Connectivity-Based Space Filling Curve Construction Algorithms in High Genus 3D Surface WSNs

CHEN WANG, HONGBO JIANG, and YAN DONG, Huazhong University of Science and Technology

Many applications in wireless sensor networks (WSNs) require that sensor observations in a given monitoring area are aggregated in a serial fashion. This demands a routing path to be constructed traversing all sensors in that area, which is also needed to linearize the network. In this article, we present SURF, a Space filling cURve construction scheme for high genus three-dimensional (3D) surFace WSNs, yielding a traversal path provably aperiodic (that is, any node is covered at most a constant number of times). SURF first utilizes the hop-count distance function to construct the iso-contour in discrete settings, and then it uses the concept of the Reeb graph and the maximum cut set to divide the network into different regions. Finally, it conducts a novel serial traversal scheme, enabling the traversal within and between regions. To the best of our knowledge, SURF is the first high genus 3D surface WSN targeted and pure connectivity-based solution for linearizing the networks. It is fully distributed and highly scalable, requiring a nearly constant storage and communication cost per node in the network. To incorporate adaptive density of the constructed space filling curve, we also design a second algorithm, called SURF$^+$, which makes use of parameterized spiral-like curves to cover the 3D surface and thus can yield a multiresolution SFC adapting to different requirements on travel budget or fusion delay. The application combining both algorithms for in-network data storage and retrieval in high genus 3D surface WSNs is also presented. Extensive simulations on several representative networks demonstrate that both algorithms work well on high genus 3D surface WSNs.

Categories and Subject Descriptors: C.2.2 [**Computer-Communication Networks**]: Network Protocols

General Terms: Design, Algorithms, Performance

Additional Key Words and Phrases: 3D surface WSNs, high genus, space filling curve, iso-contour, in-network data storage and retrieval, connectivity-based algorithms

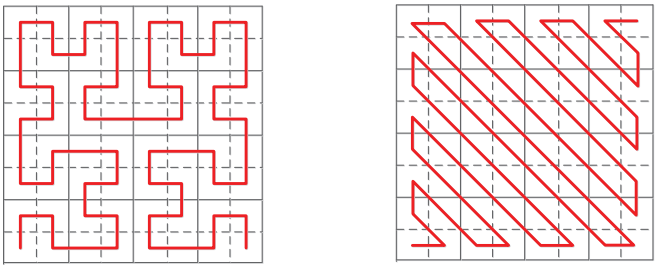
ACM Reference Format:
Chen Wang, Hongbo Jiang, and Yan Dong. 2016. Connectivity-based space filling curve construction algorithms in high genus 3D surface WSNs. ACM Trans. Sen. Netw. 12, 3, Article 22 (August 2016), 29 pages.
DOI: http://dx.doi.org/10.1145/2907947

1. INTRODUCTION

The wireless sensor network (WSN) community has envisioned a large variety of ubiquitous monitoring and actuation applications [Liu et al. 2013; Yang 2014; Lin et al.

This work was supported in part by the National Natural Science Foundation of China under Grants No. 61572219, No. 61502192, No. 61271226, No. 61272410, No. 61202460, and No. 61471408; by the National High Technology R&D Program ("863" Program) of China under Grants No. 2014AA01A701 and No. 2015AA011303; by the National Natural Science Foundation of Hubei Province under Grant No. 2014CFA040; by the China Postdoctoral Science Foundation under Grant No. 2014M560608; by the Fundamental Research Funds for the Central Universities under Grants No.2015QN073, No.2016YXMS297 and No.2016JCTD118; and by the Science and Technology Plan Projects of Wuhan City under Grant No. 2015010101010022. An earlier version of this work appeared as Wang and Jiang [2015].
Authors' addresses: C. Wang, H. Jiang, and Y. Dong (corresponding author), School of Electronic Information and Communications, Huazhong University of Science and Technology, P. R. China; emails: {cwangwhu, hongbojiang2004, eiyandong2015}@gmail.com.

DOI: http://dx.doi.org/10.1145/2907947



(a) The Hillbert curve. (b) The Cantor curve.

Fig. 1. Two typical 2D SFCs.

2015]. Although two-dimensional (2D) planar and simple 3D volume settings are assumed in most earlier studies on WSNs [Sarkar et al. 2013; Tan et al. 2014; Li et al. 2014; Jiang et al. 2015], there have been emerging interests in scenarios where sensors are typically deployed in complex-connected 3D surfaces [Yu et al. 2012, 2013; Wang et al. 2015]. Examples of such applications include the fire prevention in the corridors of buildings [Fischer and Gellersen 2010], the monitoring of coal mine tunnels for disaster warning [Li and Liu 2009], as well as the monitoring in underground tunnels used in gas, water or sewer systems [Akhondi et al. 2010]. These kinds of WSNs, often of a complex-connected 3D setting with non-trivial topology, are modeled as high genus 3D surface WSNs [Yu et al. 2012, 2013; Wang et al. 2015], where genus[1] can be simply regarded as the handle on the sphere.

In this article we focus on constructing a space filling curve (SFC) to linearize a high genus 3D surface WSN, that is, "traversing" the high genus 3D surface network by a single path. In the following, we look back on related works of the SFC and its applications and constructions in WSNs, offering a full-spectrum understanding toward its advancement in WSNs, followed by our contributions.

1.1. Existing Work

1.1.1. The Concept of the SFC. The concept of the SFC came out in the late 19th century and is accredited to Peano, who proved the existence of a curve that passes through every point of a closed unit square [Peano 1890]. Thereafter, various SFCs have been proposed for 2D and 3D settings [Sagan 1994]. Most of these curves are recursively constructed, for example, the Hillbert curve in Figure 1(a) with respect to three iterations, while some are non-recursive, for example, the Cantor curve in Figure 1(b). However, in continuous settings almost all existing SFCs are constructed in a square or cube region, and little work has been done to extend the SFC construction to other shapes, let alone solutions in discrete settings.

1.1.2. Applications of the SFC in WSNs. In discrete WSNs, enforcing a linear order of the sensor nodes via the SFC has broad applications, one of which is the serial operation on both sensor nodes and sensor data. In Patil et al. [2004] and Mostefaoui et al. [2015], serial fusion techniques were proposed for collaborative signal detection by the traversal along a SFC, which ensures visiting the nodes in a linear order. This kind of technique enables the detection process to stop as soon as sufficient evidences have been collected. In Viana et al. [2006], the authors presented a dual addressing space representation architecture for self-organizing networks. In Chung et al. [2011],

[1]Genus is one of the most fundamental concepts in the field of algebraic topology [Massey 1987], which can be simply regarded as the number of handles on the sphere [Hatcher 2002]. Its formal definition is presented in Section 2.1. Without leading to confusion, we interchangeably use the terms "genus" and "handle."

the authors proposed an algorithm for processing similarity search queries in WSNs. They both applied the Hilbert curve to linearize the network, establishing a bijective mapping between the SFC and the node's index/logical address, so standard 1D indexing/addressing mechanisms can be applied.

Another application of the SFC in WSNs is the motion planning of mobile beacons. Bahi et al. [2008] and Koutsonikolas et al. [2007] considered the localization of the WSN, where a single mobile beacon aware of its position was utilized to help other nodes to localize themselves by moving to cover the entire network. The paths that the mobile beacon should travel along are designed to follow the SFCs. Similar ideas can also be helpful for the battery recharge of the sensors [Xie et al. 2012] or the data collection by the data mules near the sink [Sugihara and Gupta 2011].

1.1.3. SFC Construction in WSNs. The previous works mainly focus on applying the SFC for various purposes in WSNs. How to construct the SFCs, however, is not their concentration. The only research concentrates on constructing the SFCs in WSNs is presented in Ban et al. [2013]. The proposed method can generate an SFC that densely covers any planar geometric domain, with a coverage density proportional to the length of the curve, but it cannot be directly utilized in 3D scenarios. A practically potential solution for visiting nodes in a 3D network could be based on the random walk [Spitzer 2001]. The problem, however, is that when choosing its next hop, the random walk is essentially "blind." Intuitively, a random walk visits a new node with high probability at the initial stage. But after a certain proportion of nodes have been visited, the random walk is more likely to take a (infinite) long period to aimlessly walk in the network, anchoring its hope on encountering the last few unvisited nodes earlier. Therefore, the random walk cannot ensure deterministic node covering results with restricted traversal path length.

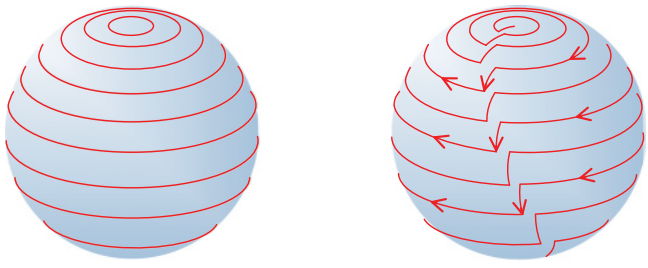
It is noted that the task of SFC construction differs significantly from the packet routing path design in high genus 3D surface WSNs. Packet routing aims at connecting two given nodes by a routing path. In high genus 3D surface WSNs, the difficulty mainly lies in naming/addressing the sensor nodes before routing among them. Therefore, existing works [Yu et al. 2012, 2013; Wang et al. 2015] focus on seeking efficient naming/addressing schemes, with different network decomposition methods. However, the goal of SFC construction is to design a path traversing all sensor nodes, which is a coverage-critical selection during the traversal. In this case, naming/addressing the nodes is no longer the primary concern; the well-designed traversal scheme is.

1.2. Challenges and Our Contributions

In this article, we present SURF, a novel Space filling cURve construction scheme for high genus 3D surFace WSNs, yielding a path provably aperiodic (any node in the path is covered/visited at most a constant number of times). The name of our proposed scheme, SURF, can be interpreted as the surface filling or surfing on 3D surfaces.

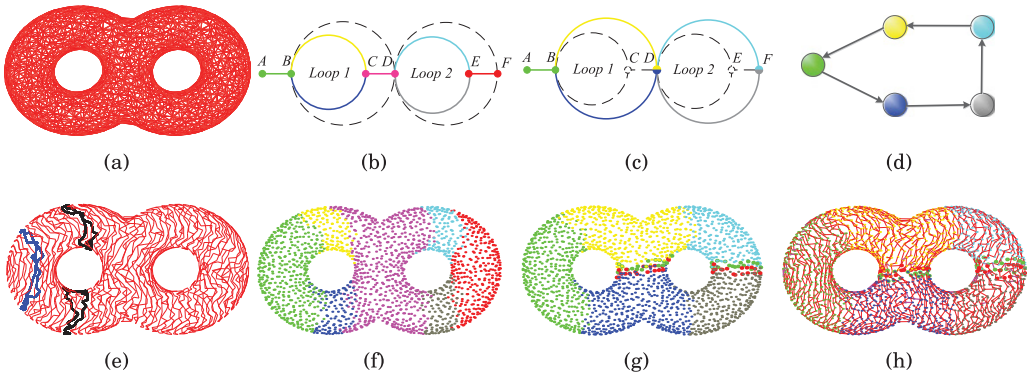
The intuition of SURF stems from the observation in the continuous domain—the iso-contours of a closed spherical (genus-0) surface in the smooth setting naturally form an embryonic form of the SFC (Figure 2(a)), and directly connecting them forms an SFC, shown in Figure 2(b). With regard to high genus surfaces, the primary challenge lies in the way we deal with the genuses, which may incur two or more contours of an iso-value (see Figure 4(a)). There is only one contour of the iso-value 1, but two contours of the iso-value 10 due to the existence of the genus.

The main idea behind SURF is to divide the surface into a set of genus-0 regions by cutting off the genuses, followed by constructing the SFC in each region separately before connecting all SFCs in a practical way. The particular problem we are facing in discrete networks, however, is more challenging: (1) How do we define the iso-contour



(a) (b)

Fig. 2. Connecting iso-contours forms a SFC of a sphere. (a) Iso-contours of a shpere. (b) A SFC of a sphere.



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

(e) (f) (g) (h)

Fig. 3. The pipeline of SURF algorithm. (a) The triangulated genus-2 torus network with 1,924 nodes; Avg deg is 17.7. (b) The Reeb graph of the network. (c) The Reeb graph after the bisection operation. (d) The traversal sequence of SURF in different regions. (e) The iso-distance contour lines of the network. (f) The regions of the Reeb graph in (b). (g) The regions of the Reeb graph in (c); the cut pairs are represented by red and green dots. (h) The SFC generated by SURF, with link edges colored in pink. Different arcs and regions of the Reeb graph are distinguished by colors.

in a discrete 3D surface network with mere connectivity information, such that the iso-contour is a connected and closed curve? (2) How do we identify genuses and further cut them off to form different regions? (3) How do we ensure that the length of the generated SFC is bounded?

To address the first challenge, we propose to use the hop-count distance function to construct the iso-contour in discrete settings (detailed in Section 3.1). For the second one, the concept of the Reeb graph and the maximum cut set are utilized to realize the network segmentation (detailed in Section 3.2). After these procedures, the network is divided into different regions. We design schemes to guarantee the traversal within and between regions, yielding an SFC that is provably aperiodic (see Theorem 4.1 in Section 3.3). The whole process, as an example, is shown in Figure 3.

To the best of our knowledge, SURF is the first high genus 3D surface WSNs targeted and pure connectivity-based solution for the SFC construction in WSNs. SURF offers several salient features. First, it requires connectivity information only, without the reliance on the location or distance measurement. Second, it does not rely on any particular communication model, only assuming a constant maximum transmission range, which is a common case in practical WSNs. Third, it is fully distributed and scalable, with a nearly constant storage and communication cost of every node.

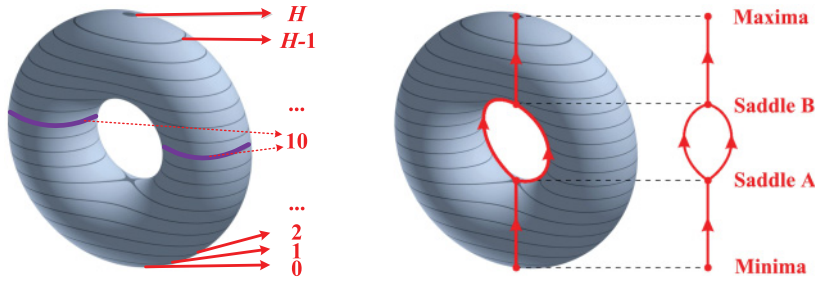


Fig. 4. Iso-distance function and its Reeb graph on a torus.

In addition, we also design a second algorithm, called SURF$^+$, that incorporates adaptive density of the constructed SFC and thus can yield a multiresolution SFC adapting to different requirements on travel budget or fusion delay. We also discuss principles and implementations of both algorithms, provide proof of their correctness and report on their performance evaluation through extensive simulations. Additional application combining both algorithms for in-network data storage and retrieval in high genus 3D surface WSNs is also presented.

The remainder of this article is organized as follows. In Section 2, we introduce some preliminary knowledge of our proposed SURF algorithm. In Section 3, we describe the implementation of SURF algorithm. Further discussions on the SURF algorithm, as well as the SURF$^+$ algorithm and the application for in-network data storage and retrieval are presented in Section 4. The performances of SURF, SURF$^+$, and the application are evaluated in Section 5. Finally, Section 6 concludes the article.

2. PRELIMINARY

Before digging into the problem of SFC construction, we first briefly introduce several notions and definitions in algebraic topology and computing geometry. For further theoretical background, we refer interested readers to Massey [1987], Hatcher [2002], and Pascucci [2011].

2.1. Cut and Genus

In algebraic topology, a *cut* C is referred to as a disjoint closed simple curve on a connected and orientable surface $\mathcal{M}$, where $\mathcal{M}$ is orientable, indicating it has two distinct sides. One notable property of a cut is its ability to *locally* disconnect the topology of $\mathcal{M}$. Suppose a set $\mathcal{C} = \{C_1, C_2, \ldots, C_n\}$ is a cut set on $\mathcal{M}$, whose cardinality is n, and then $\mathcal{C}$ is a *maximum cut set* $\mathcal{C}_{max}$ of $\mathcal{M}$ if and only if (1) any two cuts in $\mathcal{C}_{\max}$ belong to different homotopy classes, that is, one cut cannot be smoothly deformed to another without leaving the surface; (2) $\mathcal{M}\backslash(C_1 \cup C_2 \cup \ldots \cup C_n)$ is connected; and (3) $\mathcal{M}\backslash(C_1 \cup C_2 \cup \ldots \cup C_{n+1})$ is disconnected. Accordingly, the *genus* of $\mathcal{M}$ is defined as the cardinality of the $\mathcal{C}_{max}$ of $\mathcal{M}$, indicating the maximum number of cuts without rendering $\mathcal{M}$ disconnected [Erickson and Har-Peled 2004].

The notion of the genus is closely associated with the classification of orientable closed surfaces up to homeomorphism: For a given integer $n \geq 0$, there is exactly one topological type, "the surface of genus-n," which can be obtained by attaching n handles onto a simple closed 3D surface. For example, a sphere is a genus-0 surface, and any cut will render it disconnected; a torus is a genus-1 surface, as shown in Figure 4, with at most one cut on it while not leading to its disconnection.

Motivated by Yu et al. [2013] where the surface $\mathcal{M}$ is decomposed to a simple genus-0 topology using cuts, our idea is to exploit a similar method to generate a sliced surface with which SURF is able to extract the iso-distance contour and the Reeb graph, and we hereby put forward a novel serial traversal scheme for the SFC construction.

2.2. Iso-distance Contour

For a scalar, real-valued function $f : \mathcal{M} \to \mathbb{R}$, the *level set* of an *iso-value* h is the set of points $f^{-1}(h) = \{p \in \mathcal{M} | f(p) = h\}$ [Carr et al. 2004]. A *contour* is a connected component of a level set, that is, a curve along which f has a constant value [Gray et al. 2006]. Figure 4(a) illustrates a set of successive iso-distance contours on a torus with a mapping to an integer set $\{0, 1, \ldots, H\}$ of different height values.

In a discrete network with mere connectivity information, however, it is not straightforward to define the level set where no height value or geodesic distance[2] can be derived. What is more, even if we can find a metric to represent the level set, it is still unclear whether there is a connected component to form the contour.

To tackle this problem, SURF makes use of the hop-count distance, an analogy of the Euclidean/geodesic distance in continuous domains, to define the real-valued function f. Meanwhile, SURF is designed to ensure the existence of the contours (detailed in Section 3.1). Note that one challenge here is, when a level set is separated to two or more contours due to the existence of the genuses (e.g., $f^{-1}(10)$ in Figure 4(a)), SURF is supposed to have the ability to find out where the genuses are. For this purpose, the Reeb graph is used to extract a maximum cut set that cuts off the genuses of $\mathcal{M}$.

2.3. Reeb Graph and Cut Identification

A Reeb graph is a topological structure proposed in Reeb [1946]. Briefly speaking, a Reeb graph $\mathcal{R}$ of a real-valued function f explicitly reveals the evolution of its level set $f^{-1}(\cdot)$. When the number of the contours of $f^{-1}(\cdot)$ increases or decreases, the gradient of f will vanish at the separating points of the contours. Those points are called *critical points* (theoretically, there are three types of critical points, namely, *minima*, *saddles*, and *maxima* [Reeb 1946].) of f, for example, Saddle A and Saddle B in Figure 4(b), which shows an example of the Reeb graph.

Given a Reeb graph, we turn to extracting a maximum cut set $\mathcal{C}_{\max}$ from $\mathcal{M}$. Our approach is motivated by the following theorem.

THEOREM 2.1. *The Reeb graph of a closed orientable genus-n 2-manifold has exactly n loops [Cole-McLaughlin et al. 2003].*

Theorem 2.1 implies that we can first identify all loops of the Reeb graph, thereby finding a cut for each loop. Specifically, a loop in a Reeb graph is associated with two degree-3 nodes: One starts the loop and the other ends it; see the two saddles in Figure 4(b). Thus, we have

Definition 2.2. An arc of the Reeb graph of $\mathcal{M}$ is a *loop-end arc*, if it is merged from two different arcs.

See Figure 3(b) for an example where the blue and yellow arcs are merged into a loop-end one colored in pink.

COROLLARY 2.3. *Each loop in the Reeb graph of $\mathcal{M}$ corresponds to one loop-end arc.*

As such, in order to identify a cut for one loop, our method is to find the bisection in loop-end arc that disconnects this loop. Figure 3(c) shows the Reeb graph after

[2]The geodesic distance on a 3D surface can be regarded as the (locally) shortest path between two nodes on the surface.

the bisection operation. We will present the implementation in a discrete network in Section 3.2.

3. SURF ALGORITHM

Given a high genus 3D surface WSN, SURF is derived from its triangular form. However, the triangulation procedure itself is out of the scope of this work. Numerous recent studies, for example, Funke and Milosavljevic [2007], Sarkar et al. [2009], and Zhou et al. [2011], have proposed simple and distributed algorithms to obtain the triangular structure and can be used in conjunction with our approach. The triangular structure offers a shape representation of the high genus 3D surface, as shown in Figure 3(a). Without leading to confusion, hereafter we still refer to this triangular structure as the high genus 3D surface, denoted by $\mathcal{M}$, with its vertex (node) set $\mathcal{V} = \{v_i\}$ and edge set $\mathcal{E} = \{e_{ij} = (v_i, v_j) | v_j$ is called the *neighbor* of $v_i\}$. Given a triangular structure, SURF follows three steps for the SFC construction:

(1) *Contour Construction*: to lay the groundwork for regional division (see Figure 3(e)).
(2) *Maximum Cut Set Identification*: to cut off the genuses and thus divide the network to different regions (see Figure 3(g)).
(3) *Serial Traversal Scheme*: to finally construct the SFC by the traversal intra- and inter-regions (see Figure 3(h)).

3.1. Contour Construction

The first step of SURF is to construct the level set and its corresponding contour lines[3] of $\mathcal{M}$ so, in each contour line, it is trivial to locally construct a SFC. To that end, we first establish a hop count distance function $f : \mathcal{M} \to L$, where L is the integer set representing the hop count. Specifically, a randomly selected root node r initiates a flooding across the whole network. After receiving a flooded message from r, every node knows its hop count distance l to r, and then records its level index with l. Therefore, for any node v_i with the level index l, we have $f(v_i) = l$. Accordingly, the *level set* of hop count distance l is given by $f^{-1}(l) = \{e_{ij} = (v_i, v_j) \in \mathcal{E} \mid f(v_i) = f(v_j) = l\}$.

Recall that a contour in continuous settings is a connected component of a level set. In the following, we show that in discrete networks, a connected contour line of a level set also exits, which is defined based on the following notion.

Definition 3.1. The *l-neighbor* for any edge e_{ij} in $f^{-1}(l)$ is defined as $N(e_{ij}, l) = \{(e_{ij}, e_{ik}) \mid e_{ij}$ and e_{ik} have a common vertex $v_i \in \mathcal{V},$ and $e_{ij}, e_{ik} \in f^{-1}(l)\}$.

Then we introduce the concept of iso-distance contour line and its property in discrete settings.

Definition 3.2. An *iso-distance contour line* (*iso-contour* for short) of $f^{-1}(l)$, $\mathcal{O}(l)$, is defined as a neighbor graph of $f^{-1}(l)$, such that:
(1) $\bigcap_{\forall \mathcal{O}(l)} \mathcal{O}(l) = \varnothing$; and (2) $\bigcup_{\forall \mathcal{O}(l)} \mathcal{O}(l) = \bigcup_{e_{ij} \in \mathcal{E}} N(e_{ij}, l)$.

As the iso-contour of $f^{-1}(l)$ is defined as the neighbor graph of $f^{-1}(l)$, we have the following lemma.

LEMMA 3.3. *Any two nodes in an iso-contour of $f^{-1}(l)$ is l-connected, where two nodes in $f^{-1}(l)$ is l-connected, if between them there is a path the nodes on which are all in $f^{-1}(l)$.*

[3]We use the "contour line" in discrete network settings to distinguish it from the "contour" in continuous scenarios.

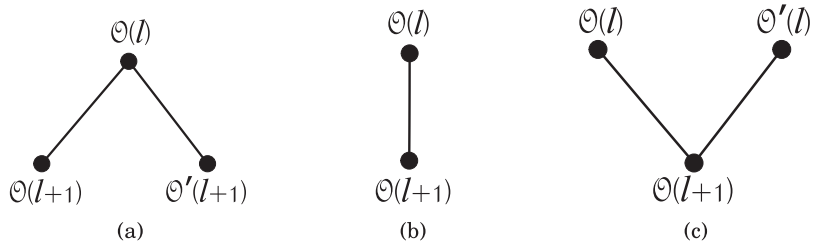


Fig. 5. (a) The iso-contour $\mathcal{O}(l)$ is connected with $\mathcal{O}(l+1)$ and $\mathcal{O}'(l+1)$. (b) The iso-contour $\mathcal{O}(l)$ is connected with $\mathcal{O}(l+1)$ only. (c) The iso-contour $\mathcal{O}(l+1)$ is connected with $\mathcal{O}(l)$ and $\mathcal{O}'(l)$.

Lemma 3.3 implies the connectivity of an iso-contour, while there is no guarantee of its closeness. In fact, there exist special nodes, so-called "dead-ends," rendering the iso-contour unclosed. They are closely related to the following parent/child relationship.

Definition 3.4. For any $e_{pc} = (v_p, v_c) \in \mathcal{E}$, if $f(v_p) = l$, $f(v_c) = l+1$, then v_p is a *parent* node of v_c; v_c is a *child* node of v_p.

Definition 3.5. A node is a *dead-end* if it has no child nodes. Correspondingly, an edge $e_{ij} = (v_i, v_j)$ is a *dead-end edge* if either v_i or v_j is a dead-end.

We then have the following theorem that guarantees the closeness of an iso-contour.

THEOREM 3.6. *An iso-contour with dead-end edges eliminated is a connected and closed cycle.*

PROOF. See Appendix A. □

So far, it is trivial to establish the discrete counterpart of a continuous contour: to treat every dead-end edge as a double-edge (detailed in Section 3.3). For high genus surfaces where there exist two or more iso-contours in a level set (see the two iso-contours colored in black of $f^{-1}(14)$ in Figure 3(e)), we utilize the Reeb graph to cut off the genuses and thus divide the network into regions. Further strategies are then proposed to guarantee the traversal intra- and inter-regions.

3.2. Maximum Cut Set Identification

Based on the hop count distance function, we next use the Reeb graph to identify the maximum cut set of $\mathcal{M}$. To that end, a distributed algorithm similar to that in Yu et al. [2013] is carried out, which evolves through four major sub-steps.

The first sub-step is to identify nodes in each iso-contour of $f^{-1}(l)$ with an iso-contour ID. This is done by randomly selecting one landmark (so-called g-landmark) in an iso-contour. After the g-landmark is selected, it performs flooding within $f^{-1}(l)$, with the messages containing its iso-contour ID and level index l. As such, all nodes in the iso-contour of $f^{-1}(l)$ have the knowledge of the iso-contour ID.

Second, all the iso-contours in $\mathcal{M}$ are composed to *regions* (arcs) of the Reeb graph. We say two iso-contours $\mathcal{O}(l)$ and $\mathcal{O}(l+1)$ are connected if there exists a node v in $\mathcal{O}(l)$ that has a neighbor v' in $\mathcal{O}(l+1)$. Then, v and v' will notify this connection to their corresponding g-landmarks. In particular, there exist three cases related to this connection, as shown in Figure 5. Correspondingly, the g-landmark notifies all nodes in $\mathcal{O}(l+1)$ if $\mathcal{O}(l+1)$ is only connected with $\mathcal{O}(l)$. In this case, the nodes in $\mathcal{O}(l+1)$ are assigned a region ID the same as that of $\mathcal{O}(l)$; otherwise, the nodes in $\mathcal{O}(l+1)$ will be assigned a new region ID.

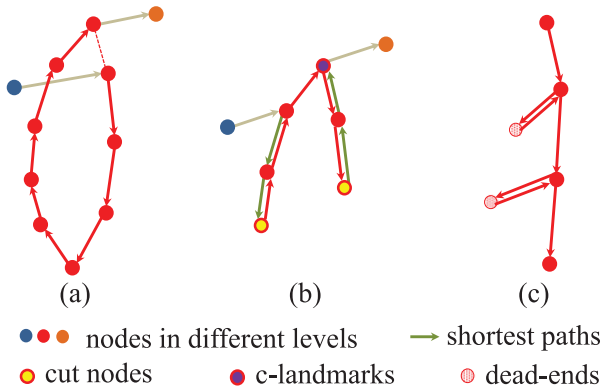


Fig. 6. Several cases in serial traversal. (a) The traversal in an uncut iso-contour. (b) The traversal in a cut iso-contour. (c) The traversal in an iso-contour with dead-ends.

With the aforementioned two sub-steps, every node has a region ID. The result of the Reeb graph is shown in Figure 3(f), where the Reeb graph regions are distinguished in colors.

Next, having the Reeb graph, all loop-end regions (arcs) can be notified directly: If the g-landmark in $\mathcal{O}(l+1)$ is notified that it is connected with $\mathcal{O}(l)$ and another iso-contour $\mathcal{O}'(l)$ (the case in Figure 5(c)), then nodes in $\mathcal{O}(l+1)$ and all other iso-contours in the same region are notified to be in a loop-end region.

Finally, to extract the maximum cut set $\mathcal{C}_{\max}$, each loop-end region performs a *bisection* operation to extract a cut. Consequently, the loop-end region I_m is bisected and a *merged region* I'_α (respectively, I'_β) is generated. Then it is simple to identify a cut C_i: Each node v in loop-end region I_m sends a message to its neighbor v' in I_m. If v' has a different region ID with v, then v and v' are notified to be *cut nodes*. Figure 5(g) depicts the result of the emerged regions and cut pairs.

3.3. Serial Traversal Scheme

After cut identification, $\mathcal{M}$ is divided into regions, each of which contains several (uncut or cut) iso-contours. Aiming at a SFC for the whole network, we first conduct the SFC in each iso-contour, thereby connecting those curves for intra- and inter-regions.

The SFC construction starts from the root node r, which randomly chooses a neighbor as the next hop, say, p, and traverses to it. Then p marks itself as visited, and the traversal path $e(r, p)$ becomes the first section of the SFC. After that, p finds its next hop node q following S-NextHop in Algorithm 1, which deals with four situations according to the spatial relationship between p and q.

Situation 1: p and q are within one uncut iso-contour. This situation happens when p is not in the loop-end region and p has an unvisited neighbor q in the same iso-contour. Recall a simple way to generate a local SFC in an iso-contour in Theorem 3.6: In this situation, if p is a link node (the first visited node in its iso-contour), then the next couple of traversals are all of this situation, until p has no unvisited neighbor in the same iso-contour. Then Situation 3 occurs. See Figure 6(a) for an example.

Situation 2: p and q are within one cut iso-contour. This situation happens when p is in the loop-end region and p has an unvisited neighbor q in the same iso-contour. It is noted that a cut iso-contour is part of an uncut iso-contour, and cut nodes are end-points of cut iso-contours. This motivates us to start the traversal from a cut node in a cut iso-contour and to "exit" the cut iso-contour from a *c-landmark* (a randomly selected node having at least one non-dead-end child). This is done by a local flooding within

the cut iso-contour from each cut node and each c-landmark after cut identification. As a result, each node in the cut iso-contour knows its next-hop node on the shortest path to each cut node and each c-landmark. When p is a link node of a cut iso-contour, it first follows the shortest path pointer to one of the cut nodes in its iso-contour and then starts traversal as in Situation 1, until p has no unvisited neighbor in the same iso-contour. After that, the last node follows the shortest path pointer to the c-landmark in the iso-contour, and then Situation 3 occus. See Figure 6(b), for instance.

Situation 3: p and q are in two different iso-contours. This situation happens when p has no unvisited neighbor in its iso-contour in Situation 1 or p is a c-landmark without unvisited neighbor in its iso-contour in Situation 2. In the former situation, p will choose an unvisited neighbor in a neighboring iso-contour as its next hop (note that if all the unvisited neighbors of p are dead-ends, then p will choose its previous hop node as its next hop to guarantee that the link node in the next iso-contour is non-dead-end). In the latter situation, as c-landmark is selected with at least one non-dead-end child, it directly chooses an unvisited neighbor in a neighboring iso-contour as its next hop. If not, then Situation 4 occurs. See the link edges connecting the two iso-contours colored in pink in Figure 3(h).

Situation 4: p and q are in two different regions. This situation happens when p has no unvisited neighbor in the same region. To achieve a traversal inter-region, we use *r-landmarks* for piloting, where the r-landmarks are randomly selected from boundary nodes (not cut nodes and having neighbors in another region) in the same region. The r-landmark selection process can be done in a region similar to g-landmark selection in an iso-contour. As a result, each region has one or more r-landmarks (depending on how many regions it borders). Next, each r-landmark in one region initiates a flooding to know the next-hop node on the shortest path to those r-landmarks in its neighboring region. Meanwhile, by doing so, the high-order topological features (i.e., regions and their connections) of $\mathcal{M}$ are identified by r-landmarks. In this situation, p will choose an unvisited r-landmark that has a minimum differential value of level index as its next hop. As an example, Figure 3(d) shows the traversal order in different regions of the network in Figure 3(a).

Note that the NextHop algorithm does not consider q as a dead-end. That is because, on the one hand, dead-ends of an iso-contour in $f^{-1}(l)$ have no effect on the l-connectivity of other nodes in the iso-contour (by Lemma A.1) and, on the other hand, in our scheme several methods are utilized to ensure the link nodes are not dead-ends. So before p finds its next node by the NextHop algorithm, it first checks whether it has an unvisited dead-end neighbor d; if it does, it just goes to d, comes back, and continues to seek for its next hop; d marks itself visited. See Figure 6(c).

The path stretches as the serial traversal goes on and, in the end, an SFC of $\mathcal{M}$ is constructed, which seems like the shell on the shape of the network topology. Figure 3(h) shows the conducted SFC of the genus-2 torus network in Figure 3(a).

4. DISCUSSIONS

4.1. Discussions on SURF

4.1.1. Node's Covered Times. From the aforementioned serial traversal scheme, it is not hard to arrive at the following theorem that ensures the validity and feasibility of SURF.

Theorem 4.1. *The SFC generated by SURF ensures that every node in $\mathcal{M}$ is covered at most* $(\max_{v_i \in \mathcal{V}} n_d(v_i) + n_r + 2)$ *times, where n_r is the number of the regions in the network, and $n_d(v_i)$ is the number of v_i's dead-end neighbors.*

ALGORITHM 1: S-NEXTHOP(p)

```
cFlag ← false; rFlag ← false;
if p is link node of a cut iso-contour then
    q ← next hop on the shortest path to one randomly chosen cut node in p's iso-contour;
else if p is cut node then
    forall the l-neighbors ngb of p do
        if ngb is unvisited then
            q ← ngb;  cFlag ← true;  break
        end
    end
    if cFlag = false then
        p follows the shortest path to the c-landmark in p's iso-contour; q ← the c-landmark;
    end
else
    forall the l-neighbors ngb of p do
        if ngb is unvisited then
            q ← ngb;  cFlag ← true;  break
        end
    end
    if cFlag = false then
        forall the unvisited neighbors ngb of p do
            if ngb.level = p.level ± 1 then
                q ← ngb;  rFlag ← true;  break
            end
        end
        if rFlag = false then
            p follows the shortest path to an unvisited r-landmark; q ← the r-landmark;
        end
    end
end
return q
```

PROOF. Without regard to the dead-ends, we consider the following four situations: the traversal (1) within an uncut iso-contour, (2) within a cut iso-contour, (3) between two consecutive iso-contours within one region, and (4) between two regions. First, the traversal within an uncut iso-contour generates a local SFC based on Theorem 3.6, and every node is covered only once. Second, the traversal within a cut iso-contour results in a local SFC, due to the cut node and the c-landmark as the "entrance" and "exit" indicator in the cut iso-contour, and every node is covered at most 3 times, for example, the c-landmark if it is also the link node of the iso-contour. Third, two consecutive local SFCs within one region are connected by the link edge, where the link node is designed non-dead-end, and thus the connection is guaranteed; note that the covered times of the nodes associated with link edges are counted in the traversal within their respective cut/uncut iso-contours. Finally, the SFCs in different regions are connected by the r-landmarks in a sorted order, and there may be some node in $\mathcal{M}$ to be covered $(n_r - 1)$ times, in the worst case, if it is shared by all the traversals inter-regions. As the whole process is well guided in a deterministic manner, every node in $\mathcal{M}$ is covered at least once and at most $(n_r + 2)$ times. When taking into account the dead-ends, the upper bound of the node's covered times becomes $(\max_{v_i \in \mathcal{V}} n_d(v_i) + n_r + 2)$, in that the current node would just conduct a back-and-forth travel to every unvisited dead-end neighbor. Thus, Theorem 4.1 holds. □

4.1.2. Storage and Communication Cost. Storage and communication cost for nodes in WSNs are important factors for an efficient SFC construction algorithm. Here, the storage cost is measured by the number of the nodes (landmarks or IDs) stored, and the communication cost is measured by the number of messages exchanged. We show the scalability of SURF by

THEOREM 4.2. *Both the storage cost and the communication cost of every node in SURF are at most $O(n_r)$, where n_r is the number of r-landmarks in the network.*

PROOF. Let us see the storage cost at first. First, every node in the network maintains its neighbors' ID with $O(1)$ storage. Second, every node in the cut iso-contours maintains a routing table pointer to every cut node and every c-landmark in its iso-contour, which incurs $O(1)$ storage cost. Third, every r-landmark maintains a routing table pointer to every other r-landmark, with the storage cost $O(n_r)$. Hence, the storage cost of every node in SURF is at most $O(n_r)$.

Next, we turn to communication cost. First, to construct the iso-contours, every node has to communicate to its neighbors to carry out the flooding initiated by the root node, and thus the communication cost is $O(1)$. Second, during the Reeb graph establishment, every node in the process of g-landmark selection, iso-contour ID notification, and region division generates $O(1)$ communication cost, respectively. Third, every node in the process of cut identification has $O(1)$ communication cost. Finally, in the serial traversal process, the communication cost is dominated by the traversal among different regions, which incurs $O(n_r)$ communication cost. Overall, the communication cost of every node in SURF is at most $O(n_r)$. □

Often in comparison with the network size, the number of r-landmarks is considered negligible. Here it is noted that we do not consider severe cases such as a narrow line-like network with many tiny genuses.

4.1.3. Time Complexity. Another important scalability measurement metric for a SFC construction algorithm is the time complexity. For SURF, we have

THEOREM 4.3. *The time complexity of SURF is at most $O(n)$, where n is the network size.*

PROOF. First, for the iso-contour construction, the time complexity results from the flooding initiated by the root node, which is $O(\sqrt{n})$ for a regularly shaped (e.g., circlelike or squarelike) network whose diameter is $O(\sqrt{n})$ and, in the worst case, $O(n)$ for a network of arbitrary shape with its diameter $O(n)$ [Nguyen et al. 2007; Tan et al. 2013; Yang et al. 2015; Liu et al. 2015]. Second, the process of the iso-contour ID notification and the region formation for the Reeb graph establishment will incur a time complexity of $O(\sqrt{n})$ and $O(\sqrt{n})$, respectively, since such a process performs only local flooding. Third, the process of cut identification requires a time complexity of $O(\sqrt{n})$ for the local flooding during the loop-end region notification and the bisection operation. Finally, the time complexity in the serial traversal process is $O(1)$ for local decisions and would be $O(n)$ in the worst case for communicating to r-landmarks in a far-away region. To summarize, the time complexity of SURF is at most $O(n)$, which is linear to the network size. □

4.1.4. Reacting to Local Failures. Sensor networks often experience local topology changes resulting from temporary or permanent local node/link failures due to battery outage or environmental changes [Sarkar et al. 2013; Lin et al. 2015]. These kinds of topology changes have a great impact on SURF, which yields SURF invalid, as the network would be no longer closed and triangulated as prerequisites. To assure the effectiveness of SURF even in the presence of such local failures, we here present a simple

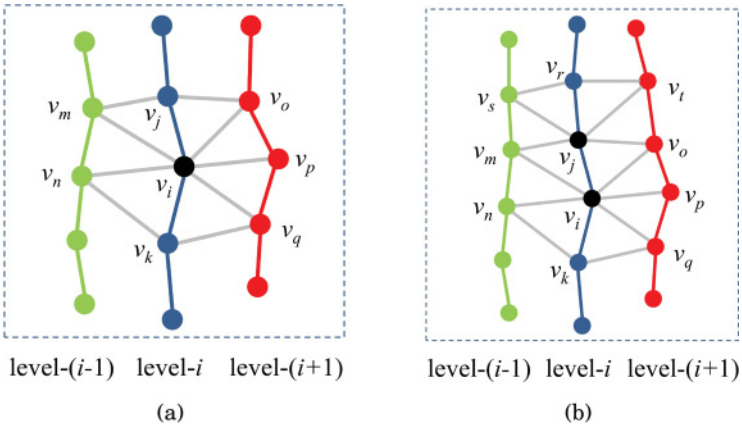


Fig. 7. Illustrations of VIFR. Node failures are represented by black dots.

virtual identity-based local failure recovery scheme (VIFR) that can be incorporated with SURF.

The basic idea of VIFR is simple. We first consider the case when only one node (say v_i) is dead. In this case, all its neighbors will set their identities as that of v_i. See Figure 7(a), for instance; suppose v_i is broken down due to energy depletion. Then each of v_i's neighbors (v_m, v_n, v_j, v_k, v_o, v_p and v_q) will find that the connection to v_i breaks of and set themselves with double identities: One represents itself and the other represents v_i. It is not hard to recognize that all v_i's neighbors form a polygon. By doing so, during the traversal when the current node is v_k (the neighbor in the same iso-contour with v_i), and v_k finds itself with double identities, it will in sequence pass through v_q, v_p, v_o, and v_j (or v_n, v_m, and v_j) and, finally, reach v_j. When there are two neighboring nodes dead (e.g., v_i and v_j in Figure 7(b)), we can also find that all neighbors of v_i and v_j form a polygon, with their double identities. Then suppose the current node is v_k; it will sequentially pass by v_q, v_p, v_o, v_t, and v_r (or v_n, v_m, v_s, and v_r) and, finally, reach v_r.

In a similar way, VIFR can work well for a small number of node failures in a local area. Note that we do not consider situations when many node failures occur in a local region, as there may be relatively large topological voids or holes. In such situations, expanding ring [Johnson and Maltz 1996] or sensor redeployment [Tan et al. 2009; Li et al. 2012] may be employed as complements, such that the traversal scheme can still proceed.

4.1.5. Energy Consumption. Energy consumption is one of the most important issues in WSNs. According to the First-Order Radio Model [Heinzelman et al. 2000], a widely used energy model in WSNs, as in Reason and Rabaey [2004], Zhang and Shen [2009], and Zhang and Li [2012], the energy consumption of a sensor node transmitting 1 bit of data over distance d is $E_t(d) = E_{elec} + E_{amp} \cdot d^{\xi}$, where E_{elec} is the energy spent by transmitter electronics, E_{amp} is the transmitting amplifier, and ξ is the path loss factor. When receiving data, only the receiving circuit is activated; thus the energy spent by receiving 1 bit of data is E_{elec}. In our study, as we have no location/distance information but only connectivity information, to qualitatively estimate the energy consumption, we assume small variances of the packet length, the path loss factor, as well as the communication distance for the data transmission. Then, for both transmitting and receiving a packet, a sensor node has a specific and bounded energy consumption. In this sense, the energy consumption of each node in SURF is positively correlated to its

communication cost (i.e., the number of packets exchanged), where the communication cost of every node in SURF is at most $O(n_r)$, as proved in Theorem 4.2 (n_r is the number of r-landmarks in the network).

It is particularly noted that, among iso-contour construction, Reeb graph establishment (including the iso-contour ID notication and the region formation), cut identication, and the serial traversal process, Reeb graph establishment consumes much more energy than other three steps due to more rounds of messages exchanged as well as more nodes involved (see Figure 16(a), for instance). Since Reeb graph establishment dominates the energy consumption of SURF and is closely related to the topology of a specific network, the energy consumption of SURF thus has a close correlation with network topology, apart from network size and network density. Actually, there is, in general, a positive correlation between the energy consumption of SURF and the complexity of the network topology, as will be explained in Section 5.1.4.

4.1.6. Communication Errors. The proposed SURF algorithm presumes a perfect link between a pair of nodes, as in most connectivity-based algorithms [Funke and Milosavljevic 2007; Lederer et al. 2009; Sarkar et al. 2013; Tan et al. 2013; Jiang et al. 2015] in WSNs. Thus possible communication errors are not taken into account in SURF. For the unreliable link where possible communication errors occur, however, SURF can still be robust against packet loss by using some simple mechanisms to achieve reliable data transmission, for example, the Automatic Repeat Query (ARQ) [Peterson and Davie 2011], where a node will (re-)transmit a message to its neighbor until it receives an acknowledgment from its neighbor before the timeout. As such, there is limited impact on the performance of our algorithms, that is, the SFC can still be constructed and the generated path length will not be longer than that in the reliable link, although unavoidable extra message cost (also more expense in energy consumption) due to retransmissions has to be induced to compensate the communication errors.

4.2. SURF$^+$ For Multiresolution SFC

The above-proposed SURF algorithm can provably cover the entire network by a single path with moderate duplicate coverage. However, one may be concerned that the constructed SFC does not have an adaptive density, while causing a nearly fixed length path. In some cases, it is undesirable to traverse an entire region before getting information from another region. Moreover, in such applications in WSNs as serial fusion and motion planning of mobile beacons, the length of a path may be restricted by the travel budget or the required fusion delay [Ban et al. 2013; Mostefaoui et al. 2015], where a multiresolution SFC is more preferable.

In this section, we present the SURF$^+$ algorithm, which makes use of parameterized spiral-like curves to cover the 3D surface and thus can generate a multiresolution SFC adapting to different requirements on travel budget or fusion delay. By doing so, one can quickly tour around the network coarsely, get a rough idea of the sensor data, and gradually refine the density when more travel budget is available or a higher delay is allowed. In the following, we first use a continuous domain for the intuition and friendly explanation and then present algorithm implementation as well as simulation results in discrete networks.

4.2.1. Principle in Continuous Settings. Consider a spiral whose parametric equations are as follows:

$$x = r(\phi)\cos\phi\cos\theta, \tag{1}$$

$$y = r(\phi)\cos\phi\sin\theta, \tag{2}$$

$$z = r(\phi)\sin\phi, \tag{3}$$

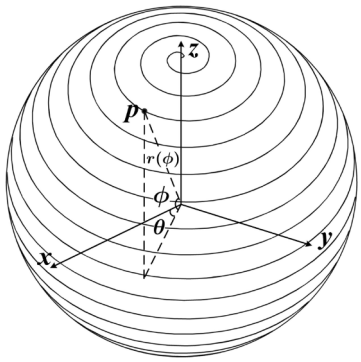


Fig. 8. Illustration of spherical spiral.

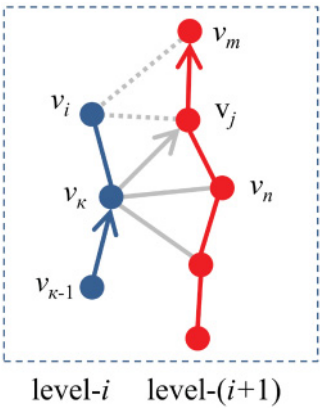


Fig. 9. Spiral-like traversal inter-level.

where θ is the angular parameter describing the spiral, $r(\phi)$ is specified by the meridian curve generating the surface, and the angle $\phi = \theta/k$. It can be easily recognized that $r(\phi) = d/\cos\phi$ is for a planar spiral lying on a plane at distance d, $r(\phi) = d/\sin\phi$ for a helix wrapping a cylinder of radius d, and $r(\phi) = d$ for a spherical spiral of radius d (see Figure 8, for example).

Now we turn to examining the factor that impacts the surface coverage of the spiral. Let us take the spherical spiral, for instance. It is not hard to find that the angle $\triangle\phi$ (with respect to the sphere center) between two consecutive intersections with a meridian must satisfy $\triangle\phi = 2\pi/k$ (since $\triangle\theta = 2\pi$). Thus the parameter k actually determines the proportion of the surface coverage of the spiral. In particular,

PROPOSITION 4.4. *The proportion of the surface coverage of the spiral is proportional to the value of k. If $k \to \infty$, then the spiral tends to cover the entire surface. If $k \to 0$, then the spiral becomes part of a great circle.*

On this basis, we can design spirals to achieve multiresolution coverage of the surface by adjusting the parameter k. However, it is nontrivial to realize this idea in discrete networks, especially when location information is unavailable. In addition, unlike continuous settings where k ranges from zero to infinity, it is challenging to define such k in discrete networks where nodes are randomly deployed in the field. Thus, the algorithm implementation in discrete networks needs elaborate designs.

4.2.2. Algorithm in Discrete Networks. We next explain the detailed algorithm implementation. The basic idea of SURF$^+$, similar to that of SURF, is to construct the local spiral-like SFC with an adaptive density in each (uncut) region, before connecting all local SFCs into a final SFC. While connecting the local SFCs is done in the same way as SURF does, the challenges of SURF$^+$ lies in the way how to construct the local spiral-like SFCs.

For constructing local spiral-like SFCs, we purposely set a parameter κ as the analogy of the parameter k in continuous settings, such that multiresolution coverage of the region can be achieved by choosing different κ values. Specifically, the constructed local spiral-like SFC is designed to cover $\kappa + 1$ nodes in one iso-contour before entering the next iso-contour to continue covering another $\kappa + 1$ nodes in a uniform direction. To that end, SURF$^+$ incorporates a spiral-like traversal scheme on the basis of contour construction and the Reeb graph-based network decomposition (see the results in Figures 3(e) and (f)). The spiral-like traversal scheme starts from one of the r-landmarks

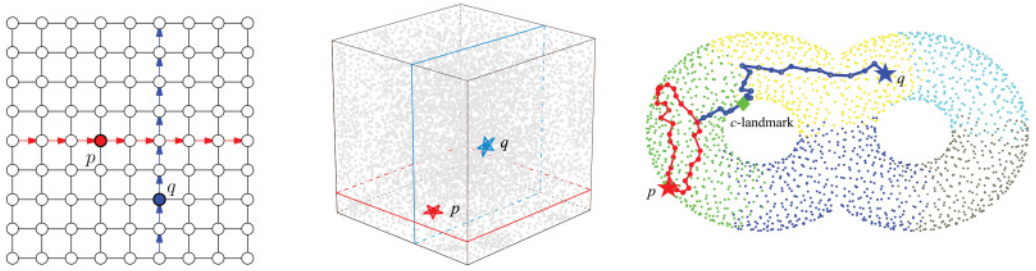
ALGORITHM 2: T-NEXTHOP(p)

```
Intra-flag ← false; Inter-flag ← false;
if the counter equals to κ then
    Inter-flag ← true;
else
    forall the neighbors of level-i ngb of p do
        if ngb is unvisited then
            q ← ngb;  Intra-flag ← true;  counter++;  break
        end
    end
end
if Intra-flag = false then
    forall the neighbors of level-(i + 1) ngb of p do
        if ngb is unvisited then
            q ← ngb;  counter ← 0;  break
        end
    end
if Inter-flag ← true then
    forall the faces fcs associated with p do
        if two nodes v_i in level-i and v_j in level-(i + 1) are in fcs, and v_i is unvisited then
            q ← v_j;  break
        end
        forall the faces fcs associated with p do
            if two nodes v_i and v_j in level-(i + 1) are in fcs, and v_i is a neighbor of p then
                v_i ← visited;  break
            end
        end
    end
return q
```

p_0 (or the root node) in the region and consists of intra- and inter-contour traversals, following T-NEXTHOP Algorithm as described in Algorithm 2.

Intra-Contour Traversal. At the beginning, p_0 randomly chooses a neighbor, say, p_1, in its contour as its next hop, sets a counter 0, and traverses to p_1. p_1 then marks itself as visited, updates the counter by adding 1, and finds an unvisited neighbor, say, p_2, in its contour as its next hop. This traversal process in the contour repeats until the current node is p_κ with the counter equals to κ or there is no unvisited neighbor in the contour. In this situation, if the current node finds a neighbor exits in its next contour with the same region ID, then the inter-contour traversal occurs.

Inter-Contour Traversal. Suppose the last node in the intra-contour traversal in contour-i is v_κ. Simply stated, the inter-contour traversal is for v_κ to choose a neighbor in contour-$(i + 1)$. Often it is desirable that the constructed curve in contour-$(i + 1)$ maintains the same direction as that in contour-i (see the arrows in Figure 9), so the curves in the two contours have less spatial correlation. Therefore, the primary task here is to determine the uniformity of the directions. We are motivated by the following observation: Denote v_κ's unvisited neighbor in contour-i as v_i, v_κ's neighbor in contour-$(i + 1)$ sharing the same face with v_κ and v_i as v_j, the node sharing the same face with v_κ and v_j as v_n, and the node sharing the same face with v_i and v_j as v_m, and then the direction v_j to v_m keeps up with the direction $v_{\kappa-1}$ to v_κ, see Figure 9. Thus for inter-contour traversal, such a node as v_j would be chosen as the next hop of v_κ. Please refer to Algorithm 2 for details, while the simulation results are demonstrated in Section 5.2.



(a) A grid network. (b) A cube network. (c) A torus network.

Fig. 10. Double rulings design methodologies. (a) From Sarkar et al. [2006, 2009]; (b) from Luo et al. [2011]; (c) our design. p is the data producer; q is the data consumer; red (respectively blue) line represents the data storage (respectively, retrieval) curve.

4.3. Application for In-Network Data Storage and Retrieval

Except for the serial operation on both sensor nodes and sensor data, and the motion planning of mobile beacons as mentioned in Section 1, we now consider another two applications our SURF and SURF$^+$ algorithms can be use for: distributed in-network data storage and retrieval for high genus 3D surface WSNs.

4.3.1. Basics on In-Network Data Storage and Retrieval. In-network data storage and retrieval has always been one of the most important topics in WSNs [Sarkar 2014]. While centralized data storage and retrieval schemes suffer from "energy hole problem" at nodes near the sink [Pottie and Kaiser 2000], distributed schemes are more desirable for the scalability and robustness.

Initial attempts for distributed in-network data storage and retrieval map data to rendezvous nodes for storage and retrieve data by geographic routing such as GPSR [Karp and Kung 2000]. A celebrated pioneer work is geographical hash table (GHT) [Ratnasamy et al. 2002, 2003], which hashes the data by its data type into geographical locations and stores the data at nodes around the so-called home node, the one closest to the hashed location. Its data retrieval is then implemented by the geographic routing, namely GPSR [Karp and Kung 2000]. The main drawback of GHT is that the rendezvous node for popular data queried by many consumers imposes a communication bottleneck. In addition, the data retrieval scheme in GHT is not distance sensitive: Even when the data consumer is close to the data producer, it may still have to go to a far-away rendezvous node.

To tackle the above issues, double rulings schemes are proposed [Sarkar et al. 2006, 2009]. The key idea is the design of two intersection guaranteed curves, namely the data replication curve and the data retrieval curve. Data are stored at nodes following the replication curve while a data request travels along the retrieval curve, so data retrieval can be guaranteed successful due to the intersection property. Figures 10(a) and (b) illustrate two typical double rulings designs in a 2D grid network and a 3D cube network. Take the grid network case, for example; the replication curves follow the horizontal lines and the retrieval curves follow the vertical lines. Invariably, the data consumer can find the desired data generated by a data producer by traveling along the vertical line.

Much research has been done recently to extend double-ruling schemes to the networks with irregular geometric shapes, for example, by GeoQuorum [Luo and He 2011], 3DQS [Luo et al. 2011], RDRIB [Lin et al. 2012], Harmonic Quorum Systems [Zhang et al. 2012], and MobiMark [Tang et al. 2014]. However, almost all these methods require the network shape be topologically equivalent to a regular shape such as a

grid or a cube. Thus, they cannot be applied to a high genus 3D surface network that apparently is topologically inequivalent to a regular shape.

As far as we know, the only double-ruling-based method that can be used for high genus 3D surface WSNs is discussed in Yang et al. [2013, 2015]. It first computes a cut graph [Munkres 2000] of the surface network, along which the surface can be cut open to a topological disk. It then embeds the planar disk to an aligned planar rectangle with discrete surface Ricci flow [Jin et al. 2008], such that regular double rulings scheme can be applied. However, computing the cut graph and the planar rectangle virtual coordinates is rather complicated. Moreover, the obtained cut graph cannot be guaranteed optimal, which possibly in turn increases the computing cost when using discrete surface Ricci flow.

4.3.2. Our Approach. We have known that the most tricky part of a double rulings design is the guaranteed intersection between the replication curve and the retrieval curve. Recall that the generated SFC by SURF$^+$ is ensured to cover every iso-contour, and then an intuitive solution for a double rulings scheme in high genus 3D surface WSNs can be developed in a straightforward manner, that is, to devise the replication curve following the iso-contour and the retrieval curve following the SFC generated by SURF$^+$ (here we use the simplest case with $\kappa = 0$[4]). To be more concrete, consider a data producer with its iso-contour ID C_i. To store the produced data, the data producer travels and leaves copies of the data at nodes whose iso-contour ID is C_i. As discussed in Section 3.3, the traversal of the data producer is in a serial manner, following the exact iso-contour C_i (see the red curve in Figure 10(c)). For data retrieval, the data consumer traverses to find the desired data as described in Section 4.2.2. The data consumer stops once it hits the node in possession of its desired data (see the blue curve in Figure 10(c)). The effectiveness is validated by simulations in Section 5.3.

5. PERFORMANCE EVALUATION

5.1. Performance Evaluation of SURF

5.1.1. SURF for Various Network Topologies. To evaluate the effectiveness of SURF, we have conducted extensive simulations on various scenarios. We first examine the performance of SURF in four 3D surface networks: a genus-1 corridor, a genus-2 bowknot, a genus-3 smile, and a genus-4 window, with the average node degree between 8 and 11. Figure 11 shows the results of SURF, as well as the statistical distribution of nodes' covered times. It is observed that SURF delivers a stable performance and successfully generates in each network an SFC that traverses the whole network with each node covered a small number of times. This is consistent with Theorem 4.1.

5.1.2. Comparisons with Other Approaches. We also evaluate another two simple SFC construction approaches for comparison, namely random walk [Spitzer 2001] and pre-oder traversal [Liang et al. 2013] (a tree-based scheme similar to a depth-first search). Note that there are very limited algorithms to compare with because previous SFC construction schemes are proposed for 2D WSNs and cannot be directly applied in high genus 3D surfaces. Hence, we additionally propose random-SURF, a variant of SURF, as an alternative comparison object. Random-SURF makes traversal first in different iso-contours of one region, and then in different regions, as is done in SURF.

[4]Note that the constructed SFC generated by SURF$^+$, as discussed in Section 4.2.2, is designed to cover $\kappa + 1$ nodes in one iso-contour before entering the next iso-contour to continue covering another $\kappa + 1$ nodes. So to guarantee the intersection between the data replication curve (i.e., the iso-contour) and the data retrieval curve (i.e., the SFC generated by SURF$^+$), one covered node is enough in each iso-contour. Therefore, we choose $\kappa = 0$.

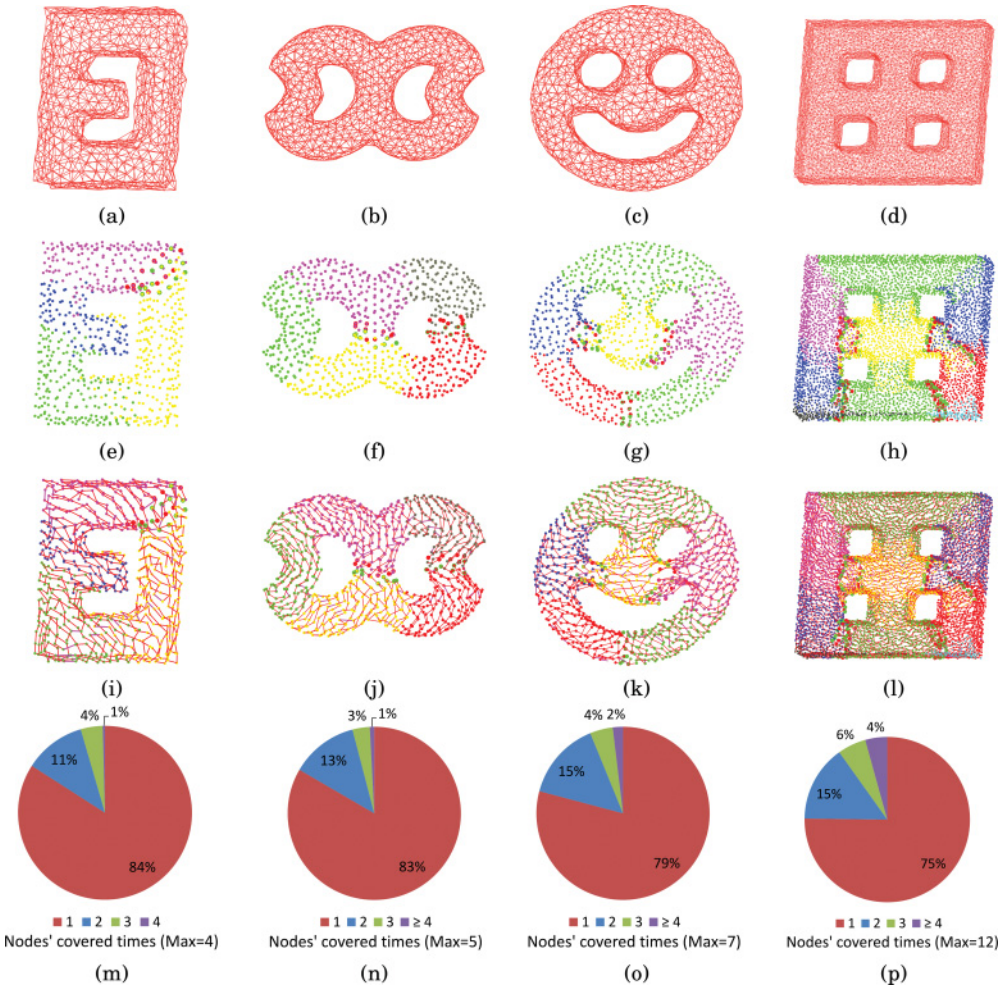


Fig. 11. Columns from left to right: (a) A genus-1 corridor network with 710 nodes; Avg deg is 8.92. (b) A genus-2 bowknot network with 837 nodes; Avg deg is 9.35. (c) A genus-3 smile network with 1,102 nodes; Avg deg is 10.01. (d) A genus-4 window network with 5,366 nodes; Avg deg is 9.63. Rows: (1) The original triangulated network. (2) The emerged regions of the Reeb graph and the maximum cut set. (3) The SFC generated by SURF. (4) The distribution of nodes' covered times.

The difference is that random-SURF chooses its next hop (within and inter regions) at random.

Figure 12 illustrates the network coverage percentage of random walk, random-SURF, preoder traversal, and SURF in each 3D network at the first time the network is fully covered by SURF. We can see that SURF yields a much faster traversal speed than random walk and random-SURF, and the superiority of SURF is even more prominent as the network becomes complex, say, in the network of genus-4 window. The better performance of SURF benefits from its region division as well as c-landmark/r-landmark piloting, while random walk and random-SURF are unrefined schemes without explicit directions on how to speedily traverse the whole network.

Figure 13 further depicts how the specific network coverage changes as the generated path moves forward by the four algorithms in different networks. It can be seen that

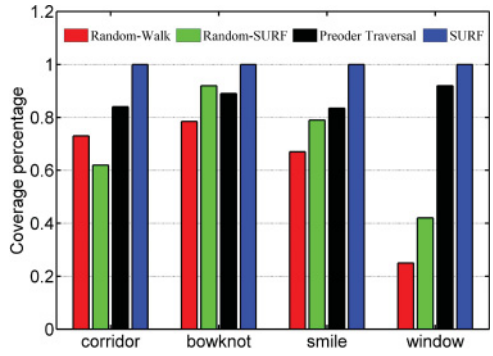


Fig. 12. Comparison on the network coverage.

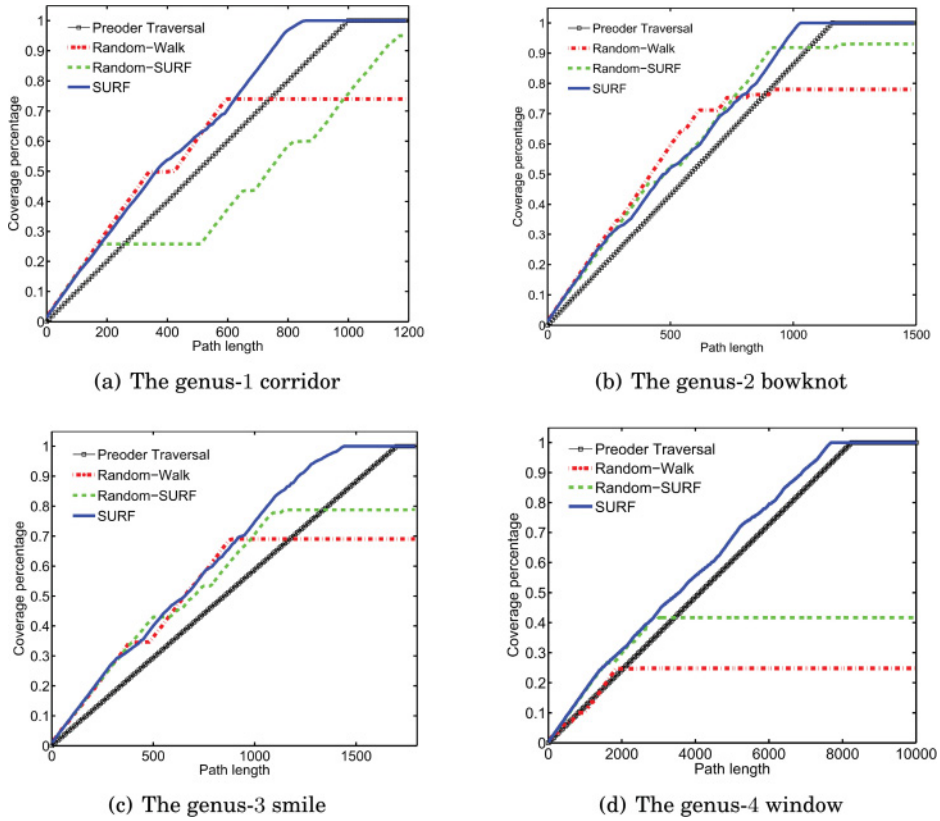
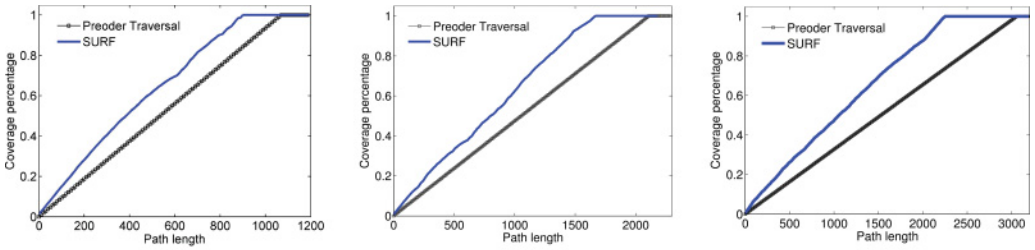


Fig. 13. The coverage vs. the path length.

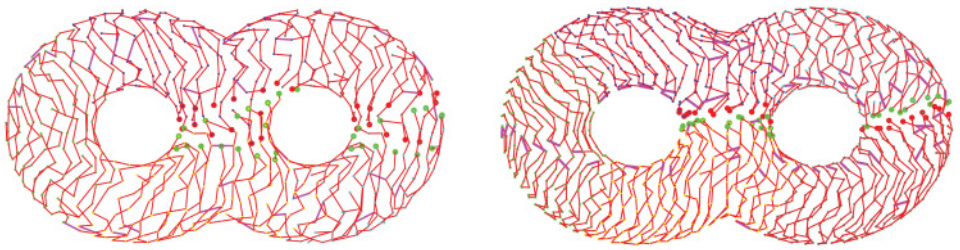
SURF always leads to 100% network coverage however complex the network is, and also with the fastest pace, while random walk cannot achieve full coverage in a competing pace. This is not surprising as SURF is a well-guided algorithm while random walk is somewhat blind and luck dependent.

It is also found that, in comparison with random walk, random-SURF performs better in simpler topologies, say, the networks of genus-2 bowknot and genus-3 smile. The reason is that, in contrast to random walk's complete blindness, random-SURF



(a) Avg deg is 7.2. (b) Avg deg is 12.5. (c) Avg deg 17.7.

Fig. 14. Comparisons between preorder traversal and SURF for torus networks with different node degrees.



(a) A network with 790 nodes; Avg deg is 7.2. (b) A network with 1,314 nodes; Avg deg is 12.5.

Fig. 15. Performance of SURF under different network densities.

traverses the network in a more global way: Before aimlessly choosing its next hop, it has to find an unvisited neighbor in its iso-contour or in its region, which, to some extent, alleviates its blindness.

In more complex scenarios, for example, in the genus-4 window, however, random-SURF is still far less effective than SURF with respect to the coverage rate and the coverage speed. After all, random-SURF is a randomized algorithm. Without the guidance of c-landmarks in cut iso-contours or g-landmarks inter-regions, random-SURF inevitable runs into locally infinite loop at a certain time. That is the reason why, in Figure 13(d), the coverage percentage of random-SURF nearly keeps steady when the path length is greater than roughly 3,000.

As a matter of fact, in our tests in the genus-4 window, both random walk and random-SURF require a path length of more than 50,000 to reach 50% coverage. This is because after a certain fraction of nodes have been visited, random walk and random-SURF are more inclined to aimlessly find the last few unvisited nodes for a long time, as discussed in Section 1.1.3.

It is noted that preoder traversal has a comparable covering speed with SURF, and, in fact, it always has 100% network coverage as shown in Figure 13. However, as a tree-based approach, the nodes near the root node (or the father nodes) are shared by different branches and thus are more likely to be subjected to more duplicated visits during the traversal process [Kuhn et al. 2003; Gao and Zhang 2006; Zhang and Shen 2009; Lam and Chen 2013], especially when the nodes near the root node (or the father nodes) have relatively large node degrees. Figure 14 shows the comparison results on Torus networks shown in Figure 15(a), Figure 15(b), and Figure 3(a), with its average node degrees of 7.2, 12.5, and 17.7, respectively. We can see that preorder traversal has a roughly 24%, 32%, and 39% longer path length than that of SURF, from left to right, which means that SURF is more robust to networks with different node degrees.

5.1.3. Performance Under Different Network Densities. Most connectivity-based algorithms in WSNs [Funke and Milosavljevic 2007; Lederer et al. 2009; Tan et al. 2013; Jiang et al. 2015] require a relatively high network density, so the shape of the network can be well represented. In contrast, SURF works well for both dense and sparse networks as long as the network is connected (as well as closed and triangulated).

To verify the impact of node density on the performance of SURF, we conduct further simulations. In particular, we fix the communication range and vary the number of the genus-2 torus network in Figure 3(a) to generate two sparser networks with different average node degrees, as shown in Figure 15. It can observed that SURF is insensitive to different network densities and can achieve stable full network coverage even for a sparse network with average node degree of 7.2 (see Figure 15(a)). In addition, coupled with the result in Figure 3(h), we can see that higher network density leads to an SFC that captures the topological features of the underlying network better, as the distance of two sensor nodes is more accurately approximated by the hop count distance in discrete networks when the network is denser.

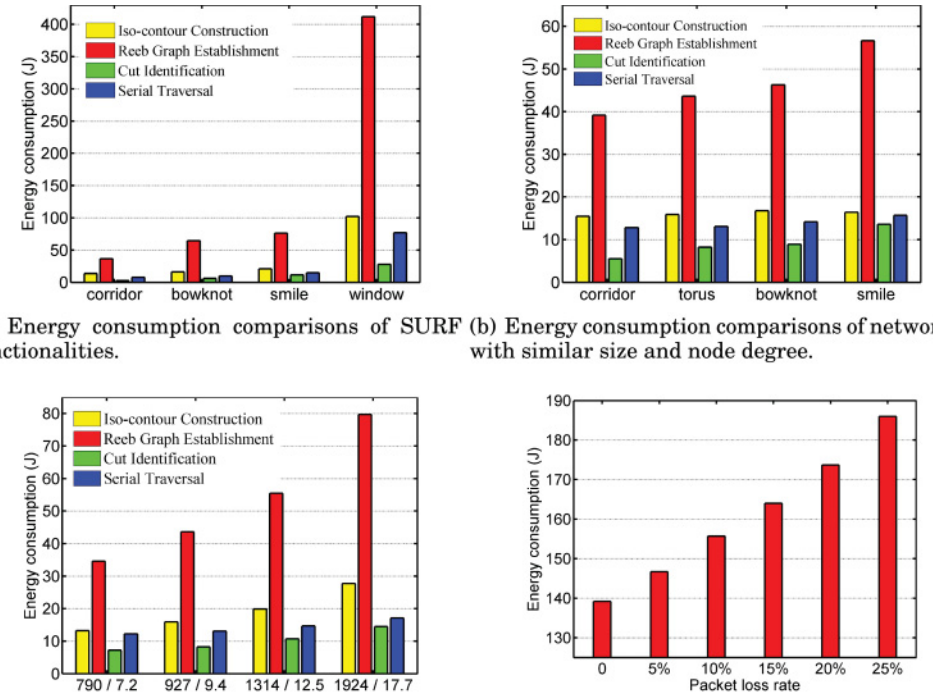
5.1.4. Energy Consumption of SURF. We conduct further simulations to illustrate how energy consumption of SURF changes. According to Shnayder et al. [2004] and Jiang et al. [2011], we set one node's energy consumption for transmission (respectively, receiving) to 80 (respectively, 30) mJ per message, which is based on widely used hardware configurations of MICA2 [Crossbow 2003]. It is noted that here we set the energy consumption to be identical to reflect its changes in trend; the exact value may vary in light of different parameters, for example, transmission distance and the path loss factor, in practice.

Figure 16(a) shows the energy consumption comparisons of different SURF functionalities in the four networks in Figures 11(a)–(d). As mentioned, Reeb graph establishment consumes the most energy. Moreover, it can be also observed that energy consumption of SURF increases as the network size gets larger. Figure 16(b) describes the energy consumption comparisons of networks with similar sizes and node degrees, where the network sizes/node degrees of corridor, bowknot, smile and window are 902/9.4, 927/9.4, 928/9.5, and 941/9.7, respectively. We can find that even with similar network sizes and node degrees, networks with different topologies have different energy consumptions, which mainly result from the major energy consumer—Reeb graph establishment. Next, we choose a fixed network topology, the torus, and vary its size as well as node degree. The result is presented in Figure 16(c), which, as expected, exhibits a positive correlation of energy consumption and network size/node degree.

For an unreliable link, the number of packet transmissions necessary to ensure the successful transfer of a single packet follows a geometric distribution of parameter p, according to Banerjee and Misra [2002], and its statistical mean value is $1/(p-1)$, where p is the packet error rate. Figure 16(d) demonstrates the energy consumption of the torus network in Figure 3(a) with increasing packet loss rate, and the result is in agreement with the theoretical analysis.

5.2. Performance Evaluation of SURF$^+$

In this section, we report the simulation results of SURF$^+$ on the four 3D surface networks in Figures 11(a)–(d). We mainly focus on how the coverage percentage changes with the parameter κ. As there is no comparison counterpart, we only evaluate the performance of SURF$^+$, with the results exhibited in Figure 17. We can observe that the coverage percentage is monotonically increasing with the value of parameter κ, in all cases, and leads to a 100% network coverage as κ becomes a relatively large number, which is well in accordance with Proposition 4.4. The results thus validate that SURF$^+$ has an adaptive density, with a tunable path length.



(a) Energy consumption comparisons of SURF functionalities.

(b) Energy consumption comparisons of networks with similar size and node degree.

(c) Energy consumption comparisons of torus network with increasing size and node degree

(d) Energy consumption comparisons of torus network with increasing packet loss rate.

Fig. 16. Comparisons of energy consumption of SURF in different situations.

It can be also noted that the coverage changing rate decreases gradually as the value of κ increases. This is because different contours contains different number of nodes. At first, when κ is small, there are lots of nodes unvisited in most of the contours. Thus, when κ is increased by one, the number of increased covered nodes roughly equals to the total number of contours in the network. However, as κ becomes larger, all nodes in some contours are visited. In this case, when κ is increased by one, the number of increased covered nodes would be less than the total number of contours in the network. This kind of disparity becomes notable when κ gets even larger, until the whole network is covered.

5.3. Performance Evaluation of SFC-Based In-Network Data Storage and Retrieval

To evaluate the performance of our approach, we conduct simulations on the genus-2 torus network in Figure 3(a) with our approach, the cut-graph-based scheme [Yang et al. 2013, 2015], and GHT [Ratnasamy et al. 2002, 2003]. We first report the results on the costs of data storage and retrieval, as shown in Figure 18. For double-ruling-based schemes, there is usually a balance between the data storage and retrieval costs: the more data replication, the fewer data retrieval costs and vice versa. From the results, we can see that our approach has a comparable performance with the cut-graph-based scheme, with fewer data storage costs but a little more data retrieval costs.

To test the distance sensitivity of data retrieval, we randomly select 200 pairs of data producer and consumer and show the average results in Figure 19. It is observed that, while the consumer cost is nearly the same in accessing local data as that for accessing remote data, the two double-ruling-based schemes have much better distance

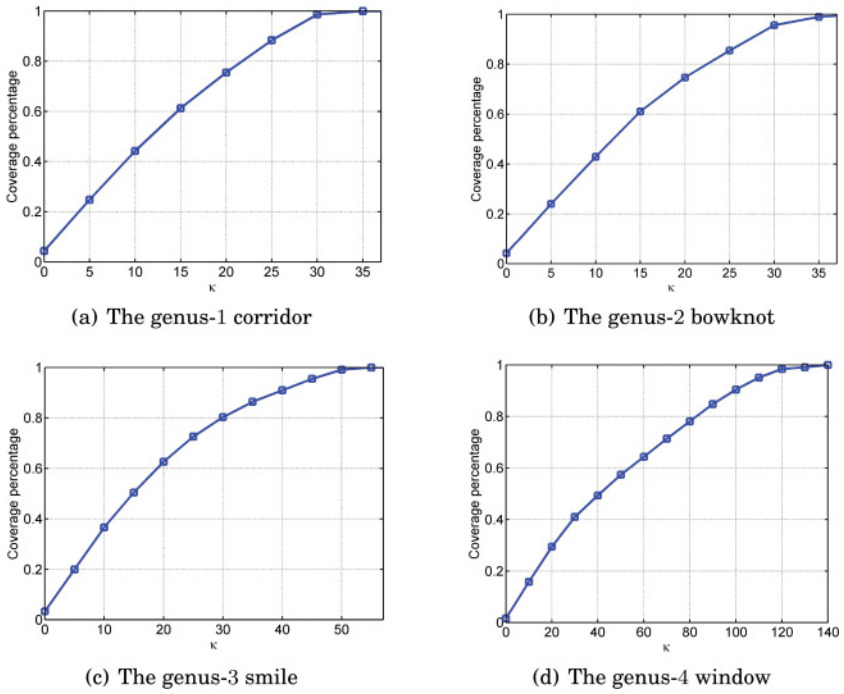


Fig. 17. The coverage v.s. κ.

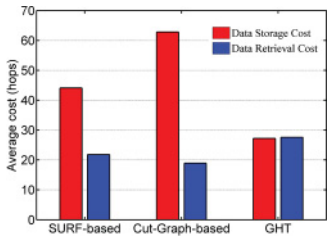


Fig. 18. Comparisons of storage/retrieval cost.

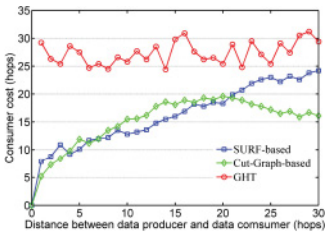


Fig. 19. Consumer cost vs. distance between data producer and data consumer.

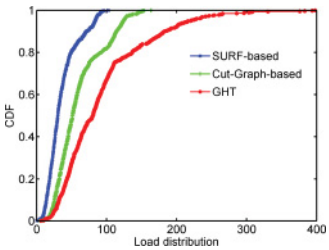


Fig. 20. Comparisons of load distribution.

sensitivity of data retrieval. In addition, the consumer cost of double rulings schemes is invariably smaller than that of GHT. It is not surprising that our approach has a comparable performance with the cut-graph-based scheme, as the cut-graph-based scheme is in essence the same way as our approach but using an embedded into a rectangle that eventually is mapped back to the original network.

At last, Figure 20 shows the traffic load distributions of the three schemes in a 500-data consumers test. For GHT, the majority of the consumers have to pass the nodes near the producer, and thus a small fraction of nodes have much higher loads than the rest. For other two schemes, there is no such hot spot as GHT, and therefore the load is much more balanced. Our approach is even more balanced than the cut-graph-based scheme, because the Ricci flow method often embeds a relatively uniform network into a space with nonuniform node distribution. Therefore, nodes in the denser region are more likely to suffer an overload than those in the sparser area.

6. CONCLUSION AND FUTURE WORK

In this article, we have presented a novel distributed algorithm for SFC construction in high genus 3D surface WSNs. This algorithm requests the connectivity information only and does not require advance knowledge of location or distance information. It is also scalable, with a nearly constant storage and communication cost per node in the network. To incorporate adaptive density of the constructed SFC, we also designed teh SURF$^+$ algorithm and present its application for in-network data storage and retrieval in high genus 3D surface WSNs. Simulations on several representative networks demonstrate that both algorithms work well on high genus 3D surface WSNs.

We are interested in several directions in the future. First, we plan to evaluate other triangulation techniques for our algorithm. As some special cases of iso-contours (e.g., when there are dead-ends) are closely related to the triangulation methods, we believe improvement is possible when appropriate approaches are adopted to lower the chances that the special cases occur. Second, we find that SURF does not work well with some severe cases (e.g., a surface with multiple genus close to each other), where it could be difficult to identify the accurate Reeb graph. Therefore, a way to deal with this kind of special settings needs further in-depth exploration. Last, we only consider in this article the orientable closed surface (compact and without boundaries). It would be more challenging to design linearization schemes for more general 3D surface networks, say, those with holes.

APPENDIX

A. PROOF OF THEOREM 3.6

A.1. Lemma A.1

The proof of Theorem 3.6 is based on the following Lemma.

LEMMA A.1. *Dead-ends of an iso-contour in $f^{-1}(l)$ have no effect on the l-connectivity of the non-dead-end nodes within the same iso-contour.*

PROOF. According to the size of the l-degree (defined as the number of its neighbors in its iso-contour), a dead-end can be classed as one of the following two categories: with l-degree 1, and with l-degree more than 1. Figure 21 shows the two cases of dead-ends.

For dead-ends with l-degree 1, when they are removed, the l-connectivity of the non-dead-end nodes in the iso-contour is guaranteed by Lemma 3.3. For dead-ends with l-degree more than 1, we first claim that the neighbor nodes of a dead-end in the iso-contour is directly connected (otherwise it contradicts our assumption of a triangulated surface). So, in Figure 21(b), nodes v_i and v_j are directly connected. Then it can be seen that this kind of dead-end cannot negate the l-connectivity of the non-dead-end nodes in the iso-contour. That is, Lemma A.1 holds. □

A.2. Proof of Theorem 3.6

PROOF. On the one hand, Lemma 3.3 guarantees that any two nodes in the same iso-contour are l-connected, and from Lemma A.1 we can see that dead-ends have no effects on the l-connectivity of the iso-contour. So an iso-contour with dead-end edges eliminated is l-connected and is also connected.

On the other hand, we prove that an iso-contour with the dead-end edges eliminated is a closed cycle. That can be done by mathematical induction. We start from the root node $r = f^{-1}(0)$. Recall that the network is a closed surface, so $f^{-1}(1)$ contains one or more closed curves, that is, any one of the iso-contours of $f^{-1}(1)$ is a closed cycle. Note that there is no dead-end edge in the iso-contour(s) of $f^{-1}(1)$. Assume any one of

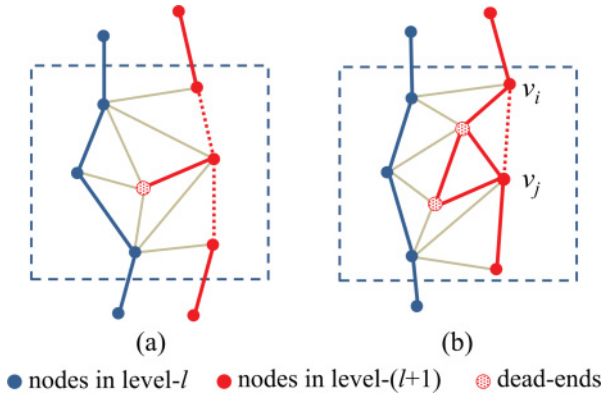


Fig. 21. Two cases of dead-ends.

the iso-contours of $f^{-1}(l)$ with the dead-end edges eliminated is a closed cycle. We next prove that any one of the iso-contours of $f^{-1}(l+1)$ with the dead-end edges eliminated is as well a closed cycle. Three cases need to be considered: the iso-contour of $f^{-1}(l+1)$ (1) without dead-ends, (2) with dead-ends of l-degree 1, and (3) with dead-ends of l-degree more than 1. In all the cases, each node in $f^{-1}(l)$ must correspond to at least one neighbor in $f^{-1}(l+1)$. As nodes in the iso-contour of $f^{-1}(l+1)$ are l-connected, suppose that the iso-contour is not a cycle; then there must be two successive nodes (in the direction of clockwise or counter-clockwise) disconnected as two endpoints, each of which connects to its parent node in $f^{-1}(l)$. Then a polygon containing the two endpoints, their parents and their children $f^{-1}(l+2)$, or a void (or a hole) if $f^{-1}(l+1)$ is the last level set, appears. This contradicts our assumption either that the surface network is closed or is triangulated. Thus, in all the cases, the iso-contour of $f^{-1}(l+1)$ must be a cycle. So far in all cases, any one of the iso-contours of $f^{-1}(l+1)$ with the dead-end edges eliminated is provably a closed cycle, and, therefore,Theorem 3.6 holds. □

REFERENCES

M. R. Akhondi, A. Talevski, S. Carlsen, and S. Petersen. 2010. Applications of wireless sensor networks in the oil, gas and resources industries. In *Proceedings of IEEE AINA*. 941–948.

J. M. Bahi, A. Makhoul, and A. Mostefaoui. 2008. Hilbert mobile beacon for localisation and coverage in sensor networks. *Int. J. Syst. Sci.* 39, 11 (2008), 1081–1094.

X. Ban, M. Goswami, W. Zeng, X. Gu, and J. Gao. 2013. Topology dependent space filling curves for sensor networks and applications. In *Proceedings of IEEE INFOCOM*. 2166–2174.

Suman Banerjee and Archan Misra. 2002. Minimum energy paths for reliable communication in multi-hop wireless networks. In *Proceedings of ACM MobiHoc*. 146–156.

H. Carr, J. Snoeyink, and M. van de Panne. 2004. Simplifying flexible isosurfaces using local geometric measures. In *Proceedings of IEEE VIS*. 497–504.

Y.-C. Chung, I. Su, and C. Lee. 2011. An efficient mechanism for processing similarity search queries in sensor networks. *Inform. Sci.* 181, 2 (2011), 284–307.

K. Cole-McLaughlin, H. Edelsbrunner, J. Harer, V. Natarajan, and V. Pascucci. 2003. Loops in reeb graphs of 2-manifolds. In *Proceedings of ACM SoCG*. 344–350.

Crossbow. 2003. MICA2 wireless measurement system datasheet.

J. Erickson and S. Har-Peled. 2004. Optimally cutting a surface into a disk. *Discr. Comput. Geom.* 31, 1 (2004), 37–59.

C. Fischer and H. Gellersen. 2010. Location and navigation support for emergency responders: A survey. *IEEE Perv. Comput.* 9, 1 (2010), 38–47.

S. Funke and N. Milosavljevic. 2007. Network sketching or: 'How much geometry hides in connectivity? – part II'. In *Proceedings of ACM-SIAM SODA*. 958–967.

J. Gao and L. Zhang. 2006. Load-balanced short-path routing in wireless networks. *IEEE Trans. Parallel Distrib. Syst.* 17, 4 (2006), 377–388.

A. Gray, E. Abbena, and S. Salamon. 2006. *Modern Differential Geometry of Curves and Surfaces with Mathematica* (3rd ed.). Chapman & Hall/CRC, London.

A. Hatcher. 2002. *Algebraic Topology*. Cambridge University Press, Cambridge.

W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. 2000. Energy-efficient communication protocol for wireless microsensor networks. In *Proceedings of 33rd HICSS*. 4–7.

Hongbo Jiang, Shudong Jin, and Chonggang Wang. 2011. Prediction or not? an energy-efficient framework for clustering-based data collection in wireless sensor networks. *IEEE Trans. Parallel. Distrib. Syst.* 22, 6 (2011), 1064–1071.

H. Jiang, T. Yu, C. Tian, G. Tan, and C. Wang. 2015. Connectivity-based segmentation in large-scale 2-d/3-d sensor networks: Algorithm and applications. *IEEE/ACM Trans. Network.* 23, 1 (2015), 15–27.

M. Jin, J. Kim, F. Luo, and X. Gu. 2008. Discrete surface ricci flow. *IEEE Trans. Vis. Comput. Graph.* 14, 5 (2008), 1030–1043.

D. B. Johnson and D. A. Maltz. 1996. *Dynamic Source Routing in Ad Hoc Wireless Networks*. The Kluwer International Series in Engineering and Computer Science, Vol. 353. Kluwer Academic Publishers, Amsterdam, Chapter 5, 153–181.

B. Karp and H. T. Kung. 2000. GPSR: Greedy perimeter stateless routing for wireless networks. In *Proceedings of ACM MobiCom*. 243–254.

D. Koutsonikolas, S. M. Das, and Y. C. Hu. 2007. Path planning of mobile landmarks for localization in wireless sensor networks. *Comput. Commun.* 30, 13 (2007), 2577–2592.

F. Kuhn, R. Wattenhofer, and A. Zollinger. 2003. Worst-case optimal and average-case efficient geometric ad-hoc routing. In *Proceedings of ACM MobiHoc*. 267–278.

S. S. Lam and Q. Chen. 2013. Geographic routing in d-dimensional spaces with guaranteed delivery and low stretch. *IEEE/ACM Trans. Network.* 21, 2 (2013), 663–677.

S. Lederer, Y. Wang, and Jie. Gao. 2009. Connectivity-based localization of large-scale sensor networks with complex shape. *ACM Trans. Sens. Network.* 5, 4 (2009), 1–32.

F. Li, J. Luo, S. Xin, W. Wang, and Y. He. 2012. LAACAD: Load balancing k-area coverage through autonomous deployment in wireless sensor networks. In *Proceedings of IEEE ICDCS*. 566–575.

F. Li, C. Zhang, J. Luo, S.-Q. Xin, and Y. He. 2014. LBDP: Localized boundary detection and parametrization for 3-D sensor networks. *IEEE/ACM Trans. Network.* 22, 2 (2014), 567–579.

M. Li and Y. Liu. 2009. Underground coal mine monitoring with wireless sensor networks. *ACM Trans. Sens. Network.* 5, 2 (2009), 10:1–10:29.

W. Liang, P. Schweitzer, and Z. Xu. 2013. Approximation algorithms for capacitated minimum forest problems in wireless sensor networks with a mobile sink. *IEEE Trans. Comput.* 62, 10 (2013), 1932–1944.

C.-H. Lin, J.-J. Kuo, B.-H. Liu, and M.-J. Tsai. 2012. GPS-free, boundary-recognition-free, and reliable double-ruling-based information brokerage scheme in wireless sensor networks. *IEEE Trans. Comput.* 61, 6 (2012), 885–898.

S. Lin, G. Zhou, M. Al-Hami, K. Whitehouse, Y. Wu, J. A. Stankovic, T. He, X. Wu, and H. Liu. 2015. Toward stable network performance in wireless sensor networks: A multilevel perspective. *ACM Trans. Sens. Network.* 11, 3 (2015), 42:1–42:26.

W. Liu, H. Jiang, Y. Yang, X. Liao, H. Lin, and Z. Jin. 2015. A unified framework for line-like skeleton extraction in 2D/3D sensor networks. *IEEE Trans. Comput.* 64, 5 (2015), 1323–1335.

Y. Liu, X. Mao, Y. He, K. Liu, W. Gong, and J. Wang. 2013. CitySee: Not only a wireless sensor network. *IEEE Network* 27, 5 (2013), 42–47.

J. Luo and Y. He. 2011. GeoQuorum: Load balancing and energy efficient data access in wireless sensor networks. In *Proceedings of IEEE INFOCOM*. 616–620.

J. Luo, F. Li, and Y. He. 2011. 3DQS: Distributed data access in 3D wireless sensor networks. In *Proceedings of IEEE ICC*. 1–5.

W. S. Massey. 1987. *Algebraic Topology: An Introduction*. Springer, New York.

A. Mostefaoui, A. Boukerche, M. A. Merzoug, and M. Melkemi. 2015. A scalable approach for serial aata fusion in wireless sensor networks. *Comput. Networks* 79 (2015), 103–119.

J. R. Munkres. 2000. *Topology* (2nd ed.). Prentice Hall, Upper Saddle River, NJ.

A. Nguyen, N. Milosavljevic, Q. Fang, J. Gao, and L. J. Guibas. 2007. Landmark selection and greedy landmark-descent routing for sensor networks. In *Proceedings of IEEE INFOCOM*. 661–669.

V. Pascucci. 2011. *Topological Methods in Data Analysis and Visualization*. Springer, Berlin.

S. Patil, S. R. Das, and A. Nasipuri. 2004. Serial data fusion using space-filling curves in wireless sensor networks. In *Proceedings of IEEE SECON*. 182–190.

G. Peano. 1890. Sur une courbe, qui remplit toute une aire plane. *Math. Ann.* 36, 1 (1890), 157–160.

L. L. Peterson and B. S. Davie. 2011. *Computer Networks: A Systems Approach* (5th ed.). Morgan Kaufmann.

G. J. Pottie and W. J. Kaiser. 2000. Wireless integrated network sensors. *Commun. ACM* 43, 5 (2000), 51–58.

S. Ratnasamy, B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin, and F. Yu. 2003. Data-centric storage in sensornets with GHT, a geographic hash table. *Mobile Network. Appl.* 8, 4 (2003), 427–442.

S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker. 2002. GHT: A geographic hash table for data-centric storage. In *Proceedings of ACM International Workshop on Wireless Sensor Networks and Applications*. 78–87.

J. M. Reason and J. M. Rabaey. 2004. A study of energy consumption and reliability in a multi-hop sensor network. *ACM SIGMOBILE Mobile Comput. Commun. Rev.* 8, 1 (2004), 84–97.

G. Reeb. 1946. Sur les points singuliers d'une forme de pfaff completement intgrable ou d'une fonction numrique. *Compt. Rend. Acad. Sances, Paris* 222 (1946), 847–849.

H. Sagan. 1994. *Space-filling Curves*. Springer-Verlag, New York.

R. Sarkar. 2014. *Geometric Methods of Information Storage and Retrieval in Sensor Networks*. Springer, Berlin, Chapter 14, 465–493.

R. Sarkar, X. Yin, J. Gao, F. Luo, and X. D. Gu. 2009. Greedy routing with guaranteed delivery using ricci flows. In *Proceedings of ACM/IEEE IPSN*. 121–132.

R. Sarkar, X. Zhu, and J. Gao. 2006. Double rulings for information brokerage in sensor networks. In *Proceedings of ACM MobiCom*. 286–297.

R. Sarkar, X. Zhu, and J. Gao. 2009. Double rulings for information brokerage in sensor networks. *IEEE/ACM Trans, Network,* 17, 6 (2009), 1902–1915.

R. Sarkar, X. Zhu, and J. Gao. 2013. Distributed and compact routing using spatial distributions in wireless sensor networks. *ACM Trans, Sens, Network,* 9, 3 (2013), 32:1–20.

Victor Shnayder, Mark Hempstead, Bor-rong Chen, Geoff Werner Allen, and Matt Welsh. 2004. Simulating the power consumption of large-scale sensor network applications. In *Proceedings of 2nd ACM SenSys*. 188–200.

F. Spitzer. 2001. *Principles of Random Walk*. Springer, Berlin.

R. Sugihara and R. K. Gupta. 2011. Path planning of data mules in sensor networks. *ACM Trans. Sens. Network.* 8, 1 (2011), 1:1–1:27.

G. Tan, S. A Jarvis, and A.-M. Kermarrec. 2009. Connectivity-guaranteed and obstacle-adaptive deployment schemes for mobile sensor networks. *IEEE Tran. Mobile Comput.* 8, 6 (2009), 836–848.

G. Tan, H. Jiang, J. Liu, and A.-M. Kermarrec. 2014. Convex partitioning of large-scale sensor networks in complex fields: Algorithms and applications. *ACM Trans. Sens. Network.* 10, 3 (2014), 41:1–41:23.

G. Tan, H. Jiang, S. Zhang, Z. Yin, and A.-M. Kermarrec. 2013. Connectivity-based and anchor-free localization in large-scale 2D/3D sensor networks. *ACM Trans. Sens. Network.* 10, 1 (2013), 6:1–6:21.

Y.-J. Tang, J.-J. Kuo, and M.-J. Tsai. 2014. Double-ruling-based location-free data replication and retrieval scheme in mobile ad hoc networks. In *Proceedings of IEEE ICCCN*. 1–8.

A. C. Viana, M. Dias de Amorim, Y. Viniotis, S. Fdida, and J. F. De Rezende. 2006. Twins: A dual addressing space representation for self-organizing networks. *IEEE Trans. Parallel Distrib. Syst.* 17, 12 (2006), 1468–1481.

C. Wang and H. Jiang. 2015. SURF: A connectivity-based space filling curve construction algorithm in high genus 3D surface WSNs. In *Proceedings of IEEE INFOCOM*. 981–989.

C. Wang, H. Jiang, T. Yu, and J. C. S. Lui. 2015. SLICE: Enabling greedy routing for large-scale 3D sensor networks with general topologies. *IEEE/ACM Trans. Network.* (2015), to appear.

L. Xie, Y. Shi, Y. T. Hou, and H. D. Sherali. 2012. Making sensor networks immortal: An energy-renewal approach with wireless power transfer. *IEEE/ACM Trans. Network.* 20, 6 (2012), 1748–1761.

K. Yang. 2014. *Wireless Sensor Networks*. Springer, Berlin.

Y. Yang, M. Jin, Y. Zhao, and H. Wu. 2013. Cut graph based information storage and retrieval in 3D sensor networks with general topology. In *Proceedings of IEEE INFOCOM*. 465–469.

Y. Yang, M. Jin, Y. Zhao, and H. Wu. 2015. Distributed information storage and retrieval in 3-D sensor networks with general topologies. *IEEE/ACM Trans. Network.* 23, 4 (2015), 1149–1162.

T. Yu, H. Jiang, G. Tan, C. Wang, C. Tian, and Y. Wu. 2013. SINUS: A scalable and distributed routing algorithm with guaranteed delivery for WSNs on high genus 3D surfaces. In *Proceedings of IEEE INFOCOM*. 2175–2183.

X. Yu, X. Yin, W. Han, J. Gao, and X. Gu. 2012. Scalable routing in 3D high genus sensor networks using graph embedding. In *Proceedings of IEEE INFOCOM*. 2681–2685.

C. Zhang, J. Luo, L. Xiang, F. Li, J. Lin, and Y. He. 2012. Harmonic quorum systems: Data management in 2D/3D wireless sensor networks with holes. In *Proceedings of IEEE SECON*. 1–9.

H. Zhang and H. Shen. 2009. Balancing energy consumption to maximize network lifetime in data-gathering sensor networks. *IEEE Trans. Parallel. Distrib. Syst.* 20, 10 (2009), 1526–1539.

Y. Zhang and W. Li. 2012. Modeling and energy consumption evaluation of a stochastic wireless sensor network. *EURASIP J. Wireless Commun. Network.* 2012, 1 (2012), 1–11.

H. Zhou, H. Wu, S. Xia, M. Jin, and N. Ding. 2011. A distributed triangulation algorithm for wireless sensor networks on 2D and 3D surface. In *Proceedings of IEEE INFOCOM*. 1053–1061.

Received June 2015; revised March 2016; accepted March 2016