Resilient Routing for Wireless Sensor Networks on High Genus Surfaces

Buri Ban, Hongyi Wu, and Miao Jin

Abstract—This paper considers a fundamental problem of designing routing scheme resilient to node or link failures for wireless sensor networks deployed on a surface of a complex-connected three-dimensional (3D) setting. Instead of heuristically detouring around the failed path, we borrow homotopy, an important topological concept, to effectively create and evaluate the diversity of alternative paths. We propose a tessellation-free and GPS-free method to compute paths with different homotopy types on surface networks. A source node greedily forwards a packet to its destination based on the computed nodes' virtual planar coordinates. When the current path fails, the source node can flexibly choose another greedy path from a different homotopy type to deliver the packet. The proposed algorithms are distributed and scalable to both the size and genus number of a surface network. We evaluate the performance of the proposed routing scheme under three different failure models. Simulation results show that our method achieves the best performance under geographically correlated failure models compared with other resilient routing schemes. We also compare our routing scheme with existing state-of-the-art ones specifically designed for surface networks when a network is failure free. Our method achieves the lowest stretch factor.

Index Terms—Surface sensor network, resilient routing, multipath routing, homotopy, scalable, distributed

1 Introduction

This paper focuses on a fundamental problem of designing routing algorithm resilient to node or link failures in a large-scale wireless sensor network deployed on the surface of a complex-connected three-dimensional ($3D$) setting, which generally has multiple handles forming a high genus surface [1]. Applications for such wireless sensor networks on high genus surfaces have been demonstrated for Structural Health Monitoring (SHM), e.g., at the Golden Gate Bridge [2] and National Stadium of China [3], where sensors are deployed on megastructures to gauge changes in materials or geometric properties that could hinder the systems performance. Other applications can be found at coal mine tunnels for disaster prevention and rescue, along the corridors of buildings for fire detection, and in water, sewer or gas systems for monitoring underground pipelines as introduced in [4]. We consider densely-deployed sensors that operate on high radio frequency and extremely low transmission power, which together result in short communication range. Only nearby sensors along the 3D surface can communicate with each other, whereas the wireless links connecting remote sensors across the space are negligible and can be removed via a simple preprocessing. As a result, the dense sensors form a 3D surface network.

1.1 An Overview of Resilient Routing

Routing is essential to sensor networks. In particular, node and link failures are unavoidable in a large-scale distributed sensor system. For example, nodes may die out of battery and communication links can be temporarily or permanently disabled due to attacks or interference. The routing scheme should be not only efficient and scalable but also resilient to sudden node or link failures, in order to adapt to network dynamics.

A few routing protocols have been proposed for 3D volume sensor networks [5], [6], [7], [8], [9], [10], [11], [12], [13], [14]. They are largely based on projection, mapping, or greedy embedding, which cannot be directly or efficiently applied for network routing on high genus surfaces. To this end, several greedy routing methods [1], [4], [15] have been developed recently to address the routing problem on high genus surfaces. [1], [4] propose to cut the original surface network open and then embed it to two-dimensional ($2D$) spaces. Each sensor node stores the embedded coordinates and use them to enable greedy routing. Both methods are scalable and can achieve guaranteed delivery. [15] proposes a two-level routing scheme by first decomposing the network and then realizing the route with greedy steps. However, none of them support resilience to node or link failures.

On the other hand, resilient routing has been discussed in conventional 2D sensor networks. For example, Directed Diffusion [16] is a well known routing algorithm that establishes gradients and uses gradual reinforcement of better paths to allow path recovery, enabling these systems to be robust to certain levels of network and sink dynamics. Multipath routing is also an efficient strategy to increase network resilience to node failures. Algorithms have been proposed for Internet multipath routing [17], [18], [19], [20], [21] and sensor network multipath routing [22], [23]. In [22], the authors consider two different approaches to construct multipaths between two nodes. One is node-disjoint multipath, where the alternate paths do not intersect with each other. The other approach builds many braided paths - partially disjoint alternate paths. In [23], a branch-aware flooding scheme is utilizes to construct a spanning tree and discover a set of node-disjoint paths for each sensor node back to the base station. Later, tree structure is applied for multipath routing [24], [25]. The general idea of these methods is to detour around the path with node and link failures. They can be extended to 3D surface sensor networks

- *Buri Ban and Miao Jin are with the Center for Advanced Computer Studies, University of Louisiana, Lafayette, LA 70503. E-mail: banburi.811@gmail.com, miao.jin@louisiana.edu*
- *Hongyi Wu is with the Center for Cybersecurity, Old Dominion University, Norfolk, VA 23529. E-mail: h1wu@odu.edu*

to improve resilience. However, they are generally best-effort heuristics, without a solid metric to evaluate the path diversity and to judge the level of resilience they can support.

1.2 Resilient Routing Based on Homotopy Types

Instead of heuristically detouring around the failed path, we borrow homotopy, an important topological concept and tool, to effectively create and evaluate the diversity of alternative paths. Informally speaking, two paths are said to be homotopic to each other if one can continuously deform to the other. The simplest case to illustrate the concept is a network deployed on a planar region with a lake inside. Two paths connecting a pair of source and destination are homotopic to each other if they are on the same side of the lake; otherwise they belong to different homotopy types because one path cannot continuously deform to the other without crossing the lake.

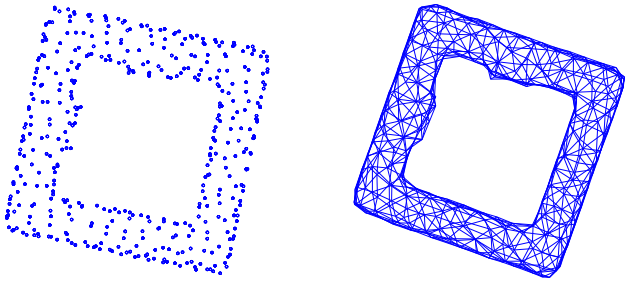


Fig. 1. (a) A wireless sensor network is deployed on a one-layer coal mine tunnel to monitor and prevent disaster. Note that we consider densely-deployed sensors that operate on high radio frequency and extremely low transmission power, which together result in short communication range. (b) Since only nearby sensors along the 3D surface can communicate with each other, whereas the wireless links connecting remote sensors across the space are negligible and can be removed via a simple preprocessing. We show a triangulation along the surface of the coal mine tunnel extracted from the connectivity graph of the sensor network.

Fig. 1(a) shows a wireless sensor network deployed on the surface of a one-layer coal mine tunnel to monitor and prevent disaster. The conventional underground mines communication often relies on Through-the-Earth (TTE) signaling [26] that uses ultra-low frequency ($300-3000Hz$) waves with long communication range to penetrate dirt and rock. However, its low data transmission rate limits the system performance, especially for modern bandwidth-hungry and time-sensitive applications. In this research, we consider densely-deployed sensors that operate on higher radio frequency (e.g., around $2.4GHz$ with $5MHz$ channel bandwidth) and extremely low transmission power, which together result in short communication range. The size of tunnel is much larger than the sensors communication range, therefore only nearby sensors along the 3D surface can communicate with each other, whereas the wireless links connecting remote sensors across the space are negligible and can be removed via a simple preprocessing. As a result, the dense sensors form a 3D surface network. Fig. 1(b) shows a triangulation along the surface of the coal mine tunnel extracted from the connectivity graph of the sensor network. We use colors to mark different routes from the central mine area to the entrance as shown in Fig. 2(a). Paths $P2$ and $P4$ are homotopic to each other, while paths $P1$, $P2$, and $P3$ all have different homotopy types. Similarly, Fig. 2(b) shows a wireless sensor network deployed on the surface of a two-layer coal mine tunnel. All routes from the central mine area of the second layer to the entrance have different homotopy types.

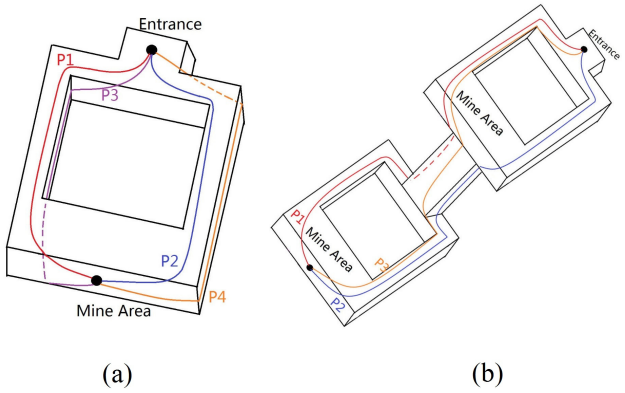


Fig. 2. (a) Among the four routes of the sensor network from the central mine area to the entrance, $P1$, $P2$, and $P3$ have different homotopy types. (b) A wireless sensor network is deployed on a two-layer coal mine tunnel. The three routes from the central mine area of the second layer to the entrance have different homotopy types.

As we can see, paths with different homotopy types between a given pair of source and destination are *strongly* disjoint. Since wireless sensor networks are generally deployed over harsh environments, regional uncontrollable disasters such as fires, or natural disasters such as floods, or collateral (non-targeted) damage in an attack cause geographically correlated node and link failures. The property that one path cannot continuously deform to another one ensures that geographically correlated node and link failures affecting one path cannot propagate easily to another one with different homotopy type. The chance that these paths fail simultaneously is extremely low under geographically correlated failures. Therefore, we propose to apply these paths for routing to achieve the desired resilience of surface sensor networks. As to be shown later, although they are not necessarily the shortest routes, the proposed approach results in negligible cost in terms of stretch ratio.

1.3 Tessellations in Hybrid Spaces

A recent work [27] applies the concept of homotopy to classify paths of a network deployed on $2D$ plane with uncovered holes. The authors propose to periodically embed such planar network into hyperbolic space. Paths connecting a source and tessellated target nodes in hyperbolic space correspond to paths with the same pair of source and target but different homotopy types on the plane.

It is obvious that the homotopy types of a network deployed on high genus surfaces are more complicated than on a planar region. The method in [27] cannot be directly applied on surface networks. However, we can extend our previous work in [28] to design a complicated tessellation-based routing algorithm to compute paths with the same pair of source and target but different homotopy types on surface networks.

Figure 3 illustrates the basic idea. Specifically, each sensor node needs to install two sets of different algorithms. One is Euclidean and the other is hyperbolic based according to Uniformization Theorem [29]. After deployment, a sensor network needs an algorithm to detect the topology, i.e., the genus number. For a network shown in Fig. 1(a), the algorithm identifies that it is topologically equivalent to a genus one surface shown in

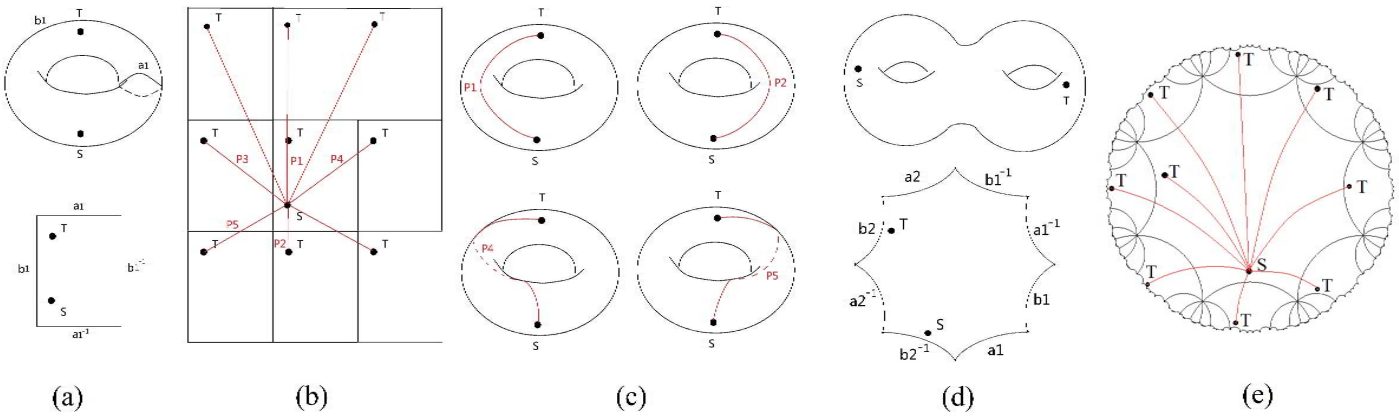


Fig. 3. Tessellations in hybrid spaces: (a) Top: A wireless sensor network deployed on a one-layer coal mine tunnel shown in Fig. 2 (a) is topologically equivalent to a genus 1 surface. The surface is marked with two closed loops a_1 and b_1. Bottom: the surface is cut open to a topological rectangle and conformally mapped to plane. Note that the source S and target T on the original surface are mapped to plane too. (b) Copies of the topological rectangle can be tessellated on the plane. Shortest paths connect S in the center domain to the other Ts. (c) Shortest paths shown in (b) correspond to homotopy different routes from S to T on the original surface. (d) Top: A wireless sensor network deployed on a two-layer coal mine tunnel shown in Fig. 2 (b) is topologically equivalent to a genus 2 surface. Bottom: The surface is cut open to a topological 8-polygon and conformally mapped to hyperbolic space. (e) Copies of the 8-polygon can be infinitely tessellated on hyperbolic space. Shortest paths connecting the S in the central domain to other Ts induce homotopy different routes from S to T on the original surface.

the top of Fig. 3(a). Intuitively, both surfaces have exactly one handle. Sensor nodes pick the set of Euclidean-based algorithms. The surface is cut open to a topological rectangle and conformally mapped to Euclidean plane as shown in the bottom of Fig. 3(a). The mapped rectangle is tessellated on plane as shown in Fig. 3(b). Since conformal mapping is one-to-one and continuous, the source and target nodes S and T on the original surface are also mapped to plane with planar coordinates, respectively. Each copy has a pair of S and T. Straight lines connecting S in the central rectangle and other Ts correspond to paths with different homotopy types on the original surface shown in Fig. 3(c).

However, for a network shown in Fig. 2(b), it is topologically equivalent to a genus two surface shown in the top of Fig. 3(d), i.e., surface with two handles. Sensor nodes need to pick the set of hyperbolic-based algorithms. The surface is cut open to a topological 8-polygon and conformally mapped to hyperbolic space as shown in the bottom of Fig. 3(d). The 8-polygon is tessellated on hyperbolic space as shown in Fig. 3(e). Each copy has a pair of S and T. Arcs connecting S in the central polygon and other Ts correspond to paths with different homotopy types on the original surface.

A more serious problem is that such tessellation-based algorithms including the one in [27] are not scalable as the stored information at each sensor node is proportional to the genus number of a surface network. Denote g the genus number of a surface. Each sensor node needs to store either $16g^2 - 8g$ number of the positions of different copies of the destination node, or $4g$ translations for later computation of the position of each copy of the destination node in hyperbolic space.

1.4 Our Approach in Euclidean Plane

We propose a totally different method to compute paths with different homotopy types on surfaces with tessellation free. What's more, the computation is fully based on Euclidean plane.

Here we use two examples to briefly illustrate the basic idea of how we compute paths with different homotopy types in Euclidean plane. Later, we will explain the necessary concepts of topology in a more formal way in Sec. 2.

For a network shown in Fig. 1(a), we first extract a triangulation from the network connectivity graph. The computed triangular surface is topologically equivalent to a genus one surface shown in Fig. 4(a). Intuitively, both surfaces have exactly one handle. Since topologically equivalent surfaces share the same number of homotopy types of paths, we use the genus one surface in Fig. 4(a) to show how our algorithm computes paths with different homotopy types on surface.

Considering homotopy is an equivalence relation for paths with fixed endpoints on surface, and paths with different homotopy types form a group. We first compute a complete set of generators of the homotopy group of surface. They are two closed loops denoted by a_1 and b_1 for genus one surface shown in the top of Fig. 4(a). We cut the surface open along a_1 and b_1 to a topological disk and then map it one-to-one and continuously to a unit one centered at the origin $(0,0)$ in Euclidean plane shown in the bottom of Fig. 4(a). The circumference of disk is a_1, b_1, a_1^{-1}, and b_1^{-1} sequentially, as each node along a loop is virtually split to two and each one is along one side of the cut loop. Each sensor node then takes the computed planar coordinates as its virtual coordinates.

Given a pair of source and target nodes in the network, denoted as S and T, a straight line connecting S and T in the unit disk induces the primary path P_1 shown in the first column of Fig. 4(b). If the primary path fails, S will send a packet along a path with different homotopy type. The first row of Fig. 4(b) shows paths with different homotopy types. The second row of Fig. 4(b) shows how S uses greedy forwarding to forward the packet to T based on virtual coordinates. Different greedy forwarding paths in the second row of Fig. 4(b) induce paths with various homotopy types in the first row of Fig. 4(b). Note that paths with different homotopy types correspond to various combinations of the generators of homotopy group. Here we use $P2$ as an example to explain. Path $P2$ crosses the homotopy group generator b_1. S forwards a packet along the shortest path to the arc of unit circle where b_1 is mapped, until the packet reaches a node in b_1. Since the node stores two sets of virtual planar coordinates corresponding to sides b_1 and b_1^{-1} respectively, we pick the planar coordinates corresponding to

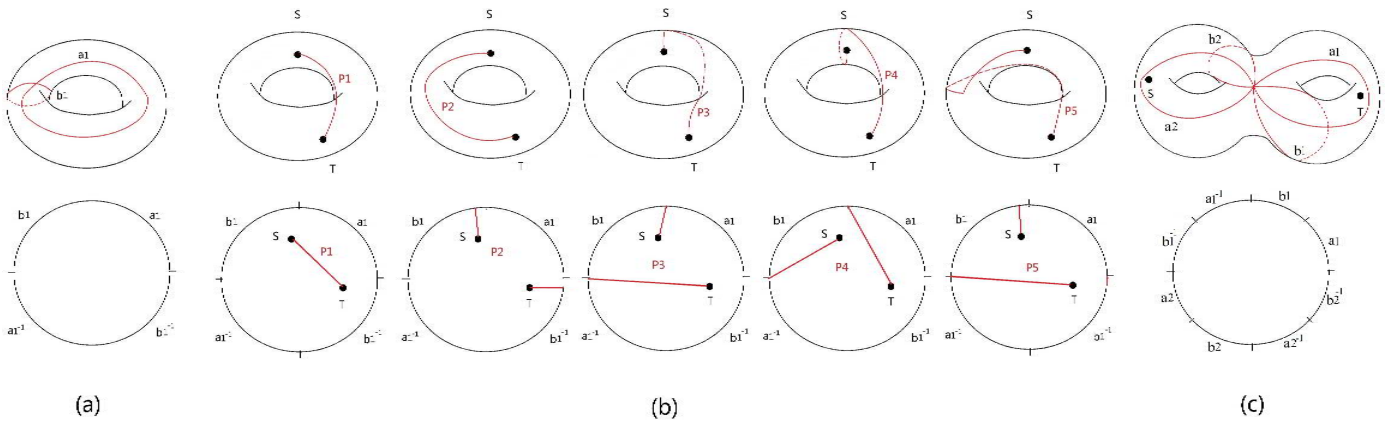


Fig. 4. Our approach: (a) Top: a genus 1 surface is marked with two closed loops a_1 and b_1. Bottom: the surface is cut open along the two loops and mapped to a unit disk centered at the origin $(0,0)$ in Euclidean plane. The map is continuous and one-to-one. The source and target nodes S and T on the original surface are mapped to plane too. Each node takes the computed planar coordinates as its virtual coordinates. (b) Different greedy forwarding paths in unit disk from S to T induce routes with different homotopy types from S to T on the original surface. Here we show five paths with various homotopy types. (c) Top: a genus 2 surface is marked with four closed loops a_1, b_1, a_2, and b_2. Bottom: The surface is cut open along the four loops and mapped to a unit disk centered at the origin $(0,0)$ in Euclidean plane.

side b_1^{-1} and start from the coordinates to forward the packet to T. Similarly, Paths $P3$ and $P4$ cross the homotopy group generator a_1 from two directions. Path $P5$ crosses the homotopy group generator b_1 first, and then a_1.

A network shown in Fig. 2(b) is topologically equivalent to a genus two surface shown in Fig. 4(c). Both surfaces have two handle. We cut a genus two surface open along four loops to a topological disk. We then map it one-to-one and continuously to a unit one centered at the origin $(0,0)$ in Euclidean plane shown in the bottom of Fig. 4(c). Paths from source to target with different homotopy types can be computed in a similar way as Fig. 4(b).

Our Contributions: Based on this idea, we propose the first routing scheme that achieves resilience to node or link failures in a large-scale wireless sensor network deployed on a surface of a complex-connected $3D$ setting. Specifically, our contributions are summarized as follows:

- We design a greedy routing scheme specifically for wireless sensor networks deployed on surfaces of complex-connected $3D$ settings. The routing algorithm does not require GPS information for individual sensor node. Greedy forwarding between a pair of nodes is always guaranteed based on computed nodes' virtual planar coordinates.
- The proposed routing scheme is the first one to achieve resilient routing for networks on surfaces of complex-connected $3D$ settings. Paths are classified by their homotopy types. A source can flexibly choose one greedy path from one homotopy type to deliver packet to its destination.
- The routing scheme is distributed and scalable to both the size and the genus of a network. Each sensor node only requires a limited and constant storage.

The rest of this paper is organized as follows: Sec. 2 introduces some closely related concepts in topology. Sec. 3 provides in detail the proposed resilient routing algorithm for wireless sensor networks deployed on high genus surfaces. Sec. **??** presents simulation results. Sec. 5 concludes the paper and discusses the future works.

2 Topology Background

Before giving the details of the proposed resilient routing algorithm on surface networks in Sec. 3, we introduce in an intuitive and informal way of the concepts in topology that are necessary to the algorithm. For rigid and formal definitions of these concepts, please refer classical textbooks in algebraic topology [30].

2.1 Orientable Surface and Its Genus

A surface is orientable if it has two distinct sides. General surfaces in real world are orientable surfaces. Denote M a connected and orientable surface embedded in 3D, and L a loop on M. L is surface separating if it can be expressed as the symmetric difference of boundaries of topological disks embedded in surface as shown in Figure 5 with L_1 and L_2; otherwise it is non-separating as shown in Figure 5 with L_3.

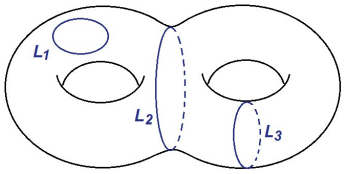


Fig. 5. Loops on a genus two surface: L_1 and L_2 are surface separating loops; L_3 is non-separating loop.

The genus of M, denoted as g, is the maximum number of disjoint non-separating loops $L_1, L_2, \cdots, L_g$ in M; that is, any L_i and L_j have no topological intersection if $i \neq j$, and $M \backslash (L_1 \cup \cdots L_g)$ is connected. The genus number is the most basic topology information of a surface and equals to the number of handles. For example, a torus is a genus one surface, and a double torus shown in Figure 5 is a genus two surface.

2.2 Homotopy Group and Its Generators

Definition 1 (Paths and Homotopy). Let $I = [0,1]$. A path in a space X is a continuous map

$$f : I \to X.$$

We say that f is a path between x_0 and x_1 if $f(0) = x_0$ and $f(1) = x_1$. A homotopy of paths in X is a continuous map

$$F : I \times I \to X$$

such that $f_t(s) = F(s,t)$ is a path between x_0 and x_1 for each $t \in I$. Two paths γ_0 and γ_1 are said to be homotopic if there exists a homotopy F such that

$$f_0 = \gamma_0, f_1 = \gamma_1.$$

They are denoted by $\gamma_0 \cong \gamma_1$.

Fig. 6 shows four paths between points x_0 and x_1 on a genus two surface. Only γ_0 and γ_1 are homotopic. Intuitively, only γ_0 and γ_1 can continuously deform to each other on the surface.

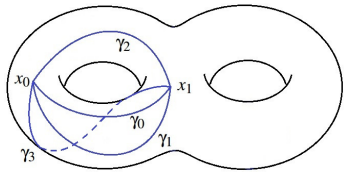


Fig. 6. Paths between points x_0 and x_1 on a genus two surface: only γ_0 and γ_1 are homotopic to each other.

Proposition 1. The relation of being homotopic is an equivalence relation on paths with fixed endpoints.

We denote the equivalence class of a path γ by $[\gamma]$. We define the product of equivalence classes of paths as

$$[f][g] = [f \cdot g].$$

It can be verified that the product of path classes is well defined, namely, it is independent of the representative path among the equivalence class. It can also be verified that the multiplication of equivalence classes of paths is associative

$$([f][g])[h] = [f]([g][h]).$$

We say f is a *closed path (or a loop) based at* p, if $f(0) = f(1) = p \in M$. We define $\varepsilon_p : I \to M$ as the constant path - a point, that is $\varepsilon_p(t) = p$. Then it is obvious that

$$[f][\varepsilon_p] = [f] = [\varepsilon][f].$$

We further define the inverse of a path $f^{-1}(t) = f(1-t)$, then it is obvious

$$[f][f^{-1}] = [\varepsilon_p] = [f^{-1}][f].$$

We denote the set of equivalence classes of closed paths based at $p \in M$ by $\pi(M,p)$.

From the above discussion, we see that $\pi(M,p)$ form a group, which is called the *fundamental group* or the *homotopy group* of M.

Let $p,q \in M$, if there is a path γ from p to q, then groups $\pi(M,p)$ and $\pi(N,q)$ are isomorphic, $u_\gamma : \pi(M,p) \to \pi(M,q)$,

$$u_\gamma[g] = [\gamma^{-1} \cdot g\gamma].$$

Therefore, we can omit the base point.

A canonical set of generators of the homotopy group of M consists of a set of $2g$ non-separating loops on M:

$$a_1, b_1, a_2, b_2, \cdots, a_g, b_g$$

such that any one of them intersects and only intersects with all the other generators at a single base point. These loops are generators of the homology group of the surface M.

Given a canonical set of the homotopy group generators $\{a_1, b_1, a_2, b_2, \cdots, a_g, b_g\}$, we can slice M along the loops and get a topological $4g$-polygon with boundary $a_1 b_1 a_1^{-1} b_1^{-1} a_2 b_2 a_2^{-1} b_2^{-1} \cdots a_g b_g a_g^{-1} b_g^{-1}$, where a_i^{-1} or b_i^{-1} simply indicates the inverse of the loop a_i or b_i. The $4g$-polygon is called a canonical fundamental domain of M.

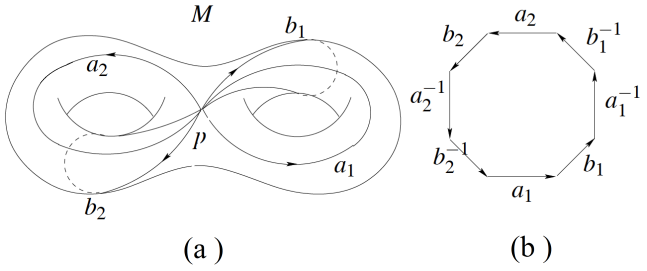


Fig. 7. (a) A canonical set of homotopy group generators marked on a genus two surface. (b) The surface is cut open to a $4g$-polygon, the canonical fundamental domain of the surface, along the set of loops.

Fig. 7 shows a canonical set of homotopy group generators marked on a genus two surface. The surface is cut open to a $4g$-polygon, the canonical fundamental domain as shown in Fig. 7(b), along the set of loops.

3 Algorithm of Resilient Routing

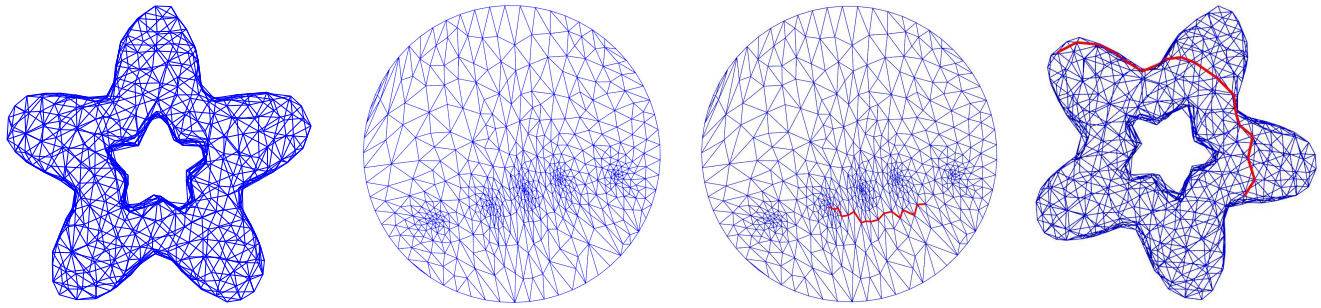
This section describes the proposed algorithm in three steps: preprocessing, mapping to unit disk, and routing.

Sec. 3.1 introduces the first step, preprocessing. Fig. 8 (a) shows a network model with extracted triangular structure after preprocessing, where vertices of the triangular mesh are the set of sensor nodes, and an edge between two neighboring vertices indicates the communication link between the two sensors.

Sec. 3.2 introduces the second step, computing virtual planar coordinates for each sensor node of the network. Fig. 8 (b) visualizes a planar embedding of the triangular mesh of the network given in Fig. 8 (a) onto a unit disk. Each sensor node stores the computed planar coordinates as its virtual coordinates.

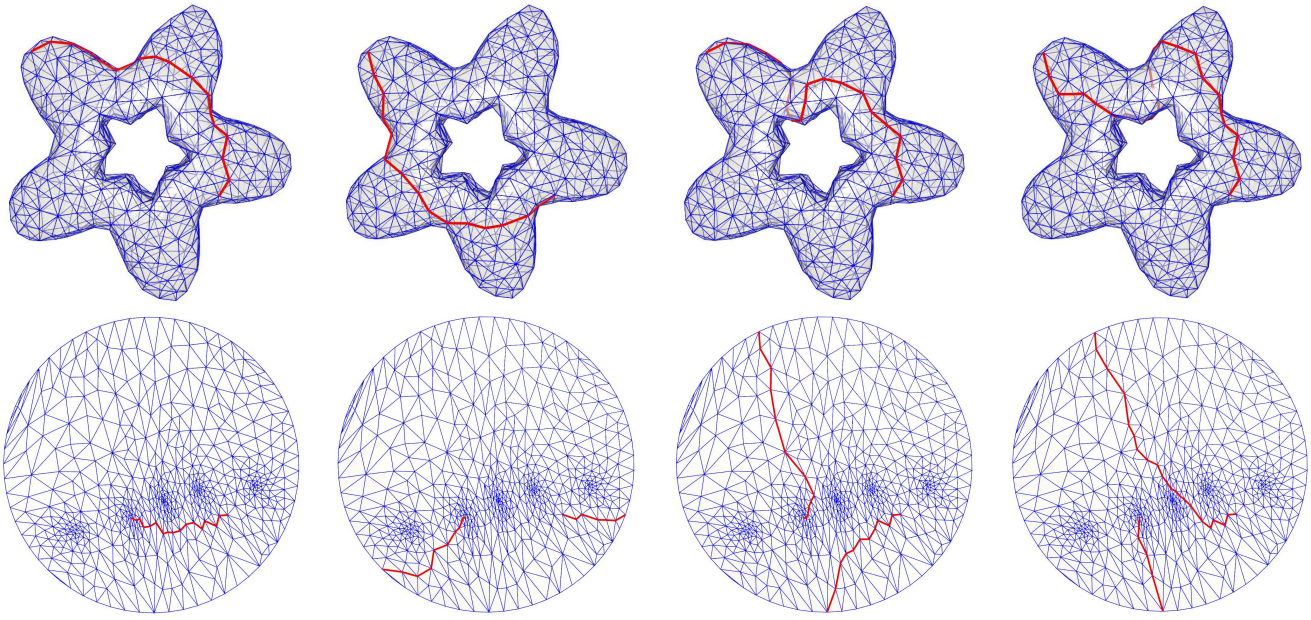
Sec. 3.3 explains the third step, a resilient routing strategy. A greedy forwarding path is computed and visualized based on the virtual planar coordinates stored at each sensor node as shown in Fig. 8 (c). Fig. 8 (d) shows the greedy path on the original network model. If the greedy forwarding path fails, the source node can flexibly choose an alternative path with different homotopy type based on the computed homotopy group generators. The source node then greedily forwards packets along the new path to the destination node. The first and second rows in Fig. 9 show paths with the same pair of source and target nodes but different homotopy types on the original network model and mapped planar disk, respectively. Note that the actual computation of all paths with different homotopy types is based on the mapped virtual planar coordinated visualized in the second row.

Later, we discuss some implementation issues in Sec. 3.4.



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

Fig. 8. (a) A network model with extracted triangular structure. (b) The embedding of the triangular mesh of the network onto plane. (c) A greedy forwarding path is computed and visualized based on the virtual planar coordinates stored at each sensor node. (d) The greedy path computed in (c) is shown on the original network model.



(a) Path I (b) Path II (c) Path III (d) Path IV

Fig. 9. The first and second rows show paths with the same pair of source and target nodes but different homotopy types on the original network model and mapped planar disk, respectively. Note that the actual computation of paths with different homotopy types is based on the mapped virtual planar coordinated visualized in the second row. We change the rendering method of the network model to better visualize the portion of a greedy path at the backside of the surface network model.

3.1 Preprocessing

Given a wireless sensor network deployed on a high genus surface, we apply the algorithm proposed in [31] to extract a triangular mesh. The algorithm takes the connectivity graph of the network as input. Each node constructs a local coordinates system based on the measured distances between nodes within one-hop communication range. Each edge computes a weight that measures the number of triangles shared by the edge and various local neighbor sets including neighboring nodes and connecting edges in the initial graph. Then an iterative algorithm keeps removing edges based on their weights until each edge in the final graph has weight exactly two shared by two triangles only. Note that the algorithm has no constraint on the communication model.

With the constructed triangular mesh M, we apply the algorithm introduced in [32] to find a set of canonical homotopy group generators. The algorithm starts from a randomly chosen triangle and grows triangles with a width first way. At each step of the growing, all the marked triangles always form a topological disk, and the marked edges form the boundary of the disk. When all the triangles have been traversed, the marked edges, i.e., the boundary, form a graph. The shortest loop on the graph is detected, corresponding to one generator of the surface homotopy group. Pick one node on the loop and then cut M open along the loop. The node is split to two boundary nodes. The shortest path connecting the two boundary nodes forms another generator of the surface homotopy group. The two loops actually correspond to one handle of the surface. Continuously, the algorithm detects all handles of the surface and their corresponding generators. It returns a set of marked $2g$ closed loops, where g is the genus number of M. The $2g$ loops are disjoint on the surface except for their common endpoint. These loops are generators of the surface homotopy group.

3.2 Mapping to Unit Disk

We virtually cut M open to a topological $4g$-polygon along the computed $2g$ closed loops. Practically, each node along the $2g$ loops is virtually split to two, and each one is along one side of the cut open loop (so $2g$ loops correspond to $4g$ sides and two sides of one loop form a pair). The exception is the base node where the $2g$ loops intersect. The base node is split to $4g$ ones - each corner node of the $4g$-polygon.

We then apply discrete harmonic map to map the $4g$-polygon to a unit disk centered at the origin $(0,0)$ in Euclidean plane. An intuitive way to understand discrete harmonic map is to model the cut open M as a spring system. Each edge is a spring with stretching energy. We fix the boundary of M to a unit circle centered at the origin $(0,0)$ and then let the non-boundary vertices settle in equilibrium. When the spring system is stable - none of the inner vertices is moving, the whole system achieves the minimum stretching energy and the position of each node is its mapped position in unit disk. Discrete harmonic map is proved a guaranteed diffeomorphism with planar convex shape boundary condition [33], [34].

Specifically, a randomly chosen corner node or the one with the lowest ID, initiates a message with an ID and one counter $(ID, count)$ that records the side ID of the $4g$-polygon and the size of boundary vertices along the current side. The corner node initiates the message as $(1,0)$ and forwards it to one of its neighbors along the boundary. Once a boundary node receives the message, it increases *count* by 1 and records the information in the message. If the boundary node is also a corner node, it forwards the *count* information back to boundary nodes along the same side of the $4g$-polygon, namely, nodes sharing the same *ID* as the corner node. The corner node then resets *count* to 0 and increases *ID* by 1, and then forwards the message to the next neighboring boundary node. The message keeps being forwarded until it comes back to the initial corner node. Assume the initial corner node is mapped to $(1,0)$, and the remaining boundary nodes are mapped uniformly and sequentially to the unit circle in a counter-clockwise direction. Each boundary node can easily compute its position along the unit circle based on its side ID, its count value and the size of boundary vertices along the side.

With boundary vertices uniformly and sequentially mapped to a unit circle, we initiate the positions of all inner vertices at the origin $(0,0)$. Then in each iteration, a non-boundary vertex node updates its position as the average of positions of its direct neighboring vertices. Each vertex receives a unique position in the unit disk when each non-boundary vertex has stopped updating its position. The fundamental domain of M is one-to-one and continuously mapped to a unit disk in plane. We then assign the computed planar coordinates as virtual coordinates of each node. Note that a boundary node has two sets of virtual planar coordinates. We set the threshold of the termination of harmonic map as $1e-6$. A non-boundary vertex stops updating its position if the difference with the previously computed one is less than the threshold. Such precision is enough to guarantee the mapping a diffeomorphism.

Algorithm 1 provides the detailed steps of mapping the cut open M to a unit disk using discrete harmonic map.

Algorithm 1 Algorithm using discrete Harmonic Map

Input: A $4g$-polygon M
Output: M mapped to a unit disk in plane

A corner node with the lowest ID, denoted as v_0 initiates a message with a segment ID and a counter as $(1,0)$ and forwards it to one of its neighbors along the boundary.
while The message not returned to v_0 **do**
  A boundary node v_i receives the message and set $counter++$.
  v_i stores segment ID and counter.
  if v_i is a corner node **then**
    v_i sends the $counter$ information back to boundary nodes with the same segment ID.
    v_i sets $ID++$ and $counter = 0$
  end if
  v_i forwards the message to its next neighboring boundary node.
end while
v_0 sends a message with the received segment ID (the total number of segments) to its next neighboring boundary node
while The message not returned to v_0 **do**
  A boundary node v_i receives the message and stores the ID.
  v_i forwards the message to its next neighboring boundary node.
end while
Each boundary node v_i computes its position along a unit circle denoted as u_i:

$$u_i = (cos(\frac{2\pi}{D}(d-1+\frac{n}{N})), sin(\frac{2\pi}{D}(d-1+\frac{n}{N})))$$

{For a boundary node, denote the stored segment ID, number of segments, position in the segment, and number of nodes in the segment as d, D, n, and N, respectively.}
For all non-boundary node v_i, initial $u_i = (0,0)$
while true **do**
  bool continue = false;
  for all Each non-boundary node v_i **do**
    $\acute{u}_i = \sum u_j$
    $\{v_j \in$ direct neighbors of $v_i\}$
    if $|\acute{u}_i - u_i| > \varepsilon$ **then**
      continue = true;
    end if
    $u_i = \acute{u}_i$;
  end for
  if !continue **then**
    return u_i as virtual coordinates of v_i
  end if
end while

3.3 Routing

Given a pair of source S and destination T nodes in network, the primary path is a straight line connecting S and T in the unit disk. S first sends a package storing the planar coordinates of T along the primary path. A node always forwards the packet to one of its neighbors, which is the closest to the destination T of the packet based on the computed virtual planar coordinates. Since the boundary of the mapped fundamental domain in plane is circle, greedy forwarding will never get stuck at boundary.

If the primary path fails, S will choose an alternative path with different homotopy type. We can generate infinite number of paths with different homotopy types, but we always choose the next shortest one based on approximated path length as discussed in Sec. 3.4.1. Assume the alternative path crosses side a_i. Based

on the side ID, S computes the closest point, denoted by P, on the arc where side a_i is mapped. S then forwards the package including the side ID, the planar coordinates of P and T to P. Later the package is forwarded to a boundary node closest to P along side a_i. Since the boundary node stores two sets of planar coordinates corresponding to sides a_i and a_i^{-1} respectively, we pick the planar coordinates corresponding to side a_i^{-1} and start from the coordinates to greedy forward the packet to T.

Algorithm 2 provides the detailed steps of greedy forwarding a packet to an alternative path when the primary one fails.

Algorithm 2 Algorithm of choosing an alternative path

Input: M with planar virtual coordinates stored at each node, a pair of S and T

Output: An alternative path from S to T with different homotopy type

1: Assume we have chosen a homotopy type represented as a sequence of homotopy group generators $L = a_i b_i$.
2: **while** L not empty **do**
3: Remove the first generator a_i from L.
4: Greedy forward a packet to the arc where segment a_i is mapped.
5: The packet is forwarded to a closest boundary node v_i on segment a_i.
6: Pick the planar coordinates stored at v_i corresponding to side a_i^{-1} as the current position of the packet.
7: **end while**
8: Greedy forward the packet to T.
{Greedy forwarding is purely based on planar virtual coordinates.}

3.4 Discussions

3.4.1 Sorting

Giving a pair of S and T nodes on M, we can generate infinite number of paths with different homotopy types if we allow the path to cross any combinations of the generators of the homotopy group of M. However, considering the efficiency and security of a network, we have to choose a limited number of paths among them to avoid unnecessary retransmissions when the primary path fails.

A straightforward idea is to sort and choose paths with the shortest lengths. Paths with the shortest length will not only consume less energy of the network, but also have better resilience to network failures. In general, the more complicated homotopy type a path has, the longer the path will be with a given pair of S and T. So we first sort paths that cross only one generator of the homotopy group of M. If it is necessary, we then sort paths that cross combinations of two generators of the homotopy group of M.

Assume a path crosses the generator a_i. It is similar for b_i. The path can cross loop a_i on M from its left or right direction, which corresponds in the embedded fundamental domain that the path starting from S crosses side a_i or a_i^{-1}. S computes the distances to the two arcs where sides a_i and a_i^{-1} are mapped, denoted by d_0 and d_1 respectively. S also computes the distances of T to the two arcs, denoted by d_2 and d_3 respectively. The sum of d_0 and d_3 is the approximated length of a path crossing a_i from its left direction. Similarly, the sum of d_1 and d_2 is the approximated length of a path crossing a_i from its right direction. S compares the two paths and chooses the one with the shorter length. Usually crossing a generator from two directions would generate paths of totally different lengths, so the approximation error caused by distortion can be remedied. Finding the closest point on a generator sometimes might lead to high traffic near the endpoints of the generator. One solution is that if such node has been used in building other source-target paths for several times, then it can be virtually disabled to future path setting requests.

3.4.2 Greedy Forwarding

Greedy forwarding based on the computed virtual planar coordinates will always be successful along the boundary since the boundary of the fundamental domain is mapped to a unit circle in plane. However, greedy forwarding may get stuck at some non-boundary node because some triangle around that node may be mapped to an obtuse one in plane. Although all angles are acute for all the mappings we obtained in our simulations, we can handle such situation by allowing information exchange of the stuck node with its 2-hop neighbors. The packet can then jump out local minimum and keep greedy forwarding. In rare case when a packet still gets stuck with its 2-hop neighbors' information, face routing [35] can be applied to guide the packet out of the local minimum.

3.4.3 Time Complexity and Communication Cost

Given a network with n nodes, we now discuss the steps that dominate the computing time and communication cost of the overall algorithm. Specifically, the time complexity to extract a triangular mesh M from the connectivity graph of the network is $O(n)$. The time complexity to find a canonical set of fundamental group generators of M is $O(gn)$, where g is the genus number of the triangular mesh. The time complexity of computing harmonic map can be measured by the number of iterations given by $O(nlogn)$.

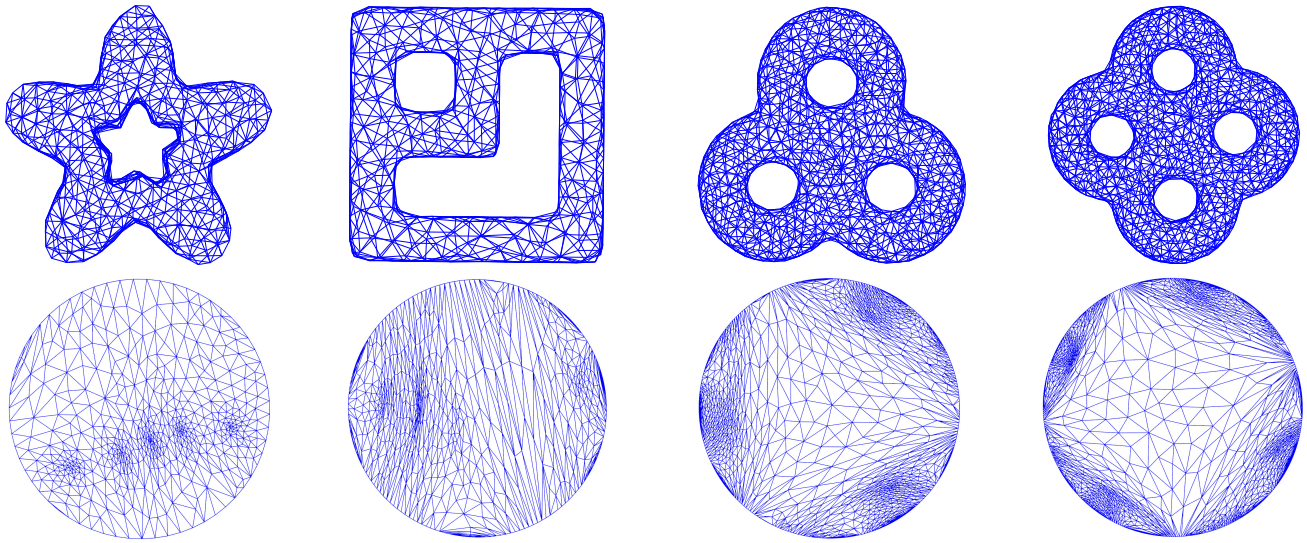
The communication cost, measured by the number of messages, is $O(dnlogn)$ for computing harmonic map where d is the average vertex degree of M, since each vertex only needs to exchange messages with its direct neighbors in each iteration. The communication cost of other steps of the algorithm is linear to n since it is a completely local algorithm.

Although the time complexity of the majority of existing multipath algorithms [16], [22], [23] is linear to a network size, these algorithms need to recompute for a new pair of source-destination nodes. While our algorithm only needs to compute virtual coordinates for each sensor node once. Assume every node of a network can be a source or destination, the overall time and communication complexity of existing multipath routing algorithms is $O(n^3)$. In contrast, the time and communication complexity of our algorithm is $O(nlogn)$ and $O(dnlogn)$ respectively, scalable to both the size and genus number of a network.

Our algorithm is also fully distributed. The only broadcasting through the whole network is the genus number after we compute a canonical set of homotopy group generators. Then a node only needs to communicate with its direct neighbors to compute virtual coordinates for later greedy routing.

4 Simulations

The proposed resilient routing algorithm applies for large-scale sensor networks deployed on surfaces of coal mine tunnels for disaster prevention and rescue, or corridors of buildings for fire



(a) Network I (b) Network II (c) Network III (d) Network IV

Fig. 10. The first row shows various network models with extracted triangular structures. The second row shows the embedding of the triangular mesh of each network onto plane. Greedy forwarding is based on the computed virtual planar coordinates stored at each sensor node.

detection, or sewer or gas systems for monitoring underground pipelines. These surfaces generally have complex shapes and multiple handles, i.e., genus numbers.

To this end, we create multiple representative surface models with various genus numbers ranging from one to four. Sensor nodes varying from 500 to 1000 are randomly deployed on these surface models to carry out simulations. As we discussed in Sec. 3.1, we apply the algorithm proposed in [31] to extract a triangular mesh. The algorithm has requirement on the node density but no constraint on the communication model. Therefore, we assume a surface model is initially fully covered by sensors, and the average neighboring degree of each node is at least six. All sensors have a common transmission range and communication model. The first row of Fig. 10 shows the extracted triangular meshes from the connectivity graph of networks based on locally estimated distances between nodes. The second row of Fig. 10 shows the embedding of the triangular mesh of each network onto plane. Greedy forwarding is based on the computed virtual planar coordinates of each sensor node.

In our simulations, we check three different failure models of a network: independent node failure, where each node in network has an equal and independent probability of failure during some time interval; small geographically correlated failure, where all nodes within a certain fixed radius fail simultaneously and locations of the centers of these regions are along the primary path; and giant geographically correlated failure, in which all nodes within a large radius fail simultaneously such that the surface network is disconnected at some handle.

We use three important metrics to evaluate the performance of a resilient routing algorithm. The first metric is the successful delivery ratio. Assume the source and destination nodes can be any pair of network nodes. The successful delivery ratio measures the probability of an algorithm can successfully find an alternate path within a limited number of retransmissions when the primary path between a randomly chosen pair of source and destination nodes is broken. The successful delivery ratio indicates the resilience of a routing algorithm. The second metric is the average number of switching to alternate paths before a message is forwarded to destination successfully. This metric reflects the consumed energy of a successful delivery, and we expect a small number. The third metric is delay ratio. We divide the end-to-end hop count from source to target including the cost of forwarding messages, informing source of failed paths, and local detours, by the hop count of primary path.

We compare our method with other multipath resilient routing methods including idealized node-disjoint multipath [22], directed diffusion [16], and N-to-1 multipath routing [23]. To avoid unlimited number of trying alternate paths, and control the energy consumption and delay cost, we require that all multipath methods can try at most a constant number of alternate paths for a given pair of source and destination nodes when the primary one fails. The number is application sensitive and depends on the size of a network. We set it to 5 in our simulations.

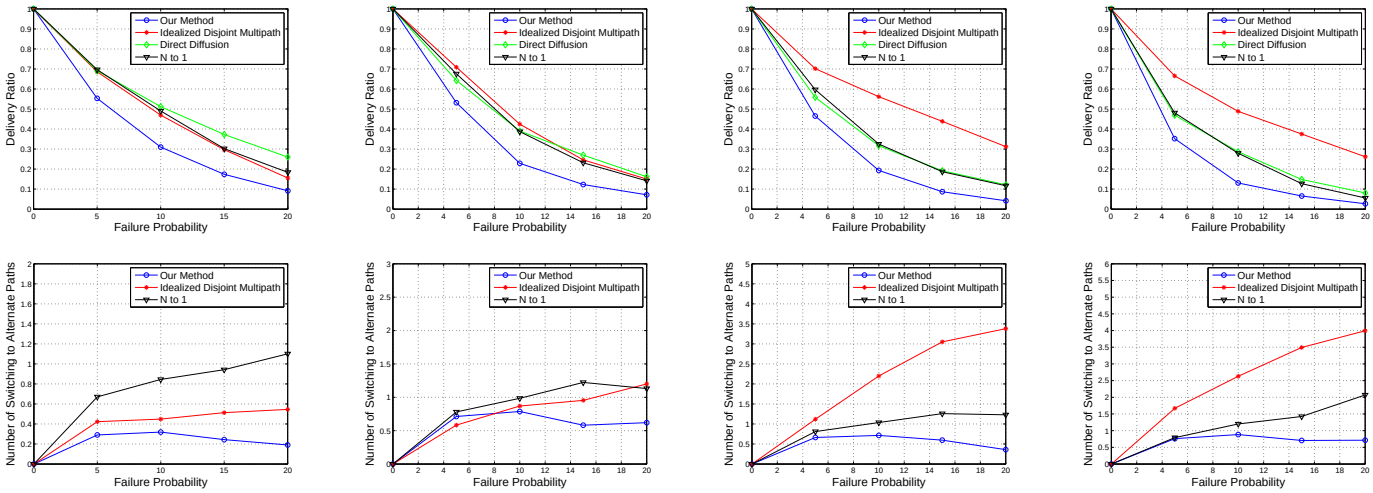
For each network model, we randomly choose 100 source and destination nodes, respectively, and then construct a total of 1×10^4 source and target pairs. Simulation results given in Secs. 4.1, Secs. 4.2, and Secs. 4.3 show that our method achieves the best performance under geographically correlated failures including small and giant failure models.

In the case when a network is failure free, we compare our algorithm with existing routing schemes specifically designed for surface networks with guaranteed delivery. Simulation results given in Sec. 4.4 show a consistently low stretch factor of our algorithm over previous routing schemes.

In Sec. 4.5, we analyze the computing time of the proposed resilient routing algorithm.

4.1 Independent Node Failure

The first row of Fig. 11 compares the successful delivery ratio of different methods as a function of the probability of independent node failure for networks shown in Fig. 10. As the probability increases, the successful delivery ratio of all methods decreases dramatically. When the probability of node failure in network is around 10%, almost half of the sampled source-destination pairs



(a) Network I (b) Network II (c) Network III (d) Network IV

Fig. 11. Independent Node Failure: the first row gives the successful delivery ratio as a function of the probability of independent node failure. The second row shows the average number of switching to alternate path before a message is forwarded to destination successfully.

fail. For the same set of network models, the second row of Fig. 11 shows the average number of switching to alternate path for a successful delivery. Considering that it is difficult and also unfair to count the number of the alternate paths for directed diffusion, we exclude it from the comparison. Since the successful delivery ratios are too low for all comparison methods, the discussion of delay ratio of a successful delivery is unnecessary.

Fig. 11 clearly shows that switching to an alternate path does not help when the probability of node failure is high. The reason is that an alternate path is always longer than the primary one for any multipath method, so the failure probability of the alternate path is always higher than the primary one assuming each node has an equal failure probability. All multipath methods are intrinsically low resilient to independent node failures and perform no better than a local detour one.

Therefore, we set a local detour method as an initial resilient routing strategy, but at the same time, each node periodically updates its neighboring nodes and local mesh structure. Either a single or a set of connected nodes failure will result in holes. Given a mesh structure, a non-boundary edge is shared by exactly two triangle faces, and a boundary edge is shared by only one. The two ending vertices of a boundary edge are boundary vertex. Boundary edges and vertices along newly generated holes can be easily detected in a mesh structure. A boundary node initiates a message with a counter set to 0 and forwards it to one of its neighbors along the boundary. When a boundary node receives the message, it increases the counter by 1. The message keeps being forwarded until it comes back to the initial boundary node. If the size of the hole is large or keeps growing, the initial boundary node will send out a message to the whole network to alert a multipath strategy for routing.

4.2 Small Geographically Correlated Failure

The first row of Fig. 12 compares the successful delivery ratio of different methods as a function of the radius of geographically correlated node failure. As the radius of a failure region increases, the successful delivery ratio of directed diffusion decreases dramatically. However, our method consistently performs the best over other multipath methods.

The second row of Fig. 12 shows the average number of switching to alternate path for a successful delivery. With the increase of the radius of failure region, the average number of switching to alternate path increases correspondingly for all methods. It is reasonable since all methods need to find an alternate path geographically spread out to avoid the failure region. The difference is that our method requires the least number of switching to achieve the highest successful delivery ratio.

The third row of Fig. 12 shows the average delay ratio of a successful delivery. It is clear that paths with different homotopy types are not necessarily short routes. However, the highest successful delivery ratio of our algorithm can justify such moderate delay cost.

4.3 Giant Geographically Correlated Failure

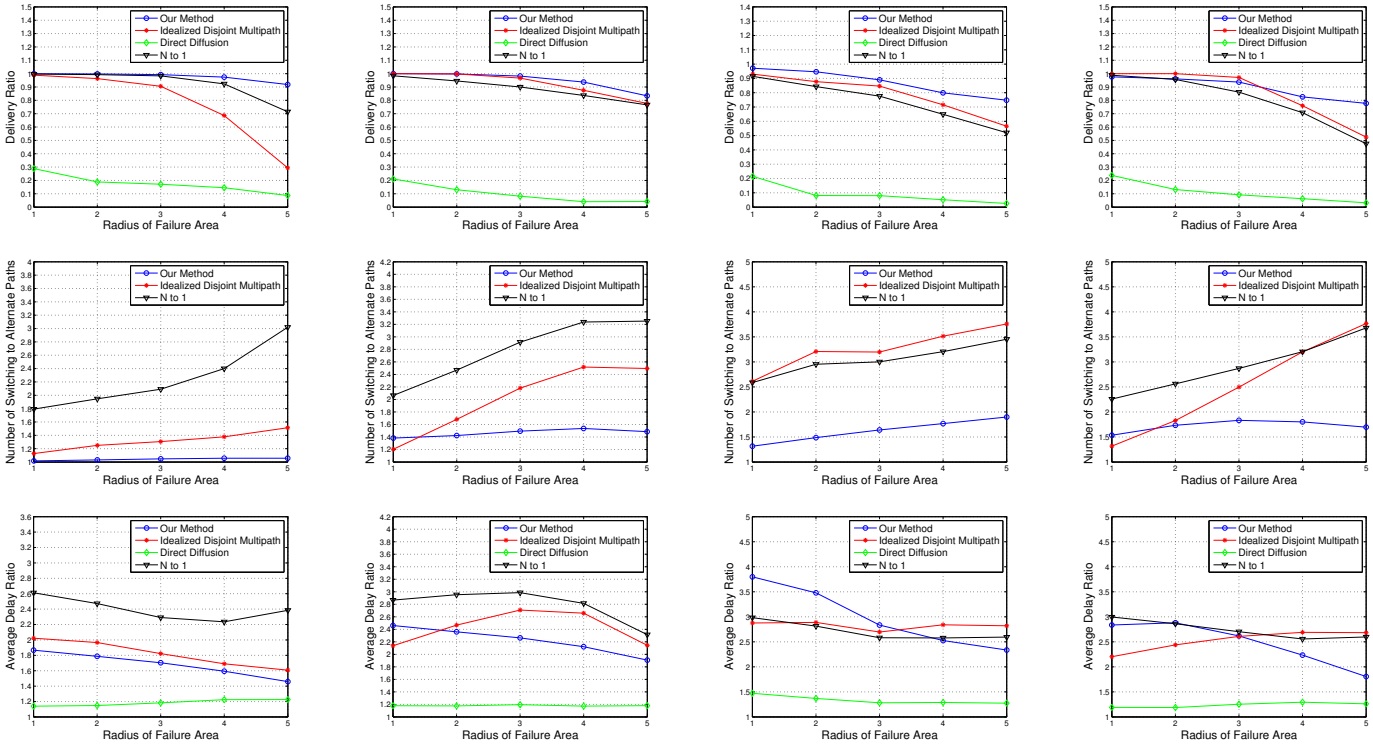
Fig. 13 (a) compares the successful delivery ratio of different methods when at least one handle of a network is disconnected. Directed diffusion cannot tolerate such large area of node failure. Its successful delivery ratio drops to 0 for all models. Idealized node-disjoint multipath and N-to-1 multipath routing perform not well either. On the contrary, our algorithm can jump far enough to avoid a huge failure area.

Fig. 13 (b) shows the average number of switching to alternate path for a successful delivery. Since the maximal number of switching to alternate path is set to 5, we can see that both Idealized node-disjoint and N-to-1 multipath routing have to try at least 3 alternate paths on average to find a successful one. Our method achieves the highest successful delivery ratio with the lowest number of switching to alternate path.

Fig. 13 (c) shows the average delay ratio of a successful delivery. Since our algorithm requires a far lower number of switching to alternate paths than others under such failure model, our algorithm achieves consistently the least delay ratio.

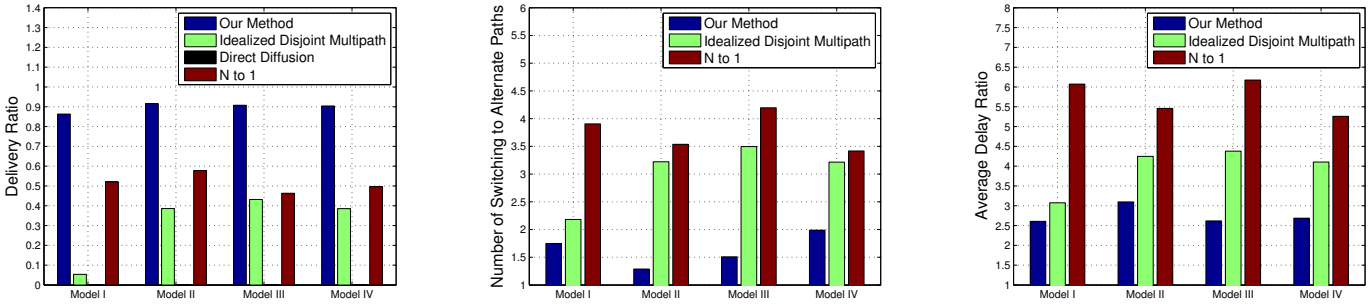
4.4 Surface Routing

If a network is failure free, our algorithm guarantees 100% delivery rate between all pairs of nodes. Since previous greedy routing schemes including GPSR [36], its variants, and BVR



(a) Network I (b) Network II (c) Network III (d) Network IV

Fig. 12. Small Geographically Correlated Failure for network models: the first row gives the successful delivery ratio of different methods as a function of the radius of geographically correlated node failure. The second row shows the average number of switching to alternate path before a message is forwarded to destination successfully. The third row shows the average delay ratio compared to primary path.



(a) (b) (c)

Fig. 13. Giant Geographically Correlated Failure: (a) the average successful delivery ratio of different methods. (b) the average number of switching to alternate path before a message is forwarded to destination successfully. (c) the average delay ratio compared to primary path.

[37] cannot guarantee successful delivery in surface networks, we compare ours with existing routing schemes specifically designed for surface networks with guaranteed delivery.

We compare our method with two other routing methods. One is the SINUS method [4] - the most recent routing algorithm specifically designed for WSNs on high genus surfaces. Their method cuts a closed high genus surface network open to a topological annulus and then maps to a planar annulus. Greedy routing is carried out based on the planar virtual coordinates of each sensor node. The other is the Random-Walk (RW) algorithm [9] working well for general $3D$ networks including surface networks. In our simulations, we pick RW-GRG algorithm that applies greedy forwarding in the original network graph instead of the dual one, because it achieves better performance on $3D$ surface network routing than other RW algorithms.

Note that a source node S in our method has the flexibility to choose a particular target node T in universal cover to deliver a package. In our comparison, we let S to choose one from a group of Ts including the one in the central domain and the $4g$ ones in one layer of the central domain based on the shortest distance in the embedded universal cover. Figure 10 gives the embedded one-layer universal cover of the testing models in Euclidean plane and $2D$ hyperbolic space.

4.4.1 *Stretch Factor*

We run routing queries on $20,000$ randomly selected source-destination pairs on each network model. Since all the three

methods including ours, the SINUS method, and the Random-Walk algorithm are guaranteed for successful delivery, the delivery rates are all 100%. We then measure the stretch factor that is the ratio of lengths between the routed path and the shortest one for each route. Figure 14(a) compares the average stretch factors of the three methods on the network models.

It is obvious that our method shows a consistently low stretch factor for each model. However, the SINUS method shows an increased stretch factor with the increased number of genus of the testing model. With more genus numbers, more cuts are required to slice each handle open to cut the originally closed high genus surface to a topological annulus. Pairs with source and target nodes close to each other in original network become far away when source and target nodes locate on each side of the cuts introduced by the SINUS method. The stretch factor of the Random-Walk algorithm, on the contrary, depends on the shape of a network model instead of its genus number. The reason is that the performance of Random-Walk algorithm is mostly decided by the number of local minimums along source-target pairs. If a network model has many local concave shapes, the Random-Walk algorithm has to trigger the expensive recover step frequently. So the stretch factors of the Random-Walk algorithm vary a lot for different models.

4.4.2 Network Load

For each model, we measure the load of each node for the same routing queries we randomly chosen for computing the stretch factors. Figures 14(b) and (c) compare the average load and the maximum load per node of the three methods on the network models.

Both the average load and the maximum load per node of the Random-Walk algorithm vary a lot for different models, which shows again a strong dependency on the shape of a network model. Local minimums at concave regions require frequent local recover steps of the Random-Walk algorithm, which induces heavy network traffic around those concave regions.

Our method has consistently lower average load and maximum load per node than the SINUS method for each model. Both methods show an increased pattern of the maximum load per node with the increased genus number of the network models. However, the SINUS method has a much higher maximum load per node. The reason is that nodes around the inner boundary of the mapped planar annulus take much heavier traffic of a network.

4.5 Computing Time

Given a network with the size of sensor nodes n, the time complexity of our overall algorithm is linear to n. We discuss steps that dominate the computing time of the overall algorithm.

The time complexity of the algorithm to extract a triangular mesh M from the connectivity graph of the network based on locally measured distances is $O(n)$. The time complexity of the algorithm to find a canonical set of fundamental group generators of M is $O(gn)$, where g is the genus number of the triangular mesh.

The time complexity of harmonic map is measured by the number of iterations. In our experiments, we set the step length to 0.5, and the threshold of the maximal moving distance of non-boundary nodes to $1e-3$. Fig. 15 gives the convergence rate of discrete harmonic map on different networks. The maximal moving distance of non-boundary nodes at each iteration, denoted as error, decreases exponentially fast, so the convergence time is less than 4 seconds for all network models.

5 Conclusion and Future Works

The routing scheme we proposed in the paper is the first work to achieve resilient routing for networks deployed on a surface of a complex-connected $3D$ setting. Paths are classified by their homotopy types. A source node can flexibly choose one greedy path from a homotopy type to deliver packet to its destination. Our routing scheme does not require GPS information for individual node. Greedy forwarding along a route is always guaranteed based on the computed nodes' virtual planar coordinates. The proposed algorithms are distributed with information exchanged between neighboring nodes. They are also scalable to both the size and genus number of a surface network with a small and constant storage at each node. Simulation results show that our resilient routing scheme achieves the best performance under geographically correlated failure models compared with the state-of-the-art resilient routing algorithms. We also compare our routing scheme with existing ones specifically designed for surface networks in the case when a network is failure free. Simulation results show that our routing scheme achieves the lowest stretch factor compared with all existing state of the art surface network routing schemes. In our future works, we will work on resilient routing algorithms to achieve a high delivery rate for surface networks under independent node failure model.

Acknowledgments

Buri Ban and Miao Jin are partially supported by NSF CCF-1054996 and CNS-1320931. Hongyi Wu is partially supported by NSF CNS-1320931.

References

[1] X. Yu, X. Yin, W. Han, J. Gao, and X. Gu, "Scalable routing in 3d high genus sensor networks using graph embedding," in *INFOCOM'12, mini-conference*, pp. 2681–2685, 2012.

[2] S. Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, and M. Turon, "Health monitoring of civil infrastructures using wireless sensor networks," in *2007 6th International Symposium on Information Processing in Sensor Networks*, pp. 254–263, 2007.

[3] Y. Shen, P. Yang, P. Zhang, Y. Luo, Y. Mei, H. Cheng, L. Jin, C. Liang, Q. Wang, and Z. Zhong, "Development of a multitype wireless sensor network for the large-scale structure of the national stadium in china," *International Journal of Distributed Sensor Networks*, vol. 2013, pp. 1–16, 2013.

[4] T. Yu, H. Jiang, G. Tang, C. Wang, C. Tian, and Y. Wu, "Sinus: A scalable and distributed routing algorithm with guaranteed delivery for wsns on high genus 3d surfaces," in *INFOCOM'13*, pp. 2175–2183, 2013.

[5] T. F. G. Kao and J. Opatmy, "Position-Based Routing on 3D Geometric Graphs in Mobile Ad Hoc Networks," in *Proc. of CCCG*, pp. 88–91, 2005.

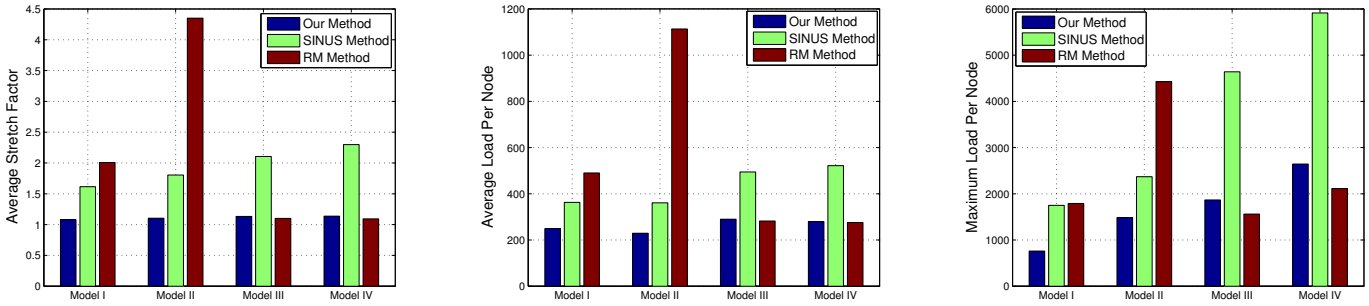
[6] J. Opatrny, A. Abdallah, and T. Fevens, "Randomized 3D Position-based Routing Algorithms for Ad-hoc Networks," in *Proc. of Mobiquitous*, pp. 1–8, 2006.

[7] F. Li, S. Chen, Y. Wang, and J. Chen, "Load Balancing Routing in Three Dimensional Wireless Networks," in *Proc. of ICC*, pp. 3073–3077, 2008.

[8] S. Durocher, D. Kirkpatrick, and L. Narayanan, "On Routing with Guaranteed Delivery in Three-Dimensional Ad Hoc Wireless Networks," in *Proc. of International Conference on Distributed Computing and Networking*, pp. 546–557, 2008.

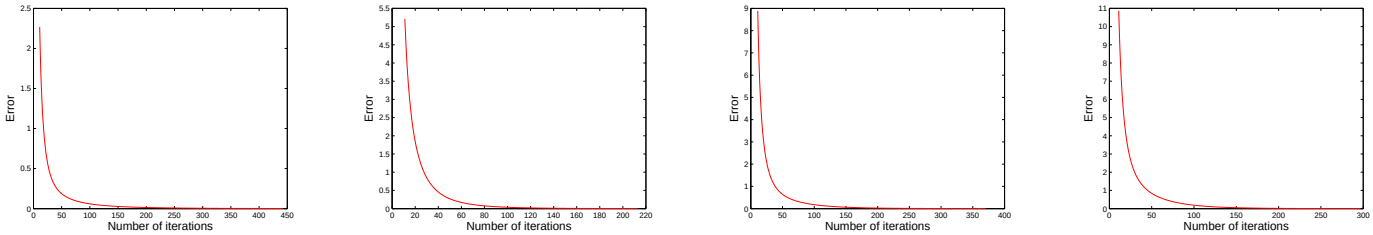
[9] R. Flury and R. Wattenhofer, "Randomized 3D Geographic Routing," in *Proc. of INFOCOM*, pp. 834–842, 2008.

[10] C. Liu and J. Wu, "Efficient Geometric Routing in Three Dimensional Ad Hoc Networks," in *Proc. of INFOCOM*, pp. 2751–2755, 2009.



(a) Average routing stretch factor (b) Average load per node (c) Maximum load per node

Fig. 14. Comparison of our proposed method, the SINUS method, and the Random-Walk algorithm on network models shown in Fig. 10.



(a) Network I (b) Network II (c) Network III (d) Network VI

Fig. 15. Convergence rate of discrete harmonic map on different networks with step length 0.5 and error threshold $1e-3$. The convergence time is less than 4 seconds for all network models.

[11] J. Zhou, Y. Chen, B. Leong, and P. S. Sundaramoorthy, "Practical 3d geographic routing for wireless sensor networks," in *SenSys '10*, pp. 337–350, 2010.

[12] S. Xia, X. Yin, H. Wu, M. Jin, and X. Gu, "Deterministic greedy routing with guaranteed delivery in 3d wireless sensor networks," in *Proc. of MobiHoc*, pp. 1–10, 2011.

[13] S. S. Lam and C. Qian, "Geographic routing in d-dimensional spaces with guaranteed delivery and low stretch," in *SIGMETRICS '11*, pp. 257–268, 2011.

[14] S. Xia, M. Jin, H. Wu, and H. Zhou, "Bubble routing: A scalable algorithm with guaranteed delivery in 3d sensor networks," in *Proc. of SECON*, pp. 245–253, 2012.

[15] X. Yu, X. Yin, W. Han, J. Gao, and X. Gu, "Scalable routing in 3d high genus sensor networks using graph embedding," in *INFOCOM*, pp. 2681–2685, 2012.

[16] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed Diffusion: a Scalable and Robust Communication Paradigm for Sensor Networks," in *Proc. of MobiCom*, pp. 56–67, 2000.

[17] M. Shand and S. Bryant, "Ip fast reroute framework," 2009.

[18] A. Atlas and A. Zinin, "Basic specification for ip fast reroute: Loop-free alternates," 2008.

[19] C. Reichert, Y. Glickmann, and T. Magedanz, "Two routing algorithms for failure protection in ip networks," in *Proc. of ISCC*, pp. 97–102, 2005.

[20] S. Ray, R. Guérin, K.-W. Kwong, and R. Sofia, "Always acyclic distributed path computation," *IEEE/ACM Transactions on Networking*, vol. 18, no. 1, pp. 307–319, 2010.

[21] Y. Ohara, S. Imahori, and R. V. Meter, "MARA: Maximum Alternative Routing Algorithm," in *Proc. of INFOCOM*, pp. 298–306, 2009.

[22] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin, "Highly-resilient, energy-efficient multipath routing in wireless sensor networks," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 5, no. 4, pp. 11–25, 2001.

[23] W. Lou, "An efficient n-to-1 multipath routing protocol in wireless sensor networks," in *Proc. of MASS*, pp. 664–672, 2005.

[24] M. Motiwala, M. Elmore, N. Feamster, and S. Vempala, "Path splicing," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 4, pp. 27–38, 2008.

[25] J. Gao and D. Zhou, "Resilient and low stretch routing through embedding into tree metrics," in *Proc. of WADS*, pp. 438–450, 2011.

[26] Wikipedia, "Through-the-earth mine communications — wikipedia, the free encyclopedia," 2015. [Online; accessed 19-July-2015].

[27] W. Zeng, R. Sarkar, F. Luo, X. Gu, and J. Gao, "Resilient routing for sensor networks using hyperbolic embedding of universal covering space," in *Proc. of INFOCOM*, pp. 1694–1702, 2010.

[28] M. Jin, J. Kim, F. Luo, and X. Gu, "Discrete Surface Ricci Flow," *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, vol. 14, no. 5, pp. 1030–1043, 2008.

[29] W. Abikoff, "The Uniformization Theorem," *The American Mathematical Monthly*, vol. 88, no. 8, pp. 574–592, 1981.

[30] J. R. Munkres, *Elements of Algebraic Topology*. Addison Wesley Co., 1984.

[31] H. Zhou, H. Wu, S. Xia, M. Jin, and N. Ding, "A Distributed Triangulation Algorithm for Wireless Sensor Networks on 2D and 3D Surface," in *Proc. of INFOCOM*, pp. 1053–1061, 2011.

[32] F. Lazarus, M. Pocchiola, G. Vegter, and A. Verroust, "Computing a canonical polygonal schema of an orientable triangulated surface," in *Proc. of SoCG*, pp. 80–89, 2001.

[33] H. Kneser, "Losung der aufgabe 41," *Jahresbericht der Deutschen Mathematiker-Vereinigung*, vol. 35, pp. 123–124, 1926.

[34] G. Choquet, "Sur un type de transformation analytique generalisant la representation conforme et define au moyen de fonctions harmoniques," *Bulletin des Sciences Mathematiques*, vol. 69, pp. 156–165, 1945.

[35] P. Bose, P. Morin, I. Stojmenović, and J. Urrutia, "Routing with guaranteed delivery in ad hoc wireless networks," in *Proceedings of the 3rd International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, DIALM '99, pp. 48–55, 1999.

[36] B. Karp and H. T. Kung, "Gpsr: Greedy perimeter stateless routing for wireless networks," in *Proc. of MobiCom*, pp. 243–254, 2000.

[37] R. Fonseca, S. Ratnasamy, J. Zhao, C. T. Ee, D. Culler, S. Shenker, and I. Stoica, "Beacon vector routing: Scalable point-to-point routing in wireless sensornets," in *Proc. of NSDI*, pp. 329–342, 2005.

Buri Ban Buri Ban is a software engineer in WePay inc. He received the B.S. and M.S. degrees from the School of Electronic and Information Engineering, Beijing Jiaotong University in 2009 and 2012, respectively, and Ph.D. degree in computer science from the Center for Advanced Computer Studies (CACS) at University of Louisiana, Lafayette in 2018. His Ph.D. dissertation title is Network Resilience Against Dynamic Changes.

Hongyi Wu is the Batten Chair in Cybersecurity and the Director of the Center for Cybersecurity Education and Research at Old Dominion University (ODU). He is also a Professor in Department of Electrical and Computer Engineering. Before joining ODU, he was an Alfred and Helen Lamson Endowed Professor at the Center for Advanced Computer Studies (CACS), University of Louisiana at Lafayette (UL Lafayette). He received the B.S. degree in scientific instruments from Zhejiang University, Hangzhou, China, in 1996, and the M.S. degree in electrical engineering and Ph.D. degree in computer science from the State University of New York (SUNY) at Buffalo in 2000 and 2002, respectively. His research focuses on networked cyber-physical systems for security, safety, and emergency management applications, where the devices are often light-weight, with extremely limited computing power, storage space, communication bandwidth, and battery supply. He received NSF CAREER Award in 2004 and UL Lafayette Distinguished Professor Award in 2011.

Miao Jin is an associate professor in the Center for Advanced Computer Studies (CACS), University of Louisiana at Lafayette (UL Lafayette). She received the B.S. degree in computer science from Beijing University of Posts and Telecommunications, Beijing, China, in 2000, and the M.S. and Ph.D. degrees in computer science from the State University of New York at Stony Brook, Stony Brook, NY, USA, in 2006 and 2008, respectively. Her research interests lie at the boundary of geometry and broad engineering fields including Mobile and Wireless Networks, Computer Vision, Computer Graphics, and Machine Learning. Her research results have been used as cover images of mathematics books and licensed by Siemens Healthcare Sector of Germany for virtual colonoscopy. She received NSF CAREER Award in 2011, Jack & Gladys Theall/BoRSF Professorship in 2013, Lockheed Martin Corporation/BoRSF Professor in 2016, and UL Lafayette College of Science Distinguished Professor Award in 2020.