Stony Brook University

Computing Teichmüller Maps and Applications of Conformal Geometry to Sensor Networks

A Dissertation Presented

by

Mayank Goswami

to

The Graduate School

in Partial Fulfillment of the

Requirements

for the Degree of

Doctor of Philosophy

in

Applied Mathematics and Statistics

Stony Brook University

August 2013

Stony Brook University
The Graduate School

Mayank Goswami

We, the dissertation committee for the above candidate for the
Doctor of Philosophy degree, hereby recommend
acceptance of this dissertation.

Joseph S.B. Mitchell – Dissertation Advisor
Professor, Department of Applied Mathematics and Statistics

Michael A. Bender – Chairperson of Defense
Associate Professor, Department of Computer Science

Xianfeng Gu – Dissertation Co-Advisor
Associate Professor, Department of Computer Science

Xiangmin Jiao
Associate Professor, Department of Applied Mathematics and Statistics

Jie Gao–External Member
Associate Professor, Department of Computer Science

This dissertation is accepted by the Graduate School

Charles S. Taber
Interim Dean of the Graduate School

Abstract of the Dissertation

Computing Teichmüller Maps and Applications of Conformal Geometry to Sensor Networks

by

Mayank Goswami

Doctor of Philosophy

in

Applied Mathematics and Statistics

Stony Brook University

2013

By the Riemann-mapping theorem, one can always bijectively map an n-polygon P to another n-polygon Q conformally; in fact, any two simply connected domains can be conformally mapped to each other. However, this mapping need not necessarily map the vertices of P to the vertices of Q. In this case, one wants to find the "best" mapping between these polygons, i.e., one that minimizes the maximum angle distortion over all points in the base polygon, and maps the vertices of P to the vertices of Q. Such maps are called Teichmüller maps or extremal quasiconformal maps.

In the first part of this work we present a variational approach in the continuous setting; we use the Hamilton-Krushkal, Reich-Strebel, necessary and sufficient condition for optimality to get an iterative method that successively improves the maximum distortion of the starting map. This procedure is proven to converge to the unique extremal quasiconformal or Teichmüller map. We then provide a numerical method for computing such maps by using this continuous construction and employing methods from convex optimization. Our method works not only for the polygon problem mentioned above, but also for the general extremal problem on punctured spheres.

In the second part of this work we present some of the ways in which we use conformal geometry to help resolve some real-world problems encountered in the field of Wireless Sensor Networks (WSNs). We 1) use Möbius transformations to

navigate through the path-space of a sensor network, 2) use harmonic measure to analyze the traffic pattern resulting from random routing, and 3) use the relation between Teichmüller theory and billiards to get dense curves in WSNs which are useful for many applications like serial data fusion, motion planning for data mules and node indexing.

Contents

List of Tables

List of Figures

Acknowledgements

Most graduate students would agree that six years of graduate study is a phase of life for which the Dickensian phrase "It was the best of times, it was the worst of times..." seems pretty apt. However, I had the privilege of meeting many interesting and remarkable people during my studies at Stony Brook, which rendered the number of "worst of" times negligible. I would like to take this opportunity to thank all of them.

First, I would like to thank Professor Xianfeng (David) Gu for being what I consider a perfect advisor. This dissertation would not be possible without his constant support and his never-ending belief in my abilities. If not for him, I would have not been exposed to the beautiful theory of Teichmüller maps, which also led me to explore more about the connections between Teichmüller theory and billiards. I would also like to thank Professor Jie Gao for providing me an opportunity to work with her; she is one of the most efficient people I have worked with and is a true professional, around whom everyone's productivity increases exponentially. I would also like to thank NSF for providing the grants to David and Jie from which I was funded.

Many thanks to Professor Joseph S.B. Mitchell, Professor Michael A. Bender and Professor Xiangmin (Jim) Jiao for always being present whenever I needed their help. Their understanding of the subject matter is deep and firm, and graduate students like me need examples of professors like them for guiding us in the early stages of academia. Discussions with Joe and Michael have opened up further avenues of research for me, and I would derive great satisfaction if I could help them solve some of the many open problems we have discussed.

For the Teichmüller maps problem (chapter 2 of this thesis), I would like to thank Professor Irwin Kra, Professor Frederick Gardiner, Professor Christopher

Bishop and Vamsi Pingali. The results obtained would not be possible were it not for the numerous discussions I had with them.

I would also like to thank my co-authors Xiaomeng Ban, Wei Zeng, Rui Shi and Ruirui Jiang, without whose help Chapters 3,4 and 5 of my thesis would not exist.

I have collaborated on projects which have not been mentioned in this thesis. Working with Džejla, Pablo and Michael on the sorting problem was a pleasure; Džejla's dedication to her work, Michael's multifaceted skills, and Pablo's precision made the entire experience very enjoyable. I would also like to thank Prof. Svetlozar Rachev and Prof. Pradeep Dubey for making me familiar with the fields of finance and game theory, respectively. I hope to work more with them in the future.

Special thanks to all my officemates—Kan Huang, Yinghua Li, Irina Kostitsyna and Shane D'Mello from the math tower, and Džejla Medjedovic, Roozbeh Ebrahimi, Golnaz Ghasemiesfeh, Pablo Montes Arango and Akshay Patil from the computer science building, for providing a lively atmosphere during work, and for all the problem discussions. I hope we can collaborate on multiple projects in future.

I would like to express my gratitude to all my friends from Stony Brook—Vatsa, Abhinav, Biligbaatar, Bruno, Ranaji, Shishir, Kaya, Tamara, Maryam, Budassi, Lala, Garnet, Rao, Rajul, Vaibhav and Chakku. Their company made my stay in Stony Brook pleasurable, making the six years seem shorter.

Last but not the least, I would like to thank my parents Lokesh and Manjula, and my brother Aman for being selfless and always trusting. It would be impossible to finish this work without their never-ending love and understanding.

Chapter 1

Introduction

1.1 Conformal geometry

Conformal geometry is one of the oldest fields in mathematics, abound with elegant theories developed by some of the greatest mathematicians. It started with the study of conformal (angle-preserving) transformations of the complex plane, and has developed into a field which lends and borrows ideas, to and from surprisingly varying fields, such as differential geometry, complex analysis, algebraic topology and dynamics.

In the last half a century, conformal geometry has found various applications of practical significance. It is widely used in computer graphics, computational geometry, sensor networks, optimization, and also in many theories in physics, such as super string theory and the theory of modular spaces.

In pure mathematics, often it happens that proving existence of quantities which optimize a certain functional suffices for achieving the immediate goals. However, many such quantities are later found to be very important practically, and one must then turn the existence proofs into constructive ones, and then devise numerical methods for computing them. In the first half of this work, we will be concerned with such a problem—computing Teichmüller maps.

Teichmüller theory is the study of quasiconformal maps, which are maps that stretch angles but by a bounded amount. It is an active area of research, and while some may think it an an obscure branch of complex analysis, its deep connections

to topology are evident from the fact that recently it has been used by Lipman Bers to give a simpler proof of Thurston's classification theorem for surface homeomorphisms.

Teichmüller maps, also called extremal quasiconformal maps are homeomorphisms between Riemann surfaces (Definition 2) which are the closest to being conformal. While their existence was proven by O. Teichmüller in 1939 ([117] and [118]), constructing them has been a long-standing open problem. The first half of this work will address the issue of computing these maps.

In the second half of this work, we will describe some applications of conformal geometry to the field of wireless sensor networks, thus reinforcing our claim that conformal geometry has and will continue to play an important part in fields of practical significance. We will encounter several problems such as analyzing traffic from random routing, devising multiple source-destination paths in a sensor network, and will play "billiards" on an appropriate table to obtain dense curves in a sensor network.

We begin by outlining some basic concepts in conformal geometry which will be used in all the subsequent chapters. All of the material in this chapter is classical and can be found in books on complex analysis and Riemann surfaces, such as [3, 29, 30, 35].

1.1.1 Holomorphic functions and conformal maps

Let U be an open set contained in the complex plane $\mathbb{C}$. A function $f : U \to \mathbb{C}$ is said to be holomorphic at a point $z_0 \in U$ if the complex derivative

$$f^{'}(z_0) = \lim_{h \to 0} \frac{f(z_0 + h) - f(z)}{h}$$

exists. Note that h is a complex number, and the limit is the same for all complex sequences converging to 0. If f is holomorphic at every point $z_0 \in U$, we say f is holomorphic on U.

If one writes $f = u + iv$, where u and v are real functions, one finds that holomorphic functions satisfy the Cauchy-Riemann equations

$$\frac{\partial u}{\partial x} = \frac{\partial v}{\partial y} \text{ and } \frac{\partial u}{\partial y} = -\frac{\partial v}{\partial x}.$$

Conversely, if u and v have continuous first partial derivatives and satisfy the Cauchy-Riemann equations, then f is holomorphic. Defining the operator

$$\frac{\partial}{\partial \bar{z}} = \frac{\partial}{\partial x} + i\frac{\partial}{\partial y},$$

one can summarize the above equations as $\frac{\partial f}{\partial \bar{z}} = 0$. Examples of holomorphic functions include all polynomials in z and the exponential function e^z. [97] is an excellent book for an introduction to properties of holomorphic functions.

If f is holomorphic and in addition the derivative is non-zero, f is called conformal. Conformal maps preserve the Euclidean angle between tangent vectors at a point. Sizes, however, may be distorted by conformal maps.

Examples of conformal maps—Möbius transformations: Möbius transformations are rational functions defined on the complex plane $\mathbb{C}$. The general form of a Möbius transformation is

$$f(z) = \frac{az + b}{cz + d}.$$

Here $a, b, c, d \in \mathbb{C}$ and satisfy $ad - bc \neq 0$. If $c \neq 0$ we can extend this mapping to the Riemann Sphere (or the extended complex plane, i.e., with a point of infinity) $\widehat{\mathbb{C}} = \mathbb{C} \cup \{\infty\}$ by specifying $f(-d/c) = \infty$ and $f(\infty) = a/c$. In the case when $c = 0$, we specify $f(\infty) = \infty$.

Here are the important properties of Möbius transformations:

1. Möbius transformations are all the bijective holomorphic mappings from $\widehat{\mathbb{C}}$ to itself. This also implies that they are conformal.

2. Möbius transformations carry circles and lines (which can be regarded as circles passing through ∞) to circles and lines. Thus, given a circular domain, any Möbius transformation will map it to another circular domain.

3. To every Möbius transformation one can associate a matrix

$$M_f = \begin{bmatrix} a & b \\ c & d \end{bmatrix}.$$

Any other matrix which is a (nonzero) scalar multiple of this matrix represents the same Möbius transformation. Composition of two Möbius transformations is equivalent to matrix multiplication, i.e., $M_{f \circ g} = M_f \cdot M_g$.

4. Given distinct $z_1, z_2, z_3 \in \widehat{\mathbb{C}}$ and distinct $w_1, w_2, w_3 \in \widehat{\mathbb{C}}$, there is a *unique* Möbius transformation f satisfying $f(z_i) = w_i$, $i = 1, 2, 3$. In other words, if one considers the unique circles C_1 and C_2, defined by z_1, z_2, z_3 and w_1, w_2, w_3 respectively, then the transformation f maps the circle C_1 to C_2 and is unique. Determining f explicitly is equivalent to finding determinants of four 3×3 matrices.

Let $\mathbb{D}$ denote the unit disk centered at the origin in $\mathbb{C}$. The set of all conformal maps from $\mathbb{D}$ to itself is a subgroup of the group of Möbius transformations, where $a, b, c, d \in \mathbb{R}$. It is well-known that any such map is of the form

$$f(z) = e^{i\theta} \frac{z - z_0}{1 - \bar{z_0} z},$$

for some $\theta \in (0, 2\pi)$ and some $z_0 \in \mathbb{D}$.

Riemann Mapping: One of the most important results in complex analysis, the Riemann mapping theorem, will be used extensively in this work.

Theorem 1 (Riemann Mapping)**.** *Let Ω be a simply connected domain in the complex plane $\mathbb{C}$, not equal to the entire complex plane. Then there exists a biholomorphic map $f : \mathbb{D} \longrightarrow \Omega$. Further, f is unique up to composition by a Möbius transformation.*

That f is biholomorphic implies it is conformal. One can therefore state as a corollary that any two simply connected domains in $\mathbb{C}$ (not equal to $\mathbb{C}$) can be mapped conformally and bijectively to each other.

1.1.2 Riemann surfaces

The concept of a Riemann surface originated as a tool to deal with the problem of multi-valued functions. Generally, analytic continuation of a starting function element along different paths leads to different values (called branches) of the function. Riemann resolved the problem by extending the domain of the function to many copies of the complex plane; in this way the function becomes analytic on this extended covering surface.

Later it was observed that a Riemann surface is nothing but patches from the complex plane glued together in a special way, so as to get a two-manifold with a special structure. This is the widely accepted definition of a Riemann surface today, which we make precise next.

Let M be a two dimensional real manifold. A complex chart on M is a homeomorphism ϕ from an open subset $A \subset R$ to an open subset $B \subset \mathbb{C}$. Let $\phi_1 : A_1 \to B_1$ and $\phi_2 : A_2 \to B_2$ be two complex charts. ϕ_1 and ϕ_2 are said to be compatible if the map

$$\phi_2 \circ {\phi_1}^{-1} : \phi_1(A_1 \cap A_2) \to \phi_2(A_1 \cap A_2)$$

is biholomorphic.

A complex atlas on M is a system of charts which cover M, and in which any two charts are compatible. Two complex atlases are regarded equivalent if all charts in the union of the atlases are pairwise compatible.

Definition 2 (Riemann surface)**.** *A Riemann surface R is a pair (M, σ), where M is a connected two-manifold and σ is an equivalence class of complex atlases on M.*

Examples of Riemann surface include the complex plane, domains in the complex plane, the Riemann sphere $\hat{\mathbb{C}}$ and all Riemannian manifolds (oriented two-manifolds with a Riemannian metric).

1.2 Quasiconformal geometry

Given two Riemann surfaces M and N, a map $f : M \to N$ is *conformal* if its restriction on any local conformal parameters is *holomorphic*. Geometrically, a conformal map preserves angles, and transforms infinitesimal circles to infinitesimal circles, as shown in Figure 1 frame (a),(b) and (c).

1.2.1 Quasiconformal maps

A generalization of conformal maps are *quasiconformal* maps, which are orientation preserving homeomorphisms between Riemann surfaces with bounded conformality distortion, in the sense that their first order approximations takes small circles to small ellipses of bounded eccentricity, as shown in Fig.1 frame (d) and (e). Mathematically, $f\colon \mathbb{C} \to \mathbb{C}$ is quasiconformal provided that it satisfies the Beltrami equation:

$$f_{\bar{z}} = \mu(z) f_z. \tag{1}$$

for some complex-valued function μ satisfying $||\mu||_\infty < 1$. μ is called the ***Beltrami coefficient***, and is a measure of the non-conformality of f. In particular, the map f is conformal around a small neighborhood of p when $\mu(p) = 0$. As shown in Figure 2, the orientation of the ellipse is double the argument of μ. The *dilatation* of f is defined as the ratio between the major axis and the minor axis of the infinitesimal ellipse. The maximal dilatation of f is given either by:

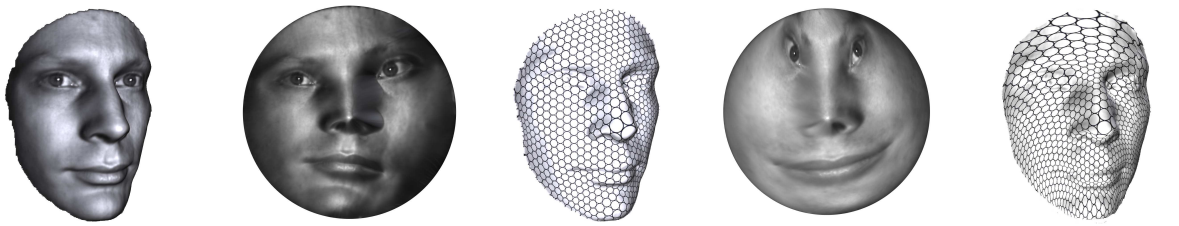
$$k_f = ||\mu_f||_\infty, \tag{2}$$

or by

$$K(f) = \frac{1 + ||\mu||_\infty}{1 - ||\mu||_\infty}. \tag{3}$$

A homeomorphism with dilatation less than or equal to K is called a *K-quasiconformal mapping*.

The inverse problem of obtaining a quasiconformal map from a Beltrami coefficient is addressed by the mapping theorem (see Theorem 8 in Chapter 1).



(a)Original surface (b) Conformal mapping (c) Conformal mapping (d) Qc mapping (e) Qc mapping

Figure 1: Conformal and quasiconformal mappings from a human face surface to the planar disk.

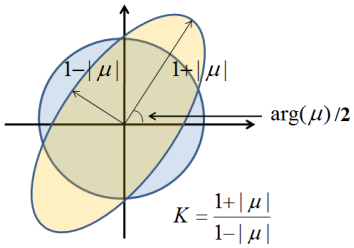


Figure 2: Beltrami coefficient.

1.3 Teichmüller theory

After defining a Riemann surface as in Definition 2, one can talk about conformal and quasiconformal maps between Riemann surfaces. Thus diffeomorphisms between closed, compact, genus g Riemann surfaces are quasiconformal homeomorphisms.

Two Riemann surfaces R and S are conformally equivalent if they can be mapped biholomorphically onto each other. In general, given two surfaces of the same topology, or even two diffeomorphic surfaces, there may not exist a biholomorphic map between them.

Let R be a Riemann surface, and consider all Riemann surfaces in the same quasiconformal class as R, i.e., surfaces that can be mapped to R via a quasiconformal homeomorphism. If R is closed and compact, this is just the space of Riemann surfaces with the same topology as R. Quotienting this space by the equivalence

relation of biholomorphism gives us the moduli space of R, $\mathcal{M}(R)$. For the simplest of Riemann surfaces too, $\mathcal{M}(R)$ is a finite dimensional complex variety and not a manifold.

1.3.1 Teichmüller space

The study of moduli spaces has attracted the interest of mathematicians for decades, and this study can be made simpler by introducing a "cover" of the moduli space—the Teichmüller space.

Consider a fixed Riemann surface R, and two quasiconformal maps f_1 and f_2 from R to Riemann surfaces R_1 and R_2, respectively. f_1 and f_2 will be considered equivalent if there is a biholomorphism ϕ from R_1 to R_2 such that $\phi \circ f_1$ is homotopic to f_2 via a homotopy h_t consisting of quasiconformal maps. The set of equivalence classes so formed, denoted as $\mathcal{T}(R)$, is called the Teichmüller space of R. One can thus view it as the set of equivalence classes of "marked" Riemann surfaces under the biholomorphism relation, where a "marking" refers to a choice of generators of the first fundamental group on a Riemann surface. The Teichmüller equivalence class of f, denoted as $[f]$, is the set of all quasiconformal maps equivalent to f.

1.3.2 Extremal quasiconformal maps

Given a quasiconformal map f between Riemann surfaces R and S, one can define its dilatation $K_z(f)$ at a point z (as in Section 1.2.1) by

$$K_z(f) = \frac{1 + |\mu_f(z)|}{1 - |\mu_f(z)|},$$

since this definition is independent of the chart z lies in. One then defines $K(f)$ to be the essential supremum of $K_z(f)$ over all $z \in R$. $K(f)$ is called the maximal dilatation of f, and describes the deviation of f from being conformal.

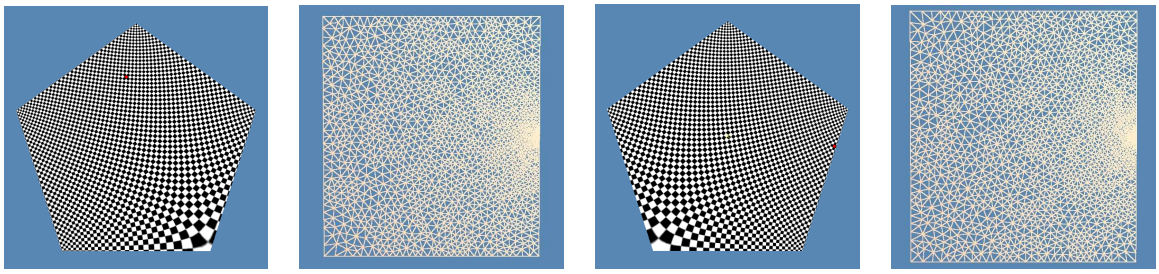
Consider the Teichmüller equivalence class $[f]$ of a given quasiconformal map f. As a result of properties of quasiconformal maps, there exists $f^* \in [f]$ such that $K(f^*)$ is minimum among all quasiconformal maps in this class. This f^*, which in a sense is the "best" way to map R to S—every other quasiconformal map must stretch angles at some point by more than $K(f^*)$, is called an extremal

quasiconformal map.

Teichmüller's metric: Given two equivalence classes f and g (or equivalently, two points in $\mathcal{T}(R)$), the distance between them is defined as $\frac{1}{2}\log K^*$, where K^* is the minimal dilatation of a quasiconformal map in the same class as $f \circ g^{-1}$. This metric on the Teichmüller space is called Teichmüller's metric.

If R is compact and of genus $g \geq 2$, Teichmüller proved that $\mathcal{T}(R)$ is a complete metric space of real dimension $6g - 6$. It turns out that in the cases relevant for applications, namely Riemann surfaces of finite analytic type, the extremal quasiconformal maps have a special characterization, which is closely related to the space of holomorphic quadratic differentials on R. This space is also used to put a complex structure on $\mathcal{T}(R)$ via the *Bers embedding*, which can be found in [35].

1.3.3 Quadratic differentials



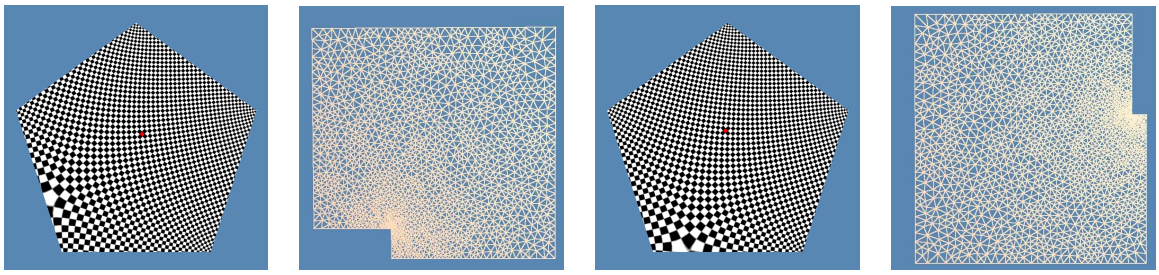
(a)ϕ_1 (b)R_1 (c) ϕ_2 (d) R_2

Figure 3: Holomorphic quadratic differential bases on a pentagon.

Definition 3. *[Holomorphic quadratic differential] A holomorphic quadratic differential on a Riemann surface R is an assignment of a function $\phi_i(z_i)$ on each chart z_i such that if z_j is another local coordinate, then $\phi_i(z_i) = \phi_j(z_j)(\frac{dz_j}{dz_i})^2$.*

We will denote the space of such differentials on R as $A(R)$. By the Riemann-Roch theorem, the complex dimension of this vector space for a genus g closed compact surface with n punctures is $3g - 3 + n$.

The Riemann surfaces of primary importance to us are the punctured Riemann sphere and the unit disk. For the unit disk, there is only one chart z, and therefore



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

Figure 4: Holomorphic quadratic differentials on a pentagon. (a) and (b) show $[0.2(\phi_1')^2 + 0.8(\phi_2')^2]dz^2$, (c) and (d) show $[-0.2(\phi_1')^2 + 1.2(\phi_2')^2]dz^2$.

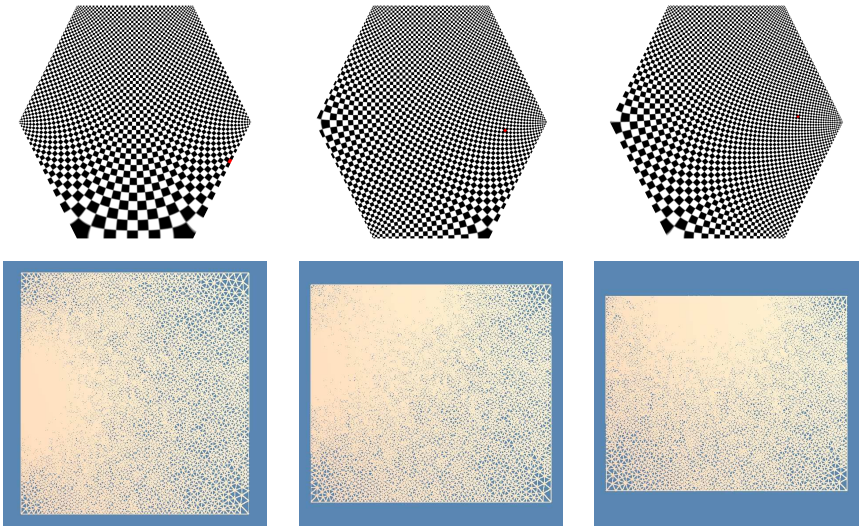


Figure 5: Holomorphic quadratic differential bases on a hexagon.

any function holomorphic in the interior of the disk can be viewed as a quadratic differential (the transition condition is vacuous). For $R = \hat{\mathbb{C}} \setminus \{0, 1, \infty, z_1, ...z_{n-3}\}$ (the Riemann sphere with n punctures),

$$\phi_i(z) = \frac{z_k(z_k - 1)}{z(z-1)(z - z_k)}, \quad 1 \leq k \leq n-3, \tag{4}$$

form a basis of $(n-3)$ dimensional complex vector space $A(R)$.

Another vector space of importance to us is the space of polygon differentials. Let P be a polygon in the plane, normalized so that 0, 1 and ∞ are three vertices

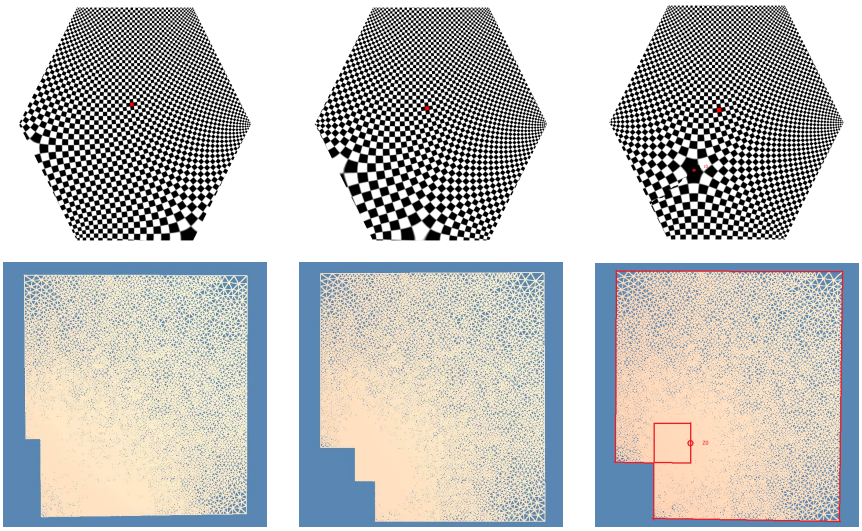


Figure 6: Holomorphic quadratic differentials on a hexagon.

of P. Suppose $\phi_k : P \to D_k$ is the conformal mapping, where D_k is a planar rectangle, such that ϕ_k maps $\{0, 1, \infty, z_k\}$ to the four corners of the rectangle D_k. Then

$$\{(\phi_1')^2 dz^2, (\phi_2')^2 dz^2, \cdots, (\phi_{n-3}')^2 dz^2\}$$

form the bases of $A(R)$. As shown in Figure 3, the Riemann surface R is a pentagon with vertices $\{z_1, z_2, z_3, z_4, z_5\}$, ϕ_1 maps R to planar rectangles R_1, such that $\{z_1, z_2, z_3, z_4\}$ are mapped to four corners. The checkerboard texture on R_1 is pulled back to R and shown in (a). Similarly, ϕ_2 maps $\{z_1, z_2, z_3, z_5\}$ to a rectangle R_2. Then $\{(\phi_1')^2 dz^2, (\phi_2')^2 dz^2\}$ form the bases of all holomorphic quadratic differentials on the pentagon. Figure 4 shows the linear combinations of these bases. Figures 5 and 6 show the bases and certain linear combinations of the bases on a hexagon, respectively.

An excellent book for studying Quadratic differentials in further detail is [113].

1.3.4 Teichmüller maps and relation to holomorphic quadratic differentials

When $\mathcal{T}(R)$ is finite dimensional, one can show that every equivalence class has an extremal quasiconformal map f whose Beltrami coefficient has the form

$$\mu_f = k\frac{|\phi|}{\phi},$$

where ϕ is a holomorphic quadratic differential satisfying $\int\int_R |\phi| = 1$, and $0 \leq k < 1$. Such Beltrami coefficient are said to be in Teichmüller form, and the solutions to their Beltrami equations are called *Teichmüller maps*.

In the sequel we only deal with Riemann surfaces R for which $\mathcal{T}(R)$ is finite dimensional. This includes all Riemann surfaces which are either compact or a compact Riemann surface minus a finite set of points.

1.4 Problems considered

In this section we describe briefly and informally the problems that are considered in this work. A formal description of the problems and proposed solutions constitutes the remaining four chapters.

1.4.1 Computing Teichmüller maps

Extremal quasiconformal maps are objects which are important to both mathematicians and computer scientists due to the following reasons:

1. The study of moduli space becomes easier by introducing the Teichmüller space. Some of the long-standing open problems in conformal geometry and geometric function theory can be rephrased in terms of extremal quasiconformal maps.

2. All surfaces (two-manifolds) we see in daily life can be regarded as Riemann surfaces, since they are Riemannian manifolds. The fields of computer vision and computer graphics aim to register such surfaces and provide a one-to-one correspondence between them. The correspondence is usually required to

have some "nice" properties; for starters, it is usually demanded to be bijective (homeomorphism) and sometimes, conformal. When it is impossible for the domain to be mapped to the target conformally, extremal quasiconformal maps come into play. Further, they are very flexible, and allow for boundary values to be prescribed.

Our general problem of interest is computing Teichmüller maps between closed, compact Riemann surfaces. With the advancements in $3D$ technology, one can get very accurate scans of surfaces in day-to-day life. These surfaces have a natural metric on them—the induced metric from $\mathbb{R}^3$. Using this metric, they can be regarded as Riemann surfaces.

Given a pair of surfaces, one can then either choose a marking on each surface (see Section 1.3.1), or choose a starting quasiconformal homeomorphism from one surface to the other. The objective is to compute the Teichmüller map which sends the marking on the first surface to a marking homotopic to the marking on the second surface, or a Teichmüller map homotopic to the starting quasiconformal homeomorphism, respectively. Surfaces are represented as triangulated meshes, and by "computing" the Teichmüller map we ask to get the image of a dense sample of points on the domain under the Teichmüller map.

In this thesis, we will relax our Riemann surfaces to be simple planar domains.

Computing Teichmüller maps between planar domains Let P and Q be two n-polygons[1] in the plane. The Riemann mapping theorem (see Theorem 1) implies that P can be mapped to Q conformally. This map, which is a-priori defined on the interior of the polygon, can be extended continuously to the boundary. Generally, this boundary extension will not map the vertices of one polygon to the vertices of the other. If the vertices of the two polygons are ordered and one only considers homeomorphisms which map the vertices of P to those of Q in an order preserving way, in the generic case some angle distortion is inevitable.

Let $\{v_i\}_{i=1}^n$ and $\{v_i'\}_{i=1}^n$ be an ordering of the vertices of P and Q, respectively.

[1] We allow for ∞ to be a vertex of the polygon.

The problem we study can now be stated simply as find

$$\arg\min_{f} ||\mu_f||_\infty,$$

where f is a quasiconformal homeomorphism from P to Q such that $f(v_i) = v_i'$. $k_f = ||\mu_f||_\infty$ is also referred to as the dilatation of f.

From classical theory, f is known to be unique. While several characterizations of f exist, there does not exist an explicit construction of the extremal map, even in the classical continuous setting.

Consider the following problem next:

Given $\{z_1, ...z_{n-3}, z_{n-2} = 0, z_{n-1} = 1, z_n = \infty\}$, $z_i \in \mathbb{C}$, $\{w_1, ...w_{n-3}, w_{n-2} = 0, w_{n-1} = 1, w_n = \infty\}$, $w_i \in \mathbb{C}$, and $h : \hat{\mathbb{C}} \to \hat{\mathbb{C}}$ such that $h(z_i) = w_i$, find $f : \hat{\mathbb{C}} \to \hat{\mathbb{C}}$ satisfying:

1. f is a quasiconformal homeomorphism of $\hat{\mathbb{C}}$ to itself.
2. f is isotopic to h relative to the points $\{0, 1, \infty, z_1, ..z_{n-3}\}$, i.e. $f(z_i) = w_i$.
3. μ_f, the Beltrami coefficient of f, satisfies $||\mu_f||_\infty \leq ||\mu_g||_\infty$ for all g satisfying (1) and (2) above.

Note that in the above problem we require h as input, as otherwise there is no unique extremal map (uniqueness is guaranteed as long as one stays within the homotopy class). The first problem (referred to henceforth as the polygon mapping problem) can be reduced to this problem (referred to as the punctured sphere extremal map problem).

In Chapter 2 we address the issue of computing Teichmüller maps between polygons or between punctured spheres. We present an iterative procedure in the continuous setting first. We prove that this procedure converges to the unique extremal map. We then discretize our procedure and give all the necessary details. We end with some discussions on the complexity and the approximation guarantee of our method.

1.4.2 Exploration of path-space using sensor network geometry

Consider a sensor network distributed densely in a planar domain (possibly with holes). A single path from source to destination may give limited throughput due to bandwidth constraints, hop length, wireless interference or other transmission failures. If there is a lot of data to be delivered, it is natural to consider using multiple disjoint paths such that different data segments can be simultaneously delivered to the destination. With multi-path routing one obtains higher throughput and lower delay. Such multi-path routing can also be used to enhance data security. For example, sensor data can be encoded such that different codewords are sent along different paths. Therefore a single compromised node stays on at most one path and with its captured data segments it is unlikely to reconstruct the original data. Multiple disjoint paths also provide one with a resilient routing scheme, which is immune to node or link failures.

Both multi-path routing and resilient routing are challenging problems for a general graph setting, especially if each node cannot afford to have the global knowledge. In this work we use a geometric approach to allow efficient exploration of the path space with very little overhead. We are motivated by the recent development on regulating sensor network geometry using conformal mapping [101, 102], in which any sensor network can be embedded to be circular (and any possible hole is made circular as well), and greedy routing guarantees delivery. In this work we explore the freedom of a Möbius transformation inherent to this conformal mapping. By applying a Möbius transformation we can get an alternative embedding with the same property such that greedy routing generates a different path. We present distributed algorithms using local information and limited global information (the positions and sizes of the holes) to generate disjoint multi-paths for a given source destination pair or switch to a different path 'on the fly' when transmission failure is encountered. The overhead of applying a Möbius transformation simply boils down to four parameters that could be carried by a packet or determined when needed at the source. Demonstrated by simulation results, this method compares favorably in terms of performance and cost metrics with centralized solutions of using flow algorithms or random walk based decentralized solutions in generating alternative paths.

1.4.3 Traffic analysis and source-location privacy under random walks

Random walk on a graph is a Markov chain and thus is "memoryless" as the next node to visit depends only on the current node and not on the sequence of events that preceded it. With these properties, random walk and its many variations have been used in network routing to "randomize" the traffic pattern and hide the location of the data sources.

In this work we show a myth in common understanding of the memoryless property of a random walk applied for protecting source location privacy in a wireless sensor network. In particular, if one monitors only the network boundary and records the first boundary node hit by a random walk, this distribution can be related to the location of the source node. For the scenario of a single data source, a very simple algorithm which says the simple integration along the network boundary would reveal the location of the source. We also develop a generic algorithm to reconstruct the source locations for various sources that have simple descriptions (e.g., multiple source locations, sources on a line segment, sources in a disk). This represents a new type of traffic analysis attack for invading sensor data location privacy and essentially re-opens the problem for further examination.

1.4.4 Topology dependent space filling curves for sensor networks and applications

We consider a sensor network that densely covers a planar domain, possibly with multiple holes in the network. In this work we develop algorithms to linearize the network, i.e., covering the sensor network by a single path. By enforcing a linear order on the sensor nodes one can carry serial logical definitions and serial operations on both the sensor nodes and the sensor data. Some applications include serial data fusion, motion planning of data mules and sensor node indexing.

In this work 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 proportionally 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 versus standard space filling curves and random walks.

1.5 Layout and References

This thesis is organized as follows. In Chapter 2 we will present our proposed solution to the problem of computing Teichmüller maps. Our main focus will be on planar domains, namely polygons or punctured Riemann spheres, i.e., on attacking Problem 1.4.1.

In Chapter 3 we will outline the work of [46], where we propose a method to compute multiple disjoint paths in a sensor network lying in a domain with or without holes. This constitutes our proposed solution to Problem 1.4.2. The author of this thesis is a co-author of [46], and has permission from all the other co-authors to include any part of the material in [46].

In Chapter 4 (based on [109]) we address the fallacy behind the belief that random routing brings with itself source privacy. We use the concept of harmonic measure to analyze traffic from random routing, and show how an intelligent adversary can detect the location of the source, or even multiple sources, pretty accurately. The author of this thesis is a co-author of [109], and has permission from all the other co-authors to include any part of the material in [109].

In Chapter 5 we show how to obtain "dense" curves in any sensor network deployed in a domain with or without holes. Our technique first uses a conformal map called the slit map to transform the domain into an appropriate space. This space then become a "table" for billiards—the billiards curve is the proposed dense curve. The author of this thesis is a co-author of [10], and has permission from all the other co-authors to include any part of the material in [10].

Chapter 2

Computing Teichmüller Maps

2.1 Introduction

Surface registration refers to the process of finding an optimal one-to-one correspondence between surfaces. It plays a fundamental role in many fields such as computer vision, computer graphics and medical imaging. In computer vision [133], it has been applied to deformable surface matching and tracking, shape analysis; in medical imaging [124], registration is always needed for statistical shape analysis [132], morphometry and processing of signals on brain surfaces (e.g., denoising or filtering). In computer graphics, surface registration is needed for texture mapping [70], which maps the surface onto planar domains, and pulls back the texture images onto the surface to improve the visualization. Developing an effective algorithm for registration is therefore very important.

Surface registration has the following important requirements:

1. *Existence:* The mapping should exist with mild topological constraints.
2. *Uniqueness:* The mapping should be unique under specific boundary conditions or topological constraints.
3. *Bijective:* The mapping needs to be bijective, and in general, differentiable.
4. *Least Distortion:* The mapping should preserve the surface geometric structures and reduce the distortions as much as possible.

The results of this work help propose a novel surface registration method based on extremal quasiconformal mapping.

A *conformal mapping* between surfaces preserves angles. In general, given two Riemann Surfaces, there does not exist a conformal map between them, i.e., all mappings between them will induce angle distortions. If the angle distortions are bounded, then the mapping is called *quasiconformal*. Conformal and quasiconformal mappings have been vastly applied in computer graphics.

In mathematics, the study of conformal equivalence class of surfaces (moduli spaces) has a long and rich history with major contributions by Teichmüller [117], [118], Ahlfors [2], Grötzch [39], Hamilton, Krushkal, Reich [94] and Strebel [113], and is currently an active area of research. The welding procedure (arising from considering the Riemann maps from the interior and exterior of a closed curve) was used in [107] to get a numerical procedure for obtaining "fingerprints" of a closed curve.

Among all the quasiconformal mappings from a marked surface to another, the one with minimal angle distortion is called the *extremal quasiconformal mapping*. In most of the cases, an extremal quasiconformal mapping exists and is unique, and it can be represented as

$$\text{conformal} \circ \text{affine} \circ \text{conformal}.$$

While extremal maps have long been studied in the field of pure mathematics, their explicit construction has not gained particular attention. Extremal quasiconformal maps between two planar disks with Dirichlet boundary conditions were recently explored in [125] for purposes of surface parametrization. However, the energy functional they minimize is non-convex and highly non-linear, and because of this there is no guarantee on how far the solution is from the extremal one. Our method on the other hand, uses clues from an explicit construction in the continuous setting to obtain a discrete approximation. At the heart of our discrete algorithm we solve a convex optimization program, which does not pose the aforementioned difficulties.

This work is intended to be an introduction to this rich subject from a computational perspective, and we certainly feel that a lot of computationally challenging

open problems lie hidden.

2.2 Problem statements

In this section we first describe the extremal quasiconformal map problem between planar domains. We then enlist some properties of the extremal map, and end with different techniques on how to represent such maps.

Let P and Q be two n-polygons[1] in the plane. A fundamental result in complex analysis, the Riemann mapping theorem (see Theorem 1 in Chapter 1), states that any simply connected domain can be mapped bijectively in an angle preserving way (conformally) to the open unit disk. This implies that P can be mapped to Q conformally. This map, which is a-priori defined on the interior of the polygon, can be extended continuously to the boundary. Generally, this boundary extension will not map the vertices of one polygon to the vertices of the other.

If the vertices of the two polygons are ordered and one only considers homeomorphisms which map the vertices of P to those of Q in an order-preserving way, in the generic case some angle distortion is inevitable. The problem we consider is to find the map in this family that minimizes the maximum angle distortion (over all points in the base domain).

The quantification of angle distortion is best represented when we view the homeomorphism as a homeomorphism $f(z)$ between domains in the complex plane $\mathbb{C}$ instead of $\mathbb{R}^2$, via a *Beltrami coefficient*. Intuitively, if one considers the unit circle in the tangent space at a point, the differential of the map (which is a linear transformation on this tangent space) maps this circle to an ellipse. The Beltrami coefficient μ_f is related to the eccentricity of this ellipse. For details, please see Section 1.1 in Chapter 1.

[1] We allow for ∞ to be a vertex of the polygon.

2.2.1 Polygon mapping

Let $\{v_i\}_{i=1}^n$ and $\{v_i^{'}\}_{i=1}^n$ be an ordering of the vertices of P and Q, respectively. The problem we study can now be stated as computing

$$\arg\min_f ||\mu_f||_\infty$$

where f is a quasiconformal homeomorphism from P to Q such that $f(v_i) = v_i^{'}$. $k_f = ||\mu_f||_\infty$ is also referred to as the dilatation of f.

We will make precise what we mean by computing or representing such an f. Different definitions can be given to handle different applications, and we explore them in Section 2.2.5.

Since the polygons are conformally equivalent to the disk, each of the polygons can be conformally mapped to the unit disk $\mathbb{D}$. Assume that z_i and w_i are the images (under a conformal map) of the vertices of P and Q, respectively. The above problem then translates to:

Problem 4. *[Polygon mapping problem] Given $\{z_1, ...z_n, w_1, ...w_n\} \in \partial\overline{\mathbb{D}}$, find $f : \overline{\mathbb{D}} \to \overline{\mathbb{D}}$ satisfying:*

1. *f is a quasiconformal homeomorphism of $\overline{\mathbb{D}}$ to itself.*
2. *$f(z_i) = w_i$, $i \in \{1, ...n\}$*
3. *μ_f (the Beltrami coefficient of f) satisfies $||\mu_f||_\infty \leq ||\mu_g||_\infty$ for all g satisfying (1) and (2) above.*

Such an f will be called an extremal map between the polygons P and Q. From classical theory, f is known to be unique (see Section 2.2.4).

2.2.2 Punctured sphere mapping

There are several generalizations of Problem 4. We consider one in which the points are placed anywhere in the plane. Let $\hat{\mathbb{C}}$ denote the Riemann sphere (the complex plane $\mathbb{C}$ union the point at infinity).

Problem 5. *[Punctured sphere problem] Given* $\{z_1, ...z_{n-3}, z_{n-2} = 0, z_{n-1} = 1, z_n = \infty\}$, $\{w_1, ...w_{n-3}, w_{n-2} = 0, w_{n-1} = 1, w_n = \infty\}$, *and* $h : \hat{\mathbb{C}} \to \hat{\mathbb{C}}$ *such that* $h(z_i) = w_i$, *find* $f : \hat{\mathbb{C}} \to \hat{\mathbb{C}}$ *satisfying:*

1. f *is a quasiconformal homeomorphism of* $\hat{\mathbb{C}}$ *to itself.*
2. f *is isotopic to* h *relative to the points* $\{0, 1, \infty, z_1, ..z_{n-3}\}$, *i.e.* $f(z_i) = w_i$.
3. μ_f, *the Beltrami coefficient of* f, *satisfies* $||\mu_f||_\infty \leq ||\mu_g||_\infty$ *for all* g *satisfying (1) and (2) above.*

Note that in the above version we require h as input, as otherwise there is no unique extremal map (uniqueness is guaranteed as long as one stays within the homotopy class). See Section 2.2.4 for details. We will prove later (Section 2.3.1) that the polygon mapping problem can be reduced to the punctured sphere problem.

2.2.3 Discrete polygon mapping

From a computational geometry viewpoint, one could define a completely discrete analogue of the polygon mapping problem (Problem 4). Given two triangles T_1 and T_2, there always exists an affine mapping $\sigma : T_1 \to T_2$. Such a map can be written in the form $\sigma(z) = az + b\bar{z}$, and the Beltrami coefficient of σ equals b/a.

Given a mapping from a triangulated polygon P to another polygon Q such that the image of the triangulation of P is a triangulation of Q, its discrete Beltrami differential can be defined piecewise on every triangle.

Problem 6. *[Discrete polygon mapping]*

- ***Input:*** *An ordered list of vertices* $\{v_i, v_i^{'}\}_{i=1}^n$ *of two* n*-polygons* P *and* Q, *and an initial triangulation* T *of a set of* m *points* $\{s_i\}_{i=1}^m$ *contained in the interior of* P.
- ***Output****:* $\{f(s_j)\}_{j=1}^m$, *where* f *satisfies*
  1. $f(v_i) = v_i^{'}$,
  2. $f(T)$ *is a triangulation of* Q *with the same combinatorics as* T,

3. $\max_{t\in T} |\mu_f(t)| \leq \max_{t\in T} |\mu_g(t)|$ *for any* g *satisfying (1) and (2) above (here* $\mu_f(t)$ *is the Beltrami coefficient of the affine mapping between the triangles* t *and* $f(t)$*, and* $|.|$ *denotes its magnitude),*

if such an f *exists.*

When considering Problem 6, existence and uniqueness of f are non-trivial concerns. Although we do not attack Problem 6, one can see that it is close in spirit to the polygon mapping problem.

2.2.4 Teichmüller mapping

Let f be the extremal map for the polygon mapping problem. From the classical theory of quasiconformal mappings ([3, 35, 117]), it is known that f is unique. Moreover, f is of *Teichmüller form*. This means that f determines a pair of holomorphic quadratic differentials (see Section 1.3.3 in Chapter 1) ϕ and ψ on $\mathbb{D}$. $\phi(z)dz^2$ (resp. $\psi(z)dz^2$) is real along $\partial\mathbb{D} \setminus \{z_1, ...z_n\}$ (resp. $\partial\mathbb{D} \setminus \{w_1, ...w_n\}$) and has at most simple poles at the z_is (resp. w_is). The trajectories of ϕ and ψ partition the disk into finitely many horizontal strips R_j and R_j'. They are mapped by the integrals $\Phi(z) = \int \sqrt{\phi(z)}dz$ and $\Psi(z) = \int \sqrt{\psi(z)}dz$ onto Euclidean horizontal rectangles

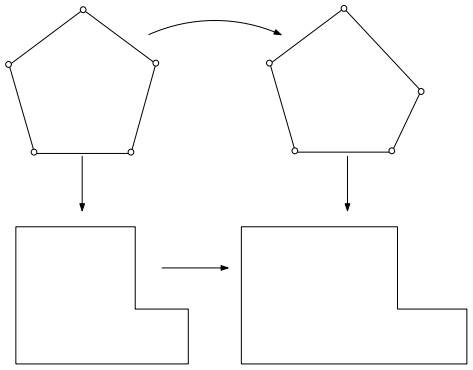


Figure 7: Teichmüller map between two pentagons.

$$\Phi(R_j) = \{(x, y) \in \mathbb{R}^2 : 0 < x < a_j, 0 < y < b_j\}$$

$$\Psi(R_j^{'}) = \{(x,y) \in \mathbb{R}^2 : 0 < x < a_j^{'}, 0 < y < b_j^{'}\}$$

and the extremal mapping f satisfies

$$\Psi \circ f \circ \Phi^{-1}(x+iy) = Kx + iy$$

where

$$K = \frac{1 + ||\mu_f||_\infty}{1 - ||\mu_f||_\infty}$$

This also means that

$$\mu_f = k(\frac{\bar{\phi}}{|\phi|}), \tag{5}$$

where $k = \frac{K-1}{K+1}$. Figure 7 demonstrates a Teichmüller map between two pentagons. A similar statement holds for the punctured sphere extremal map problem. We refer reader to [35].

From now on we use the terms extremal quasiconformal map and Teichmüller map interchangeably, as the Beltrami coefficients of the solutions to Problems 4 and 5 are of Teichmüller form (Equation 5).

2.2.5 Representing the Teichmüller map

There are two ways to to represent the solutions to Problems 4 and 5.

Computing the values of the function at a dense set of points: One could place a set $\mathcal{M}$ of m points inside the base polygon P (resp. the base punctured sphere) and compute $f(z)$ for $z \in \mathcal{M}$, where f is the Teichmüller map to the target polygon Q (resp. the target punctured sphere). Although the values of f at points in $\mathcal{M}$ do not uniquely characterize f, this technique suffices for all the intended applications as long as the set $\mathcal{M}$ can be generated to any desired resolution.

Representing the Beltrami coefficient of the extremal map: A basis for the space of holomorphic quadratic differential on a polygon or on a punctured sphere can be written down explicitly (see Section 1.3.3 in Chapter 1), e.g, for $R = \hat{\mathbb{C}} \setminus$

$\{0, 1, \infty, z_1, ...z_{n-3}\}$ (the Riemann sphere with n punctures),

$$\phi_i(z) = \frac{z_k(z_k - 1)}{z(z-1)(z-z_k)} \quad k \in \{1, 2, ...n-3\} \tag{6}$$

form a basis of $(n-3)$ dimensional complex vector space of holomorphic quadratic differentials.

Let f be a solution to either Problem 4 or Problem 5. By Equation 5, $\mu_f = k(\frac{\bar{\phi}}{|\phi|})$, where ϕ can now be written in terms of the given basis as $\phi = \sum_{i=1}^{n-3} \lambda_i \phi_i$.

Since μ_f uniquely characterizes f up to a Möbius transformation, the set of parameters $\{k, \lambda_1, \cdots, \lambda_{n-3}\}$ completely determines the extremal map. Therefore, once a basis for the space of holomorphic quadratic differentials has been agreed upon, these $n-2$ parameters can be used for representing f.

Our method: We adopt the first method to represent an extremal map. We explain how we generate points inside the polygon to ϵ accuracy in Section 2.5.1. Note that although the second approach is a more compact way of representing f, for applications one will need to compute f by solving the Beltrami equation for μ_f, and all known methods use a discretization of the domain (see Section 2.5.2 for details).

2.3 Preliminaries

2.3.1 Relation between polygon mapping and punctured sphere mapping

We claim that an algorithm for Problem 5 can be used to give a solution to Problem 4. We will start with the data for a polygon mapping problem and convert it to a data for the punctured sphere problem.

Given polygons P and Q, and an ordering of their vertices v_i and v_i', let h_0 be any quasiconformal homeomorphism mapping P to Q, such that $h_0(v_i) = v_i'$. Note that the disk and the upper half plane $\mathbb{H}$ are conformally equivalent. By conformally mapping P and Q to $\mathbb{H}$ (denote the maps by π_P and π_Q), we get a quasiconformal self-homeomorphism h_u of $\mathbb{H}$, satisfying $h_u(z_i) = w_i$, where z_i and w_i are images

(under π_P and π_Q) of v_i and v_i', respectively. Furthermore h_u can be normalized to fix 0, 1 and ∞.

Let $\overline{\mathbb{H}}$ denote the lower half plane, and define a quasiconformal self-homeomorphism h_ℓ of $\overline{\mathbb{H}}$ by $h_\ell(z) = \overline{h_u(\bar{z})}$. Now h_u and h_ℓ agree on $\mathbb{R}$, and can be pieced together to get a quasiconformal self-homeomorphism h of $\hat{\mathbb{C}}$ satisfying $h(z_i) = w_i$. Note that h fixes 0, 1 and ∞.

Theorem 7. *Let f be the solution to Problem 5 when it is fed the input data $\{z_i, w_i, h\}$ as above. Then:*

1. $\mu_f(z) = \overline{\mu_f(\bar{z})}$.
2. *Let f_u denote the restriction of f to $\mathbb{H}$. Then $(\pi_Q)^{-1} \circ f_u \circ \pi_P$ is the solution to Problem 4 with data P and Q.*

Proof. We first prove that for all $z \in \mathbb{C}$, $f(z) = \overline{f(\bar{z})}$. Define another homeomorphism g as $g(z) = \overline{f(\bar{z})}$. It is straightforward to check that g is a self homeomorphism of $\hat{\mathbb{C}}$ and satisfies $g(z_i) = w_i$.

Now $||\mu_f||_\infty = ||\mu_g||_\infty$. By uniqueness of the extremal quasiconformal mapping, f is unique, and so must satisfy $f = g$ everywhere. Thus $f(z) = \overline{f(\bar{z})}$, which implies $\mu_f(z) = \overline{\mu_f(\bar{z})}$.

To prove the second assertion, let f^* denote the solution to Problem 4 with data P and Q. Using the above construction, we get a self-homeomorphism h^* of $\hat{\mathbb{C}}$ which satisfies the same properties as h and f. Uniqueness of f now implies that $f = h^*$. □

From now on, we will focus mainly on solving Problem 5. Our construction will need certain ingredients from the classical theory, which we outline next.

2.3.2 Ingredients of the proof

We start with a theorem which explains the dependence of a Beltrami coefficient to the solution of its Beltrami equation.

Mapping Theorem [[35], Theorem 1, Page 10] Let $\mu(z)$ be a measurable complex-valued function defined on a domain Ω for which $||\mu||_\infty = k < 1$. Consider the Beltrami equation,

$$f_{\bar{z}}(z) = \mu(z) f_z(z). \tag{7}$$

Theorem 8. *Equation 7 gives a one to one correspondence between the set of quasiconformal homeomorphisms of $\hat{\mathbb{C}}$ that fix the points $0, 1$ and ∞ and the set of measurable complex-valued functions μ on $\hat{C}$ for which $||\mu||_\infty < 1$. Furthermore, the normalized solution f^μ of Equation 7 depends holomorphically on μ and for any $r > 0$ there exists $\delta > 0$ and $C(r) > 0$ such that*

$$|f^{t\mu}(z) - z - tV(z)| \leq C(r)t^2 \; for \; |z| < r \; and \; |t| < \delta, \tag{8}$$

where

$$V(z) = -\frac{z(z-1)}{\pi} \int\int_{\mathbb{C}} \frac{\mu(\zeta) d\xi d\eta}{\zeta(\zeta - 1)(\zeta - z)}, \tag{9}$$

and $\zeta = \xi + i\eta$.

Composition of Quasiconformal Maps Let μ, σ and τ be the Beltrami coefficients of quasiconformal maps f^μ, f^σ and f^τ with $f^\tau = f^\sigma \circ (f^\mu)^{-1}$. Then

$$\tau = \left(\frac{\sigma - \mu}{1 - \bar{\mu}\sigma} \frac{1}{\theta} \right) \circ (f^\mu)^{-1}, \tag{10}$$

where $p = \frac{\partial}{\partial z} f^\mu(z)$ and $\theta = \frac{\bar{p}}{p}$. In particular, if f^σ is the identity, that is, if $\sigma = 0$, then

$$\tau = -\left(\mu \frac{p}{\bar{p}} \right) \circ (f^\mu)^{-1}.$$

Equivalence relations on Beltrami coefficients Let $R = \hat{\mathbb{C}} \setminus \{0, 1, \infty, z_1, \cdots, z_{n-3}\}$ denote the n times punctured sphere. Denote by $B(R)$ the space of all Beltrami differentials on R, i.e. all complex valued measurable functions on R.

Global equivalence

This relation is defined only on Beltrami differentials of norm less than 1, i.e. those that belong to the unit ball

$$B_1(R) = \{\mu \in B(R) : ||\mu||_\infty < 1\}$$

Given two such differentials μ and υ, denote the solution to their respective normalized[2] Beltrami equations as $f^\mu : R \to R_0$ and $f^\upsilon : R \to R_1$. Both R_0 and R_1 are punctured spheres.

Definition 9. *[Global equivalence] The differentials μ and υ are called globally equivalent (denoted as $\mu \sim_g \upsilon$) if:*

- $f^\mu(z_i) = f^\upsilon(z_i)\ \forall i$.
- *The identity map from R_0 to R_1 is homotopic to $f^\upsilon \circ (f^\mu)^{-1}$ via a homotopy consisting of quasiconformal homeomorphisms.*

Thus, the punctured sphere extremal map problem can be restated as: Given $\mu \in B_1(R)$, find $\upsilon \sim_g \mu$, such that $||\upsilon||_\infty$ is the least in this global equivalence class.

Infinitesimal Equivalence

This relation is defined on all of $B(R)$.

Definition 10. *[Infinitesimal equivalence] Two Beltrami differentials μ and υ are infinitesimally equivalent (written $\mu \sim_i \upsilon$) if*

$$\int_R \mu\phi = \int_R \upsilon\phi$$

for all $\phi \in A(R)$, with $||\phi||(:= \int_R |\phi|) = 1$.

Note that on any Riemann surface, the above pairing between a Beltrami differential and a quadratic differential gives a density, hence the integral is well defined. In our case, we have an explicit basis for $A(R)$ (see Equation 4 in Chapter 1) and so this is equivalent to

$$\int_R \mu\phi_i = \int_R \upsilon\phi_i \quad \forall i \in \{1, 2...n-3\}$$

[2] fixing the points 0,1 and ∞. Hence the freedom of Möbius tranformation is accounted for.

This equivalence will play a key role in our algorithm. Infinitesimal equivalence tells us when two Beltrami differentials are the same tangent direction in the Teichmüller space.

Definition 11. *[Infinitesimally extremal] A beltrami differential υ is called infinitesimally extremal if $||\upsilon||_\infty \leq ||\mu||_\infty$ for all $\mu \sim_i \upsilon$.*

Definition 12. *[Trivial Beltrami differential] A beltrami differential υ is called trivial if it is globally equivalent to 0.*

Definition 13. *[Infinitesimally trivial Beltrami differential] A beltrami differential υ is called infinitesimally trivial if it is infinitesimally equivalent to 0.*

The following lemma relates infinitesimally trivial Beltrami coeffcients to globally trivial ones.

Lemma 14. *[Variational lemma][[35], Theorem 6, Page 140] μ is an infinitesimally trivial Beltrami differential if, and only if, there exists a curve σ_t of trivial Beltrami differentials for which $\sigma_t(z) = t\mu_z + O(t^2)$ uniformly in z.*

Teichmüller contraction The principle of Teichmüller contraction states that given a Beltrami coefficient μ, its distance to the globally extremal μ^* is of the same order as its distance to the infinitesimally extremal υ. For a full statement and proof of the principle, see [35], Theorem 10, page 103.

We will restate the part of the principle relevant to us. Let $k_0 = ||\mu^*||_\infty$ be the dilatation of the extremal Beltrami coefficient in the same global class as μ, and let υ be the infinitesimally extremal Beltrami coefficient in the infinitesimal class of μ. Fix $0 < k_1 < 1$. Then

$$\frac{||\mu||_\infty - k_0}{4} \leq \frac{2}{(1-k_1)^2}(||\mu||_\infty - ||\upsilon||_\infty) \leq \frac{2}{(1-k_1)^4}(||\mu||_\infty - k_0). \qquad (11)$$

Hamilton-Krushkal, Reich-Strebel condition

Theorem 15. *[Hamilton-Krushkal, Reich-Strebel necessary-and-sufficient condition for extremality] A quasiconformal map f has minimal dilatation in its Teichmüller class if and only if its Beltrami coefficient μ is extremal in its infinitesimal class.*

2.4 Continuous construction

In this section we describe the basic idea behind our algorithm. We will focus on the punctured sphere problem for now and later describe the analogous procedure for the polygon mapping problem.

2.4.1 Summary of the construction

Our continuous construction $\mathcal{A}_c$ can be described as follows. At step i, Given a starting map $h_i : R \to S$ where R and S are n times punctured Riemann sphere, let υ_i denote the infinitesimally extremal Beltrami coefficient in the infinitesimal class of $\mu_i := \mu_{h_i}$. Observe that $\mu_i - \upsilon_i$ is infinitesimally trivial (see Definition 13).

1. Use Section 2.4.2 to construct a quasiconformal self-homeomorphism g_i of R such that
$$\mu_g = t(\mu_i - \upsilon_i) + O(t^2),$$
for some $t > 0$ and such that μ_g is trivial (see Definition 12).

2. Using Section 2.4.3, form $h_{i+1} = h_i \circ (g_i)^{-1}$ such that h_{i+1} has smaller dilatation than h_i.

3. Reiterate with h_{i+1} as the starting map.

We now expand on Steps 1 and 2 above.

2.4.2 Constructing self homeomorphisms g_i

Given a starting map h_i, we show how to construct the self homeomorphism g_i of R used in our construction. We will reduce notation by suppressing the i, keeping in mind that this is the ith step of the procedure. Thus μ_h and μ_g will denote the Beltrami differentials of h_i and g_i, respectively. Also, υ_h is the infinitesimally extremal Beltrami differential in the infinitesimal class of μ_h.

Let $\alpha = \mu_h - \upsilon_h$. Now α is infinitesimally trivial, and essentially we want to construct the curve of trivial Beltrami differentials $\mu_g(t) = t\alpha + O(t^2)$, described in Lemma 14.

Let $t > 0$ be small enough and let $f^{t\alpha}$ be the normalized solution to the Beltrami equation for $t\alpha$. We observe next that $f^{t\alpha}$ moves the points z_i only by a distance $O(t^2)$.

Theorem 16. *Let $r = \max_{1\leq i\leq n-3} |z_i|$, and let $f^{t\alpha}$ be as above. Then there exists a constant C_r depending only on r, and a constant $\delta > 0$ such that for all i,*

$$|f^{t\alpha}(z_i) - z_i| \leq C_r t^2, \ \ \forall t < \delta. \tag{12}$$

Proof. By Equation 4 in Chapter 1,

$$\phi_i(\zeta) = \frac{z_k(z_k - 1)}{\zeta(\zeta - 1)(\zeta - z_k)}$$

for $1 \leq i \leq n-3$ is a basis for the space of quadratic differentials on R. Let $\zeta = \xi + i\eta$. Infinitesimal equivalence of $t\alpha$ now implies that

$$\int\int_{\mathbb{C}} \frac{t\alpha d\xi d\eta}{\zeta(\zeta - 1)(\zeta - z_k)} = 0 \tag{13}$$

Now we use the mapping theorem (Theorem 8 in Section 2.3.2). In the notation of the theorem, $V(z_i) = 0$ by Equation 13. Existence of δ, C_r and Equation 12 now follow from the statement of the mapping theorem.

□

Denote $f^{t\alpha}(z_i)$ by $z_i^{'}$. We will first construct another homeomorphism $f_{\mathbf{v}}$ from $\hat{\mathbb{C}}$ to itself which satisfies $f_{\mathbf{v}}(z_i^{'}) = z_i$. We then define the required self homeomorphism $g_i = f_{\mathbf{v}} \circ f^{t\alpha}$. The construction of $f_{\mathbf{v}}$ will be via a vector field method.

Construction of $f_{\mathbf{v}}$ by vector field method: Let $\{D_1, \cdots, D_{n-3}\}$ denote disjoint open disks centered at z_i. Choosing the radius of each disk to be $d/2$, where $d = \max_{1\leq i,j\leq n-3} |z_i - z_j|$ ensures disjointness. We will fix these disks once and for all, and assume they are of equal radius r.

A single disk: We first construct a self homeomorphism $f_{\mathbf{v}}^i$ of $\hat{\mathbb{C}}$ which is the identity outside D_i, and maps $z_i^{'}$ to z_i. Now $z_i \in \mathbb{R}$, and by a rotation we can

assume that $z_i^{'}$ is real and greater than z_i. Consider the vector field

$$X(z) = p(z)(z_i^{'} - z_i)\frac{\partial}{\partial x},$$

where $p(z)$ is a C^∞ function identically zero outside D_i, and identically 1 inside the disk of radius $r/2$ around z_i, denoted as $D_i^{'}$. Let γ be the one parameter family of diffeomorphisms associated with this vector field. We denote the time parameter by s and note that the diffeomorphism γ_1 sends $z_i^{'}$ to z_i. We denote this diffeomorphism γ at $s = 1$ by $f_{\mathbf{v}}^i$.

Now define $f_{\mathbf{v}} = f_{\mathbf{v}}^{n-3} \circ f_{\mathbf{v}}^{n-2} \cdots \circ f_{\mathbf{v}}^1$.

Getting g_i by composing $f^{t\alpha}$ and $f_{\mathbf{v}}$: Define $g_i = f_{\mathbf{v}} \circ f^{t\alpha}$. In the following proof, we will simplify notation: V_t will denote the vector field diffeomorphism, and s_t will denote the solution to the Beltrami equation for $t\alpha$.

Theorem 17. *Let μ_g be the Beltrami coefficient of g_i. Then*

1. *μ_g is globally equivalent to 0.*
2. *$\mu_g = t\alpha + O(t^2)$*

Proof. First, note that since $f^{t\alpha}(z_i) = z_i^{'}$ and $f_{\mathbf{v}}(z_i^{'}) = z_i$, g_i is a self quasiconformal homeomorphism of $\hat{\mathbb{C}}$ which fixes z_i point-wise. Hence μ_g is globally equivalent to 0.

In what follows, we wish to prove that $\mu_g(t) = t\alpha + O(t^2)$ i.e., we want to prove that $\mu_g^{'}(t = 0) = \alpha$. Recall that $g_t(z, \bar{z}) = (V_t \circ s_t)(t, z, \bar{z})$ where $V(w, \bar{w})$ is the diffeomorphism induced by the vector field method and s_t is the solution of the Beltrami equation $\partial_{\bar{z}}(s_t) = t\alpha\partial_z(s_t)$.

The Beltrami coefficient of g_t is $\mu_g(z)$ which is

$$\begin{aligned}\mu_{g_t}(z, \bar{z}) &= \frac{\partial_{\bar{z}}(V_t \circ s_t)}{\partial_z(V_t \circ s_t)} \\ &= \frac{\partial_z V_t \partial_{\bar{z}} s_t + \partial_{\bar{z}} V_t \partial_{\bar{z}} \bar{s}_t}{\partial_z V_t \partial_z s_t + \partial_{\bar{z}} V_t \partial_z \bar{s}_t}\end{aligned}$$

To find $\mu_g^{'}(0)$ we note that

$$\begin{aligned}\partial_{\bar{z}}(s_0) &= 0\\ \partial_{\bar{z}}(\bar{s_0}) = \partial_z(s_0) &= 1\\ \partial_{\bar{z}}(V_0) &= 0\\ \partial_{\bar{z}}(\bar{V_0}) = \partial_z(V_0) &= 1\end{aligned}$$

Hence

$$\begin{aligned}\mu_g^{'}(0) &= (\partial_{\bar{z}}(s_t))^{'}(t=0) + (\partial_{\bar{z}}(V_t))^{'}(t=0)\\ &= \mu_{s_t}^{'}(0) + (\partial_{\bar{z}}(V_t))^{'}(t=0)\\ &= \alpha + (\partial_{\bar{z}}(V_t))^{'}(t=0)\end{aligned}$$

We claim that $(\partial_{\bar{z}}(V_t))^{'}(t=0) = 0$. Indeed, $V_t(z,\bar{z}) = \gamma(s=1,t,z,\bar{z})$ where

$$\frac{d\gamma}{ds} = X(\gamma, t)$$

where the vector field $X(z,\bar{z},t) = \sum_i \rho(z,\bar{z})(\tilde{z}_i(t) - z_i)$. Recall that $(\tilde{z}_i)^{'}(t=0) = 0$, $\tilde{z}_i(0) = z_i$, $\gamma(z,\bar{z},t=0) = z$, and $X(z,\bar{z},t=0) = 0$. Hence,

$$\begin{aligned}\frac{d(\partial_{\bar{z}}\gamma)^{'}(t=0)}{ds} &= (\partial_{\bar{z}}X(\gamma,\bar{\gamma},t))^{'}(t=0)\\ &= (\partial_z X \partial_{\bar{z}}\gamma + \partial_{\bar{z}}X\partial_{\bar{z}}\bar{\gamma})^{'}(0)\\ &= (\partial_{\bar{z}}X)^{'}(0)\\ &= \sum_i (\rho_{\bar{z}}(\gamma,\bar{\gamma})(\tilde{z}_i(t) - z_i))^{'}(0)\\ &= 0\end{aligned}$$

This means that $(\partial_{\bar{z}}\gamma)^{'}(t=0)$ at $s=1$ (which is what we want) is the same as $(\partial_{\bar{z}}\gamma)^{'}(t=0)$ at $s=0$ wherein it is 0. □

Note that we did not prescribe how to choose the t involved in the above construction, which needs to be chosen while implementing the algorithm. We will discuss this in Section 2.4.4.

2.4.3 The reduction lemma

Define $h_{i+1} = h_i \circ (g_i)^{-1}$. The next lemma guarantees that if t was chosen appropriately, then h_{i+1} has smaller dilatation than h_i .

The proof is similar to that of the Hamilton-Krushkal, Reich-Strebel necessary-and-sufficient condition for extremality (see Theorem 15 in Section 2.3), published in a sequence of papers. We refer the reader to [35] for a combined proof of this celebrated result, which is the one we adapt.

For the purposes of this lemma and to reduce notation, let h be any quasiconformal homeomorphism between R and S which is a valid input to Problem 5, and μ_h denote its Beltrami differential.

Lemma 18. *Let υ_h be the infinitesimally extremal Beltrami differential in the infinitesimal class of μ_h. Let $\mu_g(t)$ be a curve of Beltrami differentials with the following properties:*

1. *$\mu_g(t)$ is trivial.*
2. *$\mu_g(t) = t(\mu_h - \upsilon_h) + O(t^2)$.*

Then $\exists\ \delta > 0$ such that $\forall t < \delta$, the map $h_t = h \circ (g_t)^{-1}$ has smaller dilatation than h

Proof. Note that by the formula for composition of quasiconformal maps (Equation 10 in Section 2.3),

$$\mu_{h_t}(g_t(z)) = \frac{\mu_h - \mu_g(t)}{1 - \overline{\mu_g(t)}\mu_h}\frac{1}{\theta_t}, \tag{14}$$

where $\theta_t = \overline{p_t}/p_t$ and $p_t = \frac{\partial g_t}{\partial z}$. Equation 14 implies

$$|\mu_{h_t} \circ g_t|^2 = \frac{|\mu_h|^2 - 2\text{Re}(\mu_h\overline{\mu_g(t)}) + |\mu_g(t)|^2}{1 - 2\text{Re}(\mu_h\overline{\mu_g(t)}) + |\mu_g(t)\mu_h|^2} \tag{15}$$

Using the fact that

$$||\mu_g(t) - t(\mu_h - \upsilon_h)||_\infty = O(t^2)$$

and differentiating Equation 15 with respect to t once and putting $t = 0$, we get that

$$|\mu_{h_t} \circ g_t| = |\mu_h| - t\frac{1-|\mu_h|^2}{|\mu_h|}\text{Re}(|\mu_h|^2 - \mu_h\overline{\upsilon_h}) + O(t^2) \tag{16}$$

Let $k_0 = ||\upsilon_h||_\infty < k = ||\mu_h||_\infty$. Define

$$S_1 = \{z \in R : |\mu_h(z)| \leq (k + k_0)/2\}$$

and

$$S_2 = \{z \in R : (k + k_0)/2 < |\mu_h(z)| \leq k\}$$

Clearly, $S_1 \cup S_2 = R$. Since in S_1 the starting value of this curve at $t = 0$ is $|\mu|$, which is certainly less than k, 14 implies there exists $\delta_1 > 0$ and $c_1 > 0$ such that for $0 < t < \delta_1$,

$$|\mu_{h_t} \circ g_t(z)| \leq k - c_1 t \quad \text{for } z \in S_1 \tag{17}$$

For z in S_2 the coefficient of t in 16 is bounded below by

$$\frac{1-k^2}{k}\left[\left(\frac{k+k_0}{2}\right)^2 - k_0 k\right] = \frac{1-k^2}{k}\left(\frac{k-k_0}{2}\right)^2 > 0$$

Therefore, 16 implies there exists $\delta_2 > 0$ and $c_2 > 0$ such that for $0 < t < \delta_2$,

$$|\mu_{h_t} \circ g_t(z)| \leq k - c_2 t \quad \text{for } z \in S_2 \tag{18}$$

Putting together 17 and 18, we find that $||\mu_{h_t}||_\infty < k$ for sufficiently small $t > 0$, proving the lemma.

□

To the best of the author's knowledge, the above lemma is the first result that describes, given a starting map, how to get a map with a smaller dilatation.

2.4.4 Choosing a good value of t

In what follows, we denote h_N (the quasiconformal map after N iterations of the algorithm) by h for convenience. In fact, we drop the subscript N altogether. Recall

from Section 2.4.1 that from μ_h, we construct a new quadratic differential ν_h that is infinitesimally extremal.

We then construct g, which is a self homeomorphism of the base punctured sphere R. There exists a constant C such that the Beltrami coefficient μ_g of g satisfies $|\mu_g - \alpha_t| \leq Ct^2$ when $t < \delta$.

Notice that C and δ could potentially depend on N.

Our aim will be to find an "optimal" t that is small enough so that everything in Theorem 18 happens, yet is large enough so that the dilatation reduces a lot in the $N+1$ step, i.e., for $h_{N+1} = h \circ g_t^{-1}$.

In what follows, $k < 1$ is the dilatation of h and $k_0 = ||\nu||_\infty \leq k$.

Theorem 19. *[Decrease in one step] The supremum of $|\mu_{h_{N+1}} \circ g_t(z)|$ over all z is less than $k - d$,where*

$$d = \min\left(\frac{k-k_0}{4}, \frac{(k-k_0)^2 t(1-k^2)}{8}\right), \tag{19}$$

whenever

$$t \leq \min(1, \delta, \frac{\epsilon}{4}, \frac{\sqrt{\epsilon}}{\sqrt{2C}}, \frac{(k-k_0)^2(1-k^2)}{1-k^2+C}), \tag{20}$$

where $\epsilon \leq \min(\frac{1}{2}, \frac{k-k_0}{8})$.

Proof. : As in the proof of Lemma 18, let S_1 be the region where $|\mu_h| \leq \frac{k+k_0}{2}$ and S_2 be such that $\frac{k+k_0}{2} \leq |\mu_h| < k$. For now, assume $t < \delta$.

On S_1, if $t < \min(1, \delta, \frac{\epsilon}{4}, \frac{\sqrt{\epsilon}}{\sqrt{2C}})$,

$$\begin{aligned}
|\mu_{h_{N+1}} \circ g_t(z)| &= \frac{|\mu_h - \mu_g|}{|1 - \mu_h \bar{\mu_g}|} \\
&\leq \frac{|\mu_h - t(\mu_h - \nu_h)|}{|1 - \mu_h \bar{\mu_g}|} + \frac{Ct^2}{|1 - \mu_h \bar{\mu_g}|} \\
&\leq \frac{1}{1-\epsilon}\left(\frac{k+k_0}{2} + 2t + Ct^2\right) \\
&\leq \frac{1}{1-\epsilon}\left(\frac{k+k_0}{2} + \epsilon\right)
\end{aligned}$$

where the last inequality follows by requiring $|\mu_h \bar{\mu_g}| < 2t + Ct^2 < \epsilon$, which is true

for the assumed value of t. Notice that $|mu_h|, |\nu_h|$ are less than 1.

Therefore, on S_1,

$$\begin{aligned} k - |\mu_{h_{N+1}|} &\geq \frac{k - k_0 - 2\epsilon(1+k)}{2(1-\epsilon)} \\ &> \frac{k-k_0}{4} \end{aligned}$$

if $\epsilon \leq \frac{k-k_0}{8}$.

On S_2,

$$\begin{aligned} \frac{|\mu_h - \mu_g|}{|1-\mu_h\bar{\mu_g}|} &\leq \frac{|\mu_h - t(\mu_h - \nu_h)|}{|1-\mu_h\bar{\mu_g}|} + \frac{Ct^2}{|1-\mu_h\bar{\mu_g}|} \\ &\leq \frac{|\mu_h - t(\mu_h - \nu_h)|}{|1-\mu_h t(\bar{\mu_h} - \bar{\nu_h})| - |\mu_h|Ct^2} + \frac{Ct^2}{|1-\mu_h\bar{\mu_g}|} \end{aligned}$$

Now,

$$\begin{aligned} k - |\mu_{h_{N+1}}| &\geq |\mu_{h_N}| - |\mu_{h_{N+1}}| \\ &\geq \frac{|\mu_h|(|1-\mu_h t(\bar{\mu_h} - \bar{\nu_h})| - |\mu_h|Ct^2) - |\mu_h - t(\mu_h - \nu_h)|}{|1-\mu_h t(\bar{\mu_h} - \bar{\nu_h})| - |\mu_h|Ct^2} - \frac{Ct^2}{1-\epsilon} \\ &\geq \frac{A-B}{|1-\mu_h t(\bar{\mu_h} - \bar{\nu_h})| - |\mu_h|Ct^2} - \frac{Ct^2}{1-\epsilon} \end{aligned} \tag{21}$$

where $A = |\mu_h|(|1-\mu_h t(\bar{\mu_h} - \bar{\nu_h})|)$ and $B = |\mu_h|Ct^2 - |\mu_h - t(\mu_h - \nu_h)|$. Using

$$A - B = \frac{A^2 - B^2}{A+B}$$

$$A + B < 4$$

$$\begin{aligned} A^2 - B^2 &= (1-|\mu_h|^2)(2\Re(t\mu_h(\bar{\mu_h} - \bar{\nu_h})) - t^2|\mu_h|^2|\mu_h - \nu_h|^2(1+|\mu_h|^2)) \\ &\geq (1-|\mu_h|^2)(t(k-k_0)^2 - t^2|\mu_h|^2|\mu_h - \nu_h|^2(1+|\mu_h|^2)) \end{aligned} \tag{22}$$

Using Equation 22 in Equation 21 and the fact that $\epsilon < 1/2$ we see that

$$\begin{aligned} k - |\mu_{h_{N+1}}| &\geq \frac{1-k^2}{4}(t(k-k_0)^2 - 8t^2) - 2Ct^2 \\ &\geq \frac{(k-k_0)^2 t(1-k^2)}{8} \end{aligned} \tag{23}$$

with the last equation holding if $t < \frac{(k-k_0)^2(1-k^2)}{1-k^2+C}$, concluding the proof. □

Thus, a good value of t can be obtained from Equation 20.

2.4.5 Convergence of the continuous construction

In this section we will prove that the sequence of homeomorphisms h_i obtained in the above procedure $\mathcal{A}_c$ (see Section 2.4.1) converges to the unique extremal quasiconformal map.

Theorem 20. *The above procedure $\mathcal{A}_c$ converges to the unique extremal map which is the solution to the punctured sphere problem, i.e. the sequence of q.c.h. h_i converge in the sup-norm to the extremal map.*

Proof. Let μ_i denote the Beltrami differential of h_i. Note that all the h_i fix 0, 1 and ∞. The sequence $x_n = ||\mu_n||_\infty$ is monotonically decreasing by Lemma 18. It is bounded below trivially by 0, and above by $||\mu_0||_\infty$, the dilatation of the starting map h_0. Thus, $x = \lim_{n\to\infty} x_n$ exists. The sequence x_n is cauchy, implying that the maps $h_n \circ (h_m)^{-1}$ converge in the sup-norm to the identity map (this argument can be found in [3]). This implies that the sequence h_i converges in the sup-norm to a q.c.h. f. Let μ_x be the Beltrami differential of f, and x be its L^∞ norm.

Let f^* be the solution to the punctured sphere problem (Problem 5), and denote by μ^* its Beltrami coefficient. We claim that $||\mu^*||_\infty = \lim_{n\to\infty} x_n$.

Assume the contrary; i.e., that x does not equal $||\mu^*||_\infty$. Let υ_x be the infinitesimally extremal Beltrami coefficient in the same class as μ_x. If $\upsilon_x \neq \mu_x$, then $\alpha \neq 0$, and the self-homeomorphism g obtained in the next step of the construction is not the identity map. Composing f by g^{-1} would give another map whose dilatation is strictly less than x (by Lemma 18), which contradicts the definition of x.

Thus we get that $\upsilon_x = \mu_x$, and by the Hamilton-Krushkal, Reich-Strebel condition (see Theorem 15), μ_x is the Beltrami coefficient of the unique extremal map. Thus $f = f^*$. □

We will discuss the convergence issue more in Section 2.6.

2.4.6 Polygon mapping problem

We showed in Theorem 7 how to reduce the polygon mapping problem to the punctured sphere problem. However, the above procedure can also be directly implemented on polygons, once we have the appropriate basis for the space of quadratic differentials. We give a simple well-known description of this basis first.

Holomorphic quadratic differentials on polygons Suppose P is a polygon with vertices $\{z_0, z_1, \cdots, z_{n-1}\}$. For each $2 < k < n$, there exists a unique conformal map ϕ_k, which maps P to a rectangle $\mathcal{R} = [0, 1] \times [0, h]$, and maps z_0, z_1, z_2, z_k to the four vertices of $\mathcal{R}$. Then $\{(\phi_k')^2 dz^2\}$ form the basis of holomorphic quadratic differentials on P.

All the above proofs can be modified analogously for the polygon mapping problem with minimal effort.

2.5 Discretization

In this section we will describe how to discretize the continuous construction presented in Section 2.4. We start with the well-known methods of conformally mapping a polygon to the disk.

Conformal Mapping There are two places where we need to compute a conformal map. One is when we map the polygons to the disk, and another when we want to compute the basis of polygon differentials by mapping the disk to a rectangle (described in Section 2.4.6).

One can use the Schwarz-Christoffel mapping directly for the rectangle, and the linear time conformal mapping algorithm in [14] or the discrete Ricci flow method [47], which work for both scenarios.

2.5.1 Finding points inside polygon

Assume $\epsilon > 0$ is given (this will be the "resolution" of the mesh). Here we show how to generate a set of points $\mathcal{P} = \{p_1, \cdots, p_m\}$ inside the polygon, and a triangulation $\mathcal{T}$ of $\mathcal{P}$ such that

1. The maximum edge length of an edge in the triangulation $\mathcal{T}$ is less than ϵ.
2. There exists a constant θ such that all angles in the triangulation $\mathcal{T}$ are greater than θ.

We use the Delaunay refinement algorithm [98] for generating $\mathcal{P}$. We start by adding points as in [98], where points are added at either the circumcenter of triangles, or at mid-points of edges. This ensures that the angles obtained in the Delaunay triangulation at every step are bounded below by θ. The claimed value of θ is around 20 ([98]).

Let us assume we have a triangulation which satisfies the angle bound condition, but not the edge length condition. Consider an edge e of length greater than ϵ, and the two triangles which share this edge. We again use the same criteria for adding points as in [98]. The new point is either added to the circumcenter of one of these triangles, or it is the mid-point of e. In both cases, the new edge length is decreased. We continue this procedure until we have a set of m points which satisfy the edge length condition.

2.5.2 Solving the Beltrami equation

Once the triangulation of the set of points inside the polygons has been computed, we show how to solve the Beltrami equation for a piecewise constant Beltrami differential.

Various methods have been proposed in literature ([23], [32]) for solving the Beltrami equation. In our experiments we solve it using the auxiliary metric method with Ricci flow [131]. The key idea is to define an auxiliary metric $ds^2 = |dz + \mu d\bar{z}|^2$, and then compute the Ricci flow under the auxiliary metric. This solves the Beltrami equation efficiently. Another method is based on Beltrami holomorphic flow [74].

Notice that solving a Beltrami equation is similar to solving an elliptic PDE. From error analysis of discrete solutions of elliptic PDEs, it is known that if all edges are length $O(\epsilon)$, then there exist methods such that the error is $O(\epsilon^2)$.

2.5.3 The starting map between polygons

Here we show how to get a candidate starting map between the polygons involved in Problem 4. Let P and Q be the two polygons, with vertices $\{z_k\}$ and $\{w_k\}$ respectively. Compute a conformal map ϕ_0 which maps P to a rectangle R_0, such that $\{z_0, z_1, z_2, z_3\}$ are mapped to the corners of the rectangle. Construct ϕ_1 similarly for the polygon Q. Choose the center c_0 of the rectangle R_0 and connect c_0 to all the images $\phi_0(z_k)$. This induces a triangulation of the rectangle R_0. Triangulate the rectangle R_1 in the same way. Then map the corresponding triangles, using the defined ordering on the vertices, using linear maps. Pulling back these linear maps by ϕ_0 and ϕ_1 provides us with the starting map between the polygons P and Q.

2.5.4 The minimization program

An important step in our algorithm is to obtain the infinitesimally extremal Beltrami coefficient in a given class. In other words, we want to discretize the program:

Program 21.

$$\min \quad ||\nu||_\infty$$

$$\textit{subject to}: \int_R \nu\phi_i = \int_R \mu\phi_i \; \forall i \in \{1, 2...n-3\}$$

Here ϕ_i is a basis element of the space of holomorphic quadratic differentials on the punctured sphere R (Equation 4), μ is the Beltrami coefficient of the starting map h, and ν is any measurable Beltrami differential on R.

Program 21 is an L_∞ norm minimization subject to certain constraints. In general, norm is a convex function and so it is reasonable that methods from convex optimization should work.

We will solve the above program when ν ranges over all piecewise constant Beltrami differentials, where by a piece we mean a triangle of the triangulation

obtained in Section 2.5.1.

Discretization of Program 21 Let $\{\nu_i\}_{i=1}^T$, where T is the number of triangles in the triangulation, be a basis of piecewise constant Beltrami differentials. We write $\nu = \sum \lambda_i \nu_i$, and represent ν as a vector $\lambda = (\lambda_1, \lambda_2, \cdots, \lambda_T)$. Each λ_k is a complex number; separating into real and imaginary parts we get $\lambda_k = \lambda_{rk} + i\lambda_{ik}$. Let the analogous vector for μ be λ'. For any $\phi \in A(R)$, we have

$$\int \mu\phi = \int (\sum_{i=1}^{T} \lambda_i' \nu_i)\phi = \sum_{i=1}^{T} (\lambda_i' \int \nu_i \phi). \tag{24}$$

Equation 24 demands a discretization of the pairing between a quadratic differential and a piecewise constant Beltrami differential.

Discretization of the pairing between ϕ and ν_i: There is an analytic formula for the holomorphic quadratic differential ϕ (Equation 4) and so ϕ can be computed on the vertices of the triangulation. Let $[v_i, v_j, v_k]$ represent a triangle with three vertices v_i, v_j and v_k. Define

$$\int_R \mu\phi \approx \sum_{[v_i,v_j,v_k]\in R} \frac{1}{3}(\phi(v_i) + \phi(v_j) + \phi(v_k))\mu_{ijk} A_{ijk} \ ,$$

where A_{ijk} the area of the face. However, since ϕ has simple poles at the boundary, one needs to be careful on triangles around a pole. We will assume that for the purposes of discretization of the integral, a pole is a vertex of the triangulation. Notice that the integral of ϕ over the triangles neighboring a pole can be computed explicitly, as we can fix these triangles once and for all. Multiplying by the corresponding value of ν on each triangle gives the integral over these triangles exactly. One could also use a Gaussian quadrature rule over triangles specialized for functions with a single pole, or use a high-degree quadrature rule within these triangles.

Denote by $\sum_R \mu\phi$ the discrete integral as defined above. We then have

Lemma 22. *Let ϵ be the maximum length of an edge in the triangulation. Then*

$$|\int_R \mu\phi - \sum_R \mu\phi| = O(\epsilon)$$

for all piecewise constant μ and all quadratic differentials ϕ.

Proof. The derivatives of ϕ in the complement of the triangles around a pole are bounded, and by standard Taylor series analysis, the error in this area is $O(\epsilon)$. The integral is evaluated exactly over the triangles having the pole as a vertex, and so the error is 0 there. □

Now we describe how to discretize Program 21.

Let $d = n - 3$ denote the dimension of $A(R)$. Define a $d \times T$ matrix A as $((a_{ij})) = \int \nu_j \phi_i$, where ϕ_i is a basis element. Then the above constraint can be written as $A\lambda = b$, where $b = A\lambda'$. Using the above separation into real and imaginary parts for the matrix $A = A_r + iA_i$ and $b = b_r + ib_i$, this is equivalent to

Definition 23 (Constraints)**.**

$$A_r\lambda_r - A_i\lambda_i = b_r \tag{25}$$

$$A_i\lambda_r + A_r\lambda_i = b_i \tag{26}$$

We introduce another variable $z \in \mathbb{R}^+$, making the number of variables $(2T + 1)$. Let the vector of unknowns be $\beta = (\lambda_{r1}, \cdots, \lambda_{rT}, \lambda_{i1}, \cdots, \lambda_{iT}, z)$.

Program 24.

$$\min \quad z$$

$$\begin{aligned} \textit{subject to} \quad &: \quad \textit{Constraints } 25 \textit{ and } 26, \\ \textit{and} \quad &\lambda_{rj}^2 + \lambda_{ij}^2 - z \leq 0 \quad \forall 1 \leq j \leq T \end{aligned}$$

The last constraint uses the fact that the solution ν^* to Program 21 is of Teichmüller form, and thus $|\nu^*(z)|$ is independent of z. The absolute value of the piecewise constant Beltrami differential ν on triangle j is exactly $\lambda_{rj}^2 + \lambda_{ij}^2$, and the last constraint implies that in the program, z is minimized in parallel with $\lambda_{rj}^2 + \lambda_{ij}^2$. We note that in practice, $\lambda_{rj}^2 + \lambda_{ij}^2$ will not be exactly the same as $\lambda_{rk}^2 + \lambda_{ik}^2$ for different triangles i and j.

Quadratically constrained quadratic program with positive semi-definite matrices: The objective function of Program 24 is linear in the unknown variables. Constraints 25 and 26 are also linear. The last set of constraints can be written as

$$\beta^t P_j \beta - z \leq 0, \tag{27}$$

where P_j is a $(2T+1)$ matrix of zeroes with its $(j,j)^{th}$ and $(T+j,T+j)^{th}$ entry being 1. P_j has all but two eigenvalues as 0, and two eigenvalues are 1, implying that it is positive semi-definite.

Although solving a quadratically constrained quadratic program in general is NP-Hard, positive semi-definite instances of it are polynomial time solvable. Numerical solvers for these programs have been vastly studied, and efficient implementations exist. We refer the reader to Page 42 of [87] for a complete reference.

2.5.5 Discretization of the vector field method

The idea of deforming a map via a vector field has been used in the computer graphics and vision community before and is fairly straightforward. We do not elaborate more and refer the reader to the LDDMM method available online [1].

2.5.6 Choosing t during the algorithm

Theorem 19 gave a "good" value of t which ensures a substantial decrease in dilatation at every step. However, in order to use Equation 20 in the statement of Theorem 19, we need to find C such that $|\mu_g - \alpha_t| \leq Ct^2$ when $t < \delta$. Assume that $\delta = 1/2$. We choose a small $\ell > 0$ and discretize the interval $[0, 1/2]$ by placing the points $p_i = i\ell$, $0 \leq i \leq 1/2\ell$.

Given p_i, form the Beltrami differential $\alpha_i = p_i(\mu_h - \nu_h)$, and obtain the self homeomorphism g_i using Section 2.4.2. Compute the Beltrami coefficient of g_i, denoted as μ_g^i. Then compute $|\mu_g^i - p_i(\mu_h - \nu_h)|$ and take the maximum over all points in the interior of the punctured sphere/polygon. Denote this maximum by x_i.

Define $C = \max_{0 \leq i \leq 1/2\ell} x_i/p_i^2$. Now use Equation 20 to choose a value of t, and use it in steps 1 and 2 of the method summarized in Section 2.4.1.

2.6 Future work: complexity and approximation

Recall that in Theorem 20 in Section 2.4.5, we showed that the continuous construction converges to the unique extremal map. We now discuss the number of iterations, and the convergence of the discretized approach.

For the discrete implementation, at every iteration we solve a positive semi-definite quadratically constrained quadratic program. To the best of our knowledge, most of the methods are a variant of Karmarkar's algorithm [53], which runs in polynomial time. Thus, if our number of iterations were polynomial, our entire method would be polynomial time.

In both problems , the decrease of the dilatation at every iteration is governed by Theorem 19. If the C involved in Theorem 19 can be bounded independent of N—the current iteration (a scenario we feel is very likely), then using the principle of Teichmüller contraction, the decrease in one step can be seen to be $O(k - k^*)^4$, where k is the dilatation of h_i (the map at iteration i), and k^* is the dilatation of the extremal map.

If the discrete procedure converges, it terminates by producing a piecewise constant Beltrami coefficient which is a "fixed point" of Program 24. The space of piecewise constant Beltrami coefficients is dense in the space of all measurable (L^∞) Beltrami differentials, and one can expect that this fixed point is close to the Beltrami differential of the extremal map. Exact error analysis will then help us figure out the constant involved (if any) in our algorithm.

Chapter 3

Exploring Path Space Using Sensor Network Geometry

3.1 Introduction

Scalable routing on a sensor network has been an active research topic for the past ten years. The major challenge comes from the fundamental resource limitation of sensor nodes, in terms of storage size and communication bandwidth. The solution that requires a node to acquire the entire network topology does not scale well. In the past few years there have been a number of innovative proposals on scalable routing schemes where each node only keeps local information and a routing path can be discovered by iteratively applying greedy routing decisions. Such work has mainly focused on issues such as guaranteed delivery and low path stretch, and has been relatively successful in that regard.

In this work we move on and focus on more advanced communication primitives, in particular, routing schemes that exploit the existence of multiple paths in a sensor network. Between a source and a destination there can be multiple different paths. A single path from source to destination may give limited throughput due to bandwidth constraints, hop length, wireless interference or other transmission failures. If there is a lot of data to be delivered, it is natural to consider using multiple disjoint paths[1] such that different data segments can be simultaneously delivered to

[1] In this work we focus on node disjoint paths as wireless communication is by nature broadcast.

the destination. With multipath routing one obtains higher throughput and lower delay. Such multipath routing can also be used to enhance data security. For example, sensor data can be encoded such that different codewords are sent along different paths. Therefore a single compromised node stays on at most one path and with its captured data segments it is unlikely to reconstruct the original data.

Exploring the space of routing paths between two nodes is also helpful for fast recovery from link or node failures. In a large wireless network, there are network changes of different scales. At the node level, wireless links have high link quality variation. Nodes may fail. Interference with other nodes could also be unpredictable, e.g., as in the hidden terminal problem. At a large scale, communication links in a region can be temporarily disabled by jamming attacks, either imposed by malicious parties [128], or as a result of co-located multiple benign wireless networks interfering with each other. For example, experiments have shown that 802.15.4 sensor network interferes with existing WiFi network resulting in 54% packet loss [81]. In case of a transmission failure, it would be good to quickly discover an alternative path to the destination. A single isolated link failure can possibly be bypassed by local random detouring attempts. A large scale node or link failure, in particular one with strong spatial correlations and a geometric pattern, would need some non-trivial exploration of the path space – by making possibly a big detour from the planned path. See Figure 8 for an example. To allow such robustness and quick response to network conditions, routing schemes that find a single path are not enough and it is important to understand the 'space' of paths and efficiently navigate within this space.

Finding multiple routings in a network is a challenging problem for a large scale network, in particular if a node does not have the entire topology. On the theoretical side, one can run a flow algorithm to find a maximum number of node disjoint paths between source and destination. But the flow algorithm requires centralized knowledge and also has a high computational cost of $O(n^3)$ if the network size is n. Even if one can afford to pre-compute multiple routing paths, storing these paths at the sensor nodes will be, storage-wise, too overwhelming, which makes the centralized algorithms scale poorly with the size of the network. In literatures on

Two paths that share the same intermediate nodes may experience inter-path interference where the common node is the bottleneck.

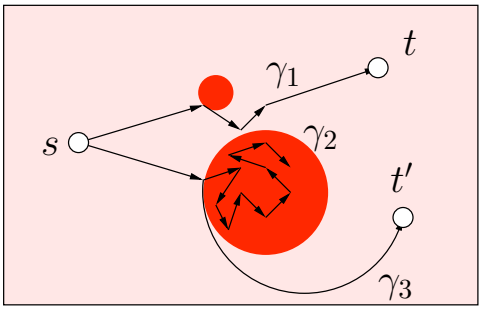


Figure 8: Consider some part of the network experiencing heavy inference (or jamming attacks), shown as the dark colored circles. Links inside these 'failure' regions have much higher loss rate. A route that hits a small failure region might be able to get around by performing some random walks in the neighborhood, as in the case of path γ_1. A route that hits a large failure region has difficulty recovering from it – as simple random walk is likely to wander around for a long time, as shown by the path γ_2. In this case a path that makes big de-tours would perform much better, as shown by the path γ_3.

mobile ad hoc networks, there have been a number of heuristic proposals to find multiple paths but the theoretical understanding of these schemes rarely exists. If one wants to use multiple paths to recover from en-route node or link failure, there is not much understanding (globally) on where the second path is going to be.

Our Approach In this work we approach the problem from a geometric angle. Since nodes are typically densely deployed in a geometric domain, the network topology is not like a general graph. We explore different embeddings of the network such that by controlling a *constant* number of parameters we can quickly switch between different network embeddings such that greedy routing in such embeddings generates different routing paths. Thus one can easily come up with multiple node disjoint paths, or even switch to an alternative path in the middle of a routing process, by spontaneously changing to a different embedding.

We are motivated by the recent development of conformal mapping of a sensor network [101, 102]. We first compute an embedding of a sensor network such that all the holes are deformed to be circular. We name this embedding to be a *circular domain*. On a circular domain, greedy routing that always delivers the message to

the node closer to the destination using the new coordinates guarantees message delivery. However, the embedding as a circular domain is not unique, and all such embeddings differ by Möbius transformations, which maps a complex plane to itself and can be represented by

$$f(z) = \frac{az + b}{cz + d},$$

where z is a complex variable and a, b, c, d are four complex numbers satisfying $ad - bc = 1$ (see Section 1.1.1 in Chapter 1 for details). A Möbius transformation always maps circles to circles. Thus applying a Möbius transformation on a circular domain essentially 're-arranges' the positions and the sizes of the circular holes and the new embedding remains to be a circular domain. Therefore there are actually *infinitely* many circular domains on each of which greedy routing guarantees delivery. The previous work as in [101, 102] only considered one such circular domain by fixing one hole to be at the center of the network. In this work we investigate all possible circular domain embeddings and the applications to multipath routing in a sensor network.

The main difficulty for efficient and scalable routing in a sensor network is due to lack of the global knowledge. Embedding the network as a circular domain makes that difficulty go away in some sense. Our routing scheme avoids the requirement for the global network topology, while the geometric information we need — the locations and shapes of the holes and the boundary – are typically of a constant size and usually remain stable. With a circular domain one can predict where the path is (subject to the assumption that the sensor nodes are sufficient dense so that the continuous path is a good approximation of the discrete path by greedy routing) and by applying a Möbius transformation we know the kind of path we will get and how different it is from the previous one. Since a Möbius transformation only uses four parameters, we can attach the parameters of the current Möbius transformation at the packet such that by applying the Möbius transformation a node can compute its coordinates under the transformation on the spot to generate the greedy path under the new embedding. In case of a link failure on the current greedy path, a node can generate a new Möbius transformation and switch to a different path immediately. The new Möbius transformation is simply attached to the packet.

Using a circular domain representation gives us the following advantages that

will be proved in the continuous case and evaluated by simulations for the discrete network setting:

- By using different Möbius transformations one generates multiple paths to the destination that are disjoint except at the source and the destination. We present algorithms for networks with or without holes.
- In case of node failures, we present an algorithm that identifies a different path to the destination. The second path takes a big circular arc type of detour that is likely to jump over correlated failure regions.

In the following we first review prior work on multipath routing. We present the theoretical proofs of our method and present simulation results afterwards.

3.2 Related work

In this section we briefly review prior work on three relevant topics: multipath routing both in theory and in practice; some of them focus on how to recover from node or link failures; and previous greedy routing schemes.

Multipath routing. Multipath routing has been investigated extensively in computer networking in order to improve routing robustness [11, 24], achieve better load balancing [26, 114, 130], reduce network congestion, reduce end-to-end delay [136] and increase network throughput [44, 123]. Between a pair of source and destination, multipath routing looks for multiple paths that are sufficiently different from each other such that node or link failures will not destroy all of them. One formulation is to look for k node disjoint or edge disjoint paths, which can be computed by flow algorithm [22]. But this is a centralized algorithm and would require the knowledge of the entire network [43]. Distributed algorithms only exist for special case of $k = 2$. In [90] two colored trees were constructed for routing such that the paths in the two trees are link or node disjoint. Relaxation of the node/edge disjointness of the multiple paths leads to the approach of braided multipath [33] in which the multiple paths are only partially disjoint.

In a mobile ad hoc network, multipath routing has also been developed to enhance the performance of on-demand routing protocols such as AODV [5,18,60,76,

93] or DSR [71,82,105] as the network topology undergoes constant changes. Prior work in this direction uses extensive message exchange or flooding to discover alternative paths to bypass a broken link. A major problem of these schemes is that they suffer from high recovery delay from node or link failures, which severely affects the performance of end-to-end QoS measurements in the transport or application layer.

Fast recovery from failures. Recently there have been a number of interesting works that study the problem of fast recovery from link or node failures, even for a centralized situation. When a link or node fails, the goal is to quickly discover an alternative path with nearly no delay, such that the current traffic is not interrupted. For the intra-domain routing protocols on the Internet, the recent IP fast re-routing (IPFRR) schemes (Loop-free alternate (LFA) [8], O2 [95, 96, 104], DIV-R [92], MARA [84] and protection routing [59]) aim to ensure fast re-covergence when node failures are detected. In general this family of work would like to find an alternative next hop when the intended next hop is not reachable. Depending on the detailed implementations, the design often suffers from one or more of the following problems: having possible transient loops, the requirement for a lower bound on node degree, computational intractability (e.g., verifying whether a graph has a protection routing or not turns out to be NP-hard [59]).

Our work is motivated by routing with multiple metrics as introduced in the path splicing idea [80], which is proposed for increasing routing reliability on the Internet. Given a weighted graph, one perturbs the weights of the edges and computes a shortest path tree on each node. These multiple shortest path trees are used in combination to generate a routing path in case of in-transit link failures. Traffic in the network can freely switch between different shortest path trees, which results in a large number of routing paths (these paths are the braided multi-paths). The overhead of switching between different trees is done by just changing a few bits in the packet header. This supports fast recovery from link or node failure and ensures low end-to-end delay. However, this is mainly for inter-domain routing on the Internet. The computation of the multiple shortest path trees is too costly for a large scale sensor network. For sensor network setting we need to have a low cost method to generate multiple metrics with great flexibility and path diversity.

Greedy routing. Our technique uses greedy routing on different network embeddings, or different metric spaces. Each of the embeddings has the property that all the holes are circular – thus delivering the message towards the neighbor closest to the destination can always get to the destination[2] [101]. In the past few years various greedy routing schemes have been proposed by using a proper embedding of the graphs [6,28,38,57,64,86,91,101,102,134]. Most of these works only focus on finding a single route to the destination [6, 28, 38, 57, 64, 86, 91, 101, 102]. In this work, we are dealing with a more sophisticated situation–multipath routing in a vulnerable sensor network. Zeng *et al.* [134] considered embedding the sensor network in the hyperbolic covering space such that a network is mapped to multiple copies glued to each other properly; greedy routing to the image of the destination in different copies will lead to homotopy different paths (i.e., these paths get around the network holes in different ways). In some sense this is also a type of multipath routing except that the multiple paths are required to be homotopy different and are not necessarily node disjoint. For the same homotopy type only one path is generated by the greedy algorithm. In our work, even for the same path homotopy type we want to get multiple node disjoint paths. We also handle dynamically appearing 'holes' or link failure regions, while the previous work in [134] assumes all the holes are given and the embedding is computed with respect to that.

For greedy routing using virtual coordinates, typically a location service is available to translate node ID to the virtual coordinates. We assume the same setting.

3.3 Algorithms

Our routing algorithm tries to find various embeddings, or mappings from the original sensor network to a circle domain, in which all holes are of a circular shape and greedy routing can guarantee delivery. Such mappings are conformal (angle preserving) and computed by Ricci flow. Then Möbius transformations could help us to find more embeddings, or controllable multiple metrics.

We first review the previous work [101] of deforming a sensor network shape

[2]This is subject to a small caveat that in certain cases routing on an 'edge' might be needed.

to make all boundaries circular in section 3.3.1. Then we describe Möbius transformations applied on such a circular domain. The algorithms for generating multiple node disjoint paths and for loop-free fast recovery from node failure are presented in section 3.3.2 and section 3.3.3 respectively.

3.3.1 Embedding into circular domains with Ricci flow

Conformal Mapping. In the continuous setting for Riemannian surfaces, let (S_1, g_1) and (S_2, g_2) be two surfaces with Riemannian metrics g_1, g_2. A mapping $\phi : S_1 \to S_2$ is called a *conformal mapping*, if the intersection angle of any two curves is preserved.

A planar domain D of connectivity m is called a *circular domain*, if all its m boundaries are circles. It is known from conformal geometry that any genus zero multiply connected planar domain can be mapped to a circular domain by conformal mappings. Such a mapping is not unique: all such mappings differ by Möbius transformations [25, 89].

To compute the conformal mapping from a surface to a circular domain, one can use Ricci flow as introduced in [48, 101]. In the case of sensor network setting, we will use the discrete version, which represents a domain by a discrete triangulation. We first give some references on how to obtain such a triangulation from a sensor network setting and then move on to the algorithm description of the discrete Ricci flow.

Sensor Network Triangulation. To apply discrete Ricci flow we require a triangulation of a planar domain. There have been quite a number of prior results on extracting a triangulation from the sensor network communication graph. We quickly go through such results. Karp *et al.* [54], Bose *et al.* [16], Gao *et al.* [34], and Li *et al.* [66] proposed distributed, localized methods to extract a planar graph from a unit disk graph. Such a planar graph can be considered as a triangulation of the planar domain where non-triangular faces are considered as network holes. Sarkar *et al.* [101] extended the algorithm in [34] for the case of a quasi-unit disk graph and also showed that sufficiently big holes are indeed captured as real holes in the triangulation. Funke *et al.* [31] worked on the case when node locations are not available and proposed a triangulation method for a quasi-unit disk graph.

For a general case when no unit disk graph or quasi-unit disk graph assumptions are available, the algorithms in Cross Link Detection Protocol proposed by Kim *et al.* [55, 56] produce a planar graph. Zhang *et al.* [135] proposed to use matching to eliminate crossing edges and produce a planar graph. This algorithm also does not require node locations.

In fact, to run Ricci flow algorithm and find the circular domain embedding, one does not require the triangulation to be a *subgraph* of the communication graph. The triangulation can be a virtual graph of the underlying domain as long as the sensor nodes jointly maintain it. In other words, each wireless node takes in charge of a group of virtual vertices of the triangulation and carries out the computation for these virtual nodes. The position of a wireless node can be made identical to any node it is in charge. This idea of 'virtualization' allows the method to be extended to a general setting when the wireless node is relatively powerful and the distribution is sparse (as a natural consequence since fewer nodes can sufficiently monitor the domain). Therefore the triangulation can be any proper triangulation of the underlying geometric domain, for example by standard meshing techniques such as Delaunay refinement methods [98, 108]. In the following discussion we assume that a proper triangulation is obtained and we focus on the embedding of the network from now on.

Discrete Ricci Flow. In the following we explain Ricci flow in the discrete setting for a triangulation of a domain with m holes. The triangulation is denoted by Σ with vertex set V, edge set E and face set F.

In the discrete setting we define a Riemannian metric by using the edge lengths on Σ:

$$l : E \to \mathbb{R}^+,$$

such that for a triangle face f_{ijk} with vertices v_i, v_j, v_k, the edge lengths satisfy the triangle inequality:

$$l_{ij} + l_{jk} > l_{ki}, \quad \forall i, j, k.$$

The lengths of the edges of the triangulation determine the corner angles of the triangles. For a triangle f_{ijk} with edge lengths $\{l_{ij}, l_{jk}, l_{ki}\}$, and the angles opposite to these edges $\{\theta_k^{ij}, \theta_i^{jk}, \theta_j^{ki}\}$ respectively, we have the following equations by cosine

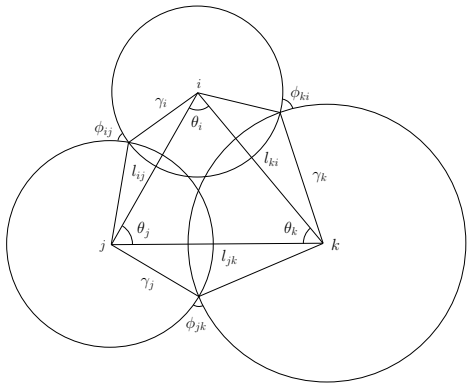


Figure 9: The circle packing metric.

law:

$$l_{ij}^2 = l_{jk}^2 + l_{ki}^2 - 2l_{jk}l_{ki}\cos\theta_k^{ij}. \tag{28}$$

Now we can define the discrete Gaussian curvature at a vertex v_i as the angle deficit:

$$K_i = \begin{cases} 2\pi - \sum_{f_{ijk}\in F}\theta_i^{jk}, & v_i \text{ is an interior vertex;} \\ \pi - \sum_{f_{ijk}\in F}\theta_i^{jk}, & v_i \text{ is at boundary.} \end{cases} \tag{29}$$

where θ_i^{jk} represents the corner angle at vertex v_i in the triangle f_{ijk}. In other words, the curvature at a vertex v is the difference of 2π or π and the total corner angles at v, for an interior vertex or a vertex on a hole boundary respectively. The curvature is 0 when it is locally flat (for interior vertices) or locally a straight line (for boundary vertices).

Ricci flow uses the circle packing metric in the discrete case, proposed by [111, 119], to approximate the conformal deformation of metrics. See Figure 9. Each vertex v_i has a circle with radius γ_i. On each edge e_{ij}, ϕ_{ij} is defined as the intersection angle of the two circles at v_i and v_j. The pair of vertex radii and intersection angles at the edges on a mesh Σ, (Γ, Φ), is called a *circle packing metric* of Σ. Two circle packing metrics (Γ_1, Φ_1) and (Γ_2, Φ_2) on Σ are *conformal equivalent*, if $\Phi_1 \equiv \Phi_2$. Therefore, a conformal deformation of a circle packing metric only modifies the vertex radii γ_i's and maintains the intersection angles ϕ_{ij}'s to be constant.

For a given mesh, the circle packing metric (Γ, Φ) and the edge lengths can be

converted to each other by cosine law as below:

$$l_{ij}^2 = \gamma_i^2 + \gamma_j^2 + 2\gamma_i\gamma_j \cos\phi_{ij}. \tag{30}$$

Thus, given a circle packing metric, we can calculate the edge lengths of the triangulation Σ and then the embedding in the plane realizing the given curvatures.

Let $u_i = \log\gamma_i$ for each vertex. The discrete Ricci flow is defined as the following differential equation:

$$\frac{du_i(t)}{dt} = (\bar{K}_i - K_i), \tag{31}$$

where K_i is the current curvature at vertex i and $\bar{K}_i$ is the target curvature at i. Define an energy function

$$f(\mathbf{u}) = \int_{\mathbf{u_0}}^{\mathbf{u}} \sum_{i=1}^{n} (\bar{K}_i - K_i) du_i, \tag{32}$$

as the *Ricci energy*, where $\mathbf{u_0}$ is an arbitrary initial metric. It has been proved by Chow and Luo [20] that the discrete Ricci flow will converge to a unique minimum of the Ricci energy. The convergence rate of the discrete Ricci flow using Equation 31 is shown to be exponentially fast, i.e.,

$$|\bar{K}_i - K_i(t)| < c_1 e^{-c_2 t}, \tag{33}$$

where c_1, c_2 are two positive constants.

The Ricci flow algorithm is naturally an iterative algorithm with all vertices adjusting local metrics and local curvatures. All the radii at the vertices are initialized to be $1/2$. That is, the circles at adjacent vertices of Σ are kept to be tangent to each other. We set the target curvature to be 0 at all interior vertices. That is, the network should be embedded to be flat in the domain. We set the target curvature at a boundary vertex to be $-2\pi/k$, if the boundary of the hole has a total number of k vertices. That is, the boundary circle should be perfectly circular. We apply the Ricci flow algorithm by changing the circle packing metric, u_i, by $\delta(\bar{K}_i - K_i)$, where δ is a constant parameter as the step size. The algorithm stops when the

current curvature is within an error bound of ϵ from the target curvature.

Since the curvature error decreases exponentially fast, the number of steps in the Ricci flow algorithm is in $O(\frac{\log(1/\epsilon)}{\delta})$, where δ is the step size in the Ricci flow algorithm. The total number of messages is thus in $O(\frac{n\log(1/\epsilon)}{\delta})$, if the algorithm is running on a network of n vertices.

In the sequel, Möbius transformations will be the main tool we use. Please see 1.1.1 in Chapter 1 for details.

3.3.2 Multipath Routing

In this section we describe how to generate multiple paths from a given source node s to a given target node t. We will give different embeddings of the domain in such a way that the route used by greedy routing in one embedding is likely to be different from the one used by greedy routing in any other embedding, and these routes are 'well spaced', a notion we will make precise soon. The algorithm we will present generates paths that are provable to be disjoint in the continuous case. In the discrete setting, we evaluate the performance by simulations.

We first present the algorithm for a network without holes. Then we discuss how to find disjoint paths in a network with holes.

3.3.2.1 Network Without Holes

Without loss of generality we can assume that the outer boundary is a circle, and that the coordinates of source s are $(-d/2, 0)$ while those of destination t are $(d/2, 0)$, so that the line segment joining s to t is horizontal and of length d. This can easily be achieved by a rotation and translation. We denote the domain by D.

Consider a continuous domain D we can easily generate many disjoint paths by essentially applying a different Möbius transformation each time. The greedy path under a Möbius transformation turns out to be a circular arc connecting s and t in the original domain D. See Figure 10. In the discrete case when the domain D is represented by a triangulation, the routing paths are found by using greedy routing in different embeddings, after proper Möbius transformations. We remark that potentially one can design greedy routing to follow any curve, i.e., as in the idea of routing along a curve [83]. But in general routing on a curve does *not* have

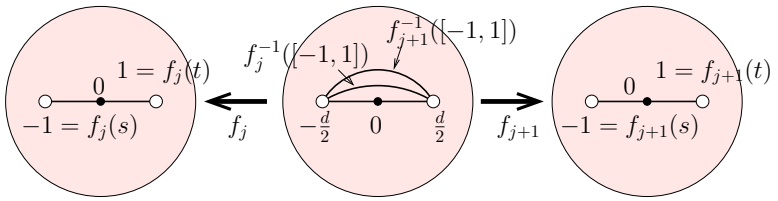


Figure 10: The multiple paths on the domain D (in the middle) are the greedy paths under transformations f_j. The figure shows two transformations f_j and f_{j+1} respectively.

any guarantee on the delivery. In our case, as we actually perform greedy routing in another circular domain after a proper Möbius transformation, this immediately implies that such a route is guaranteed to reach the destination.

In a continuous domain D, all such circular arcs are disjoint except at source and destination. In the discrete case the greedy paths are guided by the circular arcs but can definitely deviate from them due to discrete node distribution. Since typically sensor networks have upper bounded density, a major constraint on the number of node disjoint paths between s and t is due to the degree at s and t. A reasonable heuristic to minimize overlaps of multiple paths is to design paths that are evenly spread out at s and t. We use this heuristic to design our paths in the discrete setting.

Given two curves γ_1 and γ_2 joining s to t, we can define initial and final *directional spread* between γ_1 and γ_2 to be the angle between the tangent vectors of γ_1 and γ_2 at s and t respectively. We denote these by $d_i(\gamma_1, \gamma_2)$ and $d_f(\gamma_1, \gamma_2)$ respectively. See Figure 11 for an example. In the following we use Möbius transformations to generate circular arcs connecting s and t such that their directional spreads at source and destination are as even as possible.

For a given $k \geq 1$, let $\theta = \frac{\pi}{2k}$ and define $\theta_j = \frac{\pi}{2}(1 - \frac{j-1}{k})$ for $1 \leq j \leq 2k+1$. Also, let $\alpha_j = d/2\tan(\theta_j/2)$ for $1 \leq j \leq 2k+1$.

We next define a Möbius transformation $f_j(z)$ for $1 \leq j \leq 2k+1$ by

$$f_j(z) = \frac{zd - id\alpha_j}{z(-2i\alpha_j) + d^2/2}. \tag{34}$$

Theorem 25. *f_j has the following properties:*

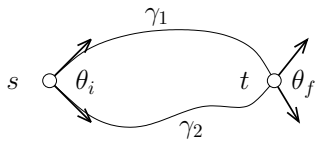


Figure 11: For two curves γ_1 and γ_2 from s to t, the initial directional spread is shown as θ_i and the final directional spread is shown as θ_f.

1. $f_j(s) = -1, \quad \forall 1 \leq j \leq 2k+1.$
2. $f_j(t) = 1, \quad \forall 1 \leq j \leq 2k+1.$
3. *Let $\gamma_j = f_j^{-1}([-1,1])$. Then γ_j is a curve joining s to t. Moreover, it is the arc of the unique circle passing through s and t, such that the tangent vectors at s and t both make an angle of θ_j with the x-axis.*
4. $d_i(\gamma_j, \gamma_{j+1}) = d_f(\gamma_j, \gamma_{j+1}) = \theta, \quad \forall 1 \leq j \leq 2k+1.$

Proof. (1) and (2) are proved by calculating $f_j(s)$ and $f_j(t)$ according to Equation 34.

To prove (3) we use the property that Möbius transformations map circles to circles. Note that the point $(0, \alpha_j)$ is represented by the complex number $z = i\alpha_j$. One can check that $f_j(i\alpha_j) = 0$. Since $-1, 0$ and 1 lie on a line, it means that s, $(0, \alpha_j)$ and t lie on a circle. Therefore $f_j^{-1}([-1,1])$ is an arc of the circle passing through these three points. One has to now verify that the tangent to this (unique) circle at s and t makes an angle of θ_j with the horizontal axis.

(4) follows from (3) and the fact that $\theta_{j-1} - \theta_j = \theta$. □

What the above calculations mean is that if we define a new embedding of the domain D by mapping a point $z \in D$ to $f_j(z) \in f_j(D)$, then the source maps to -1 and the target maps to 1. In this new embedding, the shortest path from source to target is simply the straight line from -1 to 1. Following this path for some j is equivalent to following the arc of the unique circle passing through s and t whose tangents at s and t make an angle of θ_j with the horizontal axis. Any two such arcs have an initial and final directional spread of at least $\theta = \frac{\pi}{2k}$. Hence we have

generated $2k + 1$ node disjoint θ spread paths from s to t. See Figure 10 for an example.

All such paths lie in the circle with the line segment $\overline{st}$ as its diameter. We can also consider the case $\theta_j = \frac{\pi}{2}(1 + \frac{j-1}{k})$ to get $2k - 2$ more node disjoint paths, with an angle spread of θ, getting $4k - 1$ in total. For example if $k = 3$, we can get a total of 11 paths, such that the directional difference is at least $\frac{\pi}{6}$.

We remark that the above results hold only for source s and target t pairs for which the circle with line segment $\overline{st}$ as diameter, denoted as C_{st}, is contained inside the domain (i.e., in the interior of the outer circle C). When this is not the case, the paths will 'hit' the outer boundary circle. Such paths will merge as they follow along the outer boundary and are hence not disjoint. However, this is actually a case that will be handled by the algorithm in the following section as the outer boundary is also a topological hole.

The above analysis is done in the continuous setting. We will present the multiple path routing results in a discrete setting by simulations.

3.3.2.2 Network with holes

Consider a circular domain D with k holes (including the outer hole, which can be regarded as a circle centered at ∞). In this case, finding disjoint paths is more complicated. This is precisely because if two paths both hit the same hole, they will start to follow the boundary of the hole and converge. This will create a long shared sub-path on that boundary. Therefore we would need to find paths such that either (i) they do not hit the same hole, or, (ii) when two paths hit the same hole, they hit the top and bottom of the circular hole respectively so they follow the upper boundary and lower boundary and do not converge. Take a look at Figure 12. For each hole C_i in the domain, we take three circular arcs through s and t, the one that is tangent to C_i internally (i.e., including C_i); the one that goes through the center of C_i; and the one that is tangent to C_i externally (i.e., excluding C_i). These three paths are denoted as γ_i^+, γ_i, γ_i^-, respectively. Now, any two circular arcs through s and t that fall in between γ_i and γ_i^+ (or γ_i and γ_i^-) will definitely merge on the boundary of C_i. Thus we can only allow one path selected within each angular range bounded by $[\gamma_i^+, \gamma_i]$, and $[\gamma_i, \gamma_i^-]$. In the following we present an algorithm that finds a maximum number of hole touching paths satisfying the

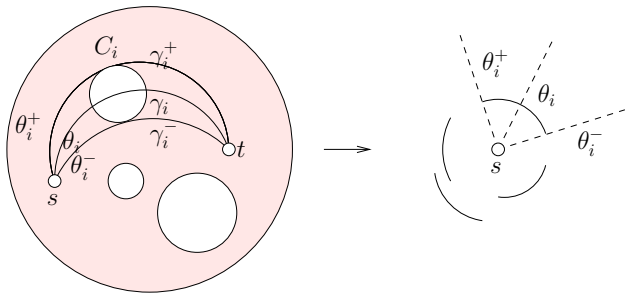


Figure 12: For a pair of source and destination, each hole C_i will produce two intervals θ_i^+ and θ_i^- such that any two paths falling in the same interval will hit the hole and share some segments of the boundary. Thus any set of disjoint paths can only select one path inside each interval.

above constraints.

Given a source s and a destination t, we can first assume without loss of generality that the x coordinate of t is larger than that of s and that there does not exist a circle which passes through s and t which is tangent to more than one of the holes. For a hole C_i with center c_i, there are three arcs of relevance: γ_i, γ_i^+ and γ_i^-, as defined earlier. Let θ_i,θ_i^+ and θ_i^- denote the angles that the initial tangent vectors to γ_i, γ_i^+ and γ_i^- make with the horizontal axis. Note that the convention is that all of them are contained in the interval $[-\pi, \pi]$.

Now we define the following angular intervals $\{T_i\}_{i=1}^{2m}$, two for each of the m holes: $T_{2h-1} = [\theta_i^-, \theta_i]$ and $T_{2h} = [\theta_i, \theta_i^+]$ ($1 \leq h \leq m$). For the outer hole, arc γ_i is simply the straight line joining s to t; while arcs γ_i^+ and γ_i^- are contained in circles that pass through s and t and are tangent to the outer boundary at the top and bottom.

Now any two circular arcs joining s to t, both of whose initial directions lie in the same interval T_{2h-1} or T_{2h}, will both traverse either the upper or the lower boundary of the hole C_i, and hence cannot be disjoint. We want to find the maximum number of arcs, all of which pass through s and t, and touch at least one hole.

Since all the T_i's lie in $[-\pi, \pi]$, we can think of them as subsets of the unit circle S^1. We can angularly sort the endpoints, to obtain a sequence $s_1, s_2, ..., s_{4k}$

where each s_i is an endpoint of T_j for some j. If some interval $[s_i, s_{i+1}]$ is not contained in any T_j, we are free to use it as there are no constraints associated to such an interval. Assume that we have collapsed all the $[s_i, s_{i+1}]$ that are not contained in any T_j, and now we are left with a sequence $s_1, s_2, ..., s_l$.

Now we want to find a maximum number of intervals such that any two intervals cannot be part of the same T_i for any i. That is, each interval in the solution can be used to generate one circular arc and all such circular arcs do not intersect with each other except at s or t.

The above problem has an optimal solution using a greedy algorithm that we now describe. Each interval $A_i = [s_i, s_{i+1}]$ now is contained in (covered by) some T_j's. We obtain a solution Q_i as follows. Q_i starts with a seed interval A_i. Move clockwise on the circle until the first interval $[s_{j_1}, s_{j_1+1}]$ which is not covered by any T_j that covers A_i. When this happens, include A_{j_1} in the solution Q_i and proceed greedily until we cannot include any more intervals to Q_i.

After this process, we have l solutions $Q_1, Q_2, ..., Q_l$. We choose the one with a maximum number of intervals. The next theorem asserts that this solution is optimal.

Theorem 26. *The number of intervals in the solution chosen above, i.e., the best amongst the $\{Q_i\}_{i=1}^{l}$, is equal to the number of intervals chosen in the optimal solution.*

Proof. Pick any interval that the optimal solution chose, say A_{j_1}. Consider what the greedy algorithm performed in Q_{j_1}. Let the next interval chosen by the greedy algorithm be G_{j_2} while the one chosen by the optimal be O_{j_2}. Assume they are different (if they are the same the argument proceeds). If there does not exist j such that G_{j_2} and O_{j_2} are both contained in T_j, then adding the interval G_{j_2} to the optimal solution increases the number of intervals in the optimal, which is a contradiction. So assume T_j contains both G_{j_2} and O_{j_2}. Therefore by the end of T_j we have not done any worse than the optimal, as both have added one interval each. The argument now proceeds in a similar fashion. G_{j_3} and O_{j_3} would both have to be contained in some T_j again, by the end of which we are again no worse than the optimal and so forth. Inductively we can show that our solution is no worse than the optimal and thus must be optimal. □

Using this greedy method we can thus find a maximum number of node disjoint paths in the domain all of which pass through a hole. We can use results of the previous section to generate node disjoint θ spread paths in the intervals $[s_i, s_{i+1}]$ which were not covered by any T_j. Thus putting together, we can find a maximum number of disjoint paths that do touch some hole and depending on the spread we can find disjoint paths that do not touch any hole using previous results.

To summarize, our multipath algorithm will generate k disjoint paths in a network with and without holes by applying different Möbius transformations, with provable results for the continuous case. When the nodes have high density, the greedy paths in the discrete case will better approximate the circular arcs. When the node density drops, the multiple paths may overlap in the middle. We evaluate in the simulation section the dependency of the performance on the network density.

3.3.3 Recovery From Failure

In this section we describe how to deal with en-route node failures or link failures. Recall that our embedding produces a circular domain that guarantees delivery when links are assumed to be reliable. When a link may fail, greedy routing no longer guarantees delivery. For example, a node may discover that all the neighbors that are closer to the destination are not reachable. In this case we aim to find an alternative path. The freedom of applying Möbius transformations on a circular domain provides great flexibility for this task.

Assume s is the source node that wants to transmit a package to t ($s, t \in D$). Let the degree of s be ν. Furthermore, assume that s has sorted its neighbors in increasing order of their distances from t such that

$$||p_1 t|| \leq ... \leq ||p_k t|| \leq ||st|| \leq ||p_{k+1} t|| \leq ... \leq ||p_\nu t||.$$

$||uv||$ is the Euclidean distance between u, v. If the link between s and any of the $\{p_i\}_{i=1}^{k}$ is functional, s routes the message to that neighbor just as in greedy routing. Assume that the only links available to s are those in $\{p_i\}_{i=k+1}^{\nu}$. Then s picks a functional link from this set, say the link to $p = p_{k+1}$. Now the idea is to find a Möbius transformation such that in the new embedding, p is closer to t than s,

so that greedy routing would then continue by using p as the next hop. The details are presented below.

As soon as node s finds that greedy routing can no longer continue, it does the following:

1. It finds the coordinates of a neighbor p. Assume that s knows the coordinates of p and the destination t.

2. s finds a Möbius transformation that maps s to -1, p to 0 and t to 1. The explicit formula for f is

$$f(z) = \frac{z(s-t) + p(t-s)}{z(2p-(t+s)) - p(t+s) + 2st}.$$

3. s then sends the packet to p along with the information about this Möbius transformation. p calculates new coordinates for all of its neighbors and for t.

4. Greedy routing then continues (since now $f(p)$ is clearly closer to $f(t)$ than $f(s)$), until we get stuck at another node. When this happens, we repeat the entire process, i.e., find another Möbius transformation and compose it with the previous one.

As will be shown later in the simulation section, our failure recovery mechanism is compared with random walk – simply pick a random 'live' link until greedy routing can be performed again. Basically our scheme makes big jumps and chooses a vastly different alternative path while random walk can only make local adjustments. This benefit of using long de-tours is significant for failures that exhibit spatial patterns.

3.4 Simulations

In the experiments, we perform greedy routing with Möbius transformations to achieve multipath routing and link failure recovery. Our simulations are performed on unit disk graph topologies potentially with holes inside, and the following are our key observations:

Multipath routing: By using Ricci flow with different Möbius transformations, we can generate a substantial fraction of node disjoint paths. With reasonable sensor density (average degree around 20), the average number of disjoint paths we find using our algorithm is consistently more than 70% of the input parameter m (the desired number of disjoint paths). We can consistently find two node disjoint paths even in very sparse networks. We also observed that when the network is sparse, the bottleneck for finding node disjoint paths is often near the source and destinations.

Recovering route from link failures: Under a spatial failure model in which the nodes in a geometric failure region have a much higher failure rate, our method of using greedy routing on the virtual coordinates in a circular domain with Möbius transformations as the recovery scheme performs consistently better than all other methods (on virtual or original coordinates, using random walk as the recovery scheme). The advantage of using Möbius transformations rather than random walk as a recovery scheme diminishes when the failure pattern is no longer spatially correlated.

3.4.1 Multipath routing

After we generate the sensor network $G = (V, E)$, we randomly choose two vertices s and t from V as the source and the destination. We then calculate the maximum number of node-disjoint paths between s and t, called the vertex connectivity $\kappa(s, t)$, as a reference for comparison, using the centralized maximum flow algorithm [22]. To test our multipath routing algorithm, we generate m (no larger than $\kappa(s, t)$) paths from the source to the destination, and count how many of them are node-disjoint. We also try different m parameters to further observe the performance of the algorithm.

Figure 13 shows the proposed routing scheme on a sensor network with 1000 vertices and 10006 links. We first apply Ricci flow to embed the network into a circular domain where each node is given a virtual coordinate. We use our multipath routing algorithm to seek $m_1 = 3$, and $m_2 = 5$ paths from s (in yellow) to t (in red) respectively (in this graph $\kappa(s, t) = 6$). Those paths are not necessarily node-disjoint, and all the shared nodes/edges are marked in purple in the figure.

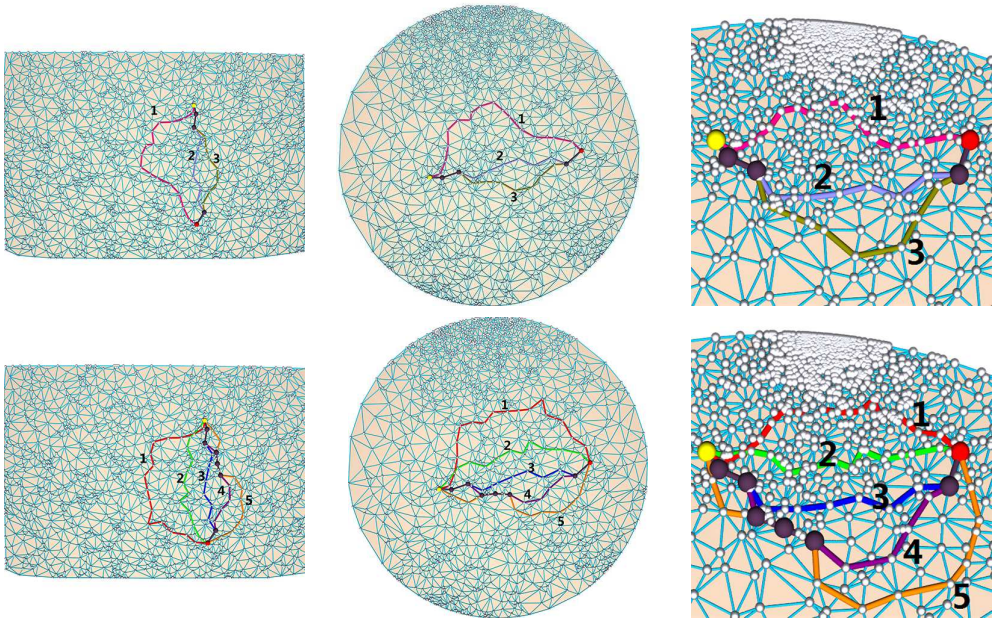


Figure 13: Multipath Routing Algorithm. Left column: original network; middle column: network applying Ricci flow; right column: network applying Ricci flow and a Möbius transformation (zoomed in). First row: $m = 3$; second row: $m = 5$.

$\kappa(s,t)$	m	Disjoint paths generated	approximation factor
6	3	2	66.7%
	5	4	
8	3	3	62.5%
	5	4	
	7	5	
11	5	4	72.7%
	9	7	
	11	8	

Table 1: Results of different sources and destinations in a uniformly distributed network with average edge links 20.

We also show the paths on the original network, and a different circular domain obtained by a Möbius transformation as well. In each of the three embeddings, the paths with the same color and number are identical. We can see that in different circular domains, the greedy paths, or the straight lines from s to t are also different,

which demonstrates that Möbius transformations together with greedy routing give us flexibility in choosing multiple paths.

More results are shown in Table 1. From the table we can see two facts. First, as a distributed algorithm, the Möbius transformation method gives us a good approximation of the number of node-disjoint paths in a dense graph. Second, the number of disjoint paths we can get is usually smaller than m, or equivalently, some paths we generate share common nodes or edges. This is due to the discrete nature of graphs. From s, we can only send packets to its neighbors, which is a restriction in choosing the first few hops of the transmission; the hops near t suffer from the similar problem. But in the middle segments, the paths generally follow the shape of a circle arc connecting s and t, which is desired. When the network becomes denser, the situation becomes more similar to that of the continuous case.

To further explore the differences between the discrete and continuous settings, we also simulate under different graphs. In graphs with different densities (with uniform sensor distribution), we randomly pick 10 pairs of sources and destinations, give the different inputs m as 3, 5 and 7, and calculate the average numbers of disjoint output paths we get. The results are shown in Table 2. From the table, we observe that the algorithm performs better in a denser graph with more links. This is reasonable since the gap between the discrete and continuous settings is smaller with a denser sensor distribution. Moreover, when the input m is smaller, the algorithm gives better approximation. This is simply because given a smaller input, the arcs span further away with each other (this result does not conflict with Table 1 where $\kappa(s,t)$ is known). We also notice that when m exceeds the average node degree, the percentage drops drastically, where the input – the number of disjoint paths we are trying to find – often exceeds the optimal value.

When the sensor distribution is non-uniform, the bottleneck of the performance lies in the sparse regions, especially when those regions cover the neighborhood of the source or the destination.

Figure 14 shows the result of a network region with holes. Some paths go along the inner boundary, but the heavy load along the boundaries is avoided.

Nodes	Average link number	Input m	Average output paths / m
1000	20.00	3	83.3%
		5	79.0%
		7	71.4%
600	12.02	3	76.7%
		5	64.0%
		7	51.4%
400	5.62	3	63.3%
		5	46.0%
		7	35.7%

Table 2: Results of graphs with different sensor densities.

3.4.2 Routing with link failures

In sensor networks, links are likely to fail, especially in an adversarial environment. Since greedy routing requires that for each step, there exists one link leading to a node closer to the destination, the performance of greedy routing will drop quickly when a link failure happens. What is more, link failures often have a property of spatial locality, which means that a group of nearby links are likely to fail at the same time. Therefore, when greedy routing hits this region, the message may 'get trapped in the mud'. We use the freedom of Möbius transformations to recover from this situation.

Based on the above observations, we adopt the setting of clustered random link failures. We first test a simple setting in which there is a region with arbitrary size and shape. All the links within this region have a link failure rate p, while all the links outside the region or crossing the boundary of the region will not fail. A message following greedy routing in the circular domain may not have guaranteed delivery as the link to the next hop can suddenly fail. Our strategy is to adopt a different Möbius transformation. We compare it with another simple strategy that recovers from failure by performing a random walk. Although random walk is simple, it is time-consuming for the routing path to jump out of a large link failure region, due to its locality and randomness. In our following experiment, we will compare greedy routing with Möbius transformations with other greedy

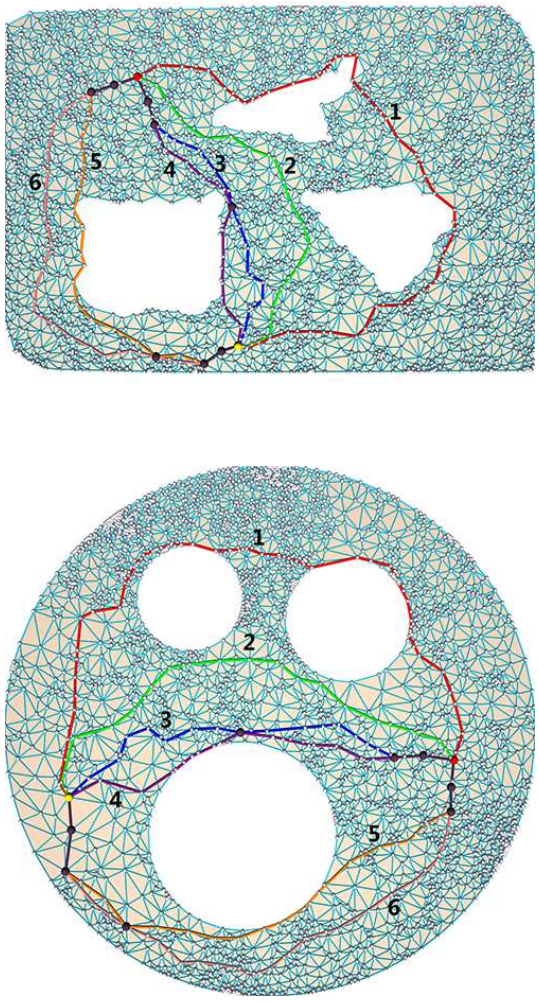


Figure 14: Multipath Routing Algorithm in a region with holes. Up: original network; Bottom: network applying Ricci flow. Here $\kappa(s,t) = 9$.

routing techniques, in terms of routing delivery rate and routing path length. The experiment network size is 1000 nodes with a varying number of links. The link failure region is a rectangle lying in the network.

In the experiments, we compared the following methods:

1. **Greedy routing on the original coordinates:** Simple greedy routing on the original coordinates, which fails to route to the destination easily due to link

failures. We call this method **Greedy** in short.

2. **Greedy routing on the virtual coordinates:** Greedily route to the destination using coordinates computed by Ricci Flow. We call this method **Ricci** in short.

3. **Greedy routing on the original coordinates with random walk:** Route based on the original coordinate set and perform random walk to recover from failure. We call this method **Gree-dyRnd** in short.

4. **Greedy routing on the virtual coordinates with random walk:** Greedily route using virtual coordinates and perform random walk to recover from link failures. We call this method **RicciRnd** in short.

5. **Greedy routing with Möbius transformations:** Our method performs greedy routing based on the virtual coordinates in a circular domain. If the route gets stuck in the middle due to link failures, it performs a Möbius transformation to get a new path towards the destination. We call this method **Möbius** in short.

Various parameters will affect the success rate of routing. In our experiment, we focus on the average degree of the network, the link failure rate and the TTL (time-to-live) of the packet. If the connectivity of networks becomes better as the average degree increases, routing will be easier for all methods. Obviously a higher link failure rate will make all methods suffer. We also include a TTL with each packet to stop a packet from roaming aimlessly in the network, in particular in the random walk method.

Routing result and analysis. In Figure 15, 16 and 17, we show the message delivery rates by varying the network density, the TTL and the link failure rate respectively. In all settings, we can see our method has a significantly higher delivery rate than the other methods, which shows that by using a new Möbius transformation, we can effectively find an appropriate path which leads the route out of the failure region. In general the performance of different methods, in decreasing order, follows the trend of

Möbius > RicciRnd > Ricci $\gg$ GreedyRnd $\simeq$ Greedy

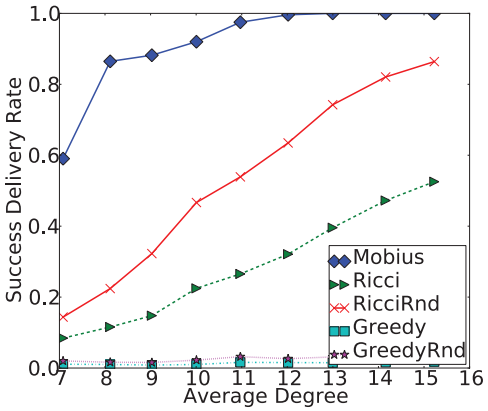


Figure 15: Routing delivery rate versus average degree (TTL = 500; link failure rate = 0.8). **Möbius** is our method. **Greedy** and **Ricci** are greedy routings on the original and Ricci flow coordinates respectively. **GreedyRand** and **RicciRand** are greedy routings on the original and Ricci flow coordinates with random walk respectively.

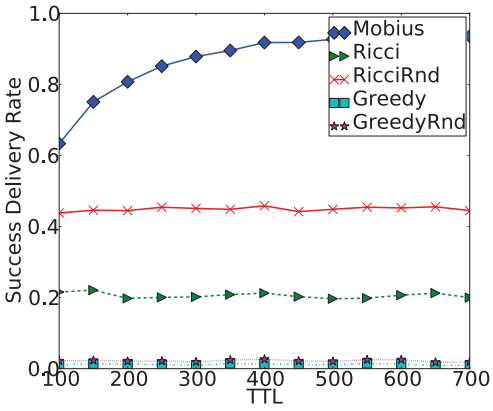


Figure 16: Routing delivery rate versus TTL (time-to-live) of packets (Avg. Degree = 10; link failure rate = 0.8).

Greedy routing using the original coordinates nearly does not work, no matter whether it is augmented with random walk or not.

Figure 15 shows the performance of all methods on networks of different node average degrees. The performance of all methods deteriorates when the network becomes sparse. Still, the Möbius method performs best. Note that in all cases

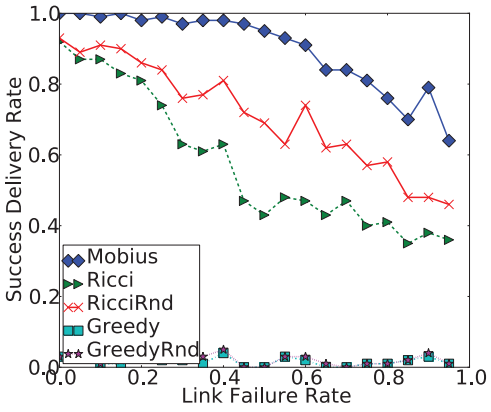


Figure 17: Routing delivery rate versus link failure rate (Avg.Degree = 10; TTL = 500).

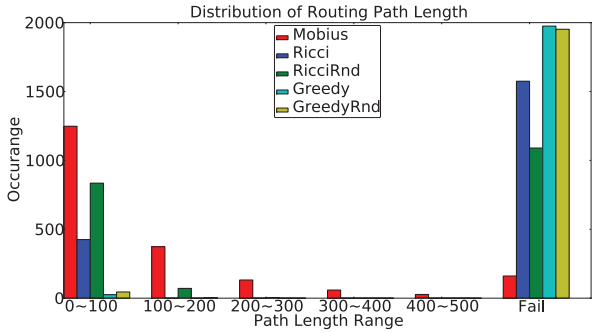


Figure 18: Distribution of routing path lengthes (Avg.Degree = 10; TTL = 500; link failure rate = 0.8).

we must first make sure that the network is connected and has a triangulation for computing the virtual embedding as a circular domain.

Figure 16 shows the importance to choose an appropriate TTL. As TTL grows, the delivery rate of our routing method grows rapidly and then stays close to 1, while the other methods do not exhibit a growing trend with TTL. This shows that the Möbius transformation scheme does recover from link failures and makes progress towards the destination, while the other methods still get stuck in the middle. Figure 18 shows the distribution of path length under different methods. Indeed our method gradually delivers more messages when TTL is increased.

p_1	p_2	Möbius	Ricci	RicciRand	Greedy	GreedyRand
0.3	0.0	0.962	0.573	0.779	0.021	0.028
0.6	0.0	0.912	0.402	0.651	0.014	0.016
0.6	0.3	0.738	0.207	0.472	0.003	0.009
0.9	0.0	0.823	0.271	0.511	0.008	0.013
0.9	0.3	0.632	0.148	0.335	0.002	0.006
0.9	0.6	0.472	0.037	0.193	0.001	0.004
0.6	0.6	0.587	0.094	0.263	0.002	0.004
0.3	0.3	0.802	0.254	0.513	0.007	0.011
0.3	0.6	0.591	0.119	0.285	0.002	0.005
0.3	0.9	0.224	0.017	0.104	0.001	0.002
0.6	0.9	0.152	0.011	0.063	0.001	0.002

Table 3: Comparison of different p_1 and p_2 settings.

Another potential factor affecting the routing performance is the distribution of link failure rates. We test the situation where links lying inside and outside the failure regions have failure rates p_1 and p_2, respectively. We evaluate the performance by varying the difference between p_1 and p_2. From the simulation results shown in Table 3, we can see that our method consistently outperforms the other methods in different settings. When p_1 and p_2 are getting close, **RicciRnd** starts to catch up. In these experiments we also vary the shape and positions of failure regions. While the exact values may vary, the trend is clear: our method makes progress towards the destination despite the existence of high link failure rate, and is unlikely to get stuck in the middle; making long detours is generally better than taking local random walks.

3.5 Conclusion

In this chapter we presented a method that uses circular domain embeddings and Möbius transformations to switch between them for multipath routing and improving routing resilience. This shows the power of a geometric transformation that regulates a network shape—difficult routing problems due to lack of global knowledge can benefit significantly from such transformations. We expect to extend the

intuition to more problems on distributed network setting in the future.

Our multipath routing algorithm in the discrete case uses a heuristic method that maximizes the angular spread of the paths at source and destination. It would be a very interesting problem to exploit the freedom of using such routing scheme to improve network throughput, or optimize energy usage, etc.

We would also like to mention that this work borrows heavily intuitions that arise from the continuous domain. Our provable results are in the continuous case and the performance of the algorithm in the discrete case is only evaluated by simulations. The problem of devising a theory addressing the gap between the continuous space and a discrete graph is open. It would be an interesting and challenging problem to come up with a suitable discrete model and derive minimum density bound. We leave this for future work.

Chapter 4

Traffic Analysis and Source-Location Privacy Under Random Walks

4.1 Introduction

Given a graph and a starting vertex, choose a neighbor of the current node at random and move to this neighbor and continue in this fashion. This sequence of nodes is called a *random walk* on the graph. A random walk is a Markov chain such that the next node to visit only depends on the current node and is independent of the history. This is often termed as the "memoryless" property of a random walk, which makes it useful for many applications in computer networking. Of particular interest to this work is the application of random walk in wireless sensor network routing for preserving source location privacy.

Source-location privacy Wireless sensor networks find many useful civilian and military applications. In many settings one would like to protect the privacy of sensor data, defined in the general sense that sensor data and its contextual information are observable by only those who are supposed to observe it [51]. Providing privacy in wireless sensor network is challenging for a number of reasons. Besides that the sensor nodes are low cost devices with limited computation and storage capacities, the fact that the sensor nodes use wireless medium make it susceptible to attacks

such as eavesdropping and traffic analysis. In the literature, privacy threats in sensor networks are classified as content-oriented privacy threats (i.e., the leaking of packet content to adversaries), that can be addressed by security and encryption mechanisms, and contextual privacy issues (i.e., the leaking of context information related to the measurement and transmission of the sensor data), of which location of the data source is a major piece of information to be protected. In particular, an adversary may be able to compromise private information of source locations without the ability of decrypting the transmitted data—by simply monitoring and analyzing the traffic pattern in the air.

A classical model formed for protecting the source location privacy is the "Panda Hunter Game" [51]. In the game, a large number of panda detecting sensors are placed in a habitat to detect panda presence. Pandas here are analogs of generic assets to be monitored by a sensor network. When a panda is observed, the nearby sensor node will report such detection data periodically to the sink through multi-hop routing methods. The data package could be encrypted such that the adversary cannot decipher the content of the message and cannot derive the location of panda right away. However, an adversary, in this case, the hunter, can monitor the traffic in the network and by timing analysis trace back the routing path to the origin of the message, i.e., the location of the data source. Clearly, simple routing schemes such as shortest path routing cannot provide data source privacy against traffic analysis attacks.

Many schemes proposed in the literature for preserving source location privacy use a common idea of introducing randomness in packet routing. The objective is to make the traffic pattern look random and uncertain, and then counteract the adversarial traffic analysis attacks. Many of them use random walk or variations of random walks as a major component in the design. Phantom routing [51], for example, first uses random walk in the network until the node is reasonably far from the source node and then uses (probabilistic) flooding method to deliver it to the source. Although a short random walk may still have the current node correlated with the origin, a long random walk will stop at a location that is independent of the packet original. It is known that if the random walk is longer than the *mixing time*, the random walk converges to its limiting distribution called the stationary distribution [73]. This is equivalent to selecting a node in the network randomly

(from the stationary distribution) and thus packet analysis afterwards will only trace back to this random location, unrelated to the true data source.

Traffic analysis on random walk In this chapter we show that it is a myth in common understanding that random walk automatically brings with it source location privacy. In other words, we present a technique which allows certain traffic analysis to infer the source location even for random walks that are as long as they want. Therefore our message is that random walk should be used carefully in protecting source location privacy.

4.2 Overview

Network Model and Attack Model We assume in this work a wireless sensor network deployed in a planar domain $\mathcal{R}$ of interest for monitoring interesting events. The event locations are of great importance for both the network owners and the adversary. When an event is detected, the nearby sensor node becomes the data source and sends the report periodically to a data sink (e.g., a base station or a mobile sink) in the network. We assume that the message is delivered by using random walk, in which the next node to visit is uniformly chosen from all neighbors of the current node. The random walk is sufficiently long to ensure that the message will be delivered to the data sink with high probability. A data source will generate data packets periodically and the delivery of these packets is completely independent of each other, i.e., they follow different random walk paths. The specific capabilities of the adversary are summarized below.

- *Monitoring traffic on network boundary.* We assume that the adversary can only monitor network traffic along the network outer boundary. This is a reasonable assumption in many settings when the domain of interest has restricted access to anyone but the network owner. It is also a realistic model of many military applications. The adversary places monitoring stations to monitor network traffic along the network outer boundary. Each monitoring station listens to the traffic in the neighborhood of a sensor node and record the signals delivered to/from the sensor node. We assume that the positions of the monitoring stations, or equivalently the network boundary, are

known. The monitoring stations are also assumed to be perfectly synchronized. The traffic data from the monitoring stations is collected and delivered to an offline base station for further analysis. We remark that the assumption puts more restriction on the adversary's power than the Panda Hunter model, in which the adversary can be anywhere inside the network and can move around as fast as possible.

- *Packets are encrypted.* We assume that the packets in the network are encrypted using symmetric encryption between the data source and the data sink and that the adversary does not have the key to decipher the content of the message. Similar to the Panda Hunter problem, the data source issues data packets periodically. We assume that the content of these data messages are different, i.e., with different time stamps. The monitoring stations can compare the messages received by different boundary nodes and conclude whether two messages received by two boundary nodes are the same or not. We assume that the chained encryption scheme used in onion routing is not feasible for sensor network, for two reasons. First the chained encryption requires that the source knows the entire path taken by the message, which is not the case for random walk. Second, chained encryption and decryption for each relay node is too heavy for resource constrained sensor nodes.
- *Non-malicious.* The adversary does not interfere with the normal functioning of the sensor networks. Otherwise it will be detected by intrusion detection schemes. The adversary does not compromise any node and does not generate or alter traffic in the network.
- *Informed.* We use the standard philosophy in security [120] that the adversary is aware of the routing methods used by the system, in our case, the random walk scheme.
- *Centralized and powerful.* The monitoring stations gather traffic received from the network boundary and then deliver all the data to an offline central station for processing. We assume the adversary has abundant computing resources and can perform complicated analysis.

Traffic analysis of random walk: We first consider a special case when the network is in a domain of disk shape and sensors are uniformly distributed inside the disk. In this case the random walk can be considered as a discrete approximation of the continuous Brownian motion inside a disk. For each message issued by the data source, through comparing the messages gathered by the monitoring stations at the network boundary we can conclude the node on the boundary that received the message for the first time. Now, since the data source generates multiple data packets, we monitor the position of the first hit on the boundary by different data packets. This constitutes a 'first hit' distribution (also called the exit distribution) ω'_x on the boundary where x is the source location. If the data source is at the center o of the disk, by symmetry the distribution ω'_x is a uniform distribution. When the data source is not at the center of the disk, the distribution has a single peak at the boundary intersected by the ray ox, and the closer the source to the boundary, the higher the peak is. See Figure 19 for an example. Therefore by monitoring the traffic pattern on the network boundary only, we obtain an observation of the first hit distribution p_x, through examining which we can infer the source location.

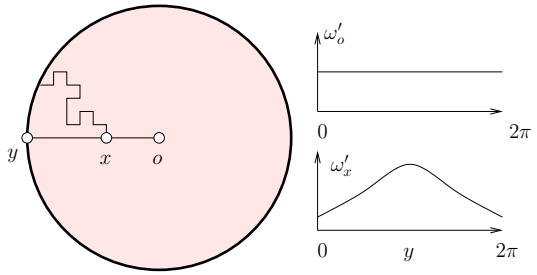


Figure 19: The first hit distribution ω'_x and ω'_o for random walk inside a unit disk starting at x and o respectively.

In general the network may not be of a disk shape and so the first hit distribution could have a complicated correlation with the source location. For a bounded domain $\mathcal{R}$ in the plane, the probability that a Brownian motion started inside a point $z \in \mathcal{R}$ hits a portion of the boundary is termed the *harmonic measure* [49], which we denote by ω_z. The first hit distribution observed from the traffic pattern ω'_z is a Monte Carlo approximation of ω_z. On simply connected planar domains, there is a close connection between harmonic measure and the theory of conformal

maps. A conformal map is a continuous one-to-one map that preserves angles. It is known that Brownian motions are conformally invariant [62]. What this means is that under a conformal map, $f : \mathcal{R} \rightarrow \mathcal{R}^{'}$, the probability for a Brownian motion starting from $x \in \mathcal{R}$ and exiting from an interval $I[a, b]$ on the boundary $\partial\mathcal{R}$ is the same as the probability of a Brownian motion starting from $f(x) \in \mathcal{R}^{'}$ and exiting from an interval $I[f(a), f(b)]$ on the boundary $\partial\mathcal{R}^{'}$. See Figure 20 for an example. Now, since any simple planar domain can be mapped to a canonical shape of a unit disk by a conformal mapping, one can obtain the harmonic measure for any simply connected domain. In particular, take the example in Figure 19, we can apply a Möbius transformation f from a disk to a disk such that the point x is now mapped to the center of the disk. Therefore the distribution ω_x can be immediately computed through f.

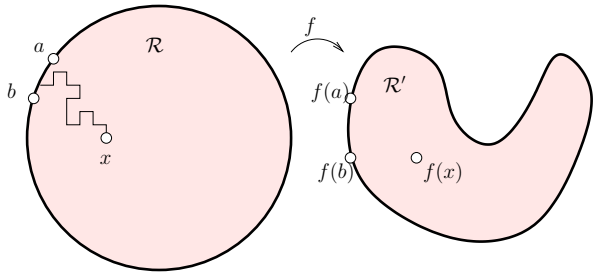


Figure 20: The probability for a Brownian motion starting from $x \in \mathcal{R}$ and exiting from an interval $I[a, b]$ on the boundary $\partial\mathcal{R}$ is the same as the probability of a Brownian motion starting from $f(x) \in \mathcal{R}'$ and exiting from an interval $I[f(a), f(b)]$ on the boundary $\partial\mathcal{R}'$.

The discussion above suggests that the exit distribution observed by the adversary along the sensor network boundary can be used to infer the source locations. In this work we present two such traffic analysis algorithms.

The first algorithm we present is for the recovery of a single data source. It is very simple, and works by integrating the position and the harmonic measure along the domain boundary, i.e., by calculating $\int_{z\in\partial\mathcal{R}} z d\omega_x(z)$. To understand this, take a look at Figure 19. If the source is at o and we integrate the position by the harmonic measure ω_o (which is uniform) along the unit circle, by symmetry this integration gives us the center of the disk. If the source is at x, the integration of

the position by ω_x must lie on the line segment oy – again by axial symmetry of ω_x with respect to oy. In fact, this integration would give precisely the position of x. And this is true not only for the case of a unit disk but for *any* planar domain. Since the first hit distribution observed from the traffic pattern, ω'_x, would be a good approximation to the harmonic measure ω_x, by using $\int_{z\in\partial\mathcal{R}} z d\omega'_x(z)$ we will get a very close approximation to x, as long as we have enough samples to be statistically meaningful.

The second algorithm is a general method using maximum likelihood estimation and it can be used for a general case when the data sources can be represented using low complexity. A number of representative scenarios include multiple data sources, data sources uniformly distributed on a line segment, as in the case of target tracking applications, or data sources distributed uniformly inside a small disk or square, as in the case when an event triggers multiple sensors to report to the sink. The results and the algorithms can be extended to a non-simple planar domain as well as a general non-planar terrain.

We present an extensive list of simulations for different network shape and different data source models as mentioned above. In particular, we presented the tradeoff between the number of messages issued by the data source vs the accuracy of our prediction of the source location.

Last we want to remark that we do not mean to claim that previous source location privacy preserving schemes using random walks are inadequate, but rather raise an alarm that their effectiveness should be reconsidered carefully given the potential attack illustrated in this work. At the end of this work we discuss variations of basic random walks and suggest ideas to defeat this particular traffic analysis attack.

4.3 Theory

In this section we first summarize the main results from the elegant theory of Brownian motions and conformal maps. We then provide the background on random walks in the discrete setting, and state our results.

4.3.1 Continuous theory

Definition 27 (Harmonic function)**.** *A harmonic function f on a domain $D \subset \mathbb{R}^2$ is a twice continuously differentiable real valued function such that $\frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} = 0$.*

Here are two useful properties of harmonic functions:

- Let $f(z) = f_1(z) + if_2(z)$ be holomorphic (see Section 1.1 in Chapter 1 for details). Then f_1 and f_2 are harmonic.
- *Mean-value property:* Let u be harmonic on the unit disk $\mathbb{D}$. Then,

$$u(0) = \int_{\partial \mathbb{D}} u(e^{i\theta}) \frac{d\theta}{2\pi}.$$

Let $\mathbb{D}$ denote the unit disk centered at the origin in $\mathbb{C}$. The group of Möbius transformations is the set of all conformal maps from $\mathbb{D}$ to itself. It is well-known that any such map is of the form $f(z) = e^{i\theta} \frac{z - z_0}{1 - \bar{z_0} z}$ for some $\theta \in (0, 2\pi)$ and some $z_0 \in \mathbb{D}$.

Definition 28 (Harmonic Measure)**.** *[13] [37] Let Ω be a simply connected domain. For any subset X of the boundary of $\Omega (X \subset \partial\Omega)$, the harmonic measure of X with respect to z is defined as $\omega(X, \Omega, z) = \frac{1}{2\pi} |f^{-1}(X)|$.*

Here $|.|$ denotes the Euclidean length of an arc on the unit circle. Note that any two conformal maps sending O to z only differ by a rotation, so this definition does not depend on the f chosen. Using harmonic measure, one can extend the mean-value property to arbitrary domains. If u is a harmonic function on an arbitrary simply connected domain Ω, $z_0 \in \Omega$ is a base point and f_{z_0} is a conformal map such that $f(0) = z_0$, then $u \circ f$ is harmonic on the disk, so that

$$u(z_0) = (u \circ f)(0) = \int_{S^1} u(f(e^{i\theta})) \frac{d\theta}{2\pi} = \int_{\partial\Omega} u(z) d\omega_{z_0}, \tag{35}$$

where $d\omega_{z_0}$ is the harmonic measure with respect to z_0.

The harmonic measure $\omega(X, \Omega, z)$ is related to a Brownian Motion started in the domain Ω from the point z. A one-dimensional Brownian Motion W_t intuitively is a scaling limit of the random walk. We define Brownian Motion next.

Definition 29 (Brownian motion)**.** *A Brownian motion W_t is a stochastic process indexed by time $t > 0$, which has the following properties :*

1. *$W_0 = x$; here $x \in \mathbb{R}$ is the starting point.*
2. *The process has independent increments, i.e. for any two disjoint intervals $[s_1, t_1]$ and $[s_2, t_2]$, where $s_i, t_i > 0$, the increment in one interval $W_{t_1} - W_{s_1}$ is independent of the increment in the other $W_{t_2} - W_{s_2}$.*
3. *$W_{t+h} - W_t$ is Normally distributed with mean 0 and variance h.*
4. *Almost surely, the function $t \longrightarrow W_t$ is continuous.*

The case $W_0 = 0$ is called Standard Brownian Motion. A two-dimensional Brownian motion is a pair $B_t = (W_t^1, W_t^2)$ of two independent one-dimensional Brownian Motions.

Harmonic measure, Brownian motion and conformal invariance An important property of the Brownian motion is that it is invariant under conformal changes, i.e. the image of a Brownian motion under a conformal map is again a Brownian motion in the image of the domain [63]. The Brownian Motion can be viewed as the limit, as $t \longrightarrow 0$, of a walk which starts at 0, chooses a direction randomly, goes a distance t in that direction, and continues this way at every point. The angle changes are preserved under conformal maps, therefore one should expect that the law of the trajectory should be invariant.

Clearly, the same is true for harmonic measure. In other words,

$$\omega(X, \Omega, z) = \omega(f(X), f(\Omega), f(z)), \tag{36}$$

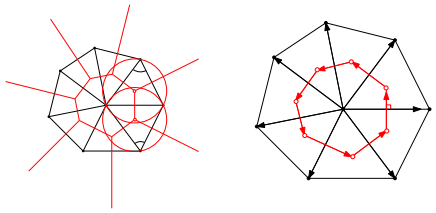
for any $X \subset \partial\Omega$ and f conformal.

4.3.2 Discrete theory

Here we summarize the related discrete theories of random walks on graphs.

Suppose G is a planar graph, embedded on the plane. Let $V = \{v_1, v_2, \cdots, v_n\}$ be the vertex set, (x_k, y_k) be the 2D position of vertex v_k and

$E = \{e_1, e_2, \cdots, e_m\}$ be the edge set. For simplicity, we assume each face of G is a triangle.



(a) (b)

Figure 21: Edge weight and vertex position—(a) shows the edge weight. (b) shows that the vertex position function is harmonic.

The following edge weight definition is motivated by the relationship of random walk and resistance of the triangulation as in an electrical network.

Definition 30 (Cotangent Edge Weight)**.** *[17, 27] Suppose edge $[v_i, v_j]$ is adjacent to two faces $[v_i, v_j, v_k]$ and $[v_j, v_i, v_l]$, then the weight on edge is given by $w_{ij} = \frac{1}{2}(\cot\theta_k + \cot\theta_l)$.*

This edge weight determines the transition probability for a random walk on graph.

Definition 31 (Random Walk on Graph)**.** *$X(t)$ is a random walk on the graph G defined as follows: if at time t the walk is at vertex v_i, then the probability of v_j being the next vertex is given by*

$$\textit{Prob}\{X(t+1) = v_j | X(t) = v_i\} = \frac{w_{ij}}{\sum_k w_{ik}}.$$

When we choose a uniform sampling and all the triangles are equilateral triangles, all the edge weights are close to 1. In this case the above definition becomes the same as the random walk with uniform distribution on all neighbors. In our simulations we choose G to be a Delaunay triangulation on a nice set of samples inside $\mathcal{R}$.

Definition 32 (Discrete Harmonic Measure)**.** *Suppose G is a planar graph with triangular faces. If the random walk $X(t)$ starts from a vertex v_i and exits at $v_k \in$*

∂G, then the discrete harmonic measure is defined as the probability

$$\omega_k(v_i) = \textit{Prob}\{X \sim v_k | X(0) = v_i\}.$$

Here $X \sim p$ means that the random walk X exits the boundary ∂G via the point p.

Definition 33 (Discrete Laplace Operator)**.** *Let $f : V \to \mathbb{R}$ be a function defined on the vertices of the graph G. The discrete Laplace operator is defined as $\Delta f(v_i) = \sum_j w_{ij}(f(v_j) - f(v_i))$.*

Definition 34 (Discrete Harmonic Function)**.** *Let $f : V \to \mathbb{R}$ be a function and Δ be the discrete Laplace operator. If $\Delta f = 0$ for all vertices, then f is called a discrete harmonic function.*

Using the above definitions, it is easy to show that discrete harmonic measures $\omega_j : V \to \mathbb{R}, \forall v_j \in \partial G$ are harmonic functions, and hence the expected position function is harmonic. Figure 21 shows the vertex position function is also harmonic. Like smooth case, discrete harmonic functions have mean-value property, which states the value at each vertex is the average of the values in the neighborhood. The mean-value property implies maximal modulus principle, which says the $\max$ and $\min$ values of a harmonic function must be on the boundary of the graph.

Definition 35 (Discrete Dirichlet Problem)**.** *Suppose $f : V \to \mathbb{R}$ is a function defined on the graph, f is harmonic, and with boundary condition $f|_{\partial G} = g$,*

$$\begin{cases} \Delta f(v_i) &= 0 & \forall v_i \notin \partial G \\ f(v_j) &= g(v_j) & \forall v_j \in \partial G. \end{cases} \tag{37}$$

From the maximum modulus principle, we get the uniqueness of the solution to the discrete Dirichlet problem. The solution to the Dirichlet problem can be explicitly given using harmonic measure.

Theorem 36 (Harmonic Measure Boundary Integration)**.** *Suppose $f : V \to \mathbb{R}$ is the solution to the Dirichlet problem (Equation 37). Then $f(v_i) = \sum_{v_j \in \partial G} g(v_j)\omega_j(v_i)$.*

Suppose a vertex v_0 at (x_0, y_0) sends messages routed by random walks. Figure 21 (b) shows the position function is harmonic. According to theorem 36, $(x_0, y_0) = \sum_{v_k \in \partial G}(x_k, y_k)\omega_k(v_0)$.. This is a linear time algorithm, given the harmonic measure

$$\omega_k(v_0) = \text{Prob}\{X \sim v_k | X(0) = v_0\}.$$

In our applications, we estimate the harmonic measure simply by the ratio between the number of messages received at v_k and the total number of messages.

The above definitions and theorems do not require the graph to be planar. In fact, these concepts can be defined on triangular meshes in $\mathbb{R}^3$. But the 3D vertex position is not harmonic. Similar to smooth case, one can apply conformal mapping [101] [41] to flatten the 3D triangulation and use the same method to estimate the source position on the 2D image. Because the Laplace matrix is solely determined by the connectivity of the graph and the corner angles, roughly speaking, discrete conformal mapping preserves angles, and therefore preserves harmonic measures. The harmonic measure can be estimated using the random walks on the 3D mesh, and applied for boundary integration to estimate the source location on the 2D image plane.

4.4 Traffic analysis on random walks

4.4.1 Settings

We assume that a sensor network W is deployed densely in a geometric domain $\mathcal{R}$. Packet routing in the sensor network is done by random walk on the network. Suppose that a data source at x generated N data messages, we record for each message the boundary node that receives this message for the first time. This frequency count can be normalized as a distribution ω'_x on the sensor network boundary. The input to the traffic analysis algorithm for the adversary is the exit distribution ω'_x, together with the geometry of the sensor network boundary $\mathcal{R}$. The adversary has no knowledge of the sensor network in the interior of $\mathcal{R}$ and would like to reconstruct the position x.

To reconstruct the source location, we assume that the sensor network is dense

and thus the random walk is a good approximation of Brownian motion in the continuous domain $\mathcal{R}$. Therefore, for each point $x \in \mathcal{R}$, define by ω_x the exit distribution of Brownian motion starting from x. We will compare ω'_x to ω_x to reconstruct the position of the source. Notice that in this setting there are two relaxations: 1) the distribution ω'_x is obtained through random walk on the (unknown) graph W; 2) the distribution ω'_x is obtained through a Monte Carlo method, i.e., based on the frequency count of N random walk samples. Thus our prediction of the source location could be a bit off from the true source location. But if random walks on the real sensor network are good approximations of the Brownian motion in $\mathcal{R}$, and that the number of samples, N, is not too small, the error in the prediction is expected to be small. This is indeed confirmed by simulations in the next section.

We will present two algorithms. The first algorithm provides a closed-form solution by simply integrating along the domain boundary $\mathcal{R}$. It works for a single source on a topological disk domain or topological disk with multiple holes. The second algorithm is based on maximum likelihood method. Basically by comparing ω' and ω (the exit distribution of Brownian motion), we find the source location y such that ω'_x and ω_y are the most similar. This is a generic framework for finding the locations of multiple data sources or any sources that can be represented in a compact way.

4.4.2 ALG1: Integration along domain boundary

Recall that if u is a harmonic function on the domain Ω, then its value at any point in the interior can be recovered by its values on the boundary, as long as one knows the harmonic measure of the boundary, i.e. $u(z_0) = \int_{\partial\Omega} u(z) d\omega_{z_0}$ where $d\omega_{z_0}$ is the harmonic measure with respect to z_0. Clearly, the identity function $u(z) = z$ is holomorphic (i.e., is differentiable in z), the real part and imaginary part are both harmonic. Hence we get $z_0 = \int_{\partial\Omega} z d\omega_{z_0}$.

For the case of a single source at position z_0, our construction algorithm is to simply multiply the coordinates of the location of a point $p \in \partial\mathcal{R}$ with its harmonic measure and add the resultants over the entire boundary. This algorithm is a linear running time algorithm with complexity dependent only on the *length* of the boundary $\partial\mathcal{R}$. The algorithm applies for all planar domains, including multiply

connected ones.

Calculating harmonic measure Now we show how to efficiently compute $\omega(X, \mathcal{R}, z)$, i.e., for any point z and any subset X of the boundary of $\mathcal{R}$, the probability that a random walk started from z will first exit the boundary from X. We first handle the (highly symmetric) case where the domain is the disk $\mathbb{D}$; X then is a subset of the unit circle and the starting point is the origin.

Computing $\omega(X, \mathbb{D}, 0)$: This is the probability that a random walk started from the origin in the disk exits the disk from the set X on the boundary. Clearly, this is uniform (by symmetry), and hence $\omega(X, \mathbb{D}, 0) = \frac{|X|}{2\pi}$. In other words this probability is just the normalized Euclidean arc-length of X.

$\omega(X, \mathbb{D}, z_0)$: To compute the harmonic measure for an arbitrary point $z_0 \in \mathbb{D}$, recall that the (conformal) Möbius transformation

$$g(z) = \frac{z - z_0}{1 - \bar{z_0} z}$$

maps the unit disk to itself and sends the point z_0 to the origin. Now, we use the property that the harmonic measure is preserved under conformal maps to obtain

$$\omega(X, \mathbb{D}, z_0) = \omega(g(X), \mathbb{D}, g(z_0)) = \omega(g(X), \mathbb{D}, 0) = \frac{|g(X)|}{2\pi}.$$

Computing $\omega(X, \mathcal{R}, z_0)$ **for arbitrary** $\mathcal{R}$**:** Here we will describe how to find the harmonic measure for an arbitrary planar domain $\mathcal{R}$. The first method only works for simply connected domains (domains with no holes) while the second works for both simply and multiply connected domains.

Method 1: Using the Riemann mapping This method uses the conformal invariance we described in Section 36. As above, let $\mathcal{R}$ be a simply connected domain, with boundary Γ a Jordan curve. In almost all practical applications, one approximates $\mathcal{R}$ by a polygon, and Γ by a polygonal chain. The first step is to compute the Riemann mapping from D to $\mathcal{R}$. For accomplishing this task, various methods have been proposed [101] [41].

So let us assume we have computed the Riemann mapping $f : \mathbb{D} \longrightarrow \mathcal{R}$. Notice that $f^{-1} : \mathcal{R} \longrightarrow \mathbb{D}$ is also conformal and once again, conformal invariance

implies that $\omega(X, \mathcal{R}, z_0) = \omega(f^{-1}(X), \mathbb{D}, f^{-1}(z_0))$ and we have shown how to compute $\omega(X, \mathbb{D}, z)$ for arbitrary $X \subset \partial\mathbb{D}$ and $z \in \mathbb{D}$ previously.

Method 2: Symm's method This method does not require one to explicitly compute the Riemann Mapping from $\mathbb{D}$ to $\mathcal{R}$, and holds for multi-holed domain. We refer the reader to [13] for a short summary of this method.

Recall from 35 that for any holomorphic function u on $\mathcal{R}$, we have the property $u(z_0) = \int_{\partial\mathcal{R}} u(z) d\omega_{z_0}$. We can discretize the boundary of $\mathcal{R}$ into n intervals $\{P_j\}_{j=1}^n$, assume that the harmonic measure is constant in each interval and look at the discrete counterpart to the above equation:

$$u(z_0) = \sum_j \int_{P_j} u(z) d\omega_{z_0} = \sum_j \frac{\omega_{z_0}(P_j)}{|P_j|} \int_{P_j} u(z) dz$$

Now if we choose n independent harmonic functions $\{u_i\}_{i=1}^n$, we get a system of n equations in n unknowns and we can solve to find $\omega_{z_0}(P_j)$.

4.4.3 ALG2: Maximum likelihood method

To apply a maximum likelihood approach (MLE), we first need the exit distribution/harmonic measure of a Brownian motion starting at a point $z \in \mathcal{R}$, which can be computed using methods in the section above. We then explain the application of MLE for different settings.

Let $f(.|\theta)$ denote a family of distributions parameterized by θ. If one observes an i.i.d. sample $x_1, x_2, ...x_n$ from one of the distributions in this family, the Maximum Likelihood method is a way to estimate the true parameter θ_0 such that this sample is most likely to come from $f(.|\theta_0)$.

Since the observations are assumed to be identically and independently distributed, the joint density function is

$$f(x_1, x_2, ...x_n|\theta) = f(x_1|\theta) f(x_2|\theta) ... f(x_n|\theta)$$

One then forms the *Likelihood Function*

$$\ell(\theta|x_1, x_2, ...x_n) = \Pi_{i=1}^n f(x_i|\theta)$$

The maximum likelihood estimate (MLE) $\hat{\theta}$ is defined to be the value of θ which maximizes the likelihood function, given the observed values x_i, i.e.

$$\hat{\theta} = \arg\max_{\theta} \ell(\theta|x_1, x_2, ...x_n)$$

For simplicity, the log-likelihood function $\hat{\ell} = \log \ell$ is also used, since $\log$ is a monotonic transformation.

From now on, $f_z := f(x|z)$ will denote the density function for the harmonic measure. Denote by X_z the exit position (the first hit position) of a random walk starting at z. It is a random variable distributed with density f_z; $\mathbb{P}(X_z \in A) = \int_A f_z(x)dx$ for all $A \subset \partial\Omega$.

- **Single source.** Suppose that $x_1x_2, ...x_N$ are the first hit positions on the boundary for the N messages sent by an unknown source $z_0 \in \mathcal{R}$ respectively. We know $f(x|z)$ from the previous section; one can now form the likelihood function and maximize.
- k **sources,** k **is known.** This boils down to the single source problem for each of the sources. Assume that the adversary cannot distinguish the data packets from different sources. Let the unknown source locations be $z_1, ...z_k$. Then what we observe is the random variable

  $$Y = X_{z_1} + X_{z_2} + ...X_{z_k}$$

  Given the z_i, the density of Y can be computed. Again one can form the likelihood function and maximize, now with respect to the vector of z_i. We also allow short-lived fake messages which are sent to a randomly selected neighbor by the relay node after a real message is relayed. Our traffic analysis is not affected if the fake messages are discarded and not relayed any further.
- **Source moving on a line.** Assuming that we have a mobile data source moving on a line. The source sends packets periodically after distance ϵ. We are interested in estimating the initial position z_0 and the direction θ in which the source is travelling. Let $z_i = z_0 + i\epsilon e^{i\theta}$. Notice here we just need to estimate 3 real parameters, and so we could expect to get good estimates with just a lot fewer data packets per source z_i.

4.5 Simulations

We conducted extensive simulation tests to examine the performance of our algorithm to find the source location, as well as how its accuracy for recovering the source-location is affected by different parameters.

The simulations were done under different settings, namely a unit disk, a planar non-disk domain, a planar domain with holes and a non-planar domain. Also for each type of domain, we conducted simulations using both a triangle mesh (TM) and a unit disk graph (UDG). In TM model, we calculated the transition probability for each node d by it's neighbors in the triangulations; for UDG model, we calculated the transition probability for d by it's neighbors in the unit disk graph. We scaled all planar domains inside a 2×2 bounding box, and scaled non planar domains inside a $2 \times 2 \times 2$ bounding box. We use the term $Error$ to measure the distance between the true source location and the location predicted by our algorithm. $Error_{ave}$ and $Error_{max}$ represent the average and maximum value of $Error$, where the unit is the same as used in the re-scaling mentioned above.. In the following, N_{domain} represents the number of nodes inside domain R, N_{msg} represents the number of messages issued at each source node.

Unit disk domain Figure 22 right and Figure 23 right show the relationship between N_{msg} with $Error_{ave}$ and $Error_{max}$ under TM disk model and UDG disk model, respectively. This is obtained by fixing N_{domain}=1K, and then randomly choosing $n = 100$ sources inside the R and issuing N_{msg} random walks from each of these chosen sources. We then calculated $Error_{ave}$ and $Error_{max}$.

We also examine how the location of source (the distance r from disk center) affects $Error_{ave}$. We uniformly sampled $0 < r < 1$ to get $\{r_1, r_2, ...r_m\}$. For each r_i we randomly chose $n_i = 100$ points whose distance to the center r_{n_i} satisfies $r_i - \varepsilon < r_{n_i} < r_i - \varepsilon$. In the simulations, we use $\varepsilon = 0.05$ and $N_{msg} = 1000$ times. Then we use our method to predict the source location according to the boundary message distribution. Based on the real source location and the one calculated by our method, we computed $Error_{ave}$ for each r_i. Figure 22 left and Figure 23 left show the relationship between r_i and $Error_{ave}$ under the TM model and UDG model, respectively. We can see that $Error_{ave}$ decreases as the distance from the

center increases.

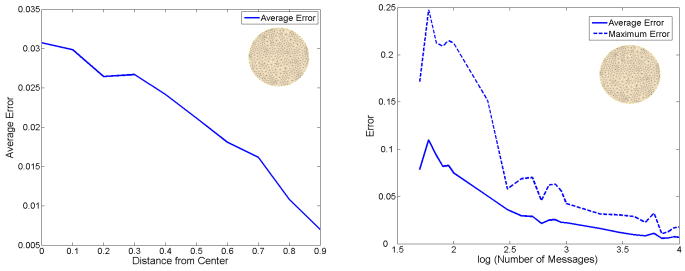


Figure 22: Left: Distance from center vs. $Error_{ave}$ under TM model. Right: N_{msg} vs. $Error_{ave}/Error_{max}$ under TM model.

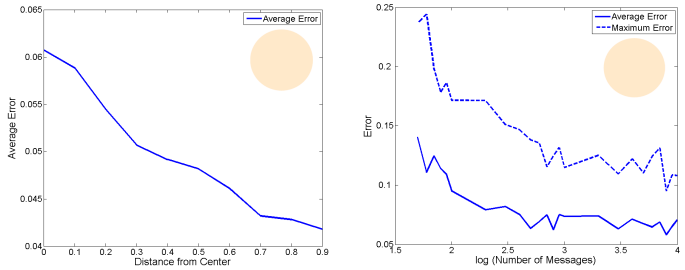


Figure 23: Left: Distance from center vs. $Error_{ave}$ under UDG model. Right: N_{msg} vs. $Error_{ave}/Error_{max}$ under UDG model.

Planar non-disk Domains We simulated similar scenarios on an irregular domain. We evaluated how N_{msg} affects $Error_{ave}$ and $Error_{max}$ by fixing $N_{domain} = 1000$. The results are shown in Figure 24. We can see that $Error_{ave}$ and $Error_{max}$ decreased while we increased N_{msg}. We obtained $Error_{ave}$ around 0.04 and 0.08 under TM model and UDG model, respectively, when 100 walks were issued.

Planar Domain with Holes Using the same setup as above, we evaluated how N_{msg} affects $Error_{ave}$ and $Error_{max}$ for a planar domain with holes. For a planar domain with holes, as long as we can monitor the inside hole boundaries as well, we can just treat them as the same as outer boundary in the calculation. The results are shown in figure 25. We obtained $Error_{ave}$ around 0.04 and 0.07 under TM model and UDG model for 100 messages.

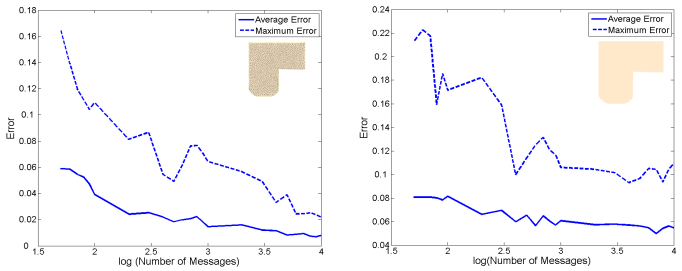


Figure 24: N_{msg} vs. $Error_{ave}/Error_{max}$. Left:TM model. Right:UDG model.

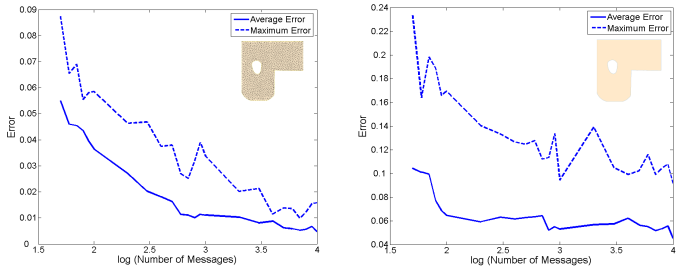


Figure 25: N_{msg} vs. $Error_{ave}/Error_{max}$. Left:TM model. Right:UDG model.

Non-planar Domain For a general non-planar domain, we first mapped it to the unit disk using conformal mapping method in [41]. Since Brownian motion is invariant under conformal mappings, we used the same method to calculate source location in the parameter domain, and then mapped it back to the original surface. The simulation results are in Figure 26. We obtained $Error_{ave}$ around 0.08 and 0.09 under TM model and UDG model for 100 messages.

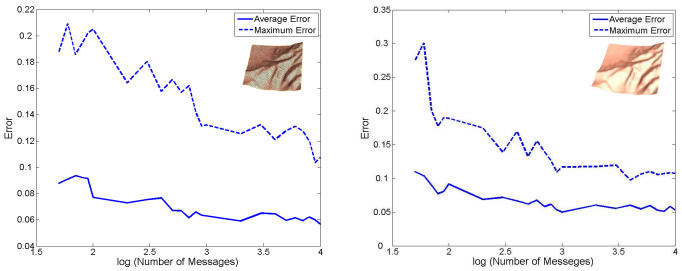


Figure 26: N_{msg} vs. $Error_{ave}/Error_{max}$. Left:TM Model. Right:UDG Model.

Visualization of Exit Distribution Following we show the exit distribution along the domain boundary. We took the non-uniform planar domain, set an arbitrary source and visualizes the exit distribution (Figure 27 left) using small disks along the boundary with **area proportional to the number of first hits**. We also show the distribution on the parameter domain, which is obtained by conformally mapping the original domain to a unit disk (Figure 27 right). The distribution on the parameter domain gives strong evidence that conformal mapping preserves Brownian motion. Namely the Brownian motion starting from source s on surface M is equivalent to the Brownian motion start from $\phi(s)$ on surface $\bar{M}$, if $\phi : M \to \bar{M}$ is a conformal mapping from M to $\bar{M}$.

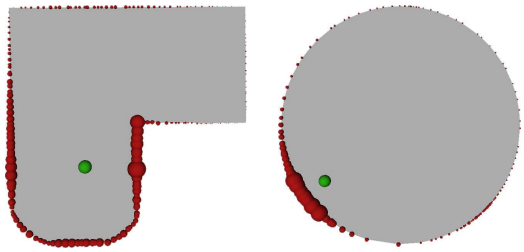


Figure 27: First Hit Distribution. Left: Original domain. Right: Parameter domain.

Network Density Versus Average Error To examine how much the network density N_{domain} affects the average distance error $Error_{ave}$, we fix N_{msg}, the number of messages. We then vary N_{domain} and observe $Error_{ave}$. The results are shown in Figure 28.

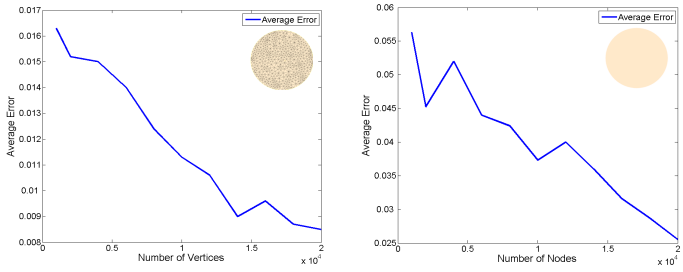


Figure 28: N_{domain} vs. $Error_{ave}$. Left:TM model. Right:UDG model.

Multiple Sources We uniformly discretized the unit square domain into a $N \times N$ grid (N=20 in our experiment), and assumed the possible location of a source is on the center of a grid. For 2 sources case, there are $N^4/2$ pairs of possible source location combinations. For each possible pair $(s_i,\ s_j)$, we issued $N_{msg} = 2000$ random walks from s_j and s_j and stored a set of first hit distributions $\{\Phi_{ij}, 0 < i, j < N\}$. We then randomly picked source-pairs $(s_1,\ s_2)$ to issue $\bar{N}_{msg}$ random walks and obtained a first hit distribution Φ_{test}. By comparing Φ_{test} with Φ_{ij} we got a p-value which stands for the probability that Φ_{test} and Φ_{ij} are the same distribution. The i, j which gave the maximum p-value directly points out the location of s_i and s_j. In this experiment, we varied N_{msg} and obtained a set of $Error_{ave}$, shown in Figure 29. We can see that $Error_{ave}$ decreased as we increased $\bar{N}_{msg}$.

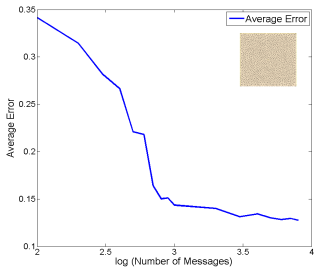


Figure 29: N_{msg} vs. $Error_{ave}$ for two sources.

4.6 Related work

Routing that preserves source anonymity has been a topic of study for a number of years. For routing on the Internet, one would like to hide the sender's identity, as phrased in anonymous routing. The most popular schemes are Chaum's mixes [19] and onion routing [115, 116]. In Chaum's scheme, the idea is to send the message in an encrypted manner to a central server called the anonymizer, which removes the source identity and then sends the message to the receiver. Thus one cannot differentiate the sources of the messages delivered by anonymizer. Onion routing uses encryption on source routing, such that the source identifies the entire routing path to the destination and encrypt the messages in layers in the order of the nodes

along the path. Each relay node decrypts the message using its own private key, which reveals the next hop and sends the message. In this way each node on the path is only aware of the immediate upstream and downstream node and is not aware of the entire path, in particular the source identity. Both schemes cannot be applied in sensor network setting since we cannot afford a central server, and public key encryption is too heavy for sensor nodes. In addition, encryption based security schemes only protect the content of the messages but cannot deal with traffic analysis attacks.

Existing schemes for preserving source location privacy is summarized in a recent survey [65]. Among them, random walk is a commonly used component. Phantom routing [51,85] first uses random walk to arrive at a node that is reasonably far away from the source and then use probabilistic flooding to deliver the message to the destination. Followup schemes such as in [67, 75, 127] use biased random walk in order to get farther away from the data source, or introduce fake data sources to further confuse the traffic pattern [51, 78]. In the next section we examine some of these variations and discuss the performance of the traffic analysis attack for these cases.

4.7 Discussions

Length of random walks Our traffic analysis scheme uses the exit distribution of random walks on the network boundary. This means that the random walks should be long enough so that they hit the network boundary with good probability before they stop. We argue that this is true as the random walks should be long enough to deliver the message to the data sink. If the data sink is at an unknown location in the network, the random walk should be long enough so that it visits every node in the network. This is termed as the *cover time*, defined as the expected number of steps for a random walk to cover all the nodes in the network [73]. For a 2D grid of n nodes the cover time is roughly in the order of $\Omega(n^2)$.

To estimate the probability that a random walk of length h hits the network boundary, we again consider a 2D grid of n nodes. Suppose X_i is the displacement vector of the i-th step of the random walk. X_i is uniformly chosen from $\{(1,0),(-1,0),(0,1),(0,-1)\}$. The position of random walk after i steps starting

from the center of the grid is simply $P_i = X_1 + X_2 + \cdots + X_i$. By the central limit theorem, P_i is a Gaussian distribution with mean $(0, 0)$ and variance $h/2\mathbb{I}$, where I is a 2×2 identity matrix. Thus the probability that P_i is outside a disk of radius r from the center is estimated as $e^{-r^2/h}$. Choose h to be $O(n^2)$ and r to be $\sqrt{n}$, the probability above is $1 - 1/n$. This means that the random walk of length $O(n^2)$ has a high probability to hit the network boundary at least once. This means that for a random walk to deliver the message to the sink, it must hit the boundary with high probability. This assures that the traffic analysis along the boundary could be performed.

Directed or biased random walk In a standard random walk, the next node to visit is chosen *uniformly randomly* from all neighbors. This is the discrete analog of Brownian motion which is isotropic. The first variation to it is to define a non-uniform probability distribution on neighbors. In Phantom routing and a number of follow-up papers, a biased random walk is often adopted in which the neighbor that is farther away from the data source is chosen with higher probability, in order to quickly get to the regions far away from the data source. For example, in sector-based directed random walk [51], a random walk from the west will be sent to a node to the east, chosen uniformly randomly. In hop-based directed random walk [51, 75], a random walk chooses the next hop uniformly randomly among only the nodes closer to the sink.

If the transition probability is non-uniform but determined (as in the two cases mentioned above), the harmonic measure as defined by the random walk will change. If the transition probability is known to the adversary, we can still calculate the harmonic measure under this change. Using the same idea presented in this work one can still infer the source location. Therefore to make a biased random walk to be a countermeasure of the traffic analysis, we need to make the transition probability to be *unknown* to the adversary. One idea is to vary this transition probability randomly and periodically. However, in this case one should be careful about the transition probability configuration to make sure that the random walk is still ergodic[1] – otherwise there is no guarantee that the random walk covers the entire

[1] A random walk is ergodic when there is a unique stationary distribution. This requires the graph (implied by the edges with non-zero transitional probability) to be connected and non-bipartite.

network and eventually delivers the message to the data sink.

Chapter 5

Topology Dependent Space Filling Curves

5.1 Introduction

We consider a sensor network that densely covers a planar domain, possibly with multiple network holes. In this chapter 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* [15, 122], 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 versus 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 [88].

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 [9,58], 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 [61]. 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.

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

5.1.1 Related work

Space filling curves In the continuous setting, various space filling curves have been defined for a square region [100].A 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 30 for an instance of the Hilbert curve. The basic recursive structure is to replace a line segment by a zigzag pattern. In a recursive step, each segment is replaced by a scaled and rotated version of this pattern.

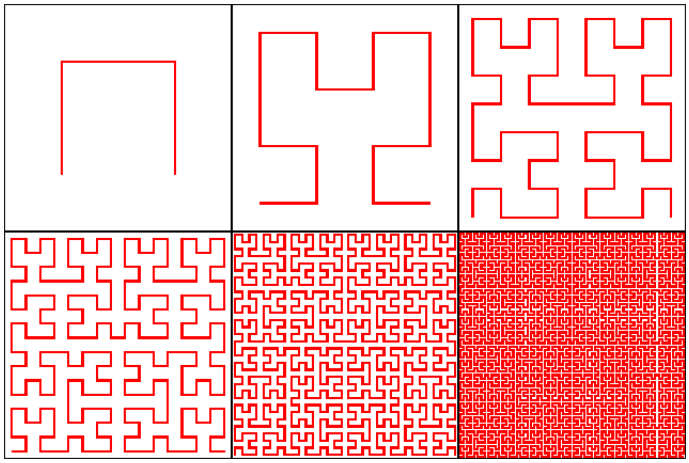


Figure 30: The Hilbert curve (source: Wikipedia).

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 [50] 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 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 [36].

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 [21] gives a $3/2$ approximation. For the Euclidean TSP, polynomial approximation schemes (PTAS) are known [7, 79] 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 [12]. Existing solutions for mule planning are all heuristic schemes [45, 52, 69, 106, 121].

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 [73]. 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.

5.1.2 Our contribution

In this work, we propose a scheme to generate a curve that (i) densely covers any geometric domain, possibly with holes; (ii) have a coverage density proportional to its length. To understand the main idea, we first consider a torus. See Figure 31. 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 31 (ii) shows a part of the curve on the torus. We 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 show that the curve will visit each point of the original domain at most twice and is provably dense.

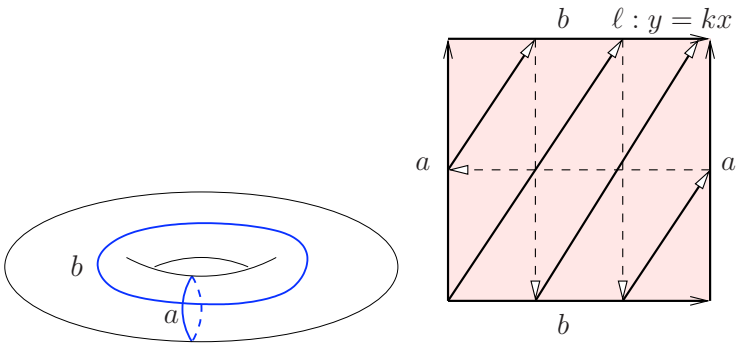


Figure 31: (i) A torus cut open along two curves a, b. (ii) The flattened torus. The line $\ell : y = kx$ is shown on the flattened torus (the top and bottom edges are the cut b, the left and right edges are the cut a). Since the top edge and bottom edge are actually the same, the line will go through 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.

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 works [46, 101, 102, 134]. We remark that the tools we use in this work are different. Our current method is based on holomorphic differentials from Riemann surface theory [40]. Imagine an electric field on a surface; 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 to be 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 Ω, 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.

5.2 Theoretical foundation

In this section, we will present the theoretical foundation, including rigorous proofs 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 present the algorithm to deform any planar domain to be one of the canonical shapes so that we can apply the dense curve computation.

5.2.1 Dense curve for simply connected domains and the annulus

Denote by S a planar domain. If S has one hole, we double cover it creating two copies, one top copy S and one bottom copy $-S$. We glue the two copies along the corresponding boundaries to form a topological 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. 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 32. For both cases, we need to design a dense curve on a torus.

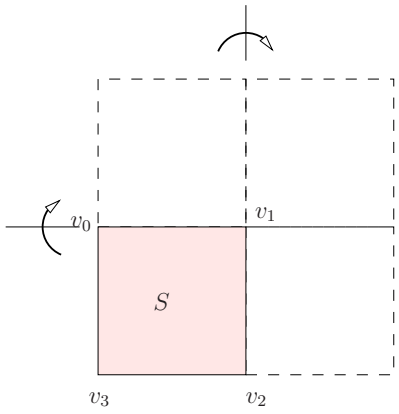
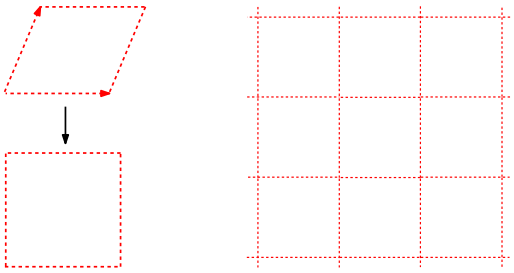


Figure 32: Reflect twice to create a torus with four copies of a square.



(a) square mapping (b) universal covering space

Figure 33: Conformal mapping for a topological 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 33, we can deform the lattice to be the regular integer lattice, namely, $\mathbf{e_1} = (1, 0)$ and $\mathbf{e_2} = (0, 1)$.

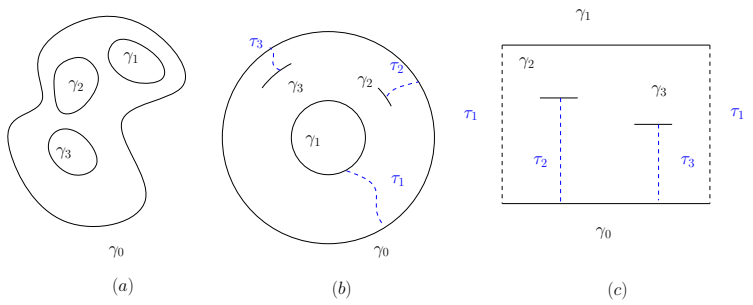


Figure 34: Multiply connected domain.

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 37 (Weyl's equidistribution theorem [110, 126])**.** *Let x be a irrational number. For $n \in \mathbb{N}$, consider the sequence $\langle nx \rangle$, where $\langle nx \rangle$ represents the fractional part of nx. Then*

- *$\langle nx \rangle$ is dense in $[0, 1]$.*
- *$\langle nx \rangle$ is equidistributed in $[0, v]$, i.e., 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 38 (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 equidistributed on T^2.*

5.2.2 Dense curve for a multiply connected domain

A relatively similar idea can be applied for multiply connected domains. As shown in Figure 34, 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, left end x-coordinates and the lengths of the slits (denoted as h_i, s_i, and l_i), respectively) are conformal invariants of the sensor network domain. In other words these are the "fingerprints" of the domain [4]. There can only be two cases here:

1. All of h_i, s_i and l_i are rational.

2. At least one of h_i, s_i and l_i is irrational.

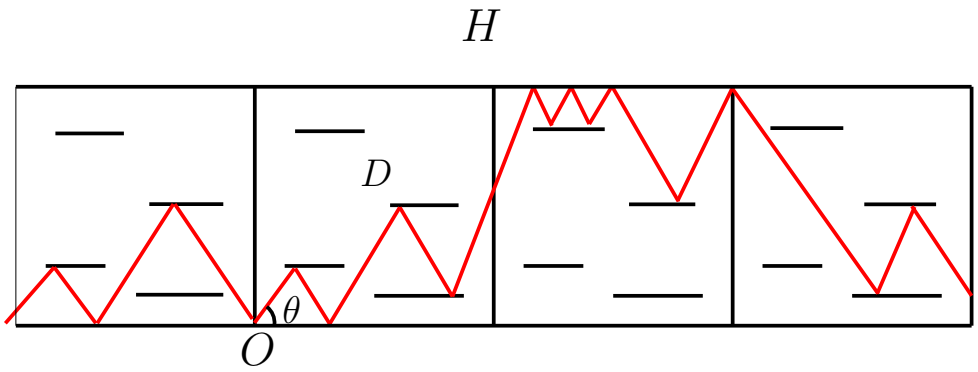


Figure 35: Tiling of H by copies of D and the curve γ_θ

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 35 for details.

Using techniques from complex dynamics, we now 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.

5.2.3 Billiards curve in the multiply connected domain

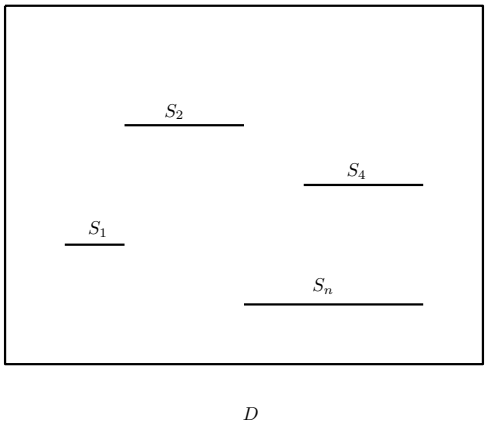


Figure 36: The slit domain D

The dense curve we construct for applications in a sensor network domain with holes is simply a “billiards" path in the transformed slit domain D (see Figure 36). The proof for the rectangular domain is identical to square domain case, which we adopt for simplicity.

Billiards in polygons is a rich and well researched field with many interesting open problems (for an excellent survey, see [42]). Most of the techniques involve the theory of Riemann surfaces, especially Teichmüller theory. In this section, we explain the construction of our curve and use these theories to derive properties of the curve.

Notation: Let I denote the unit square $[0,1] \times [0,1]$. Let H denote the infinite strip $\mathbb{R} \times [0,1]$. Let $S_i,\ 1 \leq i \leq n$ denote the horizontal slits; $S_i \subset I\ \ \forall i$. Each slit S_i is defined by a triple (h_i, s_i, l_i) where h_i is the y-coordinate, s_i is the starting

x-coordinate, and l_i is the length ($l_i < 1 - x_i$) of the slit. With the above notation, $D = I \setminus \{S_1 \cup \cdots \cup S_n\}$. We consider a tiling of H by copies of D as shown in Figure 35. Similarly, call $\widehat{H}$ the domain we get by taking copies of a reflection of D (denoted $\widehat{D}$) along the x-axis.

Curve construction: We construct a curve $\gamma_{\theta,x}$ in the following way. Shoot a ray from $x \in D$ which makes an angle θ with the positive x-axis. The ray then gets reflected at the following boundaries :

1. The line $L_1 = \mathbb{R} \times \{1\}$
2. The line $L_0 = \mathbb{R} \times \{0\}$ (the x-axis)
3. Any copy of the slit S_i, which we denote by $C_{i,t}$. By definition, $C_{i,t}$ is a slit of type $(h_i, s_i + t, l_i)$, where $t \in \mathbb{Z}$.

Let $\pi : H \longrightarrow I$ be the natural projection map. Let $\Gamma_{\theta,x} = \pi(\gamma_{\theta,x})$.For such a curve with initial direction θ , one can also define a flow $F_\theta : H \longrightarrow H$ which intuitively takes a point $z \in H$ to the corresponding point on $\gamma_{\theta,z}$ after one unit of time (assuming unit speed parametrization of $\gamma_{\theta,z}$). Such a flow is called *ergodic* if the only invariant sets are (Lebesgue) measure 0 or 1. Let O denote the origin. We are mainly interested in properties of $\Gamma_{\theta,O}$ (which is the curve we use for our purposes) and ergodicity of F_θ. For the sake of brevity, we set $\Gamma_\theta := \Gamma_{\theta,O}$ and $\gamma_\theta := \gamma_{\theta,O}$

Having defined our curve, we now define two objects which will be used in the proof of density of the billiards curve.

Quadratic differentials and flat metric: Let R be a Riemann surface (see Section 1.1.2 in Chapter 1 for details). For the definition of quadratic differentials, see Definition 3 in Chapter 1. It is well known ([113], [35]) that every quadratic differential on R provides “flat" coordinates—coordinates in which the metric of the underlying surface is flat (euclidean) everywhere except at finitely many points (corresponding to zeroes of the differential), which are also called cone singularities.

This metric is locally given by $|\phi(z)|^{\frac{1}{2}}|dz|$. On the complement of these singularities, the coordinate charts $z = (x, y)$ are such that the change of coordinates on the intersection of two charts are of the form $z \longrightarrow z + c$ or $z \longrightarrow -z + c$. In other words, the transition functions are translations and reflections in the origin composed with translations. Also, any such flat structure corresponds to a unique quadratic differential.

Furthermore, every quadratic differential naturally defines a pair of trajectories. A *horizontal trajectory* is a curve for which $\phi(z)dz^2 > 0$ and a *vertical trajectory* is a curve for which $\phi(z)dz^2 < 0$

Saddle connections: For a more detailed account of saddle connections, see [77].

Let P be the set of the endpoints of all the slits (including copies of the original slits). Hence

$$P = \{x \in H : x = (s_i + \alpha l_i + t, h_i); 1 \leq i \leq n; \alpha \in \{0, 1\}; t \in \mathbb{Z}\}.$$

Let $x, y \in \mathbb{R}^2$ and $L(x, y)$ denote the (closed) line segment joining point x to point y.

Definition 39 (*Saddle connections*)**.** *The set of Saddle connections S is the set of all possible line segments between endpoints of two distinct slits in H. Formally,*

$$S = \{L(x, y) : x, y \in P; x \neq y\}.$$

Note that in our case, S is countable. With this, we have all the building blocks for stating our theorems. In the following, $\mathbb{Q}$ denotes the field of rational numbers.

Lemma 40. *Let $h_i, s_i, l_i \in \mathbb{Q}$. Further, let θ be such that $\tan(\theta)$ is irrational. Then $\nexists \ell \in S$ such that $\ell \subset \gamma_\theta$.*

Proof. By hypothesis, $(s_i + t + \alpha l_i, h_i) \in \mathbb{Q} \times \mathbb{Q}$ for all $t \in \mathbb{Z}$ and all $\alpha \in \{0, 1\}$. Since all the endpoints of the slits are at rational coordinates, it is clear that all saddle connections have rational slopes. γ_θ has lines of only two slopes, $\tan(\theta)$ and $-\tan(\theta)$, both of which are irrational. Therefore, γ_θ cannot contain any segment from S. □

Theorem 41. *Let $h_i, s_i, l_i \in \mathbb{Q}$ and θ be such that $\tan(\theta)$ is irrational. Then :*

1. *Γ_θ is dense in D.*
2. *The flow F_θ is ergodic.*
3. *Let $x \in D$; then Γ_θ passes through x at most twice.*

Proof. The curve γ_θ has lines of two slopes, $\tan(\theta)$ and $-\tan(\theta)$. The curve on the copy in H contains only the line segments of slope $\tan(\theta)$ and once any such segment is reflected, it continues in $\widehat{H}$ appropriately, as a line with slope again $\tan(\theta)$ (Figure 37).

Consider the genus $n+1$ surface obtained by identifying a boundary of D to its corresponding reflected boundary in $\bar{D}$. In this way, we obtain a "flat surface", which has a euclidean (flat) metric everywhere except at finitely many cone singularities which occur at the end points of the glued boundaries. The charts are naturally defined from (the embedding of) D. Let ϕ be the quadratic differential associated with this flat structure. With the above construction, we now get a continuous curve V on the flat surface.

Consider the "rotated" quadratic differential $e^{i\theta}\phi$. One can see that V is a vertical trajectory of this differential. Now, we use a theorem from [77] which informally says that on a flat surface R, a trajectory of any quadratic differential which does not contain any saddle connection is dense. Furthermore, ergodicity of such directions was proved in [99]. Along with Lemma 40, these prove (i) and (ii). To prove (iii), assume the contrary. If there is a point x which Γ_θ passed through at least three times, then one of the directions θ or $(\pi - \theta)$ must be repeated, which would mean that Γ_θ is periodic; contradicting (i). □

In fact, using the above construction and [77], [99], one can also prove that

Theorem 42. *Let $h_i, s_i, l_i \in \mathbb{R}$. Then the set of directions θ for which Γ_θ is dense and ergodic is (Lebesgue) measure 1 in the set $(0, 2\pi)$.*

We end this section with some remarks:

1. In the rational case, we can find exactly the direction we need to obtain a dense curve.

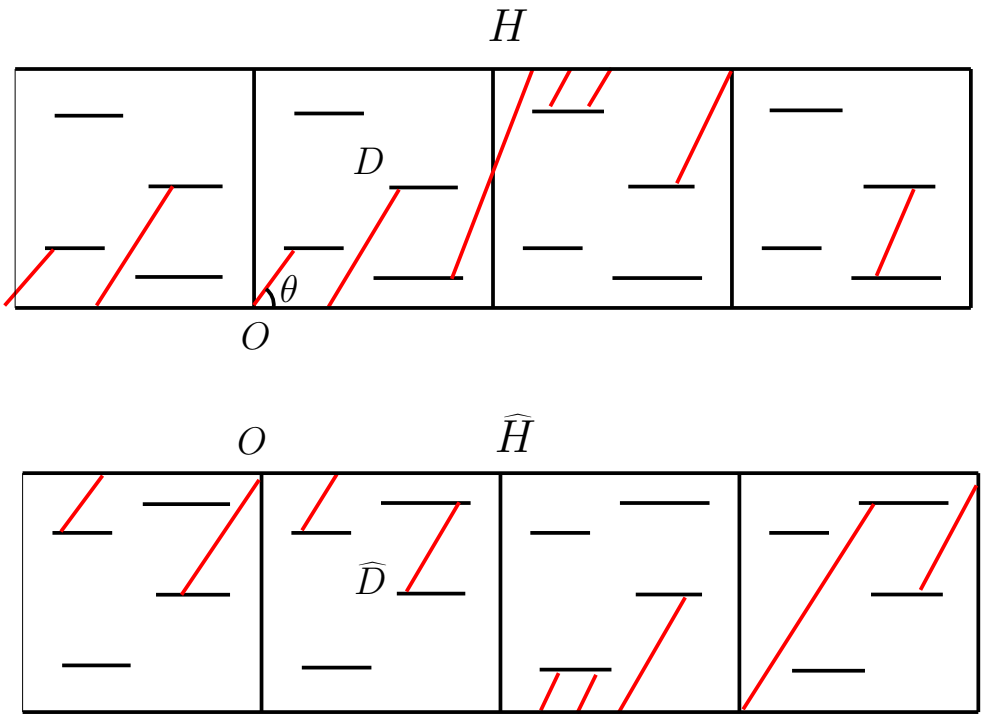


Figure 37: Decomposing γ_θ into two curves; one in H and one in $\widehat{H}$

2. In the irrational case, picking any direction uniformly at random, with probability 1 we get a dense curve.

3. Ergodicity here implies that the curve we get is *uniformly dense* in the domain D. This means that on the average, our curve will go through any two distinct equi-radii disks (contained in D) with almost same frequency. However, this does not imply uniform density in the original domain, since the conformal map we use in the next section does not preserve areas.

5.2.4 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

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

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.

5.3 Algorithms for discrete conformal mapping

In this section we go through the computational tools to deform an input domain to the 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 43 (Ahlfors [4] - Slit 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 $\phi : \Omega \to D$, 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, and $\{\phi(\gamma_k)\},\ k > 1$, are concentric circular slits. Such a conformal map 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.

Discrete exterior calculus Similar to all prior work [46, 101, 102, 134], 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 is defined on the boundary of a simplex as:

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

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 38. Slice the mesh M along η_k to get a mesh M_k, the path η_k on M corresponds to 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

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

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 ${}^*\omega_k$.

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

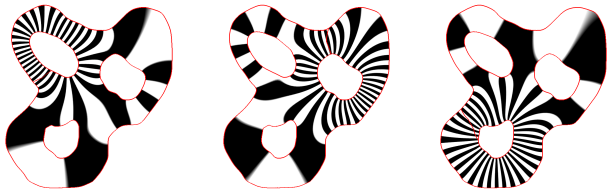


Figure 38: Exact harmonic 1-forms $\{\omega_4, \omega_5, \omega_6\}$.

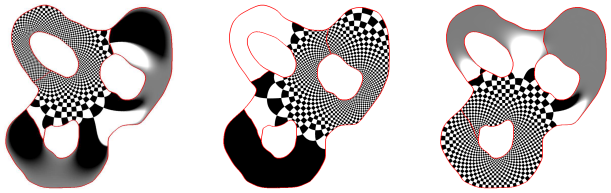


Figure 39: Holomorphic 1-forms basis $\{\tau_1, \tau_2, \tau_3\}$.

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

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

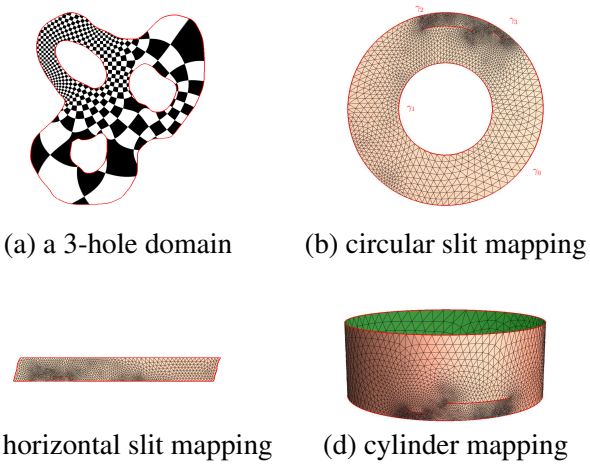
The wedge products can be computed as follows. Let $[v_i, v_j, v_k]$ be a triangle face, $e_i = [v_j, v_k]$, $e_j = [v_k, v_i]$ and $e_k = [v_i, v_j]$. Then by direct computation, we get

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

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 39 shows the basis for holomorphic 1-forms.



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

(c) horizontal slit mapping (d) cylinder mapping

Figure 40: Conformal mapping from the domain to the annulus, γ_0 is mapped to the outer circle, γ_1 is mapped to the inner circle.

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

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

and

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

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.

- 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.

- 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 [68].

- Step 3: Compute holomorphic 1-form basis. This involves completely local operations. Each node will solve a linear system only on its neighbors.

- 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-clockwisely. 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 resulting domain is a topological torus.

5.4 Simulations

5.4.1 Dense curve discretization

The aperiodic dense curve is identified as a continuous line in the universal covering space. To apply it to 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 $C(u)$ falling inside the belt $\mathcal{B}$, i.e.,

$$C(u) = \{u | u \in N(u) \text{ and } u \in \mathcal{R}\},$$

which is based on a closeness measurement. This will generate a discrete path.

5.4.2 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 41 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.

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 (UDG) model. In the experiments the networks have $5,000$ nodes with average degrees to be 7, we uniformly randomly generated 10 networks to average out the randomness.

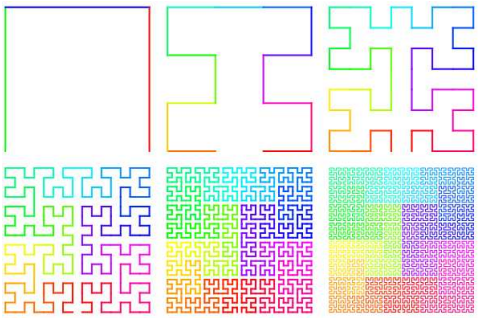


Figure 41: The Moore curve (source: Wikipedia).

Figure 42 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. Any continuation will not discover new nodes, as shown in the later part of the curve in Figure 42. 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 43 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

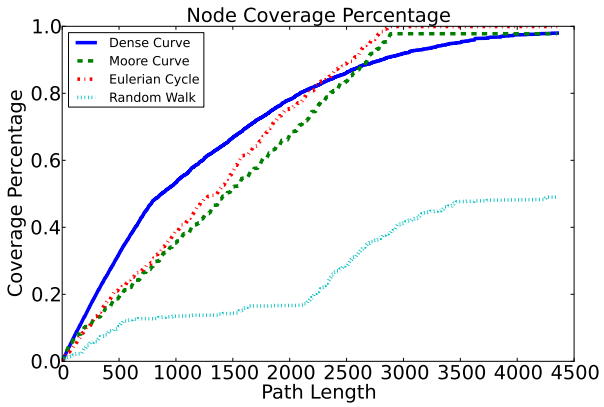


Figure 42: Comparison of network coverage

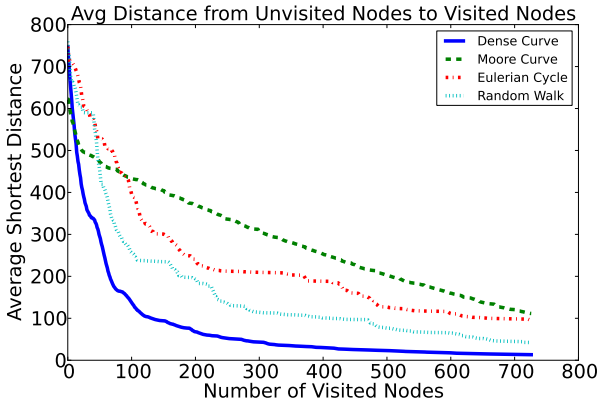


Figure 43: Comparison of average shortest distance from unvisited nodes to visited nodes

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.

5.4.3 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 44 shows a 2-hole network with its conformal mapping to circular slits and cylinder. Figure 45 shows the aperiodic dense curve on the network.

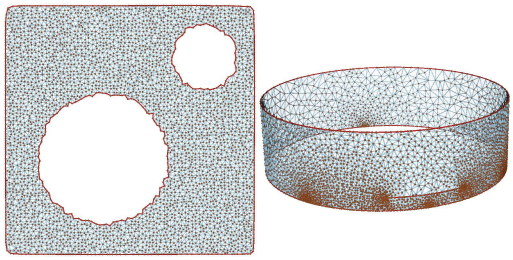


Figure 44: Conformal mapping for a network with holes.

5.5 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 46(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

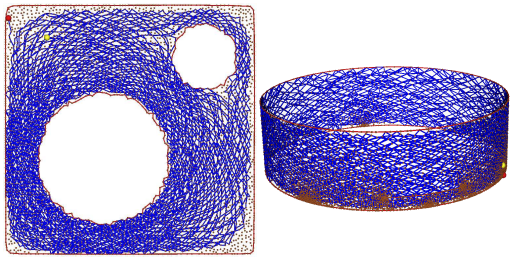
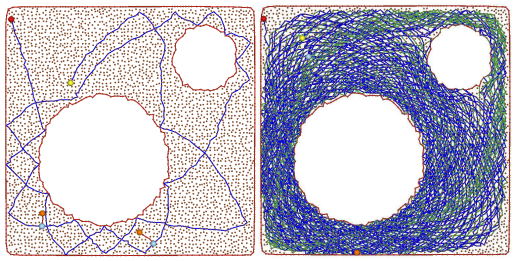


Figure 45: A dense curve on the network in Figure 44.

rulings schemes are only designed for networks of a regular shape, e.g., the horizontal/vertical lines [72, 112, 129], or proper circles (great circles through a stereographic mapping) [103]. 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 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 46(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.

Figure 46: Dense curve applications.

Bibliography

[1] http://cis.jhu.edu/software/lddmm-volume/tutorial.php.

[2] L. Ahlfors. *Lectures on Quasiconformal Mappings*. Van Nostrand Reinhold, New York, 1966.

[3] L. Ahlfors. *Lectures on quasiconformal mappings*, volume 38 of *University Lecture Series*. American Mathematical Society, Providence, RI, second edition, 2006. With supplemental chapters cy C.J.Earle, I.Kra, M.Shishikura and J.H.Hubbard.

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

[5] G. Alandjani and E. Johnson. Fuzzy routing in ad hoc networks. In *Proceedings of the 2003 IEEE International on Performance, Computing, and Communications*, pages 525 – 530, april 2003.

[6] P. Angelini, F. Frati, and L. Grilli. An algorithm to construct greedy drawings of triangulations. In *Proc. of the 16th International Symposium on Graph Drawing*, pages 26–37, 2008.

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

[8] A. Atlas and A. Zinin. Basic specification for IP fast reroute: Loop-free alternates. IETF RFC 5286, September 2008.

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

[10] X. Ban, M. Goswami, W. Zeng, J. Gao, and X. Gu. Exploration of path space using sensor network geometry. In *Proc. of the 32nd Annual IEEE Conference on Computer Communications (INFOCOM'13)*, April 2013.

[11] A. Banerjea. Simulation study of the capacity effects of dispersity routing for fault tolerant realtime channels. In *SIGCOMM '96: Conference proceedings on Applications, technologies, architectures, and protocols for computer communications*, pages 194–205, New York, NY, USA, 1996. ACM.

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

[13] C. Bishop. The riemann mapping theorem. http://www.math.sunysb.edu/ bishop/classes/math401.F09/t.pdf.

[14] C. Bishop. Conformal mapping in linear time. *Discrete and Comput. Geometry*, 44(2):330–428, 2010.

[15] 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.

[16] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. *Wireless Networks*, 7(6):609–616, 2001.

[17] S. C. Brenner and L. R. Scott. *The Mathematical Theory of Finite Element Methods*. Springer, 2002.

[18] J. Cai and W. Wu. Degraded link-disjoint multipath routing in ad hoc networks. In *ISWPC'09: Proceedings of the 4th international conference on Wireless pervasive computing*, pages 149–153, Piscataway, NJ, USA, 2009. IEEE Press.

[19] D. L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM*, 24(2):84–90, 1981.

[20] B. Chow and F. Luo. Combinatorial ricci flows on surfaces. *Journal Differential Geometry*, 63(1):97–129, 2003.

[21] 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.

[22] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press, 1994.

[23] P. Daripa. A fast algorithm to solve the beltrami equation with applications to quasiconformal mappings. *Journal of Computational Physics*, 106(2):355 – 365, 1993.

[24] S. De, C. Qiao, and H. Wu. Meshed multipath routing with selective forwarding: an efficient strategy in wireless sensor networks. *Comput. Netw.*, 43(4):481–497, 2003.

[25] T. K. Delillo, A. R. Elcrat, and J. A. Pfaltzgraff. Schwarz-christoffel mapping of multiply connected domains. *Journal d'Analyse Mathématique*, 94(1):17–47, 2004.

[26] P. Djukic and S. Valaee. Reliable packet transmissions in multipath routed wireless networks. *IEEE Transactions on Mobile Computing*, 5(5):548–559, 2006.

[27] P. G. Doyle and J. L. Snell. *Random Walks and Electric Networks*. The Mathematical Association of America, 1984.

[28] D. Eppstein and M. T. Goodrich. Succinct greedy graph drawing in the hyperbolic plane. In *Proc. of the 16th International Symposium on Graph Drawing*, pages 14–25, 2008.

[29] H. M. Farkas and I. Kra. *Riemann Surfaces*. Springer, 2004.

[30] O. Forster. *Lectures on Riemann Surfaces*. Springer-Verlag, 1981.

[31] S. Funke and N. Milosavljević. Network sketching or: "how much geometry hides in connectivity? - part II". In *SODA '07: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 958–967, 2007.

[32] D. Gaidashev and D. Khmelev. On numerical algorithms for the solution of a beltrami equation. *SIAM Journal on Numerical Analysis*, 46(5):2238–2253, 2008.

[33] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin. Highly-resilient, energy-efficient multipath routing in wireless sensor networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 5(4):11–25, 2001.

[34] J. Gao, L. J. Guibas, J. Hershberger, L. Zhang, and A. Zhu. Geometric spanners for routing in mobile networks. *IEEE Journal on Selected Areas in Communications Special issue on Wireless Ad Hoc Networks*, 23(1):174–185, 2005.

[35] F. Gardiner and N. Lakic. *Quasiconformal Teichmüler theory*. American Mathematical Society, 1999.

[36] 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.

[37] J. Garnett and D. Marshall. *Harmonc Measure*. Cambridge University Press, 2005.

[38] M. T. Goodrich and D. Strash. Succinct greedy geometric routing in r^2. Technical report on arXiv:0812.3893, 2008.

[39] H. Grötzsch. Ueber die verzerrung bei nichtkonformen schlichten abbildungen mehrfach zusammenhngender bereiche. *Leipz. Ber.*, 82:69–80, 1930.

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

[41] X. Gu and S.-T. Yau. Global conformal parameterization. In L. Kobbelt, P. Schröder, and H. Hoppe, editors, *Symposium on Geometry Processing*, volume 43 of *ACM International Conference Proceeding Series*, pages 127–137. Eurographics Association, 2003.

[42] E. Gutkin. Billiards in polygons: survey of recent results. *J. Stat. Phys.*, 83:7–26, 1996.

[43] K. Ishida, Y. Kakuda, and T. Kikuno. A routing protocol for finding two node-disjoint paths in computer networks. In *ICNP '95: Proceedings of the 1995 International Conference on Network Protocols*, page 340, Washington, DC, USA, 1995. IEEE Computer Society.

[44] F. Javadi and A. Jamalipour. Multi-path routing for a cognitive wireless mesh network. In *RWS'09: Proceedings of the 4th international conference on Radio and wireless symposium*, pages 223–226, Piscataway, NJ, USA, 2009. IEEE Press.

[45] 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.

[46] R. Jiang, X. Ban, M. Goswami, W. Zeng, J. 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.

[47] M. Jin, J. Kim, F. Luo, and X. Gu. Discrete surface Ricci flow. *IEEE Transactions on Visualization and Computer Graphics*, 14(5):1030–1043, 2008.

[48] M. Jin, J. Kim, F. Luo, and X. Gu. Discrete surface ricci flow. *IEEE Transaction on Visualization and computer Graphics*, 14(5):1030–1043, 2008.

[49] S. Kakutani. On brownian motion in n-space. *Proc. Imp. Acad. Tokyo*, 20(9):648–652, 1944.

[50] 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.

[51] P. Kamat, Y. Zhang, W. Trappe, and C. Ozturk. Enhancing source-location privacy in sensor network routing. In *Proceedings of the 25th IEEE International Conference on Distributed Computing Systems*, ICDCS '05, pages 599–608, 2005.

[52] 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.

[53] N. Karmarkar. A new polynomial-time algorithm for linear programming. pages 302–311, 1984.

[54] B. Karp and H. Kung. GPSR: Greedy perimeter stateless routing for wireless networks. In *Proc. of the ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, pages 243–254, 2000.

[55] Y.-J. Kim, R. Govindan, B. Karp, and S. Shenker. Geographic routing made practical. In *Proceedings of the Second USENIX/ACM Symposium on Networked System Design and Implementation (NSDI 2005)*, May 2005.

[56] Y.-J. Kim, R. Govindan, B. Karp, and S. Shenker. On the pitfalls of geographic face routing. In *DIALM-POMC '05: Proceedings of the 2005 joint workshop on Foundations of mobile computing*, pages 34–43, 2005.

[57] R. Kleinberg. Geographic routing using hyperbolic space. In *Proceedings of the 26th Conference of the IEEE Communications Society (INFOCOM'07)*, pages 1902–1909, 2007.

[58] 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.

[59] K.-W. Kwong, L. Gao, R. Guerin, and Z.-L. Zhang. On the feasibility and efficacy of protection routing in IP networks. In *INFOCOM'10*, March 2010.

[60] W. K. Lai, S.-Y. Hsiao, and Y.-C. Lin. Adaptive backup routing for ad-hoc networks. *Comput. Commun.*, 30(2):453–464, 2007.

[61] 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.

[62] G. Lawler. *Conformally Invariant Processes in the Plane*. Amer Mathematical Society, 2005.

[63] G. F. Lawler. Conformally invariant processes in the plane. *Mathematical Surveys and Monographs*, 114(2), 2008.

[64] T. Leighton and A. Moitra. Some results on greedy embeddings in metric spaces. In *Proc. of the 49th IEEE Annual Symposium on Foundations of Computer Science*, pages 337–346, October 2008.

[65] N. Li, N. Zhang, S. K. Das, and B. Thuraisingham. Privacy preservation in wireless sensor networks: A state-of-the-art survey. *Ad Hoc Netw.*, 7:1501–1514, November 2009.

[66] X.-Y. Li, G. Calinescu, and P.-J. Wan. Distributed construction of planar spanner and routing for ad hoc networks. In *IEEE INFOCOM*, pages 1268 – 1277, 2002.

[67] Y. Li and J. Ren. Preserving source-location privacy in wireless sensor networks. In *Proceedings of the 6th Annual IEEE communications society conference on Sensor, Mesh and Ad Hoc Communications and Networks*, SECON'09, pages 493–501, 2009.

[68] 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.

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

[70] Y. Lipman, V. G. Kim, and T. A. Funkhouser. Simple formulas for quasiconformal plane deformations. *ACM Trans. Graph.*, 31(5):124:1–124:13, 2012.

[71] C. Liu, M. Yarvis, W. S. Conner, and X. Guo. Guaranteed on-demand discovery of node-disjoint paths in ad hoc networks. *Comput. Commun.*, 30(14-15):2917–2930, 2007.

[72] 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.

[73] L. Lovasz. Random walks on graphs: A survey. *Bolyai Soc. Math. Stud.*, 2:353–397, 1996.

[74] L. M. Lui, T. W. Wong, W. Zeng, X. Gu, P. M. Thompson, T. F. Chan, and S. Yau. Optimization of surface registrations using Beltrami holomorphic flow. *Journal of Scientific Computing*, 50(3):557–585, 2012.

[75] X. Luo, X. Ji, and M. Park. Location privacy against traffic analysis attacks in wireless sensor networks. In *2010 International Conference on Information Science and Applications*, pages 1–6. Ieee, February 2010.

[76] M. K. Marina and S. R. Das. Ad hoc on-demand multipath distance vector routing. *SIGMOBILE Mob. Comput. Commun. Rev.*, 6(3):92–93, 2002.

[77] T. S. Masur H. Rational billiards and flat structures. *Hasselblatt B., Katok, A. (eds.) Handbook of Dynamical Systems*, 1A:1015–1089, 2002.

[78] K. Mehta, D. Liu, and M. Wright. Protecting location privacy in sensor networks against a global eavesdropper. *IEEE Trans. Mob. Comput.*, 11(2):320–336, 2012.

[79] 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.

[80] M. Motiwala, M. Elmore, N. Feamster, and S. Vempala. Path splicing. *SIGCOMM Comput. Commun. Rev.*, 38(4):27–38, 2008.

[81] R. Musaloiu-E. and A. Terzis. Minimising the effect of wifi interference in 802.15.4 wireless sensor networks. *International Journal of Sensor Networks*, 3(1):43–54, 2008.

[82] A. Nasipuri and S. Das. Demand multipath routing for mobile ad hoc networks. In *Proceedings of the 8 th Annual IEEE Internation Conference on Computer Communications and Networks (ICCCN)*, pages 64–70, October 1999.

[83] B. Nath and D. Niculescu. Routing on a curve. *SIGCOMM Comput. Commun. Rev.*, 33(1):155–160, 2003.

[84] Y. Ohara, S. Imahori, and R. V. Meter. Mara: Maximum alternative routing algorithm. In *Proc. IEEE INFOCOM*, 2009.

[85] C. Ozturk, Y. Zhang, and W. Trappe. Source-location privacy in energy-constrained sensor network routing. In *Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*, SASN '04, pages 88–93, 2004.

[86] C. H. Papadimitriou and D. Ratajczak. On a conjecture related to geometric routing. *Theor. Comput. Sci.*, 344(1):3–14, 2005.

[87] P. Pardalos and M. Resende. *Handbook of applied optimization*, volume 1. Oxford University Press New York, 2002.

[88] 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.

[89] P.Henrici. *Applied and Computational Complex Analysis*, volume 3. Wiley, New York, 1986.

[90] S. Ramasubramanian, H. Krishnamoorthy, and M. Krunz. Disjoint multipath routing using colored trees. *Comput. Netw.*, 51(8):2163–2180, 2007.

[91] A. Rao, C. Papadimitriou, S. Shenker, and I. Stoica. Geographic routing without location information. In *Proceedings of the 9th annual international conference on Mobile computing and networking*, pages 96–108, 2003.

[92] S. Ray, R. Guérin, K.-W. Kwong, and R. Sofia. Always acyclic distributed path computation. *IEEE/ACM Transactions on Networking*, 2009.

[93] L. Reddeppa Reddy and S. V. Raghavan. Smort: Scalable multipath on-demand routing for mobile ad hoc networks. *Ad Hoc Netw.*, 5(2):162–188, 2007.

[94] E. Reich. Extremal quasiconformal mappings of the disk. *Handbook of Complex Analysis*, 1:75–136, 2002.

[95] C. Reichert, Y. Glickmann, and T. Magedanz. Two routing algorithms for failure protection in IP networks. In *Proc. ISCC*, 2005.

[96] C. Reichert and T. Magedanz. Topology requirements for resilient IP networks. In *Proc. 12th GI/ITG Conf. on Meas., Mod. and Eval. of Comp. and Comm. Sys*, 2004.

[97] W. Rudin. *Real and complex analysis*. Mathematics series. McGraw-Hill, 1987.

[98] J. Ruppert and R. Seidel. A Delaunay refinement algorithm for quality 2-dimensional mesh generation. *J. Algorithms*, 18:548–585, 1995.

[99] J. S. S. Kerckhoff, H. Masur. Ergodicity of billiard flows and quadratic differentials. *Ann. of Math.*, 124:293–311, 1986.

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

[101] 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.

[102] 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.

[103] 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.

[104] G. Schollmeier, J. Charzinski, A. Kirstädter, C. Reichert, K. Schrodi, Y. Glickman, and C. Winkler. Improving the resilience in IP networks. In *Proc. HPSR*, 2003.

[105] C. Sengul and R. Kravets. Bypass routing: An on-demand local recovery protocol for ad hoc networks. *Ad Hoc Netw.*, 4(3):380–397, 2006.

[106] 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.

[107] E. Sharon and D. Mumford. 2 d-shape analysis using conformal mapping. *International Journal of Computer Vision*, 70:55–75, October 2006.

[108] J. Shewchuk. Delaunay refinement algorithms for triangular mesh generation. *Computational Geometry: Theory and Applications*, 22(1–3):86–95, 2002.

[109] R. Shi, M. Goswami, J. Gao, and X. Gu. Is random walk truly memoryless-traffic analysis and source location privacy under random walks. In *Proc. of the 32nd Annual IEEE Conference on Computer Communications (INFOCOM'13)*, April 2013.

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

[111] K. Stephenson. *Introduction To Circle Packing*. Cambridge University Press, 2005.

[112] 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.

[113] K. Strebel. *Quadratic differentials*, volume 5. Springer, 1984.

[114] H. Suzuki and F. Tobagi. Fast bandwidth reservation scheme with multi-link and multi-path routing in atm networks. In *Eleventh Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 2233 –2240 vol.3, may 1992.

[115] P. F. Syverson, D. M. Goldschlag, and M. G. Reed. Anonymous connections and onion routing. *IEEE Journal on Selected Areas in Communications*, 16(4):482–494, 1997.

[116] P. F. Syverson, M. G. Reed, and D. M. Goldschlag. Onion routing access configurations. In *DISCEX 2000: Proceedings of DARPA Information Survivability Conference and Exposition*, pages 34–40, January 2000.

[117] O. Teichmüller. Extremale quasikonforme abbildungen und quadratische differentiale. *Preuss. Akad. Math.-Nat.*, 1, 1940.

[118] O. Teichmüller. Bestimmung der extremalen quasikonformen abbildungen bei geschlossenen orientierten riemannschen flchen. *Preuss. Akad. Math.-Nat.*, 4, 1943.

[119] W. P. Thurston. *Geometry and Topology of Three-Manifolds*. Princeton lecture notes, 1976.

[120] W. Trappe and L. C. Washington. *Introduction to Cryptography with Coding Theory*. Prentice Hall, 2002.

[121] 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.

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

[123] H. Q. Vo, Y. Y. Yoon, and C. S. Hong. Multi-path routing protocol using cross-layer congestion-awareness in wireless mesh network. In *ICUIMC '08: Proceedings of the 2nd international conference on Ubiquitous information management and communication*, pages 486–490, New York, NY, USA, 2008. ACM.

[124] Y. Wang, J. Shi, X. Yin, X. Gu, T. F. Chan, S.-T. Yau, A. W. Toga, and P. M. Thompson. Brain surface conformal parameterization with the Ricci flow. *IEEE Transactions on Medical Imaging*, 31(2):251–264, 2012.

[125] O. Weber, A. Myles, and D. Zorin. Computing extremal quasiconformal maps. *Comp. Graph. Forum*, 31(5):1679–1689, 2012.

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

[127] Y. Xi, L. Schwiebert, and W. Shi. Preserving source location privacy in monitoring-based wireless sensor networks. *Proceedings 20th IEEE IPDPS*, 06:1–8, 2006.

[128] W. Xu, K. Ma, W. Trappe, and Y. Zhang. Jamming sensor networks: attack and defense strategies. *Network, IEEE*, 20(3):41–47, May-June 2006.

[129] 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.

[130] D. Zappala. Alternate path routing for multicast. *IEEE/ACM Trans. Netw.*, 12(1):30–43, 2004.

[131] W. Zeng, L. M. Lui, F. Luo, T. Chan, S.-T. Yau, and X. Gu. Computing quasiconformal maps using an auxiliary metric with discrete curvature flow. *Numeriche Mathematica*, 121(4):671–703, 2012.

[132] W. Zeng, J. Marino, K. Gurijala, X. Gu, and A. Kaufman. Supine and prone colon registration using quasi-conformal mapping. *IEEE Transactions on Visualization and Computer Graphics)*, 16(6):1348–1357, 2010.

[133] W. Zeng, D. Samaras, and X. D. Gu. Ricci flow for 3D shape analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(4):662–677, 2010.

[134] 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.

[135] F. Zhang, A. Jiang, and J. Chen. Robust planarization of unlocalized wireless sensor networks. In *Proc. of INFOCOM 2008*, pages 798–806, 2008.

[136] Z. Zhou and J.-H. Cui. Energy efficient multi-path communication for time-critical applications in underwater sensor networks. In *MobiHoc '08: Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing*, pages 221–230, New York, NY, USA, 2008. ACM.