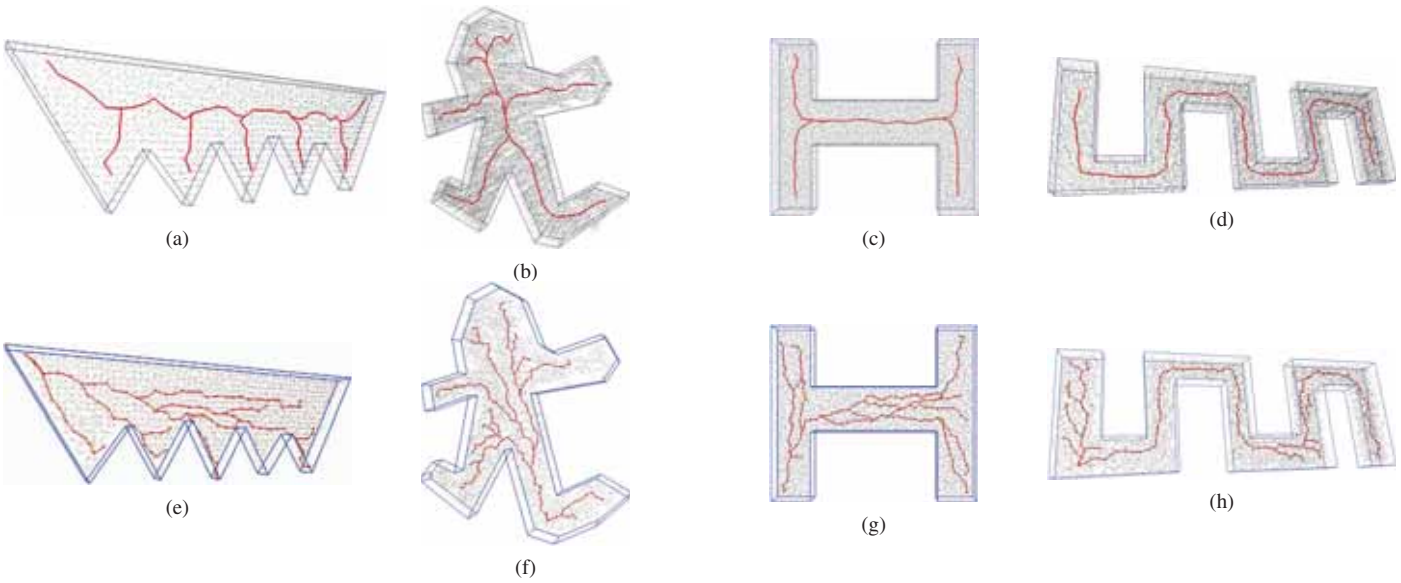


Fig. 8: Skeleton extraction under different 3D scenarios. Row 1: our algorithm; row 2: naive algorithm using flow complex. (a) Seabed, 5,537 nodes, avg. deg. 22.35; (b) Man, 15,288 nodes, avg. deg. 20.78; (c) H, 16,156 nodes, avg. deg. 22.47; (d) Snake, 19,313 nodes, avg. deg. 19.84.

IV. Performance Evaluation

A. Simulation Setup

To demonstrate the efficiency of the proposed algorithm, we conduct extensive simulations on different 2D/3D scenarios, and compare the proposed algorithm with the existing algorithms. Specifically, for 3D sensor networks, we compare with the naive algorithm using traditional flow complex. Note that the flow complex based algorithm suffers from local maxima and incurs a self-disconnected skeleton, as mentioned earlier in Section I. For fair comparison, similar with our algorithm, the naive algorithm first identifies skeleton nodes by using flow complex, and then selects the skeleton node with the largest distance transform as a root to build a skeleton tree, and finally the skeleton tree is refined based on branch similarities. And for 2D sensor networks, we compare our algorithm with MAP [5], [6] and CASE [19], [20], both of which require complete boundary information. Besides, we also study the performance of our algorithm under different scales of boundary noise.

In our simulations, nodes are randomly deployed in the sensing field. The communication radio model is unit disk graph. That is, a link between two nodes exists if their separation is less than the communication radio range. The threshold for importance measure is user-defined and in our algorithm, it is set to be 0.02 unless otherwise stated.

B. Simulation Results

1) Performance under different scenarios: Fig. 8 illustrates the results of our algorithm and naive algorithm on a variety of 3D scenarios, namely, Seabed (Fig. 8(a)(e)), Man (Fig. 8(b)(f)), H (Fig. 8(c)(g)) and Snake (Fig. 8(d)(h)), with the number of nodes ranging from 5,537 to 19,313. In Fig. 8(a)(e), the Seabed-shaped network has 5,537 nodes with five deep *valleys*, which are unlikely caused by boundary noise. Consequently, the extracted skeleton by our algorithm (see Fig. 8(a)) has five branches to reflect the details of the network, accurately capturing the main topological feature of the original network. While the skeleton by naive algorithm (see Fig. 8(e)) has many "parallel branches" because the traditional flow complex based algorithm identifies too many skeleton nodes due to the presence of parallel boundaries, and the refinement process has no guarantee to achieve a good skeleton.

Fig. 8(b)(f) describes the extracted skeleton of a Man-shaped network by our algorithm and naive algorithm, respectively. Note that there are five *logical* parts (i.e., head, two arms and two feet). Clearly, our algorithm delivers a medially-placed line-like skeleton and reflects these topological features very well, outperforming naive algorithm. Note that in Fig. 8(b), three skeleton branches occur in the head part because there the network is comparatively sparser, incurring many nodes have a feature loop that decomposes the boundary into connected components, and hence identify themselves as skeleton nodes. By setting an appropriate threshold for the importance measure, these skeleton nodes could be deleted, at the expense of other skeleton nodes at the end of arms and feet being disregards as skeleton nodes.

We can see the similar results in Fig. 8(c)(g) and Fig. 8(d)(h). Fig. 8(c)(g) illustrates the line-like skeleton of an H-shaped network by our algorithm and naive algorithm. Obviously, the extracted skeleton by algorithm is better which is well centered and homotopic to the underlying environment. In Fig. 8(d)(h), there is a long and narrow corridor, but we still achieve a line-like skeleton that captures the network topology very well, outperforming naive algorithm.

Fig. 9 depicts the comparison results of MAP, CASE and the proposed algorithm under 2D scenarios. We can clearly see that the skeletons generated by MAP (shown in Fig. 9(b)) have many spurious branches due to the existence of unstable skeleton nodes, even though the skeleton nodes with the geodesic

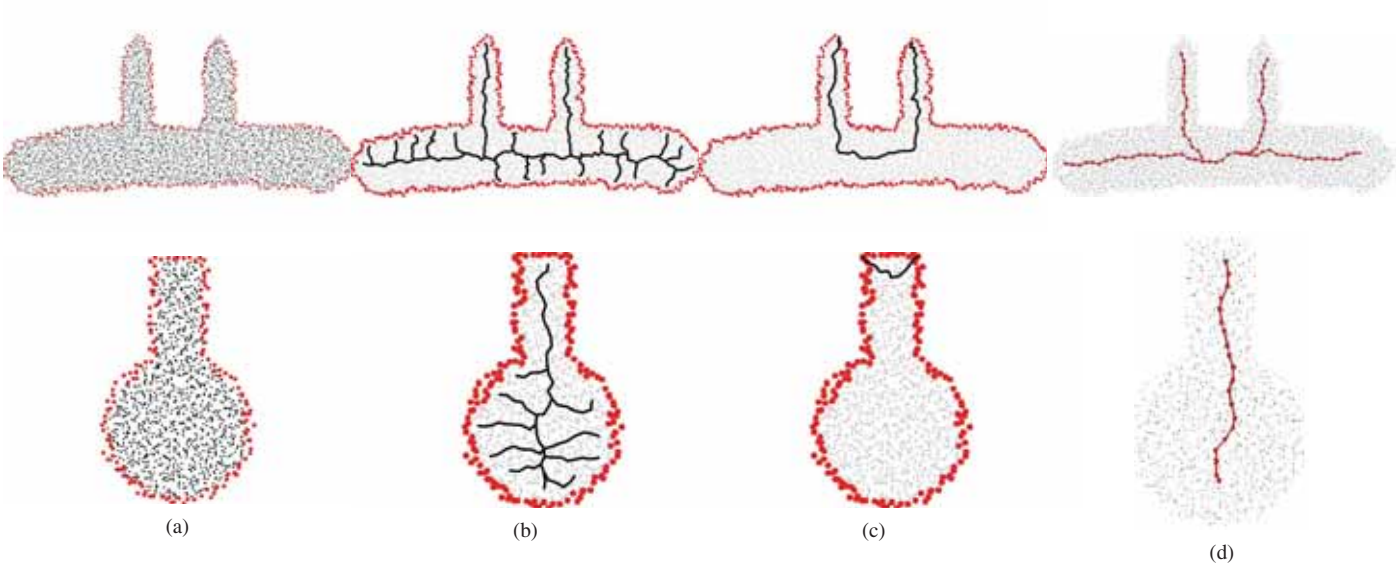


Fig. 9: Skeleton extraction under different 2D scenarios. Row 1: terminal-shaped network; row 2: bat-shaped network. (a) Original network; (b) MAP [5], [6]; (c) CASE [19], [20]; (d) Our algorithm.

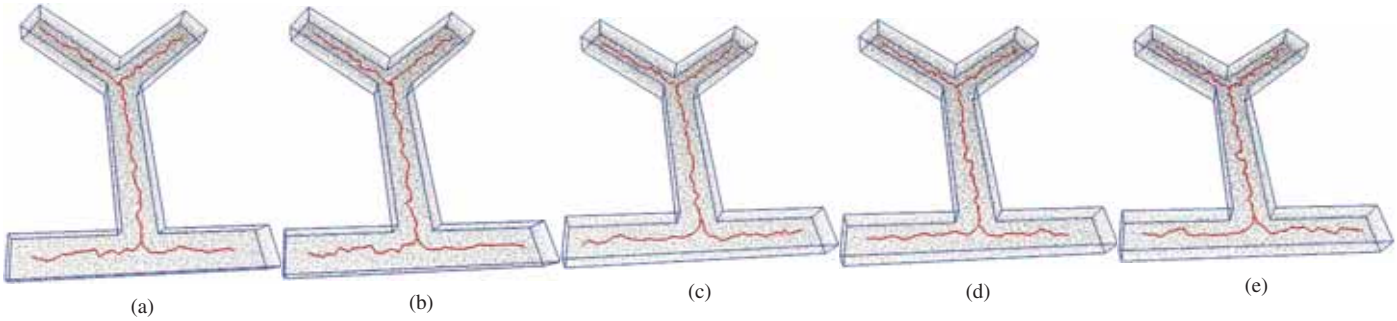


Fig. 10: The impact of boundary noise on the skeleton extraction of the 3D network in Fig. 4. The boundary noise are generated by adding to the boundary nodes random perturbations following a normal distribution with 0 mean and standard deviations in terms of the times of the communication radio range r. (a) $0.054r$; (b) $0.072r$; (c) $0.090r$; (d) $0.108r$ (e) $0.126r$.

distance (in hops) between the nearest boundary nodes less than 4 have already been regarded as unstable skeleton nodes and thus eliminated, implying that MAP is sensitive to the boundary noise; and CASE ignores some important skeleton branches, as shown in Fig. 9(c), since the true skeleton nodes, whose nearest boundary nodes locate at the same boundary branch due to the improperly chosen threshold of the corner node, did not identify themselves as skeleton nodes. Our algorithm produces the best skeletons (shown in Fig. 9(d)) which accurately capture the salient topological features of the underlying networks, outperforming the other two schemes.

2) Robustness to boundary noise: To investigate the robustness of our algorithm to boundary noise, we add to the boundary nodes of the 3D network in Fig. 4 some random perturbations following a normal distribution with zero mean and standard deviations in terms of the times of the communication radio range, as shown in Fig. 10. We can clearly see that with the increasing scale of the boundary noise, the extracted skeletons exhibit more and more irregularities, but the major topological features of the original network are still accurately reflected by the extracted skeletons, implying that our algorithm is robust to boundary noise and always yields a stable result.

V. Conclusion

We have proposed a unified framework for the line-like skeleton extraction in 2D/3D sensor networks, where an interior node identifies itself as a skeleton node if its geodesic shortest paths between the feature nodes form at least one loop to decompose the network boundary into two or more connected components, based which the importance measure of a skeleton node is derived. The importance measure is monotonically increasing such that the identified skeleton nodes are self-connected. We finally conduct a pruning process based on the proposed metric named branch similarity to yield the final skeleton. The proposed algorithm is fully distributed, scalable, with reliance on mere connectivity information. To the best of our knowledge, this is the first work on skeleton extraction in 3D sensor networks. Particularly, we conduct the first step of formalizing a unified skeleton for both 2D and 3D sensor networks.

In the future we will study its uses in applications, such as segmentation [21], [29], localization [35], navigation [37], routing, data collection [17], [18], [16] and data storage [13], [30], [31], [33], etc., especially for 3D sensor networks, which have attracted the attention of many researchers.

ACKNOWLEDGMENT

The comments from the anonymous reviewers of ICNP helped greatly for this paper. This work was supported in part by the National Natural Science Foundation of China and Microsoft Research Asia under Grant 60933012; by the National Natural Science Foundation of China under Grant 61073147, Grant 61202460 and Grant 61271226; by the National Natural Science Foundation of Hubei Province under Grant 2011CDB044; by the CCF-Tencent Open Research Fund; and by the Program for New Century Excellent Talents in University under Grant NCET-10-408 (State Education Ministry). The corresponding author of this paper is Hongbo Jiang.

REFERENCES

[1] N. Amenta, M. Bern, and M. Kamvysselis. A new voronoi-based surface reconstruction algorithm. In *Proc. of ACM SIGGRAPH*, 1998.

[2] D. Attali, J.-D. Boissonnat, and A. Lieutier. Complexity of the delaunay triangulation of points on surfaces: the smooth case. In *Proc.of the nineteenth annual symposium on Computational geometry*, 2003.

[3] O. K.-C. Au, C.-L. Tai, H.-K. Chu, D. Cohen-Or, and T.-Y. Lee. Skeleton extraction by mesh contraction. In *Proc. of ACM SIGGRAPH*, 2008.

[4] S. Bouix and K. Siddiqi. Divergence-based medial surfaces. In *Proc. of European Conference on Computer Vision*, 2000.

[5] J. Bruck, J. Gao, and A. A. Jiang. Map: Medial axis based geometric routing in sensor networks. In *Proc. of ACM MOBICOM*, 2005.

[6] J. Bruck, J. Gao, and A. A. Jiang. Map: Medial axis based geometric routing in sensor networks. *Wireless Networks*, 13(6):835–853, 2007.

[7] C. Buragohain, D. Agrawal, and S. Suri. Distributed navigation algorithms for sensor networks. In *Proc. of IEEE INFOCOM*, 2006.

[8] N. Cornea, D. Silver, and P. Min. Curve-skeleton applications. In *Proc. of IEEE Visualization*, 2005.

[9] T. K. Dey, J. Giesen, and S. Goswami. Shape segmentation and matching with flow discretization. In *In Proc. Workshop on Algorithms and Data Structures*, pages 25–36, 2003.

[10] T. K. Dey, J. Giesen, and S. Goswami. Shape segmentation and matching from noisy point clouds. In *Proc. of the First Eurographics conference on Point-Based Graphics*, 2004.

[11] T. K. Dey and J. Sun. Defining and computing curve-skeletons with medial geodesic function. In *Proc. of Symposium on Geometry Processing*, 2006.

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

[13] Q. Fang, J. Gao, and L. J. Guibas. Landmark-based information storage and retrieval in sensor networks. In *Proc. of IEEE INFOCOM*, 2006.

[14] H. Federer. *Geometric measure theory (Classics in Mathematics)*. Springer, 1996.

[15] S. P. Fekete, A. Kroller, D. Pfisterer, S. Fischer, and C. Buschmann. Neighborhood-based topology recognition in sensor networks. In *Proc. of the 1st Int. Workshop on Algorithmic Aspects of Wireless Sensor Networks*, 2004.

[16] H. Jiang, J. Cheng, D. Wang, C. Wang, and G. Tan. A general framework for efficient continuous multi-dimensional top-k query processing in sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 23(9):1668–1680, 2012.

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

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

[19] H. Jiang, W. Liu, D. Wang, C. Tian, X. Bai, X. Liu, and W. Liu. Case: Connectivity-based skeleton extraction in wireless sensor networks. In *Proc. of IEEE INFOCOM*, 2009.

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

[21] H. Jiang, T. Yu, C. Tian, G. Tan, and C. Wang. Consel: Connectivity-based segmentation in large-scale 2d/3d sensor networks. In *Proc. of IEEE INFOCOM*, 2012.

[22] H. Jiang, S. Zhang, G. Tan, and C. Wang. Cabet: Connectivity-based boundary extraction of large-scale 3d sensor networks. In *Proc. of IEEE INFOCOM*, 2011.

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

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

[25] S. Lederer, Y. Wang, and J. Gao. Connectivity-based localization of large scale sensor networks with complex shape. *ACM Transactions on Sensor Networks*, 5(4):31:1–31:32, 2009.

[26] M. Li, Y. Liu, J. Wang, and Z. Yang. Sensor network navigation without locations. In *Proc. of IEEE INFOCOM*, 2009.

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

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

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

[30] S. Ratnasamy, B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin, and F. Yu. Data-centric storage in sensornets with ght, a geographic hash table. *Mobile Networks and Applications*, 8(4):427–442, 2003.

[31] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govin-dan, and S. Shenker. Ght: A geographic hash table for data-centric storage in sensornets. In *Proc. of International Workshop on Wireless Sensor Networks and Applications*, 2002.

[32] D. Reniers, J. J. Wijk, and A.Telea. Computing multiscale curve and surface skeletons of genus 0 shapes using a global importance measure. *IEEE Transactions on Visualization and Computer Graphics*, 14(2):355–368, 2008.

[33] R. Sarkar, W. Zeng, J. Gao, and X. D. Gu. Covering space for in-network sensor data storage. In *Proc. of ACM/IEEE IPSN*, 2010.

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

[35] 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. of ACM MOBIHOC*, 2010.

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

[37] S. Xia, N. Ding, M. Jin, H. Wu, and Y. Yang. Medial Axis Construction and Applications in 3D Wireless Sensor Networks. In *Proc. of IEEE INFOCOM*, 2013.

[38] H. Zhou, Y. Wu, and M. Jin. A robust boundary detection algorithm based on connectivity only for 3d wireless sensor networks. In *Proc. of IEEE INFOCOM*, 2012.

[39] X. Zhu, R. Sarkar, and J. Gao. Shape segmentation and applications in sensor networks. In *Proc. of IEEE INFOCOM*, 2007.

[40] X. Zhu, R. Sarkar, and J. Gao. Segmenting a sensor field: Algorithms and applications in network design. *ACM Transactions on Sensor Networks*, 5(2):12:1–12:32, 2009.