Topology Dependent Space Filling Curves for Sensor Networks and Applications

Xiaomeng Ban* Mayank Goswami† Wei Zeng‡ Xianfeng Gu* Jie Gao*

* Department of Computer Science, Stony Brook University. {xban, gu, jgao}@cs.stonybrook.edu
† Department of Applied Mathematics and Statistics, Stony Brook University. mgoswami@ams.sunysb.edu
‡ School of Computing and Information Sciences, Florida International University. wzeng@cs.fiu.edu

***Abstract*—In this paper we propose an algorithm to construct a "space filling" curve for a sensor network with holes. Mathematically, for a given multi-hole domain $\mathcal{R}$, we generate a path $\mathcal{P}$ that is provably aperiodic (i.e., any point is covered at most a constant number of times) and dense (i.e., any point of $\mathcal{R}$ is arbitrarily close to $\mathcal{P}$). In a discrete setting as in a sensor network, the path visits the nodes with progressive density, which can adapt to the budget of the path length. Given a higher budget, the path covers the network with higher density. With a lower budget the path becomes proportional sparser. We show how this density-adaptive space filling curve can be useful for applications such as serial data fusion, motion planning for data mules, sensor node indexing, and double ruling type in-network data storage and retrieval. We show by simulation results the superior performance of using our algorithm vs standard space filling curves and random walks.**

I. Introduction

We consider a sensor network that densely covers a planar domain, possibly with multiple network holes. In this paper we develop algorithms to linearize the network, i.e., 'covering' the sensor network by a single path. By enforcing a linear order of the sensor nodes one can carry serial logical definitions and serial operations on both the sensor nodes and the sensor data. We list a number of such applications in the following.

Serial data fusion. When a signal is spread over an area larger than the coverage range of a single sensor, we will need to use multiple sensors to collaboratively detect the distributed signal. One type of data fusion mechanisms, called *serial fusion* [5], [29], combines sensor observations in a linear fashion to derive hypothesis. A state is maintained and passed on from sensor to sensor along a serial path, incorporating new observation at each step. This is in contrast with *parallel fusion* mechanism in which sensors independently process their data and pass the output to a centralized fusion center. There are pros and cons for serial fusion v.s. parallel fusion respectively. One particular advantage of serial fusion is that the fusion process can be stopped as long as there is enough evidence to support or reject the hypothesis, while in parallel fusion all data will be sent to the fusion center nevertheless. The implementation of the serial data fusion in a distributed network requires a path that visits all the nodes in a linear order [20].

Motion planning of data mules. Collecting data from sensor networks to a static data sink often suffers from communication bottleneck near the sink. One way to address this is to use a mobile sink, or called a data mule, implemented by a mobile device touring around the network to collect data through direct communication with a sensor in close proximity. Besides collecting sensor data, a data mule can also be helpful for sensor network maintenance such as battery recharge, beacon-based localization [3], [13], etc. A data mule moves along a path. Planning the motion of a data mule requires a path that visits the nodes in the network with minimum duplicate visits. When there are multiple data mules in the network, a flexible set of paths that can be used by the data mules with minimum coordination and minimum interference (e.g duplicate visits by different mules) will be handy.

Sensor node indexing. Another application of representing a sensor network by a linear order is for indexing sensor nodes or sensor data [14]. A number of indexing schemes for multi-dimensional input first take a space filling curve to 'linearize' the input and then apply standard 1D indexing mechanisms.

In the following we first review previous work of linearizing a two dimensional continuous domain or a discrete two dimensional network, before we present our ideas.

A. Related Work

Space filling curves. In the continuous setting, various space filling curves have been defined for a square region [21]. The narrow definition of space filling curve, in mathematical analysis, refers to a curve whose range contains the entire 2-dimensional unit square (or more generally an N-dimensional hypercube). Space filling curves were initially discovered by Giuseppe Peano and are also called Peano curves. These curves are often recursively constructed. See Figure 1 for an instance of the Hilbert curve. The basic recursive structure is to replace a line segment by a zig-zag pattern. In a recursive step, each segment is replaced by a scaled and rotated version of this pattern. The larger number of recursions used, the denser the curve becomes. Mathematically every point of the unit square is on the curve, given an infinite number of recursions. For a discrete set of points it suffices to take a sufficiently high number of recursions to generate a linear order of the points. Space filling curves in this narrow definition only apply to 2-dimensional (or N-dimensional) unit squares (hyper-cubes). When the domain is irregular and/or has holes the space filling curve will be chopped into many disconnected pieces. Very little work is known about extending the space filling curves to other shapes. The only work known is a heuristic algorithm [11] with a modified Hilbert curve for an ellipse.

Hamiltonian paths. In a discrete setting such as a graph, a natural analog of a space filling curve is a Hamiltonian

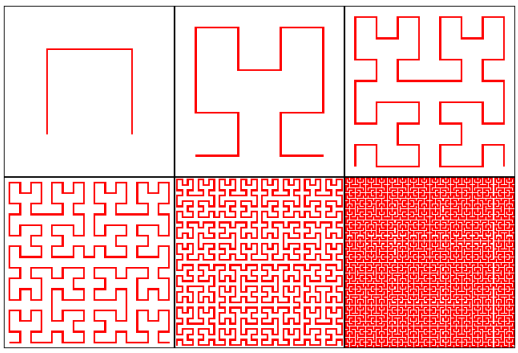


Fig. 1: The Hilbert curve (source: Wikipedia).

cycle or a Hamiltonian path, i.e., a cycle or a path that visits each vertex once and only once. Only a subset of graphs has a Hamiltonian path and determining whether a Hamiltonian path or a Hamiltonian cycle exists in a given graph (whether directed or undirected) is NP-complete, even in restricted families such as planar graphs [7].

Traveling salesman tour. When a metric is defined between any two nodes, the traveling salesman problem (TSP) asks for the shortest tour that visits each node once and only once. In our setting the distance between two nodes can be either the graph distance or the Euclidean distance. The latter becomes the Euclidean TSP. Both the metric TSP and the special case of Euclidean TSP are NP-complete. For the metric TSP, the heuristic of using the Euler tour on the minimum spanning tree gives a two-approximation. With some additional tricks, the Christofides algorithm [6] gives a $3/2$ approximation. For the Euclidean TSP, polynomial approximation schemes (PTAS) are known [2], [19] to find a $(1+\varepsilon)$ approximate solution for any $\varepsilon > 0$. Such algorithms are mostly of theoretical interest. When multiple tours are allowed (e.g., multiple data mules), the problem of minimizing the total travel distance collectively done by all tours becomes the multiple traveling salesman problem (mTSP), which is also NP-complete and does not have any efficient approximation algorithms [4]. Existing solutions for mule planning are all heuristic schemes [9], [12], [16], [25], [28].

Random walk. A practically appealing solution for visiting nodes in a network is by random walk. The downside is that we encounter the coupon collector problem. Initially a random walk visits a new node with high probability. After a random walk has visited a large fraction of nodes, it is highly likely that the next random node encountered has been visited before. Thus it takes a long time to aimlessly walk in the network and hope to find the last few unvisited nodes. Theoretically for a random walk to cover a grid-like network, the number of steps is quadratic in the size of the network [18]. For a random walk of linear number of steps, there are a lot of duplicate visits as well as a large number of nodes unvisited at all. In the case of multiple random walks, since there is little coordination between the random walks, they may visit the same nodes and duplicate their efforts.

A major problem with all the above constructions is that the curve found does not have adaptive density. A space filling curve has a fixed density, determined by the threshold of the recursion. Hamiltonian paths and TSP will generate fixed length paths. In sensor network applications such as serial fusion and data mule planning, the length of a path may be restricted by travel budget or required fusion delay. If we start with a high density curve, we spend a lot of time visiting nodes in one region of the network before we ever get information from another region. Instead, we may want to adopt a visiting scheme such that we 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. Our construction is one of this type.

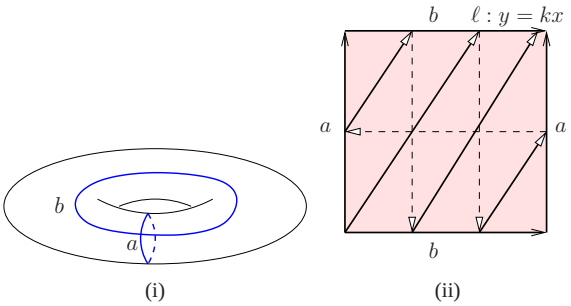


Fig. 2: (i) A torus cut open along two curves a, b. (ii) The flattened torus. The line $\ell : y = kx$ is shown on the flattened torus (the top and bottom edges are the cut b, the left and right edges are the cut a). Since the top edge and bottom edge are actually the same, the line will go through the torus as shown by the parallel lines. It will not intersect itself and can be shown to be arbitrarily close to any point on the torus.

B. Our Contribution

In this paper, we propose a scheme to generate a curve that (i) densely covers any geometric domain with possibly holes; (ii) have a coverage density proportional to its length. To understand the main idea, we first consider a torus. See Figure 2. We cut a torus open with two cuts a, b, and flatten it as a square in the plane with the top edge identified as the bottom edge and the left edge identified as the right edge.

We will consider the universal covering space by packing an infinite number of translated copies of the torus to cover the entire two dimensional plane, with the origin at the bottom left corner of one such copy. Now take a straight line ℓ with a slope k being an irrational number. Mapping back to the original torus, the line becomes a curve that spirals around the torus for infinitely long and never repeats itself. Figure 2 (ii) shows a part of the curve on the torus. We could prove that the curve has no self-intersections and the curve is dense, i.e., any point p of the torus is arbitrarily close to the curve.

With the basic construction for a torus, we will generalize it to any planar domain with holes. Specifically, for a simple domain with no holes, we will first map it one-to-one to a unit square, and then flip the square along the top edge and the right edge to get four copies, creating a torus. Then we find the dense curve on the torus. Since any point in the original domain is mapped to four copies on the torus, the curve we find will visit any point for at most four times. The property of being dense still holds. For a domain with holes, we will first double cover it, i.e., creating two copies of the network, the upstairs copy and the downstairs copy. The two copies are

glued to each other along the hole boundaries to create a multi-torus, each hole being a handle. In the same way we choose one handle to flatten the torus, and the rest of the handles are mapped to very narrow 'slits'. A line with irrational slope in the covering space, when hitting a slit, bounces back. We could show that the curve will visit each point of the original domain at most twice and is provably dense.

The mapping of a general two dimensional domain to a multi-torus is handled by conformal map. Computing a conformal map for deforming the shape of a sensor network has been done by using Ricci flow to change the network curvature, in a number of prior work [10], [22], [23], [32]. We remark that the tools we use in this paper is different. Our current method is based on holomorphic differentials from Riemann surface theory [8]. Imagine an electric field on a surface, then the equipotential lines are orthogonal to the electric field lines everywhere, the pair of electric field lines and the equipotential lines form the holomorphic 1-form. All holomorphic 1-forms on a surface form a group, which is isomorphic to the first homology group of the surface. We select a special holomorphic 1-form, such that the integration of the 1-form gives a special conformal map. Assume the network is a planar domain with multiple holes, then the conformal map transforms the domain to an annulus with concentric circular slits. Two boundaries are mapped to the inner and outer circles, the other boundaries are mapped to the slits. This type of maps can not be carried out by Ricci flow method, because Ricci flow requires the target curvature given a priori. But in this scenario, neither the position nor the radii of circular slits are known at the beginning. On the other hand, Ricci flow is a non-linear method in nature; whereas holomorphic differential method is a linear one, which is computationally more efficient.

The conformal map is computed for a given network field at the network initialization phase. The computation can be carried with only the network geometric domain $Omega$, if the sensors are densely deployed inside Ω. With the map computed the dense curve can be found and followed locally by simply specifying an irrational slope. This leads to naturally decentralized computations and planning in the network that can benefit data storage and data mule collection.

In the following we first present the theory of finding a dense curve in a continuous domain. The algorithmic details follow. We present simulation results and comparisons with space filling curves and random walks at the last.

II. Theoretical Foundation

In this section, we will present the theoretical foundation, including rigorous proof and computational methodology, of our dense curve computation. We show how to find a dense curve for a continuous planar domain with a canonical shape under different topologies, including topological quadrilaterals (i.e., simple domain without hole), topological annulus (i.e., with one hole), and topological annuli (i.e., with multiple holes). In the next section we will show the detailed algorithm

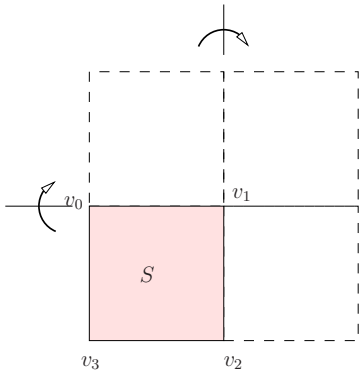
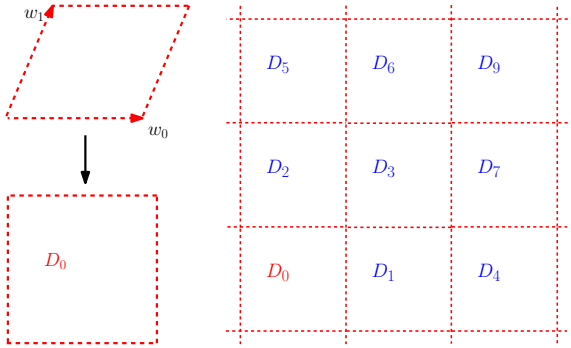


Fig. 3: Reflect twice to create a torus with four copies of a square.



(a) square mapping (b) universal covering space

Fig. 4: Conformal mapping for a topological torus.

to deform any planar domain to be one of the canonical shapes so that we can apply the dense curve computation.

A. Dense Curve For Annulus and Simply Connected Domains

Denote by S a planar domain. If S has one hole, we double cover it – creating two copies, one upstairs copy S and one downstairs copy $-S$ and glue the two copies along the corresponding boundaries to form a torus.

If S is simply connected, we first map it to a square and denote by v_0, v_1, v_2, v_3 the four corners on the outer boundary. And now take four copies of the domain, essentially first reflect along the top edge v_0v_1 and then reflect the two copies along the right edge v_1v_2. This will make it a torus. See Figure 3. For both cases, we need to design a dense curve on a torus.

A torus T^2 can be conformally and periodically mapped onto the plane, namely, the torus can be treated as

$$T^2 = \mathbb{R}^2/\Gamma,$$

where Γ is the lattice formed by

$$\Gamma := \{m\mathbf{e_1} + n\mathbf{e_2} | m, n \in \mathbb{Z}\},$$

e_1 and e_2 are linearly independent translations. By an affine transformation as shown in Figure 4, we can deform the lattice to be the regular integer lattice, namely, $\mathbf{e_1} = (1, 0)$ and $\mathbf{e_2} = (0, 1)$.

We define a dense curve on a 2D domain T^2 as an infinitely long straight line in its universal covering space $\mathbb{R}^2$, with irrational slope. It is continuous and non self-intersecting, and uniformly distributed, as demonstrated by the following theorem.

Theorem 2.1 (Weyl's Equidistribution Theorem [26], [30]). *Let x be a irrational number, $\langle nx \rangle$ represents the fractional part of nx. Then*

- *$\langle nx \rangle$ is dense in $[0, 1]$.*
- *$\langle nx \rangle$ is equi-distributed in $[0, v]$, namely for large integer N,*

$$\lim_{N \to \infty} \frac{\#\{n < N | \langle nx \rangle \in (c, c + \epsilon)\}}{N} = \epsilon.$$

The following result is a well known corollary of the equidistribution theorem.

Theorem 2.2 (Dense Curve On a Torus). *Suppose $T^2 = \mathbb{R}^2/\Gamma$, where Γ is the canonical integer lattice $\Gamma = \{(m, n) | m, n \in \mathbb{Z}\}$. Let γ be a straight line with irrational slope on $\mathbb{R}^2$, $\pi : \mathbb{R}^2 \to T^2$ is the projection, then $\pi(\gamma)$ is dense and equi-distributed on T^2.*

B. Dense Curve For Multiply Connected Domain

A relatively similar idea can be applied for multiply connected domains. As shown in Figure 5, the input network in (a) is a planar domain with $n + 1$ boundary components $\{\gamma_0, \gamma_1, \cdots, \gamma_n\}$, where γ_0 is the exterior boundary component. The domain is then conformally mapped to an annulus with concentric slits in (b), such that γ_0 is mapped to the outer circle, γ_1 is mapped to the inner circle, γ_k, $1 < k \leq n$, are mapped to concentric circular arcs. The shortest path(s) from $\gamma_k, k > 1$ to γ_0 are denoted as τ_k. By taking the complex logarithm, the annulus in (b) is mapped to a rectangle with horizontal slits, as shown in (c). Note that, the annulus is cut along τ_1, therefore the left and right vertical boundaries of the rectangle are both τ_1. Denote by the rectangular domain as D'.

The heights, starting x-coordinates and the lengths of the slits (denoted as (h_i, s_i, l_i) respectively) are conformal invariants of the sensor network domain. In other words these are the "fingeprints" of the domain [1]. There can only be two cases here:

1) All of h_i, s_i and l_i are rational.
2) At least one of the h_i, s_i and l_i are irrational.

The curve we construct is a simple "billiards" path (starting from the lower left corner of the rectangle) in the rectangular domain D'. We take a curve by starting from the bottom left corner of D' with an irrational slope. When the path hits a slit, top or bottom boundary of D', it is bounced back. The two vertical sides of the rectangle are identified and the curve continues at the point from the same height on the left vertical side when it exited the right vertical side from. See Figure 6 for details.

Using techniques from complex dynamics, we can show that:

1) In case 1, a billiards path with irrational slope is dense and ergodic.
2) In case 2, the set of initial directions (from $(0, \frac{\pi}{2})$) which ensure density of the resulting billiards path is measure 1. This means that if we pick a direction randomly, the resulting curve will be dense with probability 1.

The proofs of the above claims are omitted from this abstract due to space constraints.

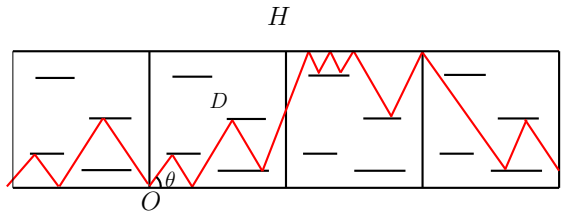


Fig. 6: Tiling of H by copies of D and the curve γ_θ

C. Comparison With Space Filling Curves

A space filling curve will be 'filling up' the square, i.e., every point of the square is on the curve. A space-filling curve must be everywhere self-intersecting in the technical sense that the curve is not injective[1]. Intuitively, a non self-intersecting curve can never fill up the square as the two have different topologies.

In our case, we generate a continuous curve which is non self-intersecting on a torus. It does not *go through* all the points but it is *arbitrarily* close to all points — any point p is within distance ε of the curve, for $\varepsilon > 0$ to be arbitrarily small. For a simple domain, since we use four copies to create a torus, our dense curve will visit each point of the original domain at most four times. For a domain with holes, we use two copies to create a torus or a multi-torus. Thus a dense curve in our construction will visit each point of the original domain at most twice. This ensures that the dense curve in the applications will eventually visit the entire network and does not visit any particular node too often.

III. Algorithms for Discrete Conformal Mapping

In this section we go through the computational tools to deform an input domain to our canonical shape of a torus or a multi-torus on which we define a dense curve. Our method is based on conformal geometry. The following theorem lays down the theoretic foundation for our method:

Theorem 3.1 (Ahlfors [1] - Slit Conformal Map). *Suppose Ω is a planar domain with multiple boundary components $\partial\Omega = \{\gamma_0, \gamma_1, \cdots, \gamma_n\}$, then there exists a conformal map $\phi : \Omega \to \mathbb{D}$, where D is a unit planar annulus with concentric circular slits, such that $\phi(\gamma_0)$ and $\phi(\gamma_1)$ are the outer and inner circles of the annulus, $\phi(\gamma_k)'s, k > 1$, are concentric circular slits. Such kind of conformal mapping ϕ is unique up to a rotation.*

The discrete algorithms for computing conformal mappings for arbitrary 2D domain are explained in details below. The pipeline is as follows: 1) Compute cohomology basis; 2) Compute harmonic 1-form basis; 3) Compute holomorphic 1-form basis; and 4) Compute the slit map.

[1]An injective function is a function that preserves distinctness: it never maps distinct elements of its domain to the same element.

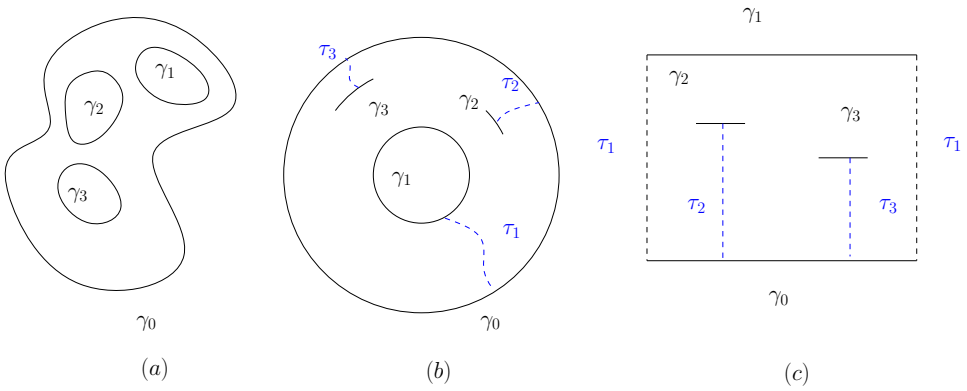


Fig. 5: Multiply connected domain.

Discrete exterior calculus. Similar to all prior work [10], [22], [23], [32], the network is represented as a discrete triangular mesh $M = (V, E, F)$, with the vertex set V, the edge set E and the face set F. An oriented edge is denoted as $[v_i, v_j]$, an oriented faces is $[v_i, v_j, v_k]$. The boundary operator takes the boundary of a simplex:

$$\begin{aligned} \partial[v_i, v_j] &= v_j - v_i, \\ \partial[v_i, v_j, v_k] &= [v_i, v_j] + [v_j, v_k] + [v_k, v_i]. \end{aligned}$$

A 0-form is a function defined on the vertex set $f : V \to \mathbb{R}$. A 1-form is a linear function defined on the edge set $\omega : E \to \mathbb{R}$. A 2-form is a linear function defined on the face set $\tau : E \to \mathbb{R}$. The discrete exterior differential operator is defined as

$$d\omega(\sigma) := \omega(\partial\sigma).$$

If ω is a closed form, then $d\omega = 0$.

Let $[v_i, v_j]$ is an interior edge, with two adjacent faces $[v_i, v_j, v_k]$ and $[v_j, v_i, v_l]$. Then the edge weight is defined as

$$w_{ij} := \cot\theta_{ij}^k + \cot\theta_{ji}^l,$$

where θ_{ij}^k is the corner angle at vertex v_k in the face $[v_i, v_j, v_k]$. If $[v_i, v_j]$ is a boundary edge, adjacent to $[v_i, v_j, v_k]$ only, then

$$w_{ij} := \cot\theta_{ij}^k.$$

The discrete co-differential operator is defined as follows. Let ω be a one-form, then $\delta\omega$ is a 0-form,

$$\delta\omega(v_i) = \sum_{[v_i,v_j]\in E} w_{ij}\omega([v_i, v_j]).$$

If $f : V \to \mathbb{R}$ is a harmonic function, then

$$\Delta f(v_i) = \delta df(v_i) = \sum_{[v_i,v_j]} w_{ij}(f(v_j) - f(v_i)) = 0, \forall v_i \in V.$$

Step 1: Compute cohomology basis. Suppose the boundary components of the mesh are $\partial M = \gamma_0 - \gamma_1 - \gamma_2 \cdots \gamma_n$, where γ_0 is the exterior boundary. Compute the shortest path from γ_k to γ_0, denoted as η_k as shown in figure 7. Slice the mesh M along η_k to get a mesh M_k, the path η_k on M corresponds to

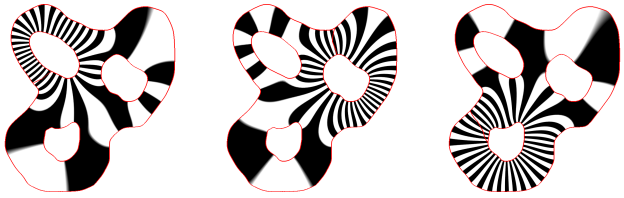


Fig. 7: Exact harmonic 1-forms $\{\omega_4, \omega_5, \omega_6\}$.

two boundary segments η_k^+ and η_k^- on M_k. Define a function $f_k : M_k \to \mathbb{R}$,

$$f_k(v_i) = \begin{cases} +1 & v_i \in \eta_k^+ \\ -1 & v_i \in \eta_k^- \\ 0 & v_i \notin \eta_k^+ \cup \eta_k^- \end{cases}$$

Assume $e \in \eta_k^+ \cup \eta_k^-$, then $df_k(e) = 0$. Therefore, the exact 1-form df_k on M_k in fact is a closed 1-form on the original mesh M. Let $\rho_k := df_k$ on M, then

$$\{\rho_1, \rho_2, \cdots, \rho_n\}$$

form a basis for the cohomology group $H^1(M, \mathbb{R})$.

Step 2: Compute harmonic 1-form basis. Given a closed 1-form ρ_k, we can find a function $g_k : M \to \mathbb{R}$, such that

$$\delta(\rho_k + dg_k)(v_i) = \sum_{[v_i,v_j]} w_{ij}\{\rho_k([v_i, v_j]) + (g(v_j) - g(v_i))\} = 0,$$

for all vertex v_i in M. Then $\omega_k := \rho_k + dg_k$ is a harmonic 1-form. Then $\{\omega_1, \omega_2, \cdots, \omega_n\}$ form the basis for the harmonic 1-form basis.

Similarly, we compute n harmonic functions $f_k : V \to \mathbb{R}$, with Dirichlet boundary condition, such that

$$\begin{cases} \Delta f_k(v_i) = 0 & \forall v_i \\ f_k(v_i) = 1 & v_i \in \eta_k \\ f_k(v_i) = 0 & v_i \in \partial M - \eta_k \end{cases}$$

Then let $\omega_{n+k} := df_k$, then $\{\omega_{n+k}\}, k = 1, 2, \cdots, n$ are exact harmonic 1-forms.

Step 3: Holomorphic 1-form basis. For each harmonic 1-form ω_i, we compute its conjugate harmonic 1-form ${}^*\omega_k$.

$${}^*\omega_i = \sum_{j=1}^{2n} \lambda_{ij}\omega_j$$

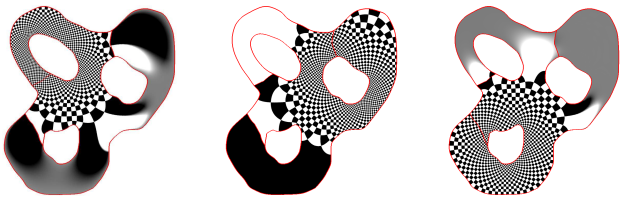


Fig. 8: Holomorphic 1-forms basis $\{\tau_1, \tau_2, \tau_3\}$.

The unknowns $\{\lambda_{ij}\}$ can be computed by solving the following linear equation system:

$$\int_M \omega_i \wedge {}^*\omega_j = \sum_{k=1}^{2n} \lambda_{jk} \int_M \omega_i \wedge \omega_k.$$

The wedge products can be computed as follows. Let $[v_i, v_j, v_k]$ be a triangle face, $e_i = [v_j, v_k]$, $e_j = [v_k, v_i]$ and $e_k = [v_i, v_j]$. Then by direct computation, we get

$$\int_{[v_i,v_j,v_k]} \omega_1 \wedge \omega_2 = \frac{1}{2} \begin{vmatrix} \omega_1(e_i) & \omega_1(e_j) & \omega_1(e_k) \\ \omega_2(e_i) & \omega_2(e_j) & \omega_2(e_k) \\ 1 & 1 & 1 \end{vmatrix}$$

and

$$\int_{[v_i,v_j,v_k]} \omega_1 \wedge {}^*\omega_2 = \tfrac{1}{2}[\omega_1(e_i)\omega_2(e_i)\cot\theta^i_{jk} + \omega_1(e_j)\omega_2(e_j)\cot\theta^j_{ki} + \omega_1(e_k)\omega_2(e_k)\cot\theta^k_{ij}]$$

Let $\tau_k := \omega_k + {}^*\omega_k\sqrt{-1}$, Then $\{\tau_1, \tau_2, \cdots, \tau_n\}$ form the basis for the holomorphic 1-form group. Figure 8 shows the basis for holomorphic 1-forms.

Step 4: Slit conformal mapping. We then search for a special holomorphic 1-form $\tau = \sum_k x_k \tau_k$, such that

$$Img(\int_{\gamma_0} \tau) = \sum_k x_k Img(\int_{\gamma_0} \tau_k) = 2\pi.$$

and

$$\int_{\gamma_i} \tau = \sum_k x_k Img(\int_{\gamma_i} \tau_k) = 0, i = 2, 3, \cdots, n.$$

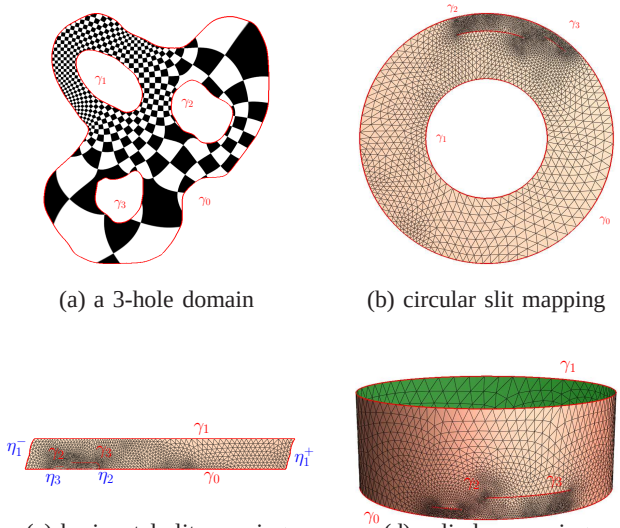
This implies $\int_{\gamma_1} \tau = -2\pi$. Then the mapping is given by

$$\phi(z) = \exp\{\int_{z_0}^{z} \tau\}.$$

where the integration path is arbitrarily chosen.

Now we summarize the computation and communication steps involved in the pipeline.

1) Step 1: Compute cohomology basis. In this step we will find shortest paths connecting the interior holes to the outer boundary. This can be done by a single flooding starting from the nodes at the inner hole boundaries simultaneously.
2) Step 2: Compute harmonic 1-form basis. In this step we compute n harmonic functions, where n is the number of holes. This uses the Dirichlet boundary condition and an iterative gossip-style algorithm, similar to the distributed algorithm used in [15].



(a) a 3-hole domain (b) circular slit mapping

(c) horizontal slit mapping (d) cylinder mapping

Fig. 9: Conformal mapping from the domain to the annulus, γ_0 is mapped to the outer circle, γ_1 is mapped to the inner circle.

3) Step 3: Compute holomorphic 1-form basis. This involves completely local operations. Each node will solve a linear system only on its neighbors.
4) Step 4: Slit conformal mapping. This involves only one round of flooding, starting from the outer boundary inward. The nodes compute their virtual coordinates.

The algorithm solves sparse linear systems. Therefore the holomorphic differential method is more efficient compared to the non-linear curvature flow methods. The algorithm handles domains with two or more holes. For other cases the algorithm is similar.

Simply connected domain. If the input network is a simply connected domain, we select four corner vertices on the boundary $\{v_0, v_1, v_2, v_3\}$ sorted counter-clock-wisely. Then we glue two copies of the network along the boundary segments between v_0, v_1 and the boundary segments between v_2, v_3. The result is a topological annulus. The algorithm above can also handle this case and will map the doubled network to an annulus. By taking the complex logarithm, the original network is mapped to a planar rectangle, such that the four corners are mapped to the corners of the rectangle.

Doubly connected domain. If the input network is a doubly connected domain, the conformal mapping gives a canonical annulus. By taking the complex logarithm, it is flattened to a periodic rectangle. Glue one copy along one boundary, the result domain is a topological torus.

IV. Simulations

A. Dense Curve Discretization

The aperiodic dense curve is identified as a continuous line in the universal covering space. To apply it in sensor networks, the curve needs to be mapped to a discrete path. There are various strategies for curve discretization. In our setting we suppose there is a sensor network G deployed on a continuous sensor domain $\mathcal{R}$. We compute the dense curve on $\mathcal{R}$ and

expand the width of the curve to get a belt region $\mathcal{B}$. The discrete path starts at an arbitrary node $s \in \mathcal{B}$. For a node u on the path, we compute its next hop from the set of neighbors falling inside the belt $\mathcal{B}$: $C(u) = \{u | u \in N(u) \text{ and } u \in \mathcal{R}\}$ based on a closeness measurement. This will generate a discrete path.

B. Comparison with Various Network Covering Approaches

We compare our method with a number of other approaches that generate a linear ordering of the sensor nodes, specifically, the space filling curves, Eulerian cycles and random walks. For space filling curve, we use the Moore curve which covers a square densely. See Figure 10 for an example. Starting from a corner, a space filling curve first visits nodes nearby exhibiting strong locality. It also does not handle network holes and may be disconnected into multiple pieces. An Eulerian cycle gives a cycle on the network nodes in which one node may appear multiple times. We build a minimum spanning tree of the network, duplicate all the edges to generate an Eulerian cycle. Compared to the aperiodic dense curve, Eulerian cycle also has spatial locality. For random walk, the next hop of the path is chosen uniformly randomly from the neighbors of the current node.

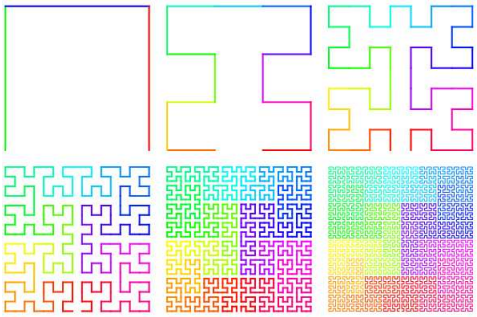


Fig. 10: The Moore curve (source: Wikipedia).

Since Moore curve only exists on a square, in our comparison the networks are deployed on square regions without holes. The sensor nodes are uniformly randomly deployed within the network region, the transmission pattern follows the unit disk graph model(UDG). In the experiment the networks have $5,000$ nodes with average degrees to be 7, we uniformly randomly generated 10 networks to average out the randomness.

Figure 11 shows the network coverage percentage as the paths move forward. The x axis is the length of the path in the number of hops, the y coordinate is the percentage of nodes covered by the path. It is obvious that aperiodic dense curves, Eulerian cycles and Moore curves are much better than random walk in terms of coverage, which is not surprising because these are well-guided curves, while a random walk is aimless. Since Moore curve is designed to cover the unit square case, the coverage grows linearly at a fast pace. However, the problem for Moore curve to be used in practice is that: it needs to choose a resolution before the curve starts. If the resolution is not appropriately selected, Moore curve may miss some nodes when it comes back to the starting point.

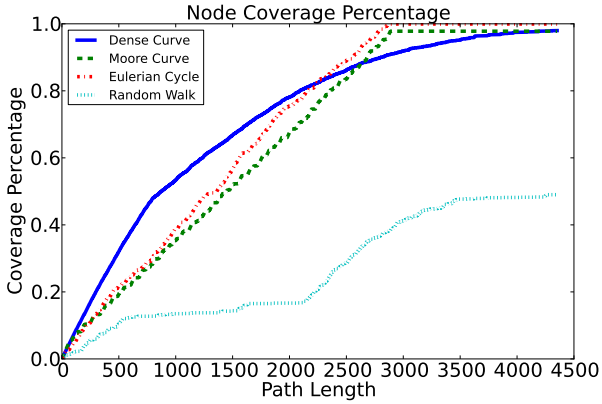


Fig. 11: Comparison on Network Coverage

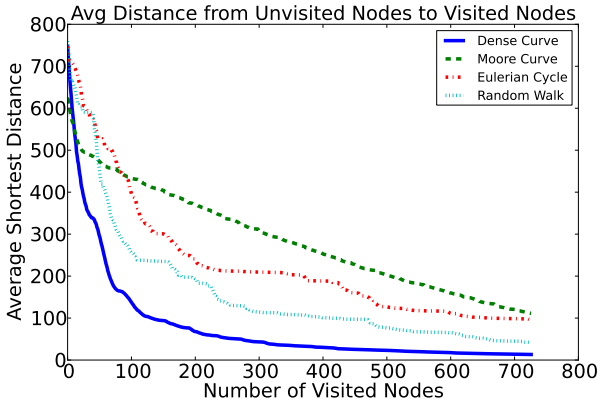


Fig. 12: Comparison on Average Shortest Distance from Unvisited Nodes to Visited Nodes

Any continuation will not discover new nodes, as shown in the later part of the curve in Figure 11. The Eulerian cycle can eventually cover all nodes by definition. Compared to all the other methods, our approach has a clear advantage at the beginning of the path. Our dense curve sets out to explore the entire domain in a coarse manner; network coverage is improved continuously when the path is longer.

Figure 12 shows the average shortest distance from the set of unvisited nodes to the set of visited nodes, this average shortest distance criteria measures the locality property of the paths. If the path visits most of nearby nodes before moving to nodes faraway, the average shortest distance can still remain relatively high even though the path visits more nodes. Compared to other methods, the average shortest distance of the aperiodic dense curve drops sharply, which means that the aperiodic dense curve visits the network in a more global way than other methods.

To conclude, the aperiodic dense curve, Moore curve and Eulerian cycle cover the network much faster than random walk. Compared to the Moore curve and Eulerian cycle, the aperiodic dense curve is able to quickly sample the whole network, which gives a good representation of the network in the early stage.

C. Covering Network with Holes

Sensor networks may have obstacles inside, which lead to holes in the sensor domain. Normal space filling curves like Moore curve would fail under such cases, because those curves only cover the unit square, and would become disconnected pieces. By performing conformal mapping to map the holes to slits, the aperiodic dense curve can be used to cover the whole sensor domain. Figure 13 shows a 2-hole network with its conformal mapping to circular slits and cylinder. Figure 14 shows the aperiodic dense curve on the network.

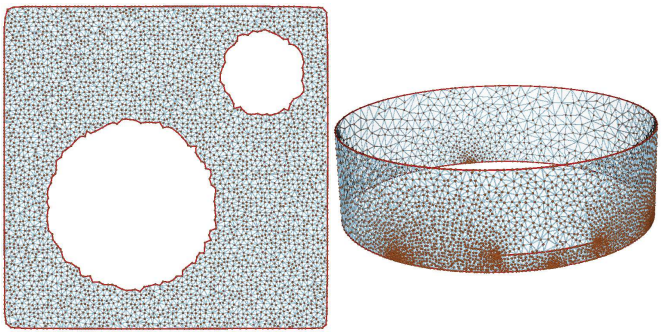


Fig. 13: Conformal mapping for a network with holes.

V. Dense Curve Applications

Multiple paths and data mule coordination. To gather network data, one could use multiple data mules simultaneously to speed up the process. To coordinate and collaborate with each other, the data mules may need to communicate during the data collection, which can be expensive or even infeasible. By using aperiodic dense curve we can reduce such coordination efforts. Each aperiodic dense curve would be able to cover the whole network in a particular pattern, and the visiting pattern is predefined by the slope and starting position. By deliberately assigning slopes and starting positions to multiple aperiodic dense curves, the data mules can collectively cover the whole network such that the overlap between different paths is small. Figure 15(ii) shows two dense curves starting from different boundary nodes O_1, O_2 with different slopes.

Double ruling. Besides the applications for data mule planning and data fusion, we can also make use of the dense curve for in-network storage and retrieval. One scheme for storing sensor data in the network, called double rulings, stores the sensor data along a storage curve and retrieves data along a retrieval curve. Data is retrieved when the retrieval curve intersects the storage curve. Previous double rulings schemes are only designed for networks of a regular shape, e.g., the horizontal/vertical lines [17], [27], [31], or proper circles (great circles through a stereographic mapping) [24]. When the network has holes, these curve are fragmented by the presence of holes. Alternative repairing schemes must be used to reconnect them.

A pair of non-parallel aperiodic dense curves give two trajectories on the network that intersect with each other. Those two trajectories form a lattice on the network, which is very

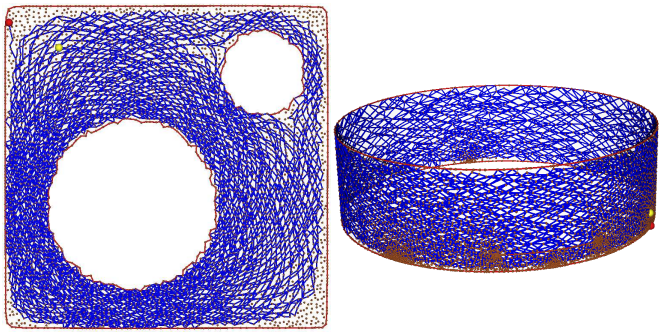
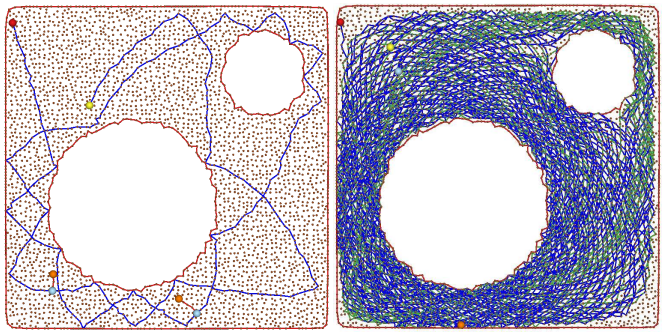


Fig. 14: A dense curve on the network in Fig. 13.

suitable for double ruling. In particular, for storage curves, we simply use the line $\ell : y = kx$ with slope k. For the retrieval curves, we use the line $\ell^* : y = x/k$, i.e., the line perpendicular to ℓ in the universal covering space. Figure 15(i) shows the double ruling result from two consumers A, B to get the data from the producer trajectory.



(i) double ruling. (ii) multiple paths.

Fig. 15: Dense curve applications.

VI. Conclusion

In this paper we propose the computation of a dense curve for any planar domain. When walking on the curve the trajectory will gradually and densely cover the domain of interest. This linearization of a 2D network can be useful for any scenarios that require a logical serial order. We hope to develop applications of this idea for data mule planning in our future work.

Acknowledgment

The authors would like to acknowledge the support from NSF through DMS-1221339 and CNS-1016829.

References

[1] L. V. Ahlfors. *Complex Analysis*. McGraw-Hill, New York, 1966.

[2] S. Arora. Polynomial time approximation schemes for euclidean traveling salesman and other geometric problems. *J. ACM*, 45:753–782, September 1998.

[3] J. M. Bahi, A. Makhoul, and A. Mostefaoui. Localization and coverage for high density sensor networks. *Comput. Commun.*, 31(4):770–781, 2008.

[4] T. Bektas. The multiple traveling salesman problem: an overview of formulations and solution procedures. *Omega*, 34:209–219, June 2006.

[5] R. Blum, S. Kassam, and H. Poor. Distributed detection with multiple sensors ii. advanced topics. *Proceedings of the IEEE*, 85(1):64 –79, jan 1997.

[6] N. Christofides. Worst-case analysis of a new heuristic for the traveling salesman problem. In J. F. Traub, editor, *Sympos. on New Directions and Recent Results in Algorithms and Complexity*, page 441, New York, NY, 1976. Academic Press.

[7] M. R. Garey and D. S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1990.

[8] X. Gu and S. Yau. *Computational conformal geometry*. Advanced lectures in mathematics. International Press, 2008.

[9] D. Jea, A. A. Somasundara, and M. B. Srivastava. Multiple controlled mobile elements (data mules) for data collection in sensor networks. In *IEEE International Conference on Distributed Computing in Sensor System (DCOSS)*, pages 244–257, 2005.

[10] R. Jiang, X. Ban, M. Goswami, W. Z. adn Jie Gao, and X. D. Gu. Exploration of path space using sensor network geometry. In *Proc. of the 10th International Symposium on Information Processing in Sensor Networks (IPSN'11)*, pages 49–60, April 2011.

[11] M. Kamat, A. Ismail, and S. Olariu. Modified hilbert space-filling curve for ellipsoidal coverage in wireless ad hoc sensor networks. In *Signal Processing and Communications, 2007. ICSPC 2007. IEEE International Conference on*, pages 1407 –1410, nov. 2007.

[12] A. Kansal, M. Rahimi, W. J. Kaiser, M. B. Srivastava, G. J. Pottie, and D. Estrin. Controlled mobility for sustainable wireless networks. In *IEEE Sensor and Ad Hoc Communications and Networks (SECON'04)*, 2004.

[13] D. Koutsonikolas, S. Das, and Y. Hu. Path planning of mobile landmarks for localization inwireless sensor networks. In *Distributed Computing Systems Workshops, 2006. ICDCS Workshops 2006. 26th IEEE International Conference on*, page 86, july 2006.

[14] J. K. Lawder and P. J. H. King. Using space-filling curves for multi-dimensional indexing. In *Proceedings of the 17th British National Conferenc on Databases: Advances in Databases*, BNCOD 17, pages 20–35, London, UK, 2000. Springer-Verlag.

[15] H. Lin, M. Lu, N. Milosavljević, J. Gao, and L. Guibas. Composable information gradients in wireless sensor networks. In *Proc. of the International Conference on Information Processing in Sensor Networks (IPSN'08)*, pages 121–132, April 2008.

[16] W. Lindner and S. Madden. Data management issues in periodically disconnected sensor networks. In *Proceedings of Workshop on Sensor Networks at Informatik*, 2004.

[17] X. Liu, Q. Huang, and Y. Zhang. Combs, needles, haystacks: balancing push and pull for discovery in large-scale sensor networks. In *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 122–133, 2004.

[18] L. Lovasz. Random walks on graphs: A survey. *Bolyai Soc. Math. Stud.*, 2, 1996.

[19] J. S. B. Mitchell. Guillotine subdivisions approximate polygonal subdivisions: A simple polynomial-time approximation scheme for geometric tsp, k-mst, and related problems. *SIAM J. Comput.*, 28:1298–1309, March 1999.

[20] S. Patil and S. R. Das. Serial data fusion using space-filling curves in wireless sensor networks. In *Proceedings of IEEE International Conference on Sensor and Ad Hoc Communications and Networks (SECON'04)*, pages 182–190, 2004.

[21] H. Sagan. *Space-Filling Curves*. Springer-Verlag, New York, 1994.

[22] R. Sarkar, X. Yin, J. Gao, F. Luo, and X. D. Gu. Greedy routing with guaranteed delivery using ricci flows. In *Proc. of the 8th International Symposium on Information Processing in Sensor Networks (IPSN'09)*, pages 97–108, April 2009.

[23] R. Sarkar, W. Zeng, J. Gao, and X. D. Gu. Covering space for in-network sensor data storage. In *Proc. of the 9th International Symposium on Information Processing in Sensor Networks (IPSN'10)*, pages 232–243, April 2010.

[24] R. Sarkar, X. Zhu, and J. Gao. Double rulings for information brokerage in sensor networks. In *Proc. of the ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, pages 286–297, September 2006.

[25] R. Shah, S. Roy, S. Jain, and W. Brunette. Data MULEs: Modeling a three-tier architecture for sparse sensor networks. In *IEEE SNPA Workshop*, May 2003.

[26] E. Stein and R. Shakarchi. *Fourier Analysis: An Introduction*. Number v. 10 in Princeton Lectures in Analysis, Volume 1. Princeton University Press, 2009.

[27] I. Stojmenovic. A routing strategy and quorum based location update scheme for ad hoc wireless networks. Technical Report TR-99-09, SITE, University of Ottawa, September, 1999.

[28] Z. Vincze and R. Vida. Multi-hop wireless sensor networks with mobile sink. In *CoNEXT'05: Proceedings of the 2005 ACM conference on Emerging network experiment and technology*, pages 302–303, New York, NY, USA, 2005. ACM Press.

[29] R. Viswanathan and P. Varshney. Distributed detection with multiple sensors i. fundamentals. *Proceedings of the IEEE*, 85(1):54 –63, jan 1997.

[30] H. Weyl. Über die Gleichverteilung von Zahlen mod Eins. *Math. Ann.*, 77:313–352, 1916.

[31] F. Ye, H. Luo, J. Cheng, S. Lu, and L. Zhang. A two-tier data dissemination model for large-scale wireless sensor networks. In *MobiCom '02: Proceedings of the 8th annual international conference on Mobile computing and networking*, pages 148–159, 2002.

[32] W. Zeng, R. Sarkar, F. Luo, X. D. Gu, and J. Gao. Resilient routing for sensor networks using hyperbolic embedding of universal covering space. In *Proc. of the 29th Annual IEEE Conference on Computer Communications (INFOCOM'10)*, pages 1694–1702, March 2010.