Greedy Routing with Guaranteed Delivery Using Ricci Flows

Rik Sarkar
Dept. of Computer Science
Stony Brook University
rik@cs.sunysb.edu

Xiaotian Yin
Dept. of Computer Science
Stony Brook University
xyin@cs.sunysb.edu

Jie Gao
Dept. of Computer Science
Stony Brook University
jgao@cs.sunysb.edu

Feng Luo
Dept. of Mathematics
Rutgers University
fluo@math.rutgers.edu

Xianfeng David Gu
Dept. of Computer Science
Stony Brook University
gu@cs.sunysb.edu

ABSTRACT

Greedy forwarding with geographical locations in a wireless sensor network may fail at a local minimum. In this paper we propose to use *conformal mapping* to compute a new embedding of the sensor nodes in the plane such that greedy forwarding with the virtual coordinates guarantees delivery. In particular, we extract a planar triangulation of the sensor network with non-triangular faces as holes, by either using the nodes' location or using a landmark-based scheme without node location. The conformal map is computed with *Ricci flow* such that all the non-triangular faces are mapped to perfect circles. Thus greedy forwarding will never get stuck at an intermediate node. The computation of the conformal map and the virtual coordinates is performed at a preprocessing phase and can be implemented by local gossip-style computation. The method applies to both unit disk graph models and quasi-unit disk graph models. Simulation results are presented for these scenarios.

Categories and Subject Descriptors

C.2.2 [**Computer-Communication Networks**]: Network protocols—*Routing protocols*; F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*Geometrical problems and computations*

General Terms

Algorithms, Design, Theory

IPSN'09, April 13-16, 2009, San Francisco, California, USA.

Keywords

Greedy Forwarding, Ricci Flow, Conformal Mapping, Sensor Networks

1. INTRODUCTION

This paper studies greedy routing in sensor networks using a form of virtual coordinates. Greedy routing has received a lot of attention since it was proposed for routing in ad hoc wireless networks [2,17], due to its simplicity and efficiency. A node forwards the packet to its neighbor whose distance to the destination is the smallest. Thus routing decision is made with only local knowledge. On a dense network without holes greedy routing typically works very well and gives close to optimal routing paths.

A well-known problem with geographical forwarding is that packets may get stuck at nodes with no neighbor closer to the destination. There have been many ways to handle this problem, the most well-known one is face routing [2,17,20]. In this paper we take the approach of constructing a map of the network and finding *virtual coordinates* for the sensor nodes such that simple greedy routing with virtual coordinates always succeeds. In the following we survey prior work along this line and briefly explain our approach.

1.1 Related Work

Using virtual coordinates for greedy routing was first used in NoGeo routing proposed by Rao *et al.* [28]. In this method the network boundary is pinned on a convex planar curve (such as a square or a circle) and the interior nodes are embedded by using the rubberband representation [23]. The rubberband representation is obtained by each node (that is not fixed) running an iterative algorithm of putting itself at the center of mass of the neighbors' current locations until convergence.

Motivated by NoGeo, a few theoretical works ask under what conditions embedding of a given graph in the plane ex-

ists such that greedy routing always works [3, 24]. Such an embedding is called a *greedy embedding*. It is known that not every graph admits a greedy embedding, for example a star with 7 leaves [24]. Some graphs are known to have greedy embeddings, for example, any graph with a Hamiltonian path, any complete graph, any 4-connected planar graph (since they are Hamiltonian [31]), and Delaunay triangulations. It still remains open to fully characterize the class of graphs that admit greedy embeddings.

Papadimitriou and Ratajczak [24] made the conjecture that any planar 3-connected graph[1] has a greedy embedding in the plane. Dhandapani discovered that any planar triangulation (without holes) admits a greedy embedding in the plane for which greedy forwarding always succeeds [6]. Recently the 3-connected graph conjecture was proved to be true by Leighton and Moitra [21], and independently by Angelini *et al.* [1]. Later the algorithm in [21] was improved such that the coordinates use $O(\log n)$ bits for a graph with n vertices [13].

A recent observation by Kleinberg [18] shows that if we use hyperbolic space then greedy routing becomes easy. He showed that any connected graph has an embedding in the hyperbolic space such that by using the hyperbolic distance greedy routing from any node to any node always succeeds. The intuition is to embed a tree in a hyperbolic space such that greedy routing works on the tree. Since any connected graph has a spanning tree, greedy routing works at all times. A similar idea was used in [7].

For all the virtual coordinates schemes, one needs to have a location service such that any node can inquire the virtual coordinate of any other node in the network. Efficient location services for sensor networks have been developed [22, 29]. Such location services can be used in routing with virtual coordinates developed in this paper. The virtual coordinate addresses in this case, as in [28], are simple euclidean coordinates of the form (x, y). The service simply needs to have tables of such coordinates for the nodes.

1.2 Our Approach: Conformal Maps

To find virtual coordinates for the sensor nodes in a network, we look at a more fundamental problem of studying maps between *spaces*. This is motivated by the fact that most practical applications of sensor networks require sufficient sensor density, for both sensing coverage and system robustness/redundancy to cope with node failure. Taking the view point of maps of spaces also introduces some insensitivity to link dynamics and local disturbances. In a wireless network, communication links are volatile and may go up and down. Nodes may also die or be replaced periodically. The shape of the geometric region for which the sensors are deployed and aim to monitor, is much more stable, provided that the sensor network still has sufficient coverage for its normal functioning.

The question we ask is, given a domain surface $\mathcal{R} \subseteq \mathbb{R}^2$, is there a continuous map $f : \mathcal{R} \to \mathcal{D}$ such that greedy routing on $\mathcal{D}$ always succeeds? If the domain $\mathcal{R}$ is a simply connected convex region (i.e., with no holes), then the identity map is good, since greedy routing is essentially straight line routing in $\mathcal{R}$ and always successfully delivers a message. If the domain $\mathcal{R}$ has holes, especially some concave ones, then we need a non-trivial map to prevent greedy routing from getting stuck at hole boundaries. In fact, we would like to map the domain $\mathcal{R}$ to a domain D such that all the holes in D are circular — since greedy routing never gets stuck at circular hole boundaries. Thus the map that achieves this will produce virtual coordinate system for the nodes with guaranteed delivery for greedy routing.

The map we will use is a *conformal map*, as shown in Figure 1. A conformal map between two surfaces preserves angles. For any two arbitrary curves γ_1, γ_2 on the surface S, a conformal map ϕ maps them to $\phi(\gamma_1)$, $\phi(\gamma_2)$ with the same intersection angle as that of γ_1, γ_2. According to conformal geometry theory, a genus zero surface with multiple boundaries (a topological multi-holed annulus) can be conformally mapped to the unit disk with circular holes, as shown in Figure 1. Such a conformal mapping is unique up to a Möbius transformation in each homotopy class of degree one mappings. Recent advances in differential geometry, in particular, on *Ricci flow* lead to computationally efficient algorithms to construct such kind of conformal mapping.

Ricci flow was introduced by Richard Hamilton for Riemannian manifolds of any dimension in his seminal work [14] in 1982. Intuitively, a surface Ricci flow is the process to deform the Riemannian metric of the surface. The deformation is proportional to Gaussian curvatures, such that the curvature evolves like the heat diffusion. It has been considered a powerful tool for finding a Riemannian metric satisfying the prescribed Gaussian curvature in mathematics and has been applied in the proof of the Poincaré conjecture on 3-manifolds [25–27]. Chow and Luo [5] proved a general existence and convergence theorem for the discrete Ricci flow on surfaces, and proved that the Ricci energy is convex. Jin *et al.* provided a computer algorithm in [16].

1.3 Our Contribution

We investigate in this paper algorithms for computing a conformal map of a sensor field and the application in enabling greedy routing. We first extract from the communication network a planar graph H such that all non-triangular faces map to network holes that will be later mapped to circular holes in the embedded domain D. Ideally these holes in H are also real holes in the network/environment. Thus the triangulated mesh H is a discrete representation and approximation of the underlying domain $\mathcal{R}$. We show that when the sensors are deployed densely in $\mathcal{R}$ such that any disk with diameter 1 has at least one sensor inside, the unit disk

[1] A graph is 3-connected if it remains connected after the removal of any 2 nodes.

graph on the nodes contains such a triangulation H of $\mathcal{R}$ that can be computed locally. Note that the density requirement here is simply a condition for detecting the topology faithfully and is not necesaary for successful routing. We present a method to construct a suitable triangulation from any unit disk graph. Similar results can be proved as well for certain quasi-unit disk graphs (when there must be a link between two nodes within distance $1/\alpha < 1$ and no link when distance is greater than 1). When node locations are not known, we use a landmark-based scheme as in [9, 10] to locally select landmarks of constant bounded density (i.e., the Voronoi diagram of each landmark has $O(1)$ nodes) and compute a planar triangulation on the landmarks. The conformal map is computed on this triangulation. The planar triangulation algorithm for a sensor network may be of interest by itself since many surface processing algorithms assumes a nice triangular mesh and now can be applied in the network setting.

The triangulation and conformal map computation are performed as a preprocessing phase during network setup. There are two major advantages of using Ricci flow method for computing the virtual coordinates: distributed nature and conformality.

1. Ricci flow algorithm is intrinsically distributed, each node only requires the information from its one-hop neighbors, and the curvature at that node can be calculated, the metric deformation is proportional to the curvature, therefore the flow can be performed completely in parallel.

2. Asymptotically, Ricci flow leads to a conformal mapping. We set the virtual edge lengths of the triangulation to be one, therefore each triangle is an equilateral one in the original triangulation.

The virtual coordinates are disseminated to every node with which they can use greedy routing for point-to-point message delivery. In our simulation section we demonstrate the efficiency of conformal map computation, in terms of the number of messages used per node, different network topologies and node density.

On a last note, we remark that the rubberband representation used in NoGeo algorithm [28] is essentially the discrete version of finding a harmonic map between the simply connected domain defined by the sensor field outer boundary and a convex planar domain. **However, harmonic maps for multi-holed annulus can not be guaranteed to be a diffeomorphism.** Therefore holes in the network are not handled properly. In particular, near a non-convex hole, the network may be folded over itself, causing the routing to fail. Our algorithm can be considered as an extension of Dhandapani's result that any triangulation (without holes) admits a greedy embedding. Our method considers triangulation of a domain with possible holes and converges to an embedding of the network with holes mapped to circles as long as such embedding exists [15]. The existence of embedding can be verified by the combinatorics of the network as explained in [5]. An embedding can always be ensured by appropriate local refinements to the triangulation. For conventional methods, it is more challenging to handle dense networks. Our method is especially good at handling dense networks. In fact, our discrete mapping converges to smooth conformal mappings with the increase of density. The proof can be found in [15].

In the next section we will briefly introduce the theory behind the Ricci flow algorithm to compute a conformal map. Readers may also choose to read the algorithm section first in which we introduce the implementation of the algorithm in a network setting and the entire pipeline for computing virtual coordinates for greedy routing.

2. THEORY

In this section, we introduce several important concepts in differential geometry and the theory of the Ricci flow.

2.1 Riemannian Metric and Curvature

Suppose a surface S is embedded in $\mathbb{R}^3$, then it has a Riemannian metric, induced from the Euclidean metric of $\mathbb{R}^3$. The metric tensor is denoted by $\mathbf{g} = (g_{ij})$.

Riemannian metric determines the Gaussian curvature K and the geodesic curvature k . Gauss-Bonnet theorem states that the total curvature is a topological invariant

$$\int_S K dA + \int_{\partial S} k ds = 2\pi\chi(S), \tag{1}$$

where ∂S represents the boundary of S, $\chi(S)$ is the Euler characteristic number of S.

Suppose $u : S \to \mathbb{R}$ is a scalar function defined on S, then it can be verified that $e^{2u}\mathbf{g}$ is another Riemannian metric on S, denoted by $\bar{\mathbf{g}}$. It can be proven that angles measured by $\mathbf{g}$ are equal to those measured by $\bar{\mathbf{g}}$. Therefore, $\bar{\mathbf{g}}$ is *conformal* to $\mathbf{g}$ and now e^{2u} is called the conformal factor. The Gaussian curvatures are related by

$$\bar{K} = e^{-2u}(-\Delta u + K), \tag{2}$$

where Δ is the Laplace-Beltrami operator under the original metric $\mathbf{g}$. Similarly, the geodesic curvatures satisfy

$$\bar{k} = e^{-u}(\partial_{\mathbf{n}} u + k), \tag{3}$$

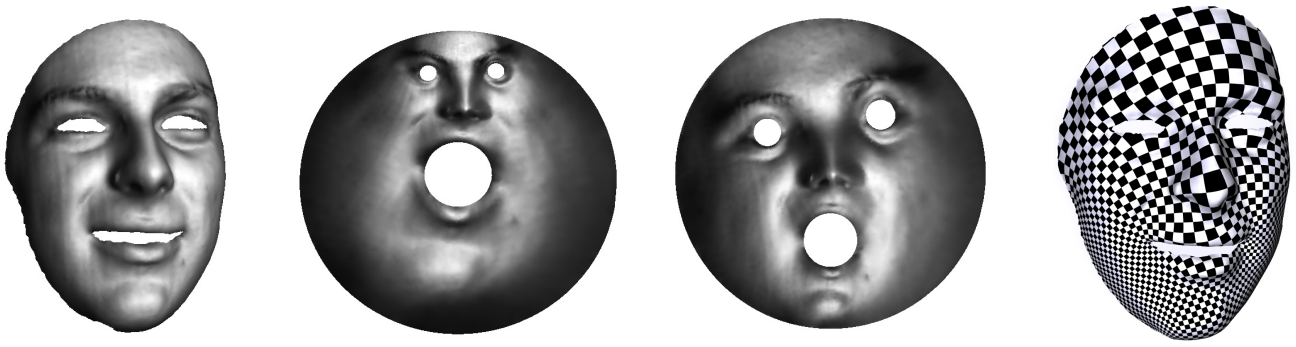
where $\mathbf{n}$ is the tangent vector orthogonal to the boundary. According to Gauss-Bonnet theorem (equation 1), the total curvature doesn't change.

2.2 Surface Ricci Flow

Suppose S is a smooth surface with Riemannian metric $\mathbf{g}$. The Ricci flow is the process to deform the metric $\mathbf{g}(t)$ according to its induced Gaussian curvature $K(t)$, where t is the time parameter

$$\frac{dg_{ij}(t)}{dt} = -2K(t)g_{ij}(t). \tag{4}$$

Suppose $T(t)$ is a temperature field on the surface. The heat diffusion equation is $dT(t)/dt = -\Delta T(t)$, where Δ is



(a) Face surface (b) Conformal Mapping (c) Möbius transformed (d) Texture Mapping

Figure 1. Conformal Mapping. Genus zero surface (a) with multiple boundaries can be mapped to the unit disk with circular holes conformally. Two such conformal mappings differ by a Möbius transformation are shown in (b) and (c). The checker board image is applied for texture mapping in (d), where all the right angles of the checkers are well preserved, therefore, the mapping is conformal.

the Laplace-Beltrami operator induced by the surface metric. The temperature field becomes more and more uniform with the increase of t, and it will become constant eventually.

In a sense, the curvature evolution induced by the Ricci flow is exactly the same as heat diffusion on the surface, as follows:

$$\frac{K(t)}{dt} = -\Delta_{\mathbf{g}(t)} K(t), \tag{5}$$

where $\Delta_{\mathbf{g}(t)}$ is the Laplace-Beltrami operator induced by the metric $\mathbf{g}(t)$. We can simplify the Ricci flow equation 4. Let $g(t) = e^{2u(t)} g(0)$, then Ricci flow is

$$\frac{du(t)}{dt} = -2K(t). \tag{6}$$

The following theorems postulate that the Ricci flow defined in 4 is convergent and leads to the conformal uniformization metric.

Theorem 2.1 (Hamilton 1982). *For a closed surface of non-positive Euler characteristic, if the total area of the surface is preserved during the flow, the Ricci flow will converge to a metric such that the Gaussian curvature is constant everywhere.*

Theorem 2.2 (Chow [4]). *For a closed surface of positive Euler characteristic, if the total area of the surface is preserved during the flow, the Ricci flow will converge to a metric such that the Gaussian curvature is constant everywhere.*

The corresponding metric $\mathbf{g}(\infty)$ is the *uniformization metric*. Moreover, at any time t, the metric $\mathbf{g}(t)$ is conformal to the original metric $\mathbf{g}(0)$.

The Ricci flow can be easily modified to compute a metric with a *prescribed* curvature $\bar{K}$, and then the flow becomes

$$\frac{dg_{ij}(t)}{dt} = 2(\bar{K} - K) g_{ij}(t). \tag{7}$$

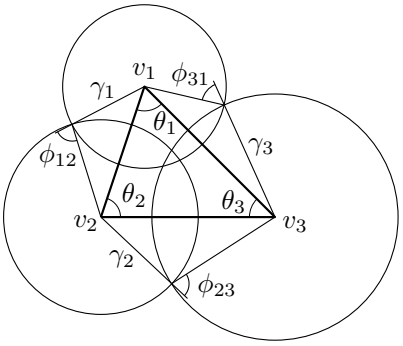


Figure 2. The circle packing metric.

With this modification, any target curvatures $\bar{\mathbf{K}}$, which are admissible with the Gauss-Bonnet theorem, can be induced from the solution metric $\mathbf{g}(\infty)$.

2.3 Discrete Ricci Flow

Smooth surfaces are often approximated by simplicial complexes (triangle meshes). We consider such a triangle mesh Σ with vertex set V, edge set E and face set F.

Discrete Riemannian Metric.

In the discrete setting, the edge lengths on a mesh Σ simply define the Riemannian metric on Σ,

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

such that for a face f_{ijk} the edge lengths satisfy the triangle inequality: $l_{ij} + l_{jk} > l_{ki}$.

The discrete metric determines the angles. Suppose we have a triangle f_{ijk} with edge lengths $\{l_{ij}, l_{jk}, l_{ki}\}$, and the angles against the corresponding edges are $\{\theta_k, \theta_i, \theta_j\}$ (see figure 2). By the cosine law,

$$l_{ij}^2 = l_{jk}^2 + l_{ki}^2 - 2 l_{jk} l_{ki} \cos\theta_k, \tag{8}$$

The discrete Gaussian curvature is defined as the angle deficit on a mesh,

$$K_i = \begin{cases} 2\pi - \sum_{f_{ijk} \in F} \theta_i^{jk}, & \text{interior vertex} \\ \pi - \sum_{f_{ijk} \in F} \theta_i^{jk}, & \text{boundary vertex} \end{cases} \quad (9)$$

where θ_i^{jk} represents the corner angle attached to vertex v_i in the face f_{ijk}.

In the discrete setting, the Gauss-Bonnet theorem (equation 1) still holds on meshes with the discrete Gaussian curvatures, as follows.

$$\sum_{v_i \in V} K_i = 2\pi\chi(M).$$

The circle packing metric was introduced [30, 32] to approximate the conformal deformation of metrics. Let us denote by Γ a function which assigns a radius γ_i to each vertex v_i.

$$\Gamma : V \to \mathbb{R}^+$$

We also define a *weight* function:

$$\Phi : E \to [0, \frac{\pi}{2}].$$

by assigning a positive number $\Phi(e_{ij})$ to each edge e_{ij}. The pair of vertex radii and edge weight functions on a mesh Σ, (Γ, Φ), is called a *circle packing metric* of Σ. Figure 2 illustrates the circle packing metric. Each vertex v_i has a circle whose radius is r_i. On each edge e_{ij}, an intersection angle ϕ_{ij} is defined by two circles of v_i and v_j, which intersect with or are tangent to each other. Two circle packing metrics (Γ_1, Φ_1) and (Γ_2, Φ_2) on a same mesh are *conformal equivalent*, if $\Phi_1 \equiv \Phi_2$. Therefore, a conformal deformation of a circle packing metric only modifies the vertex radii.

For a given mesh, its circle packing metric and the edge lengths on the mesh can be converted to each other by using cosine law.

$$l_{ij}^2 = \gamma_i^2 + \gamma_j^2 + 2\gamma_i\gamma_j \cos\phi_{ij} \quad (10)$$

Let u_i to be $\log \gamma_i$ for each vertex. Then, the discrete Ricci flow is defined as follows.

$$\frac{du_i(t)}{dt} = (\bar{K}_i - K_i) \quad (11)$$

Discrete Ricci flow can be formulated in the variational setting, namely, it is a negative gradient flow of some special energy form.

$$f(\mathbf{u}) = \int_{\mathbf{u_0}}^{\mathbf{u}} \sum_{i=1}^{n} (\bar{K}_i - K_i) du_i, \quad (12)$$

where $\mathbf{u_0}$ is an arbitrary initial metric. The integration above is well defined, and called the *Ricci energy*. The discrete Ricci flow is the negative gradient flow of the discrete Ricci energy. The discrete metric which induces $\bar{k}$ is the minimizer of the energy.

Computing desired metric with prescribed curvature $\bar{K}$ is equivalent to minimizing the discrete Ricci energy. The discrete Ricci energy is strictly convex (namely, its Hessian is positive definite). The global minimum uniquely exists, corresponding to the metric $\bar{\mathbf{u}}$, which induces $\bar{\mathbf{k}}$. The discrete Ricci flow converges to this global minimum [5].

Theorem 2.3 (Chow & Luo: Euclidean Ricci Energy). *The Euclidean Ricci energy $f(\mathbf{u})$ on the space of normalized metric $\sum u_i = 0$ is strictly convex.*

The convergence rate of the discrete Ricci flow using equation 11 is governed by the following theorem

Theorem 2.4 (Chow & Luo). *The Ricci flow 11 converges exponentially fast,*

$$|\bar{K}_i - K_i(t)| < c_1 e^{-c_2 t}, \quad (13)$$

where c_1, c_2 are two positive constants.

3. ALGORITHMS

This section describes the distributed algorithm to compute the virtual coordinates corresponding to the conformal map. The first step is to obtain a triangulation from the network that is a compact manifold with boundaries and is homeomorphic to a bounded subset of $\mathbb{R}^2$. Later, we describe the algorithm to compute the conformal map of this triangulation.

3.1 Obtain a triangulation

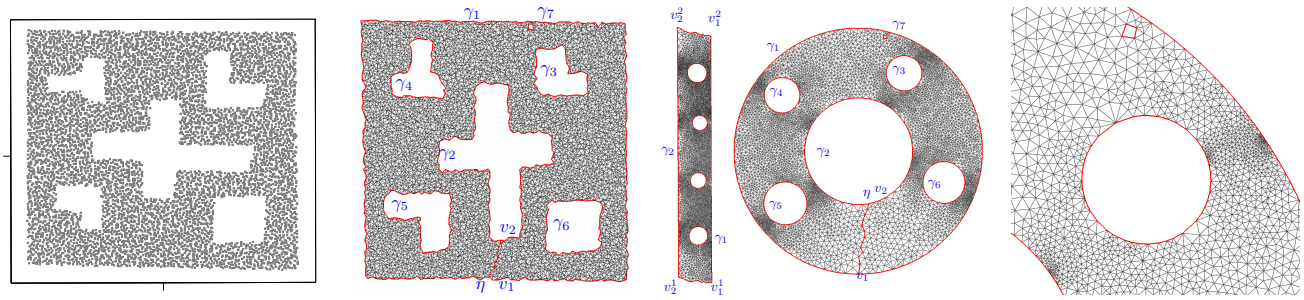
We describe methods for obtaining a triangulation in cases where the nodes are aware of their locations as well as in cases where locations are not available. The methods apply to quasi-unit disk graphs with suitable parameter. For unit disk graphs in the plane, the only requirement is that the graph should be connected. We denote the set of neighbors of a node u as $N(u)$.

3.1.1 Location-based triangulation

A method for computing a triangulation of a unit disk graph through local computations is described in detail in [12]. This graph is called the *restricted delaunay graph (RDG)*. The essential method is as follows:

1. At each node u, compute the delaunay triangulation of $N(u) \cup \{u\}$, denote this triangulation as $T(u)$.
2. For each node u, if an edge (u, v) is in $T(u)$, then it is valid if and only if $(u, v) \in T(x)$, for all common neighbors x of u, v, i.e. $\forall x \in (N(u) \cup \{u\}) \cap (N(v) \cup \{v\})$.
3. All invalid edges are removed.

It is proved in [12] that RDG is connected and planar. The essential reason being that for a pair of crossing edges of a unit disk graph, at least one of the four nodes that lie at



(a) Nodes (b) Triangulation (c) Virtual Coordinates (d) Zoomed in

Figure 3. Algorithm pipeline. Given a network in (a), a triangulation is constructed in (b). The boundaries are traced as γ_k's in (b). A cut between γ_1 and γ_2 is located as η. The triangular mesh is sliced open along η. By using Ricci flow the sliced mesh is flattened to a parallelogram in (c), then mapped to the unit disk with circular holes in (c). Local region is zoomed in as shown in (d), which shows all the triangles are with acute angles.

the end points of the edges is aware of both the edges, and hence can force the crossing to be removed. Our goal here is to obtain virtual coordinates for routing, but we would like to approximate the true topology of the network as closely as possible. The following theorem suggests that at least for dense sensor networks, the method above preserves the topology:

Lemma 3.1. *If a bounded set $\mathcal{R} \subseteq \mathbb{R}^2$ is covered by sensors P such that any disk with unit diameter centered inside $\mathcal{R}$ has at least one node inside, any non-triangular face in the RDG computed above is of distance at most $1/2$ away from the boundary of $\mathcal{R}$.*

PROOF. Denote by $D(P)$ the Delaunay triangulation on P and $D'(P)$ the graph with all the Delaunay edges on P with lengths smaller than 1. It has been shown in [12] that the restricted Delaunay graph RDG computed is a planar graph that includes $D'(P)$. Therefore, any non-triangular face in the RDG must map to a non-triangular face in $D'(P)$. In the following we argue that there can not be a non-triangular face 'deeply inside' the region $\mathcal{R}$. Consider any Delaunay triangle $\triangle uvw$ in $D(P)$, if its circumcircle C is centered inside $\mathcal{R}$, then the circumcircle has radius at most $1/2$ — otherwise the sensor density requirement will put one node inside C, which contradicts with the property that the circumcircle of a Delaunay triangle is empty of any other nodes. This shows that all the three delaunay edges of $\triangle uvw$ have length at most 1 and are then inside $D'(P)$. Therefore any Delaunay triangule $\triangle uvw$ *not* in $D'(P)$ must have its circumcircle centered outside $\mathcal{R}$. The circumcircle has radius greater than $1/2$ — otherwise the three edges are no longer than 1 hence the triangle $\triangle uvw$ is in $D'(P)$. Now we argue that all three nodes u, v, w are within distance $1/2$ from the boundary of $\mathcal{R}$. Otherwise, we can shrink the circumcircle C of $\triangle uvw$ to a unit diameter circle, completely inside C, with its center within $\mathcal{R}$. By the density requirement there must be a node inside the shrunk circle, thus inside C. This again contradicts the Delaunay triangulation property. □

The above Lemma shows that the RDG we get indeed provides a triangular mesh that covers the region $\mathcal{R}' = \{p | p \in \mathcal{R}, d(p, \partial\mathcal{R}) \leq 1/2\}$, which does not include the points within $1/2$ distance from the boundary $\partial\mathcal{R}$.

This distributed algorithm can be adapted to produce a planarization algorithm for connected quasi unit disk graphs (quasi-UDG) [2] of parameter $\alpha \leq \sqrt{2}$. This is done as follows:

1. Compute the RDG with $1/\alpha$ sized disks for neighborhoods instead of unit disk neighborhoods. The RDG algorithm applies without modification, hence produces a planar graph. The planar graph produced may not be connected. It is possible that connectivity of the quasi-UDG relied on some edges longer than $1/\alpha$.

2. Restore connectivity using edges of quasi-UDG.

The following method shows that connectivity can be restored without sacrificing planarity.

Restoring connectivity. First, observe that any two nodes that are within distance $1/\alpha$ must be in the same connected component of the RDG. Therefore, if RDG has more than one connected component, then somewhere there must be a quasi-UDG edge (u, v) longer than $1/\alpha$ where u and v belong to different components. If no edge of the current RDG crosses (u, v), we can simply add the the edge, and connect u and v.

If an existing edge (x, y) in RDG crosses (u, v) at point p, then one of the nodes x, y must be within a distance $1/\alpha$ to one of the nodes u, v, and therefore neighbors in quasi-UDG [19]. Without loss of generality, we assume that u and x are within distance $1/\alpha$. Then these nodes are in the same connected component of RDG. Note that by this argument,

[2] In a *quasi unit disk graph* with parameter $\alpha \geq 1$, if two nodes are within distance $1/\alpha$, an edge between the two exists, if they are at a distance more than 1, the edge does not exist; while for other distances, the existence of the edge is uncertain.

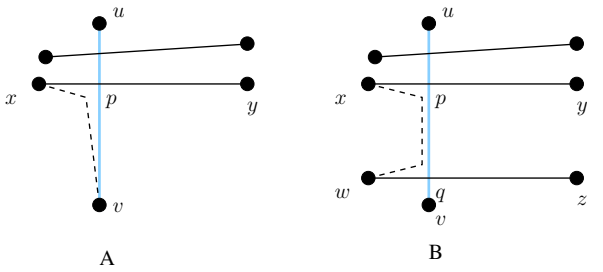


Figure 4. Restoring connectivity. Dark edges are RDG edges, light blue edge is a quasi-UDG edge not in RDG. Dotted lines are virtual edges. (A) Crossing edge belonging to connected component of u (B) Crossing edges belonging to different connected components.

for any RDG edge crossing (u, v) its endpoints must belong to the connected component of u or v.

Now, consider the scenario of 4(A), where the crossing edges belong to the connected component of u. If there are more than one such edges, we consider the edge whose intersection with (u, v) is nearest to v. Observe that by construction, no edge crossing (x, y) can exist in RDG. To restore connectivity, we insert the edge (x, v). This can be done without compromising planarity, since no RDG edge intersects (p, v) or (x, p), we can lay down an edge that is infinitesimally close to the path (x, p, v), that does not intersect any RDG edge. Note that we only need a combinatorial planar graph and do not require a straight-line planar embedding under the original coordinates.

If another crossing edge such as (w, z) exists, in the connected component of v, then we similarly choose to insert the virtual edge (x, w) via the corresponding path (x, p, q, w).

These virtual edges do not correspond to real quasi-UDG edges, communication between their endpoints are achieved in the network by routing through the quasi-UDG edge (u, v).

Orientation and degeneracy removal. We need an oriented planar triangulation to obtain a manifold on which the conformal map can be applied. Therefore, we start with detecting all the triangles (K_3) in $\mathcal{G}$. This is done simply by gathering 2-hop information at each node. Based on this, we find *boundary* edges:

Definition 3.2. Boundary Edge. *Any edge that does not belong to exactly 2 different triangles.*

The boundary cycles can now be determined by traversing connected components of boundary edges. The boundary cycles bound the *holes* in the network.

Then we start with any triangle in $\mathcal{G}$, and assign an arbitrary orientation to it. This determines orientations of all faces of $\mathcal{G}$, and are computed distributedly as follows:

1. Once a triangle is oriented, any triangle adjacent to it can compute its own orientation, by orienting the shared edge in the opposite direction.

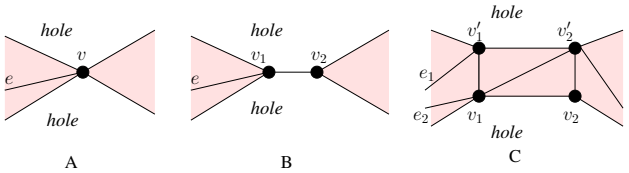


Figure 5. Handling degeneracy. (A) Holes sharing a boundary vertex, (B) Holes sharing a boundary edge and (C) triangulation using virtual nodes.

2. Once an edge on a particular boundary cycle has been identified, that determines the orientation of the hole, and is propagated along the boundary cycle.

Observe that given any vertex, the orientation of edges and faces incident on it, determine a cyclic (clockwise or counterclockwise) order of incident edges.

The planar graph consists of two types of oriented faces - triangles and holes. For our purposes, it is necessary that the union of the triangles form a 2-manifold. At this point, however, the planar graph may not contain such a triangulation. In particular, it is possible that an edge may belong to the boundary cycle of two different holes - see Figure 5(B).

We handle such a case by creating triangulation of virtual nodes (Figure 5(C)). First, the degenerate edge (v_1, v_2) is copied to a new edge (v'_1, v'_2). Then we add in edges (v_1, v'_1), (v_2, v'_2) and diagonal (v_1, v'_2). Note that to maintain a triangulation, one of the edges incident on each of v_1 and v_2 must be duplicated. For example, in Figure 5(C), the edge e has been duplicated. Each other edge incident on an original vertex is assigned to one of the copies, the assignment being uniquely determined by the cyclic order defined by the orientation, and the choice of the duplicated edge. Orientation of all new triangles and edges are also determined uniquely by the orientation of existing faces.

A degeneracy of the form of Figure 5(A) is also possible, where two holes share a common boundary vertex. In this case, we duplicate the degenerate vertex v into v_1, v_2 and connect by an edge. Then we repeat the method for the previous case.

Note that we need not assign coordinates to the new virtual nodes, since we are working purely on the graph structure, and our mapping algorithm does not require coordinates.

This method extends to chains of degenerate edges, dangling edges and to multiple holes sharing a vertex. We omit the details at this point, and skip ahead to the next topic.

3.1.2 Landmark-based triangulation

In a network where location information is not available, we have to rely only on the hop-count distance metric to obtain a triangulation. This is achieved by means of the landmark Voronoi diagram. From the landmarks, we flood messages, that measure distance of other nodes from the landmarks. This creates a set of Voronoi cells in the graph. The adjacency of these cells give rise to a dual *combinatorial de-*

launay complex (CDC).

Landmark based Voronoi and delaunay graphs have been used to do routing in [9, 10].

We follow the approach of [10, 11], to select landmarks. The idea is to choose landmarks such that:

- Any two landmarks are k hops apart (for a small $k = 5$ or 6).
- Any non-landmark node is within k hops of some landmark.

This method requires a flood from a landmark to last only k hops, hence the overall cost is linear for a network of bounded density. This produces a dense set of landmarks. But the CDC obtained from the adjacency is not planar. The following method for obtaining a planar graph from the CDC is described in detail in [11]:

An edge of CDC is valid if there is a path between the corresponding landmarks a and b such that no node on the path has a neighbor that belongs to the Voronoi cell of any landmark other than a and b. The graph formed by the set of valid edges is called the Combinatorial delaunay map (CDM).

This method is proved to produce a planar graph from a quasi unit disk graph ($\alpha \leq \sqrt{2}$). At this point we can apply the methods from the previous discussion to obtain a triangulation. While this landmark based method is more natural and intuitive for large scale networks of high density, it applies in principle to any network corresponding to a suitable quasi unit disk graph. The theory suggests that in certain cases it may be necessary to refine the triangulation further. However, this was not necessary in any of the networks we have tried.

3.2 Compute conformal map

Figure 3 shows the algorithm pipeline for computing the conformal mapping.

Algorithm description.

The conformal mapping can be achieved by using the discrete Ricci flow algorithm. The algorithm only requires the connectivity information. The locations of nodes are irrelevant, and all edges are assumed to be of length 1.

Discrete Ricci flow.

Each node v_i is associated with a disk, with radius e^{u_i}. For simplicity, the length of each link connecting v_i and v_j equals to $e^{u_i} + e^{u_j}$. The corner angles of each triangle can be estimated using cosine law by each node locally. The curvature can be computed by each node directly. Then u_i is modified proportionally to the difference between the target curvature and the current curvature. Once the curvature error is less than a given threshold, the process stops. The details can be found in the algorithm 1.

Flattening.

The virtual coordinates can be estimated by flattening triangle by triangle using the resulting metric (edge length) from the Ricci flow algorithm. First, the root face is chosen. Given three edge lengths of the root triangle $[v_0, v_1, v_2]$, the node coordinates can be constructed directly. Then the neighboring triangle of the root, e.g. $[v_1, v_0, v_i]$, can be flattened, the virtual coordinates of v_i is the intersection of two circles, one is centered at p_0 with radius l_{0i}, the other is centered at p_1 with radius l_{1i}. Furthermore, the normal of the triangle $[v_1, v_0, v_i]$ is consistent with the root triangle. In similar way, the neighbors of the newly flattened triangles can be further embedded. The whole network can be flattened by the flooding process. The details can be found in the algorithm 2.

Algorithm 1 Discrete Ricci Flow

Require: Triangular Mesh M, Target curvature for each vertex $\bar{k}_i$. Error threshold ϵ. Step length δ.

Ensure: Discrete metric (edge lengths) satisfying the target curvature.

1: for all vertex v_i, $u_i \Leftarrow 0$
2: **while** true **do**
3: Compute edge length l_{ij} for edge $[v_i, v_j]$,

$$l_{ij} = e^{u_i} + e^{u_j}$$

4: Compute the corner angle θ_i^{jk} in triangle $[v_i, v_j, v_k]$,

$$\theta_i^{jk} = \cos^{-1} \frac{l_{ij}^2 + l_{ki}^2 - l_{jk}^2}{2 l_{ij} l_{ki}}$$

5: Compute the curvature k_i at v_i

$$k_i = \begin{cases} 2\pi - \sum_{jk} \theta_i^{jk}, & v_i \notin \partial M \\ \pi - \sum_{jk} \theta_i^{jk}, & v_i \in \partial M \end{cases}$$

6: **if** max $|\bar{k}_i - k_i| < \epsilon$ **then**
7: **return** The discrete metric $\{l_{ij}\}$.
8: **end if**
9: Update u_i

$$u_i \Leftarrow u_i + \delta(\bar{k}_i - k_i)$$

10: **end while**

Conformal Mapping.

Given a triangular mesh M, which is a multi-holed annulus (genus zero surface with multiple boundaries), using above algorithm it can be mapped to a canonical unit disk with circular holes. Furthermore, the mapping is an approximation of a conformal mapping. In smooth case, the conformal mapping is unique upto a Möbius transformation as shown in Figure 1. In the following algorithm, the Möbius ambiguity is removed.

First, the boundary loops of M are traced, and sorted by

their lengths decreasingly using hop distance, denoted as γ_k, where k is from 1 to n, as shown in Figure 3 frame (b). Second, the target curvature is set, such that all interior nodes have zero curvatures, nodes on γ_1 and γ_2 are of zero curvatures also. Nodes on γ_k, $k > 2$ has the target curvature $\frac{-2\pi}{|\gamma_k|}$, where $|\gamma_k|$ denotes the length of γ_k. The edge lengths satisfying the target curvatures are computed using the Ricci flow algorithm.

Algorithm 2 Flattening

Require: Triangular Mesh M, Discrete metric $\{l_{ij}\}$ with zero curvatures on all interior vertices.

Ensure: Virtual coordinates for each vertex.

1: for each node v_i, label v_i as *un-accessed*

2: flatten the first triangle $[v_0, v_1, v_2]$, such that

$$p_0 \leftarrow (0,0), p_1 \leftarrow (l_{01}, 0), p_2 \leftarrow l_{20}(\cos\theta_0^{12}, \sin\theta_0^{12}),$$

label v_0, v_1, v_2 as *accessed*.

3: for each un-accessed node v_i, check all its neighboring faces $[v_i, v_j, v_k]$, if v_j, v_k have been accessed, then p_i is the intersection point of two circles

$$|p_i - p_j| = l_{ij}, |p_i - p_k| = l_{ki},$$

furthermore $(p_j - p_i) \times (p_k - p_i) > 0$. Label v_i as *accessed*.

Third, a shortest path η from γ_1 to γ_2 is traced, suppose η intersects γ_1 and γ_2 at v_1, v_2 respectively. M is sliced along η to form another mesh $\tilde{M}$, v_k is split to $v_k^1, v_k^2 \in \tilde{M}, k = 1, 2$. The edge lengths are copied from M to $\tilde{M}$, $\tilde{M}$ is flattened to the plane. The virtual coordinates of $v_k^1, v_k^2, k = 1, 2$ form a parallelogram. Without loss of generality, v_1^1, v_1^2 are mapped to the y-axis and their distance is h , as shown in Figure 3 frame (c). Then by the following conformal map $e^{\frac{2\pi z}{h}}$, $\tilde{M}$ is mapped to the canonical unit disk with circular holes, as shown in Figure 3 frame (c). Then the virtual coordinates of nodes are copied from $\tilde{M}$ to M. This algorithm guarantees to map γ_1 and γ_2 to concentric circles, and the only ambiguity left is a rotation. Detailed description can be found in algorithm 3.

In practice, in order to make the inner holes more circular, the target curvatures of $v_j \in \gamma_k, k > 2$ can be updated and more iterations of Ricci flow algorithm are performed. The target curvatures can be updated using the following formula

$$\bar{k}_j = -2\pi \frac{l_{j-1} + l_j}{\sum_{e_i \in \gamma_k} l_i}$$

where l_j and l_{j-1} are the current edge lengths of edges adjacent to v_j, l_i is the current edge length of e_i which is in the boundary γ_k. In general, 4 or 5 iterations are good enough. Figure 3 frame (c) and (d) show the computational result of this algorithm. In (c), all the boundaries become circular, in (d), all the triangle corners are acute, the triangulation is with good quality.

Algorithm 3 Conformal Mapping

Require: Triangular Mesh M, genus zero with multiple holes.

Ensure: Virtual coordinates p_j for each node v_j, all the boundaries are circular.

1: locate all the boundaries $\gamma_1, \gamma_2, \cdots, \gamma_n$, sorted increasingly according to their lengths using the hop distance.

2: set target curvature,

$$\bar{k}_i = \begin{cases} 0 & v_i \notin \partial M \\ 0 & v_i \in \gamma_1 \cup \gamma_2 \\ \frac{-2\pi}{|\gamma_k|} & v_i \in \gamma_k, k > 2 \end{cases}$$

3: Compute the metric using Ricci flow algorithm 1.

4: compute the shortest cut from γ_1 to γ_2, denoted as η, η intersects γ_1, γ_2 at v_1, v_2 respectively.

5: slice M along η to get another mesh $\tilde{M}$, v_k is split to two nodes v_k^1, v_k^2, where $k = 1, 2$.

6: compute the virtual coordinates of $\tilde{M}$ using algorithm 2, such that $p_1^1, p_1^2, p_2^1, p_2^2$ form a parallelogram. p_1^1, p_1^2 are along y-axis. The distance between them is h.

7: for each node $v_j \in M$, there exists a corresponding node $\tilde{v}_j \in \tilde{M}$, $p_j = (x_j, y_j)$,

$$p_j \leftarrow e^{\frac{2\pi}{h}(x_j + iy_j)}$$

Robustness.

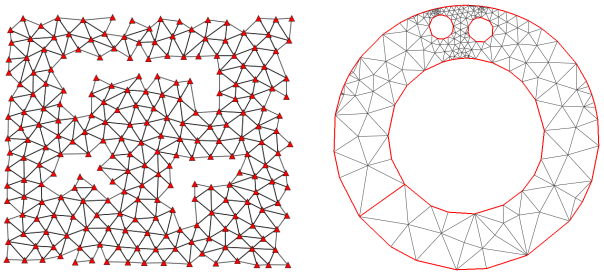
The accuracy of the computation is mainly controlled by the curvature error bound ϵ in the Ricci flow algorithm 1. According to theorem 2.4, the curvature error decreases exponentially fast. Therefore, the number of steps to reach the desired error bound is given by $O(-\frac{\log \epsilon}{\delta})$, where δ is the step size in the Ricci flow algorithm. In practice, if the number of triangles in the network is about tens of thousands and the error bound is about $1e-8$, the algorithm is stable.

3.3 Routing

Having obtained the virtual coordinates, greedy routing is straight forward. As mentioned earlier, a message cannot get stuck on the boundary of a circular hole. It has been shown in [8] that greedy routing cannot get stuck at vertices with an angle less than $2\pi/3$. In the mappings we obtained, all angles were acute. However, in a case of a large angled triangle appearing in the final embedding, the routing can be handled by routing on *edges* as follows. Suppose in $\triangle ABC$ the angle $\angle BAC > 2\pi/3$, and the current routing request to destination D arrives at A, and has no nearer neighbor. We can create a virtual node E (representing the edge BC) and the message is delivered from A to BC (or E) in the sense that the edge BC is closer to the destination. As all the non-hole faces are triangles, the triangle adjacent to $\triangle ABC$ sharing the edge BC must have another edge closer to the destination. Thus greedy routing will guarantee delivery.

Figure 6 shows the effect of greedy routing on the virtual coordinates. The routing cannot be successful under normal greedy routing (Figure 6(a)). The path under the virtual coordinates, however, easily gets past the hole to the other side (Figure 6(b)).

The domain can also be triangulated using landmarks as described in section 3.1.2. Figure 7 shows the CDM triangulation of the domain of Figure 6, and the corresponding virtual coordinates.



(a) (b)

Figure 7. (a) Landmark based triangulation of domain of fig 6. (b) The corresponding virtual coordinate map.

Routing in the landmark based scheme is achieved in the usual way. At every stage, the next Voronoi tile to visit is decided based on the virtual coordinates of the landmarks of neighboring tiles. Then a local routing scheme is applied to reach the chosen neighboring tile. This local routing can be executed in different ways. For example, since the size of the tiles are constant in a bounded density network, it is possible to store a routing table for the entire tile. Alternatively, it is possible to flood a tile from the boundary with each neighbor, and thus obtain paths to neighboring tiles from each node.

In theory, and in practice, the method also works for very fragmented networks with many holes. Figure 8 shows an example.

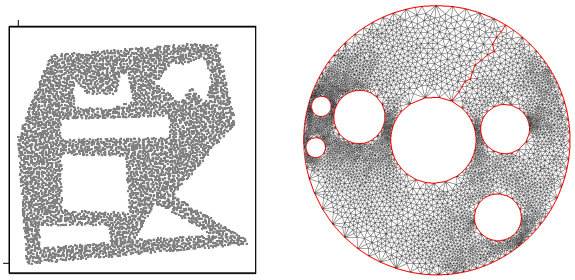


Figure 8. (a) Network of 7000 nodes with many holes (b) Virtual coordinates

4. EXPERIMENTAL RESULTS

We conducted extensive experimentation on UDG or quasi-UDG based network of Figure 6 with about 8700 nodes, and on networks of similar topology but different number of nodes. From a routing performance point of view, the following are important observations from the experiments and simulations:

- **100% Routing guarantee.** We selected 10,000 random source-destination pairs in the network, and performed greedy routing based on the real coordinates and using our virtual coordinates. With the real coordinates, the success rate of routing is only about 52.29%, while with the virtual coordinate greedy routing, we achieve 100% success rate.
- **Small routing stretch.** The path length of the virtual coordinate routing was compared with the shortest path in the graph for 5000 random source-destination pairs. The average stretch (ratio of routing path length to shortest path) was 1.59, while the maximum stretch was 3.21.

Since our mapping algorithm uses a numerical method, we carried out some tests to estimate the convergence time of the algorithm, and compared the results with the convergence of NoGeo [28]. While NoGeo does not guarantee delivery even on full convergence, the comparison is interesting, as described in the introduction. The results are shown in Figure 9. Note that NoGeo iterates on the actual node coordinates, whereas the Ricci-flow reduces the error in the curvature. The result shows that in this case, the Ricci-flow method converges faster than NoGeo, and guarantees delivery.

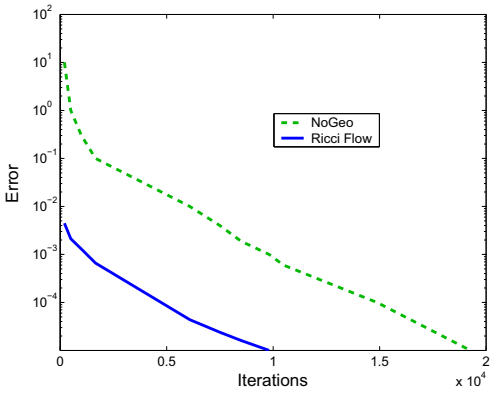
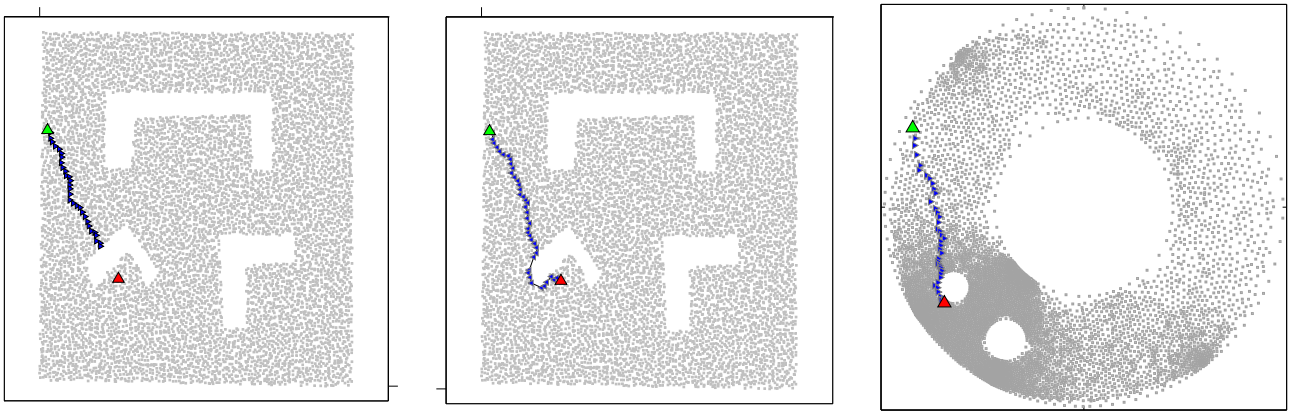


Figure 9. The solid blue curve shows the error in curvature after a number of iterations by the Ricci flow method, the dashed green curve shows the error in location after a number of iterations by NoGeo.

We further compared our method with NoGeo on routing stretch and delivery guarantee on the network of Figure 6, over 5000 source-destination pairs. The results are shown



(a) (b) (c)

Figure 6. Routing. Network of about 8700 nodes, average degree of about 20 in quasi-UDG setting, and nodes in a perturbed grid distribution. (a) Geedy routing gets stuck at a hole boundary. (b) Routing based on virtual coordinates successfully goes around the hole. (c) The routing path in the virtual coordinate space.

in table 1. Our algorithm incurs a larger stretch (ratio of routing path length to shrotest path length) than NoGeo, but guarantes delivery.

Table 1. Stretch and Delivery Comparison

Method	Delivery rate	Avg Stretch	Max Stretch
Our Method	100%	1.59	3.21
NoGeo	83.66%*	1.17	1.54

*NoGeo performs extremely well in simply connected networks and networks with convex holes, as shown in [28]. But as discussed earlier, in this case the presence of concave holes and holes of large aspect ratio affects its performance adversely.

The algorithm was executed on networks of several different sizes but of same essential topology. We measured the number of iterations required to obtain a small enough error on curvature to get a successful embedding and routing. The resulting plot is shown in Figure 10. The curvature error bound was selected as $1e-6$.

Table 2 lists some statistics about the triangulations of different networks shown in the paper.

Table 2. Experimental Statistics

Case	Nodes	Faces	Edges	Holes
Graph of Figure 7	250	378	630	3
Network of Figure 6	8714	17091	25807	3
Network of Figure 3	5299	10179	15482	6

5. CONCLUSION

This work proposes a novel method for greedy routing with guaranteed delivery based on Ricci flow algorithm. The method has solid theoretic background and competitive performance, compared with prior algorithms under the metric of preprocessing cost and routing quality. The distributed nature of the Ricci flow method makes it valuable for the practical applications on wireless sensor networks. Network dynamics such as node failures and resultant topology changes can be handled efficiently by incrementally recomputing the map starting from the previously computed one. Analysis and experiments related to this aspect are under investigation.

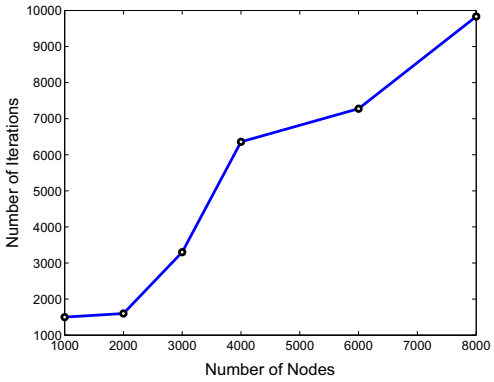


Figure 10. The number of iterations required to obtain a given error bound on the curvature.

Acknowledgement: Jie Gao and Rik Sarkar acknowledge the support from NSF CAREER Award CNS-0643687.

6. REFERENCES

[1] 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.

[2] 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.

[3] M. B. Chen, C. Gotsman, and C. Wormser. Distributed computation of virtual coordinates. In *SCG '07: Proceedings of the twenty-third annual symposium on Computational geometry*, pages 210–219, 2007.

[4] B. Chow. The ricci flow on the 2-sphere. *J. Differential Geom.*, 33(2):325–334, 1991.

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

[6] R. Dhandapani. Greedy drawings of triangulations. In *SODA '08: Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, 2008.

[7] 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.

[8] Q. Fang, J. Gao, and L. Guibas. Locating and bypassing routing holes in sensor networks. In *Mobile Networks and Applications*, volume 11, pages 187–200, 2006.

[9] Q. Fang, J. Gao, L. Guibas, V. de Silva, and L. Zhang. GLIDER: Gradient landmark-based distributed routing for sensor networks. In *Proc. of the 24th Conference of the IEEE Communication Society (INFOCOM)*, volume 1, pages 339–350, March 2005.

[10] S. Funke and N. Milosavljević. Guaranteed-delivery geographic routing under uncertain node locations. In *Proceedings of the 26th Conference of the IEEE Communications Society (INFOCOM'07)*, pages 1244–1252, May 2007.

[11] 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.

[12] 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.

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

[14] R. S. Hamilton. Three manifolds with positive ricci curvature. *Journal of Differential Geometry.*, 17:255–306, 1982.

[15] Z.-X. He and O. Schramm. Fixed points, Koebe uniformization and circle packings. *Ann. of Math. (2)*, 137(2):369–406, 1993.

[16] 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.

[17] 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.

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

[19] A. Kröller, S. P. Fekete, D. Pfisterer, and S. Fischer. Deterministic boundary recognition and topology extraction for large sensor networks. In *Proceedings of the 17th Annual ACM–SIAM Symposium on Discrete Algorithms (SODA'06)*, pages 1000–1009, 2006.

[20] F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger. Geometric ad-hoc routing: of theory and practice. In *PODC '03: Proceedings of the twenty-second annual symposium on Principles of distributed computing*, pages 63–72, 2003.

[21] 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.

[22] J. Li, J. Jannotti, D. Decouto, D. Karger, and R. Morris. A scalable location service for geographic ad-hoc routing. In *Proceedings of 6th ACM/IEEE International Conference on Mobile Computing and Networking*, pages 120–130, 2000.

[23] N. Linial, L. Lovász, and A. Wigderson. Rubber bands, convex embeddings and graph connectivity. *Combinatorica*, 8(1):91–102, 1988.

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

[25] G. Perelman. The entropy formula for the ricci flow and its geometric applications. Technical Report arXiv.org, November 11 2002.

[26] G. Perelman. Finite extinction time for the solutions to the ricci flow on certain three-manifolds. Technical Report arXiv.org, July 17 2003.

[27] G. Perelman. Ricci flow with surgery on three-manifolds. Technical Report arXiv.org, March 10 2003.

[28] 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.

[29] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker. GHT: A geographic hash table for data-centric storage in sensornets. In *Proc. 1st ACM Workshop on Wireless Sensor Networks ands Applications*, pages 78–87, 2002.

[30] K. Stephenson. *Introduction To Circle Packing*. Cambridge University Press, 2005.

[31] R. Thomas and X. Yu. 4-connected projective-planar graphs are hamiltonian. *J. Comb. Theory Ser. B*, 62(1):114–132, 1994.

[32] W. P. Thurston. *Geometry and Topology of Three-Manifolds*. Princeton lecture notes, 1976.