Article

Scalable and Fully Distributed Localization in Large-scale Sensor Networks

Miao Jin [1,*], Su Xia [2], Hongyi Wu [3], and Xianfeng David Gu [4]

[1] The Center for Advanced Computer Studies, University of Louisiana, Lafayette, LA 70504, USA; E-Mail: mxj9809@louisiana.edu

[2] Cisco Systems, Milpitas, CA 95035, USA; E-Mail: suxia.ull@gmail.com

[3] The Center for Cybersecurity, Old Dominion University, Norfolk, VA 23529, USA; E-Mail: h1wu@odu.edu

[4] Department of Computer Science, Stony Brook University, Stony Brook, NY 11790, USA; E-Mail: gu@cs.sunysb.edu

Academic Editor: name

Version May 22, 2017 submitted to Entropy

Abstract: This work proposes a novel connectivity-based localization algorithm, well suitable for large-scale sensor networks with complex shapes and non-uniform nodal distribution. In contrast to current state-of-art connectivity-based localization methods, the proposed algorithm is highly scalable with linear computation and communication costs with respect to the size of the network, and fully distributed where each node only needs the information of its neighbors without cumbersome partitioning and merging process. The algorithm is theoretically guaranteed and numerically stable. Moreover, the algorithm can be readily extended to localization of networks with one-hop transmission range distance measurement, and the propagation of the measurement error at one sensor node is limited within a small area of the network around the node. Extensive simulations and comparison with other methods under various representative network settings are carried out, showing superior performance of the proposed algorithm.

Keywords: Localization; large-scale sensor network; scalable; fully distributed

1. Introduction

Geographic location information is imperative to a variety of applications in wireless sensor networks, ranging from position-aware sensing to distributed data storage and processing, geographic routing, and nodal deployment. While global navigation satellite systems (such as GPS) have been widely employed for localization, integrating a GPS receiver in every sensor of an entire large-scale sensor network is unrealistic. Moreover, some application scenarios prohibit the reception of satellite signals by part or all of the sensors, rendering it impossible to solely rely on global navigation systems.

Even for those ranging information based localization schemes, extra equipments installed to measure the distance or the angle between nodes, can also lead to a dramatically increase of network cost. To this end, many interesting approaches have been proposed for localization with mere connectivity information. Each node only knows which nodes are nearby within its one-hop communication radio range, but does not know how far away and what direction its neighbors are.

1.1. Challenges of Previous Approaches

Previous localization methods with mere connectivity have mainly focused on dimension reduction of multidimensional data sets based on the input distance matrix, which is approximated by hop counts between each possible pair of nodes. The two major methods, multi-dimensional scaling (MDS) based [1–3] and neural network based [4,5] achieve the highest localization accuracy and yield coordinates of sensor nodes that preserve the distance matrix between the data points of the input space and the output space (i.e., a 2D plane) as much as possible.

One of the major problems for MDS based methods is their low *scalability*. The time complexity for obtaining the distance matrix is $O(nm)$ where n and m are the number of vertices and edges, respectively. The time complexity to compute the two largest eigenvalues and the corresponding eigenvectors is $O(n^2)$. With the increase of a network size n, the computational cost is prohibitive. Another issue is that they are inherently *centralized*. As a sensor network grows large in size, centralized computation has a fundamental bottleneck at nodes near the sink as each sensor node has a limited power and computation capability. So a distributed algorithm is highly preferred especially for a large-scale network. Different algorithms have been proposed to overcome these disadvantages. A basic approach is to partition the network to many subnetworks, and compute the localization of each subnetwork, and then merge these subnetworks together. This method requires delicate strategies and great caution in the merging stage.

For neural network based methods, *stability* is their major problem due to the non-convex shape of their minimized energy. Although several approaches have been proposed to increase the possibility to escape from local minima of the minimized energy, the selection of initial values are still crucial for the final localization results [5].

1.2. Our Approach

We propose a novel localization algorithm which overcomes the major difficulties of conventional MDS and neural network based methods. We first explain the basic idea using a smooth and planar surface, and then transform it to sensor network localization.

Let's consider a smooth and planar surface. It is flat everywhere, so the Gaussian curvature, which measures how much a surface is curved and can be computed based on local distance information, equals to zero at every point of the surface. Assume we only have approximated distance information instead of the exact one of the surface. Such approximated distance generates non-zero Gaussian curvatures, which induce a curved surface instead of a planar one. We can distort the approximated distance such that the deformed one generates zero Gaussian curvatures for every point that guarantee a surface can be embedded in plane. The tool we apply to distort the approximated distance is Ricci flow [6].

Given a large-scale sensor network deployed on plane, we can extract a triangular mesh structure from the connectivity graph of the network such that the mesh structure approximates well the geographic structure of the sensor network. Specifically, we uniformly select a set of landmark nodes such that any two neighboring landmarks are a fixed K hops away. Landmarks initiate local flooding to build a landmark-based Voronoi diagram of the network such that any non-landmark node is within k hops of some landmark. We can build a triangulation based on the dual of the landmark-based Voronoi diagram. Each vertex of the triangulation is a landmark node. Each edge connecting two neighboring vertices is a shortest path between the two neighboring landmark nodes.

We can consider the triangular mesh as a discrete approximation of a smooth and planar surface. The local distance information of the mesh is discrete and represented by the approximated edge lengths (i.e., a fixed K-hop). The Gaussian curvature of the mesh is also discrete and can be computed based on the approximated edge lengths. Discrete Ricci flow deforms the length of each edge such that the deformed edge lengths induce zero Gaussian curvature at each vertex of the mesh. The deformed edge lengths guarantee the triangular mesh can be embedded in plane. We call such edge lengths a flat metric. However, there exist infinite number of flat metrics. Each one of them can isometrically embed the triangular network into plane. The key is which one of those flat metrics introduces the minimal deformation to the initially approximated edge length (i.e., achieves the minimal localization error). We prove that we can obtain an optimal flat metric with the least deformation from an initially non-flat one by controlling the condition of boundary vertices. The theorem is the foundation of our paper. Computing the optimal flat metric is the key step of our algorithms. Based on the computed optimal flat metric, localization (i.e., isometric embedding of the network to plane) is straightforward.

In our algorithm, all the involved computations for each node only require information from nodes within its one-hop transmission range, therefore it is *fully distributed* without cumbersome cutting and merging process. The proposed method is *numerically stable*, free of the choice of initial values and local minima with theoretical guarantee. The computational cost and communication cost are both linear to the size of the network, so the method is *scalable*, suitable for large ad-hoc networks with thousands of highly resource-constrained sensor nodes (processor, memory, and power) that have limited communication radio range.

Although accurate distance measurement is too expensive, or even impossible for a network with thousands or even millions of sensors, it is possible for a node to approximate its distance to nearby nodes within one-hop transmission range in practice. If the range distance information is available, even with unavoidable measurement errors, our algorithm can incorporate the approximated one-hop distance information to further improve the localization accuracy. The impact of the measurement error at one node on the localization of other nodes of the network decreases dramatically in terms of the distance between them. This *limited error propagation* also contributes to the high scalability and localization accuracy of our proposed algorithms.

The rest of this paper is organized as follows. Section 2 gives a brief review of related works. Section 3 introduces the theoretic background of the proposed optimal flat metric based localization algorithms. Section 4 describes the localization algorithm with mere connectivity information step by step. Section 5 discusses the localization algorithm with one-hop distance measurement. Section 6 presents the simulation results under various scenarios and different network topologies. Extensive comparison of the proposed localization algorithm with previous methods are also conducted in Section 6. Section 7 concludes the paper and gives future directions.

2. Related Works

With merely connectivity information available, three major techniques are employed in current state-of-the-art localization schemes: multi-dimensional scaling (MDS), neural networks, and graph embedding.

MDS is a non-linear dimension reduction and data projection technique that transforms distance matrix into a geometric embedding (e.g., a planar embedding for 2D sensor network localization). MDS-based localization is originally proposed in [1]. It constructs a proximity matrix based on the shortest path distance (approximated by hop counts) between all pairs of nodes in the network. The singular value decomposition (SVD) is employed to produce the coordinates matrix that minimizes the least square distance error. Finally, it retains the first 2 (or 3) largest eigenvalues and eigenvectors as 2D (or 3D) coordinates. Subsequent improvements on MDS are made by dividing the graph into patches to enable distributed calculation [2,3]. In addition [7] proposes to apply SVD to the matrix based on a set of beacon nodes only and thus reduces complexity. A similar idea is adopted in [8], with the simplex method (instead of SVD) for error minimization.

The second method is based on neural networks [4,5], where non-linear mapping techniques and neural network models such as self-organizing map (SOM) are employed for dimension reduction of multidimensional data sets, yielding coordinates of sensor nodes that preserve the distances (also approximated by hop counts) between the data points of the input space and the output space (i.e., a 2D plane) as much as possible.

The localization algorithms based on rigid graph embedding theory [9–11] aim to create a well-spread and fold-free graph that resembles the given network. They focus on finding a globally rigid graph which can be embedded without ambiguity in plane. While with mere globally rigid structure, like a topological disk triangulation in [9], there exist infinite number of flat metrics that induce different planar embedding as long as the total Gaussian curvatures satisfy the discrete Gauss-Bonnet Theorem as discussed in Sec. 3. A brute-force way is applied to find one planar embedding of the extracted global structure, which in general can not be easily guaranteed. So compared with MDS and neural network-based approaches, the graph rigidity-based methods exhibit lower localization accuracy in general.

A special graph embedding problem is unit disk graph embedding. Assuming the transmission model of a sensor network is unit disk graph (UDG) model with transmission range 1; there exists an edge between two nodes in the connectivity graph of the network if and only if the Euclidean distance between the two nodes is no more than 1. Embedding such a unit disk graph into plane is NP-hard. Kuhn et al. in [12] proved a non-approximability result for the problem of embedding a given unit disk graph. Later, Moscibroda et al. in [13] introduced a polynomial algorithm to embed a unit disk graph into plane. The quality of the embedding, measured as the ratio of the maximum edge length to the minimum distance between two non-neighboring nodes, is $O(\log^{2.5} n\sqrt{\log\log n})$ with n the size of the network. Note that the unit disk graph embedding achieves a different goal compared with the discussed localization algorithms. Given a network modeled by UDG, a unit disk graph embedding minimizes the violation of the unit disk constraints, but not necessarily the localization error that is in general measured as the ratio of the average node distance error and the average one-hop communication radio range.

Localization with noisy distance and angle information is discussed and distributed localization methods are proposed in [14]. If only distance measurement information is available, both the multi-dimensional scaling (MDS) and neural network based methods are flexible to incorporate distance information to improve localization accuracy. Previous research assumes a given network has not only a small fraction of sensors with a priori coordinate knowledge but also distance measurement between pairs of nodes with long distances. One representative work is the distributed weighted-multidimensional scaling (dwMDS) algorithm proposed to give more weight on range measurements that are believed to be more accurate and select neighbors adaptively [15]. While in this paper, we limit the discussion to localization based on distance measurement within one-hop communication radio range to limit the communication cost of a large-scale sensor network.

3. Optimal Flat Metric

Before giving the details of the proposed optimal flat metric based localization algorithms in Sec. 4 and Sec. 5, we introduce briefly the concepts of metric and Gaussian curvature in Sec. 3.1, which provide the necessary background knowledge of the algorithms. Specifically, we introduce discrete surface Ricci flow in Sec. 3.2, a tool we apply to compute the optimal flat metric that induces a planar localization of a given sensor network with a minimal localization error. We then discuss the condition to find the optimal flat metric and give the Optimal Flat Metric Theorem that serves as the foundation of the proposed algorithms in Sec. 3.3. Table 1 summaries the symbols we use in the paper.

3.1. Discrete Metric and Gaussian Curvature

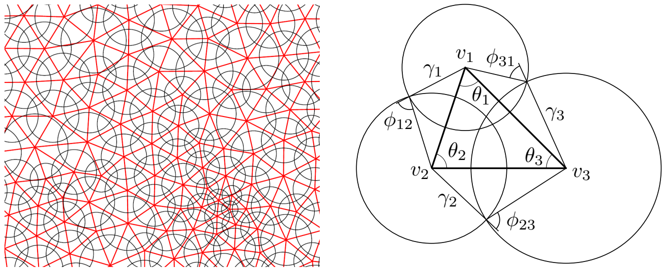
In discrete setting, we denote $M = (V, E, F)$ a triangulated surface (or mesh in short) embedded in $\mathbb{R}^3$, consisting of vertices (V), edges (E), and triangle faces (F). Specifically, we denote $v_i \in V$ a vertex with id i; $e_{ij} \in E$ an edge with two ending vertices v_i and v_j; $f_{ijk} \in F$ a triangle face with vertices v_i, v_j, and v_k.

Definition 1 (Discrete Metric)**.** *A discrete metric on M is a function $l : E \to \mathbb{R}^+$ on the set of edges, assigning to each edge $e_{ij} \in E$ a positive number l_{ij} such that all triangles satisfy the triangle inequalities $f_{ijk} \in F$: $l_{ij} + l_{jk} > l_{ki}$.*

Edge lengths of M define a discrete metric. If M can be embedded in the Euclidean plane $\mathbb{R}^2$, we call its metric *a flat metric*.

Definition 2 (Discrete Gaussian Curvature)**.** *Denote θ_i^{jk} the corner angle attached to Vertex v_i in Face f_{ijk}, and ∂M the boundary of M, the discrete Gaussian curvature K_i on $v_i \in V$ is defined as the angle deficit at v_i:*

$$K_i = \begin{cases} 2\pi - \sum_{f_{ijk}\in F} \theta_i^{ij}, & v_i \notin \partial M, \\ \pi - \sum_{f_{ijk}\in F} \theta_i^{jk}, & v_i \in \partial M. \end{cases} \tag{1}$$



(a) (b)

Figure 1. Circle Packing Metric. (a) Flat circle packing metric on a triangular mesh (b) Circle packing metric on a triangle.

We can compute corner angles directly from edge lengths, so the discrete metric solely determines the discrete Gaussian curvature of M. If M can be embedded in plane, it is intuitive and also obvious from Eqn. 1 that the flat metric induces zero Gaussian curvatures for all non-boundary vertices.

The well-known Gauss-Bonnet Theorem says that the total Gaussian curvature of M is solely determined by its topology:

Theorem 1 (Discrete Gauss-Bonnet Theorem)**.** *Denote b the number of boundaries. Denote $\chi(M)$ the Euler characteristic number of M and $\chi(M) = 2 - b$. The total Gaussian curvature of M is a topological invariant. It holds as follows:*

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

3.2. Discrete Surface Ricci Flow

Ricci flow was first introduced by Richard Hamilton in his seminal work [6] in 1982. Suppose S is a smooth surface with a Riemannian metric $\mathbf{g}$. The Ricci flow deforms the metric $\mathbf{g}(t)$ according to the Gaussian curvature $K(t)$ (induced by itself), where t is the time parameter

$$\frac{dg_{ij}(t)}{dt} = -2K(t)g_{ij}(t). \tag{3}$$

If we replace the metric in Eqn. 3 with $g(t) = e^{2u(t)}g(0)$, then the Ricci flow can be simplified as

$$\frac{du(t)}{dt} = -2K(t), \tag{4}$$

which states that the metric should change according to the curvature.

The Ricci flow can be easily modified to compute a metric with a *user-defined* curvature $\bar{K}$ as the following,

$$\frac{du(t)}{dt} = 2(\bar{K} - K). \tag{5}$$

With this modification, the solution metric $\mathbf{g}(\infty)$ can be computed, which induces the curvature $\bar{K}$.

Conformal metric deformation preserves infinitesimal circles and the intersection angles among them. Ricci flow defined in Eqn.3 is proved in [27,28] convergent. The final metric is conformal to the original one. Moreover, at any time t, the metric $\mathbf{g}(t)$ is conformal to the original one $\mathbf{g}(0)$.

Later, Chow and Luo [23] proved a general existence and convergence theorem for discrete Ricci flow on surfaces.

To briefly introduce the concept of discrete surface Ricci flow, we start from the concept of circle packing metric given by Thurston in [24] as shown in Fig. 1. We assign each v_i a circle and denote γ_i its radius. The radius function is $\Gamma : V \to \mathbb{R}^+$. The two circles at v_i and v_j of edge e_{ij} intersect with an acute angle, denoted as ϕ_{ij} and called the *weight* of e_{ij}. The edge weight function is then $\Phi : E \to [0, \frac{\pi}{2}]$.

The length l_{ij} of e_{ij} can be computed from the circle radii of its two ending vertices γ_i, γ_j and its weight ϕ_{ij} from the cosine law:

$$l_{ij}{}^2 = \gamma_i{}^2 + \gamma_j{}^2 + 2\gamma_i\gamma_j \cos\phi_{ij}. \tag{6}$$

Definition 3 (Circle Packing Metric)**.** *A circle packing metric of a mesh M includes the circle radius function and the edge weight function.*

Given a discrete surface, we use circles with finite radii to approximate the infinitesimal circles; conformal deformation of a circle packing metric to approximate continuous metric deformation. Since two circle packing metrics (Γ_1, Φ_1) and (Γ_2, Φ_2) on the same mesh are *conformally equivalent* if $\Phi_1 \equiv \Phi_2$, a *conformal deformation* of a circle packing metric only modifies the vertex radii and preserves the intersection angles on the edges.

Denote $u_i = \log\gamma_i$, $\bar{K}_i$ and K_i the target and current Gaussian curvatures of v_i respectively, and t the time. The discrete Ricci flow is defined as follows:

$$\frac{du_i(t)}{dt} = (\bar{K}_i - K_i(t)). \tag{7}$$

Discrete Ricci flow continuously deforms the circle packing metric according to the difference between the current and target Gaussian curvatures in a heat-like diffusion process. Convergence of discrete surface Ricci flow is proved in [23]. The final circle packing metric induces the one that satisfies the target Gaussian curvatures.

Discrete Ricci flow can be formulated in a variational setting, namely, it is a negative gradient flow of a special energy form. We define a mesh M with edge weight Φ a *weighted mesh*, denote as (M, Φ). We represent a circle packing metric on (M, Φ) by a vector $\mathbf{u} = (u_1, u_2, \cdots, u_n)^T$, and Gaussian curvatures at vertices by the curvature vector $\mathbf{K} = (K_1, K_2, \cdots, K_n)^T$, where n is the number of verices. For two arbitrary vertices v_i and v_j, the following symmetric relation holds:

$$\frac{\partial K_i}{\partial u_j} = \frac{\partial K_j}{\partial u_i}.$$

Let $\omega = \sum_{i=1}^n K_i du_i$ be a differential one-form [29]. The symmetric relation guarantees that the one-form is closed (curl free) in the metric space.

$$d\omega = \sum_{i,j} (\frac{\partial K_i}{\partial u_j} - \frac{\partial K_j}{\partial u_i}) du_i \wedge du_j = 0.$$

By Stokes theorem, the following integration is path independent and called *discrete Ricci energy*.

$$f(\mathbf{u}) = \int_{\mathbf{u_0}}^{\mathbf{u}} \sum_{i=1}^{n} (\bar{K}_i - K_i) du_i, \tag{8}$$

where $\mathbf{u_0}$ is an initial circle packing metric that induces the surface original metric. The discrete Ricci energy has been proven to be strictly convex in [23]. The global minimum uniquely exists, corresponding to the desired metric $\bar{\mathbf{u}}$ that induces user-defined curvature $\bar{\mathbf{k}}$. The discrete Ricci flow is a negative gradient flow of the discrete Ricci energy, converging to the global minimum.

Furthermore, the speed of convergence can be estimated by the following formula [23]:

$$|K_i(t) - \bar{K}_i| < c_1 e^{-c_2 t}, c_1, c_2 > 0,$$

where c_1 and c_2 are constant, namely the convergence is exponentially fast.

3.3. Optimal Flat Metric

Analytically, the distortion of the metric at each vertex is given by u_i. This motivates us to define the *total distortion energy* as

$$E(\mathbf{K}) = \int_{\mathbf{K_0}}^{\bar{\mathbf{K}}} \sum_{i=1}^{n} u_i dK_i,$$

where $\overline{\mathbf{K}}$ and $\mathbf{K_0}$ represent the set of target and initial vertex Gaussian curvatures, respectively. The integration is along an arbitrary path from $\mathbf{K_0}$ to the target curvature $\overline{\mathbf{K}}$. This energy is the Legendre dual to the Ricci energy given in Eqn. 8. Therefore it is also convex, and it has a unique global minimum for a given $\overline{\mathbf{K}}$.

All the possible $\mathbf{u}$'s form the *admissible metric space*, and all the possible $\mathbf{K}$'s form the *admissible curvature space*. According to the Gauss-Bonnet theory (Eq. 2), the total Gaussian curvatures of M must be $2\pi\chi(M)$. A curvature vector $\mathbf{K}$ is *admissible* if there exists a metric vector $\mathbf{u}$ on M, which induces $\mathbf{K}$.

Define

$$\Omega = \{\mathbf{K} \quad | \quad \sum_{i=1}^{n} K_i = 2\pi\chi(M) \quad \text{and} \quad K_i = 0 \quad \forall v_i \notin \partial M\}$$

the set of $\mathbf{K}$'s such that the sum of Gaussian curvatures of vertices satisfies the Gauss-Bonnet Theorem and the Gaussian curvature of all interior vertices is zero. Ω induces the set of all possible flat metrics of M.

We formulate our problem as:

$$\min_{\mathbf{K}\in\Omega} E(\mathbf{K}). \tag{9}$$

Among all possible flat metrics of M induced from the set of $\mathbf{K}$'s, we want to find the one introducing the least distortion from the initially estimated curved metric of M.

Theorem 2 (Optimal Flat Metric Theorem)**.** *The solution to the optimization problem 9 is unique, and satisfies*

$$u_j = const, \forall v_j \in \partial M. \tag{10}$$

Proof. The distortion energy $E(\mathbf{K})$ is convex. The domain Ω is a linear subspace of the original domain $\{\mathbf{K}|\sum_i K_i = 2\pi\chi(M)\}$. Therefore the restriction of $E(\mathbf{K})$ on Ω is still convex, it has a unique global optimum at an interior point. The gradient of the energy is $\nabla E(\mathbf{K}) = (u_1, u_2, \cdots, u_n)$. At the optimal point, the gradient is orthogonal to Ω. Assume $v_i \in \partial M, 1 \leq i \leq m$, then the normal vector to Ω is given by $(1, 1, \cdots 1, 0, \cdots, 0)$. Therefore the gradient is along the normal vector. So Eqn. 10 holds. □

We set the constant in Equation 10 as 1. We also set the following two conditions for the target metric of discrete Ricci flow: set the target Gaussian curvatures of all interior vertices to zero so that the final metric is a flat one; deform only the metrics of interior vertices during the process of deforming the estimated metric so that the final flat metric is an optimal flat one. We apply discrete Ricci flow to find the target metric. Since discrete Ricci flow is proved in [23] negative gradient flow of a convex shape energy, it is guaranteed to find the optimal flat metric regardless of the step length or initial values.

4. Localization with Mere Connectivity

We explain the proposed localization algorithm with mere network connectivity in this section. Specifically, the input of algorithm is a connectivity graph of a network. Vertices represent the sensor nodes. Edges represent the communication links. Each sensor node knows a set of neighboring nodes within its one-hop communication range without the knowledge of distance or angle. The output of algorithm is a set of planar coordinates of the whole network nodes that may differ the set of GPS positions of the network a scaling, translation, and rotation. We

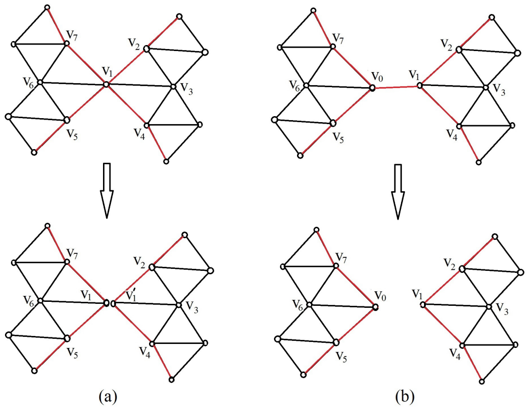


Figure 2. (a) Dangling vertex: Vertex v_1 belongs to two disconnected boundaries with boundary edges marked with red color. After vertex split, the two boundaries are merged to one. Now a boundary vertex belongs to one boundary only. (b) Dangling edge: Edge e_{01} belongs to two disconnected boundaries with boundary edges marked with red color. After edge removal, the two boundaries are merged to one. Now a boundary edge belongs to one boundary and is adjacent with one triangle.

first extract a triangular mesh from the connectivity graph and then apply discrete surface Ricci flow to compute an optimal flat metric as introduced in Sec. 4.1. We then isometric embed the network to 2D plane based on the computed flat metric in Sec. 4.2. We analyze the time complexity and communication cost of the algorithm in Sec. 4.3 and provide discussions in Sec. 4.4.

4.1. Computing Optimal Flat Metric with Mere Connectivity

Given a large-scale sensor network with mere connectivity, we apply a simple distributed scheme as discussed in [16–18] to uniformly select a set of landmark nodes. Any two neighboring landmarks are a fixed k hops away ($k = 4$ in our tests) in the connectivity graph of the network. Landmarks initiate local flooding to build a landmark-based Voronoi diagram of the network such that any non-landmark node is within k hops of some landmark. The adjacency of these Voronoi cells give rise to a dual combinatorial Delaunay complex (CDC). CDC is not necessarily planar because there may exist some crossing edges. We then apply the method introduced in [18] to remove crossing edges from the CDC. Vertices of the resulting planar graph are the set of landmarks. An edge connecting two neighboring vertices is a shortest path between the two landmarks in the network. A face of the planar graph is an n-polygon with $n \leq 3$. We can simply consider all n-polygons with $n > 3$ as inner holes of the network. In practice, we add one virtual edge (i.e., a shortest path) connecting a pair of non-neighboring vertices of a 4-polygon. A 4-polygon is then split to two triangles.

A landmark node-based graph may exist dangling vertices or edges as shown in Figure 2. A dangling vertex is a vertex belonging to more than one boundaries. We remove dangling vertices by vertex split operation. As shown in Figure 2 (a), v_1 is connected with boundary vertices v_2, v_4, v_5, and v_7 that belong to two disconnected boundaries with boundary edges marked with red color. We find a sensor node closest to v_1 and mark it as a landmark node too. Denote the newly added landmark as v'_1. v_1 is connected with v_5 and v_7, and v'_1 is connected with v_2 and v_4. The previously disconnected boundaries are merged to one. Similarly, a dangling edge is an edge belonging to more than one boundaries. We remove dangling edges by edge removal operation. As shown in Figure 2 (b), e_{01} belongs to two disconnected boundaries. We remove this edge such that the previously disconnected boundaries are merged to one. The graph now is a triangular mesh with each face a triangle, a boundary vertex or edge belonging to exactly one boundary.

Denote the constructed triangular mesh M, we assume each edge of M with unit edge length. We then construct an initial circle packing metric (Γ_0, Φ) of M. We assign each vertex v_i a circle with initial circle radius $\gamma_i = 1$, so initially $u_i = \log \gamma_i = 0$. The intersection angle of the two unit circles at v_i and v_j is the edge weight of e_{ij}. Since a boundary edge of M is adjacent with only one triangle face, we can easily detect them and then mark their vertices. For those non-marked vertices (i.e., non-boundary vertices), we set their target Gaussian curvatures $\bar{K}$ to zero.

In each iteration of discrete Ricci flow, only non-marked vertices are involved. Specifically, each non-marked v_i collects the u values from its direct neighbors and updates its adjacent edge length with $l_{ij} = e^{(u_i+u_j)}$. For each triangle f_{ijk} adjacent with v_i, v_i can easily compute the corner angle θ_i^{jk} based on the inverse cos law:

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

Then v_i computes its current discrete Gaussian curvature K_i as the excess of the total angle sum at v_i as Eqn. 1. If for every non-marked v_i, the difference between its target Gaussian curvature $\bar{K}_i$ that is set to zero and current Gaussian curvature K_i is less than a threshold (we set it to 1×10^{-5} in our tests), the discrete Ricci flow converges. Otherwise, each non-marked v_i updates its u_i: $u_i = u_i + \delta(\bar{K}_i - K_i)$, where δ is the step length (we set it to 0.1 in our tests).

When the algorithm stops, all the interior vertices have zero Gaussian curvature, which induces a flat metric of M. Since in each step of the algorithm, there is always no deformation of circle radii for boundary vertices (e.g., $u_i - u_i^0 = 0, v_i \in \partial M$). According to Theorem 1, the computed flat metric introduces the least distortion to the initially estimated metric.

4.2. Isometric Embedding

Isometric embedding is a propagation process, starting from one vertex, embedding the whole triangular mesh into plane with computed optimal flat metric (i.e., edge length) preserved. For simplicity, we let the vertex with the smallest ID (denoted as v_1) initiate the embedding process. Its planar coordinates are set to $(0,0)$, denoted as $uv(v_1)$. Then it arbitrarily selects one of its direct neighbors, e.g., v_j, and sets the coordinates of v_j to $(0, l_{ij})$. For vertex v_k, adjacent to both v_i and v_j, it calculates the intersection points of the two circles with centers at v_i and v_j, and radii of l_{ik} and l_{jk}, respectively. Then, v_j chooses one of the intersection points that satisfies $(uv(v_j) - uv(v_i)) \times (uv(v_k) - uv(v_i)) > 0$ as the coordinates of v_k. The procedure continues until all vertices of M have their planar coordinates.

If a triangular network has multiple boundaries (e.g., inner holes), we need to slice holes open to change the topology of the triangular network to a disk before embedding. For each boundary, an initiator with the smallest ID is elected. Since a boundary forms a closed loop, a boundary node always has exactly two boundary neighbors. The initiator will send out a message to a randomly chosen one of its boundary neighbors. The message contains a counter recording the size of the boundary that is initialized to one. When the neighbor receives the message, it will increase the counter by one and then forward the message to its boundary neighbor that is not the sender of the message. Eventually, the message will come back to the initiator. The initiator will then advertise the size of the boundary via a simple flooding on the triangular network. As a result, each initiator learns the sizes of boundaries of the network. Denote B_i a boundary and B_0 the longest one. For each B_i $(i > 0)$, through a local flooding, its initiator finds a shortest path L_i to B_0. Then holes are "sliced" open along the set of shortest paths $(L_1, L_2, ...)$, where each vertex on L_i is split to two virtual vertices with one on each side. Such spliced vertex will have two sets of coordinates after embedding and it will use the average as its planar coordinates.

For planar embedding of non-landmark nodes, each node n_i finds its three nearest landmarks, denoted as v_1, v_2, v_3 with planar coordinates (x_1, y_1),(x_2, y_2), and (x_3, y_3) respectively. Let d_1, d_2, and d_3 be the shortest distances (hop counts) of n_i to v_1, v_2, v_3 respectively. Then n_i computes its planar coordinates (x_i, y_i) simply by minimizing the mean square error among the distances:

$$\sum_{j=1}^{3}(\sqrt{(x_i-x_j)^2+(y_i-y_j)^2}-d_j)^2. \tag{11}$$

It is worth to note that the step of the isometric embedding of the proposed approach is fundamentally different with graph rigidity based localization methods [9]. For an extracted global planar structure, our algorithm computes first the optimal flat metric with discrete Gaussian curvature equals to zero for all interior vertices, which guarantees that the embedding process can be determined at each step for every single edge. While in [9], the extracted structure is embedded to plane by minimizing a least square energy which can't guarantee a global planar embedding and the embedded network can still curve around and self intersect.

4.3. Time Complexity and Communication Cost

The algorithm of extracting a triangular structure M from a network is completely local, so both its time complexity and communication cost are linear to the size of the network, denoted as n (the number of sensor nodes).

We then apply discrete Ricci flow to compute the optimal flat metric of M. The time complexity (the number of iterations) of discrete Ricci flow is given by $-C\frac{log\varepsilon}{\delta}$, where C is a constant, ε is a threshold of curvature error, and δ is the step length of each iteration [23]. Denote m the size of M. Since each vertex only needs to exchange u values with its direct neighbors, the communication cost is given by $O(-C\frac{log\varepsilon}{\delta}gm)$, where g is the average vertex degree of M. Note that $m << n$ for a general network, and g is six for a triangular mesh.

During the isometric embedding step, two rounds of flooding are involved to slice holes of M open with communication cost $O(m)$. The embedding of M is a propagation process, with both the time complexity and communication cost $O(m)$.

4.4. Discussions

As indicated in [18], a large k to select landmarks decreases the number of crossing edges of a CDC. In practice, we prefer a small k such that the extracted planar graph can resemble the geometry of the underlying domain of a network better. In our simulations, $k = 4$ introduces only very few crossing edges on the CDCs of the testing networks. For the resulting non-triangular face of a planar graph with the removal of the detected crossing edges, we add virtual edges (i.e., shortest paths) connecting pair of non-neighboring vertices of a non-triangular face to split it to triangle faces.

We apply the method introduced in [18] to remove crossing edges of a CDC. The method is proved to produce a planar graph if the transmission model of a given network is either unit disk graph (UDG) or quasi-unit disk graph (Quasi-UDG). In Sec. 6.2, we apply the same network to evaluate the performance of the proposed localization method with mere connectivity under different transmission models including UDG model, Quasi-UDG model, Log-Norm model, and Probability model. We assume sensor nodes of the network have identical transmission range. Legal triangular meshes can still be generated for the network with different transmission models, although the localization accuracy is lower for the network with Log-Norm model or Probability model.

5. Localization with Distance Measurement

It is possible for some network to estimate the Euclidean distance between adjacent sensor nodes that are within the one-hop transmission range of each other. Two basic approaches are Received Signal Strength (RSS) and Time-Differential-of-Arrival (TDoA). RSS measures the power of the signal at the receiver and calculates the distance according to the propagation loss model. TDoA measures the differential propagation time of the received signal to determine the distance. Multiple measurements can be averaged to obtain more accurate results. RSS is

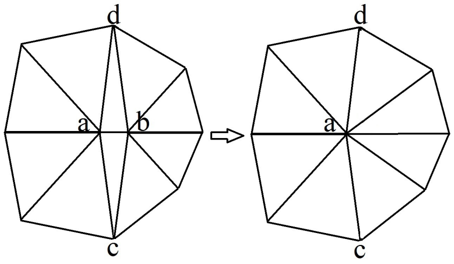


Figure 3. Short edge collapse: the operation of short edge collapse combines v_a and v_b to v_a and removes faces f_{abc} and f_{abd}.

easier to implement, while TDoA may achieve higher accuracy, but in general a measurement error is unavoidable. In our previous experiments [19], we let each sensor periodically broadcasts a beacon message containing its node ID to its neighbors in a sensor network formed with Crossbow motes. Based on received beacon messages, a node builds a neighbor list with the RSSI (received signal strength indicator) of corresponding links. The node applies RSSI to estimate the length of links by looking up an RSSI-distance table established by experimental training data. Our preliminary tests show that, under low transmission power, such estimation has an error rate around 20%.

When distance information is available, we extend the proposed localization algorithm to incorporate it to achieve higher localization accuracy than with mere network connectivity. We give the implementation details of the algorithm in Sec. 5.1 and 5.2. We then discuss time complexity and communication cost of the algorithm in Sec. 5.3. In Sec. 5.4, we show how distance measurement error at a single node of a network affects the localization performance of the whole network.

5.1. Constructing Triangulation

Given a sensor network with distance measurement within one-hop transmission range, we apply our previously proposed algorithm [21] to generate a refined triangular mesh M such that vertices of M are the set of sensor nodes. An edge between two neighboring vertices indicates the communication link between the two sensors, and the edge length is the measured distance. The triangulation algorithm is distributed with no constraints on communication model.

However such triangular mesh may have unavoidable skinny triangles especially when the node distribution is non-uniform. Note that a triangular mesh constructed based on mere network connectivity has all equilateral triangles because we assume a unit edge length for each edge and the assumption is independent of the node density and distribution. As discussed in our previous work [20], solving discrete surface Ricci flow can be considered as solving a linear equation of discrete Lapalce-Beltrami operator defined on a triangular mesh. Skinny triangles increase the condition number of the linear equation, so a triangular mesh M with many skinny triangles may affect the numerical stability and the convergence speed of surface Ricci flow. So we need the following post-processing on a constructed triangular mesh with skinny triangles.

For each f_{ijk}, we compute a ratio between the longest and shortest edge lengths of f_{ijk} and denote it as $r_{f_{ijk}}$. Triangles with extremely high r_f represent those skinny ones. To remove them, we define one operation, called short edge collapse as illustrated in Fig. 3. f_{abc} and f_{abd} have very high r_f values. Both share the short edge e_{ab}. The short edge collapse operation combines the two vertices v_a and v_b to one of them and removes faces f_{abc} and f_{abd}. In practice, we choose the vertex which results in smaller r_f values than the other one after short edge collapse. As in the example shown in Fig. 3, v_a is chosen as the combined vertex. If the operation of one short edge collapse generates a larger r_f than previous ones, we won't collapse the edge.

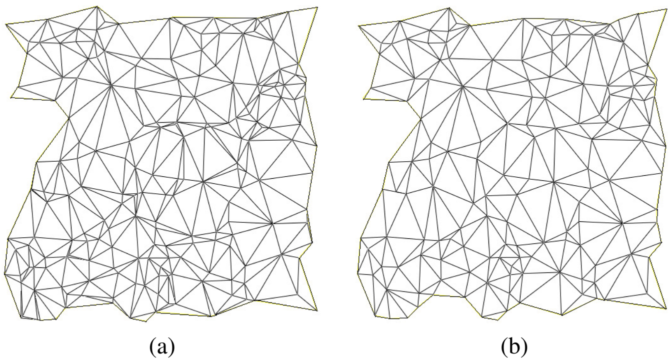


Figure 4. (a) A triangular mesh with too many skinny triangles is constructed from a network based on algorithm proposed in [21]. Its maximal r_f value is close to 27. (b) After a series of short edge collapse operations, the maximal r_f value of the triangular mesh is less than 4.

We first set a threshold of r_f. The algorithm computes the r_f value for each triangle, and then sort these r_f values and put them into a queue. Each time the algorithm picks one triangle with currently the highest r_f value and apply the short edge collapse operation. The algorithm stops when the highest r_f value is below the threshold. Fig. 4 gives one example. A triangular mesh constructed from a network with extremely non-uniform node distribution has a maximal r_f value close to 27, as shown in Fig. 4 (a). After a series of short edge collapse operations, its maximal r_f value is below 4, as shown in Fig. 4 (b).

5.2. Computing Optimal Flat Metric with Distance Measurement

We construct an initial circle packing metric (Γ_0, Φ) of M from provided distance measurements with the following procedures:

1. For each corner θ_i^{jk} attached to Vertex v_i in Face f_{ijk}, we compute a corner radius γ_i^{jk} for v_i with respect to f_{ijk}:
$$\gamma_i^{jk} = \frac{l_{ki} + l_{ij} - l_{jk}}{2},$$
where l_{ij}, l_{jk}, l_{ki} represent the distance measurements of edges e_{ij}, e_{jk}, e_{ki}, respectively.
2. For each v_i, we compute its initial circle radius γ_i by averaging its attached corner radii computed from the previous step:
$$\gamma_i = \frac{1}{m} \sum_{f_{ijk} \in F} \gamma_i^{jk},$$
where m is the number of the adjacent faces to v_i (i.e., the vertex degree of v_i).
3. For each edge e_{ij}, we compute its edge weight ϕ_{ij} (the intersection angle of the two circles centered at v_i and v_j) based on the Euclidean cosine law:
$$\cos\phi_{ij} = \frac{\gamma_i^2 + \gamma_j^2 - l_{ij}^2}{2\gamma_i\gamma_j}.$$

With the constructed initial circle packing metric (Γ_0, Φ), we apply the algorithm introduced in Sec. 4.1 to compute the optimal flat metric, and then the isometric embedding algorithm introduced in Sec. 4.2 to embed the given network to a plane.

5.3. Time Complexity and Communication Cost

To incorporate distance information into the localization algorithm introduced in Sec. 4, we need to spend some extra costs.

The method to initialize circle packing metric is local. The time complexity and communication cost are both linear to the size of the network n, the number of sensor nodes.

The algorithm to extract a triangulation from the network connectivity graph based on distance measurement is also local and fully distributed. Both the time complexity and communication cost are linear to n [21].

The time complexity and communication cost of post-processing a triangulation are bounded by the number of edges of the triangular mesh. Denote v_n the number of vertices, e_n the number of edges, and f_n the number of triangle faces of a triangular mesh M, the well known Euler-Poincaré theorem relates v_n, e_n, and f_n [22].

Theorem 3 (Euler-Poincaré Theorem)**.** *Denote b the number of boundaries of a triangular mesh M. The following equation holds:*

$$v_n - e_n + f_n = 2 - b.^1 \tag{12}$$

Based on Euler-Poincaré theorem, the following theorem shows that $e_n \approx 3v_n$.

Theorem 4. *Denote b the number of boundaries of a triangular mesh M, v_n the number of vertices, and e_n the number of edges. The following equation holds:*

$$e_n = 3v_n + (3b - 6). \tag{13}$$

Proof. Denote f_n the number of triangle faces of M, $e_{\notin \partial M}$ the number of non-boundary edges, and $e_{\in \partial M}$ the number of boundary edges. Considering the two facts that each non-boundary edge is shared by two triangle faces and each triangle face has three edges, the following equation holds:

$$2e_{\notin \partial M} + e_{\in \partial M} = 3f_n,$$

which gives

$$f_n = \frac{2}{3}e_n, \tag{14}$$

by ignoring boundary edges if

$$e_{\in \partial M} << e_{\notin \partial M}.$$

According to Euler-Poincaré Theorem 12, the following equation also holds:

$$v_n - e_n + f_n = 2 - b. \tag{15}$$

Plug Eqn. 14 into Eqn. 15, we have:

$$v_n - e_n + \frac{2}{3}e_n = 2 - b.$$

Then we have:

$$e_n = 3v_n + (3b - 6).$$

[1] Note that we assume the number of handles of M is zero.

□

So the time complexity and communication cost of post-processing a triangulation are $O(n)$.

5.4. Error Propagation

If distance measurement error is introduced only by one single node, we show that the error propagation of the proposed localization algorithms decreases with the distance to the node.

According to the discrete Ricci curvature theory, the differential discrete curvature $d\mathbf{K}$ and discrete scaling factor $d\mathbf{u}$ satisfies the discrete Poisson equation:

$$d\mathbf{K} = \Delta d\mathbf{u},$$

where Δ is the discrete Laplace-Beltrami operator.

Specifically, suppose v_i is an interior vertex, and v_j is one of its neighboring vertices. f_{ijk} and f_{jil} are the two faces adjacent to e_{ij}. The power circle of f_{ijk} is defined as a circle orthogonal to all three circles centered at v_i,v_j and v_k with radii γ_i, γ_j, and γ_k respectively. Similarly, one can define the power circle of f_{jil}. Denote d_{ij} the distance between the two power circles, l_{ij} the length of e_{ij}. For each e_{ij}, we define a weight:

$$w_{ij} = \frac{d_{ij}}{l_{ij}}.$$

Then we have

$$dK_i = \sum_{j=1}^{g} w_{ij}(du_j - du_i),$$

where g is the vertex degree of v_i.

Suppose a small perturbation of the curvature happens at v_i, modeled as $\delta(v - v_i)$. Such perturbation is due to the measurement error of neighboring distances around v_i. Then the influence of the perturbation to the entire network can be estimated by the Green's function, $G(v, v_i)$, such that $\Delta G(v, v_i) = \delta(v - v_i)$. Roughly speaking, the Green's function on the plane is $\frac{1}{2\pi} \log \frac{1}{|v - v_i|}$. Therefore, the influence of the perturbation at one node is Sublinear to others. The influence of wrong measurements at one node to others decreases proportionally with their distances.

6. Simulations and Comparison

We carry out extensive simulations under various scenarios to evaluate the performance of our algorithm on networks with different topologies and how the performance is affected by different factors such as node density, communication model (UDG, quasi-UDG, log-Norm model, and probability model), and non-uniform node distribution. We compare our algorithm with those state-of-the-art localization methods, including the centralized MDS approach (MDS-MAP) [1], the distributed MDS approach (MDS-MAP(P)) [2], the centralized (C-CCA) neural network, and the distributed (D-CCA) neural network approaches [5].

We apply the proposed optimal flat metric based method to localize the landmark nodes (points marked with red in Figs. 5, 6, 7, and 8). Those non-landmark nodes (points marked with grey in Figs. 5, 6, 7, and 8) in the network find their 3 nearest landmarks and compute their own coordinates. We then compute the difference of each node-pair distances in the original and localized networks. We calculate the localization error as the ratio of the average difference of node-pair distances and the one-hop communication radio range.

6.1. Networks with Variant Nodal Densities

In general, connectivity based localization methods favor high nodal density, because hop counts approximates well the true shortest distance. Figs. 5 (a)-(d) gives a series of reversed C-shape networks with mere connectivity

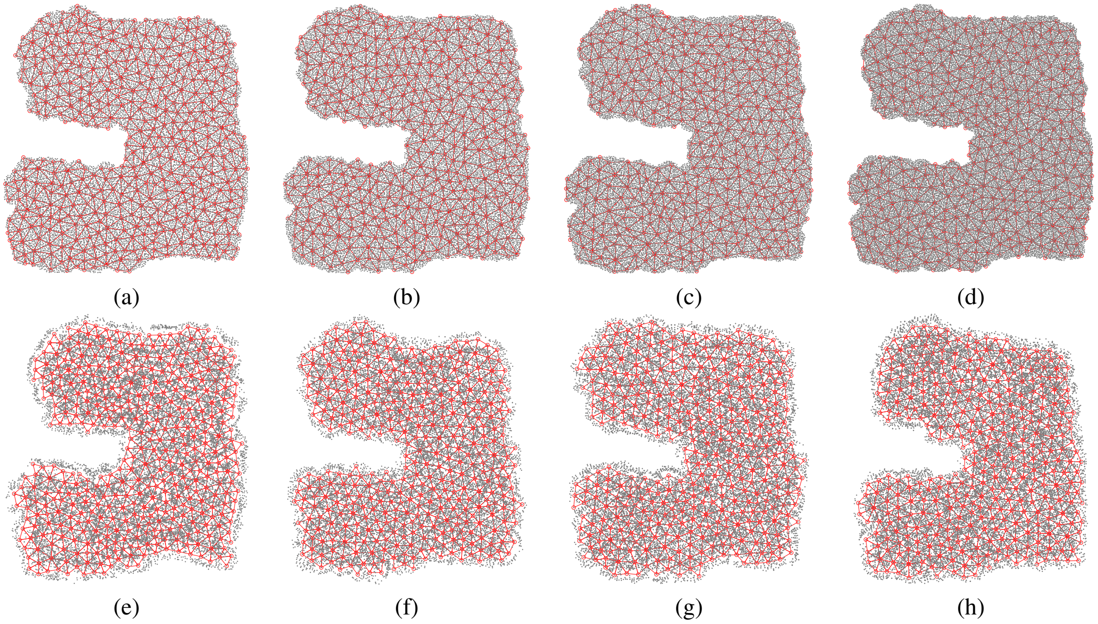


Figure 5. Networks with variant nodal density: (a)-(d) the original network with increased nodal density; (e)-(h) the localization results of our algorithm. All the networks share the same communication radio range and under the same transmission model. We compute the difference of each node-pair distances in the original and localized networks. Then we calculate the localization error as the ratio of the average difference of node-pair distances and the one-hop communication radio range. Specifically, the average nodal degrees for networks shown in (a)-(d) increase with $d = 9.4, d = 12.6, d = 15.2, d = 18.5$, respectively. The localization errors of networks shown in (a)-(d) decrease with $0.514, 0.322, 0.28, 0.246$, respectively. Figures (e)-(h) show the corresponding localization results. The localization error decreases with the increase of nodal density.

information. All the networks have the same communication radio range and under the same transmission model, but the average nodal degree increases from 9 to 18. Figs. 5 (e)-(h) shows the localization results correspondingly. The localization error decreases from 0.514 to 0.246. The trend is clear that the localization error decreases with the increase of nodal density.

6.2. Networks with Different Transmission Models

We still apply the same reversed C-shape network to evaluate the performance of the proposed localization algorithm with mere connectivity under different transmission models, with average nodal degree $d = 16.5$ (under UDG model). For a network with identical communication radio range of sensor nodes, different transmission models induce different sets of landmarks because each node has different neighborhoods. We generate different triangular meshes accordingly. In our simulations, the transmission range of a UDG model is 1. Two nodes are definitely connected when their distance is less than 1. Under the Quasi-UDG model, two nodes are definitely connected when their distance is less than $\alpha = 0.75$, definitely not connected when their distance is larger than 1, while they have a probability of $\rho = 0.5$ to be connected when the distance is between α and 1. Under the Log-Norm model [25], since the receiving power is log-normally distributed, we simplify it as when the distance between two nodes is larger than 1, they are not connected; when the distance is less than 1, they have a probability $P(d)$ to be connected, where d is the distance and $P(d)$ satisfies the log-normal distribution with $\alpha = 2$ and $\sigma = 4$. Under the Probability model, when the distance of two nodes are less then 1, they have a probability equal to a constant set to 0.65 to be connected. Fig. 6 shows the localization results of the reversed C-shape network with mere connectivity under different transmission models.

Among all the transmission models, UDG model achieves the smallest localization error.

6.3. Networks with Non-uniform Nodal Distribution

We also test the proposed optimal flat metric based method with mere connectivity on network with non-uniform node distribution. The nodal density of the reversed C-shape network shown in Fig. 7 (a) increases from the bottom to the top. Specifically, the nodal degree increases from 11.3 to 18.7. Fig. 7 (b) shows the localization result with a slightly higher localization error 0.46. The proposed algorithm can tolerate networks with a moderately non-uniform nodal distribution.

6.4. Comparison with Other Methods on Networks With Mere Connectivity

Fig. 8 lists a set of representative networks with various shapes and different topologies. These networks have only connectivity information available. For these networks, we only show the original coordinates of landmark nodes marked with red color. Grey points represent the computed coordinates of sensor nodes of the networks, including landmark and non-landmark nodes, after a least square alignment with the original networks. we associate each landmark node with a red line segment, starting from its original coordinates marked with red and ending at the computed coordinates marked with grey. Clearly, the length of the line segment indicates the localization error of that node. Overall, the more and the longer the red lines are, the worse the localization performance is.

As can be observed in Figs. 8 a(1) and c(1), C-CCA and MDS-MAP both yield large distortions for nodes on the two branches. This is because the approximated distances based on hop counts among those nodes are much longer than their actual Euclidean distances due to the reversed C shape. On the contrary, MDS-MAP(P) and D-CCA compute local maps first, and then merge them to a global map. Since local maps are "smooth" and do not have large "tentacles" in general, the shortest paths are free of significant distortions. Therefore, both methods achieve better performance (i.e. less distortion and lower errors) than the centralized approaches for networks with irregular boundary conditions. This can be clearly seen in Figs. 8 b(1) and d(1). However, if a network has smooth boundary (e.g., Figs. 8 a(2)-e(2)), the shortest paths are not seriously distorted and thus the centralized schemes perform better since they utilize more constraints to localize the network nodes.

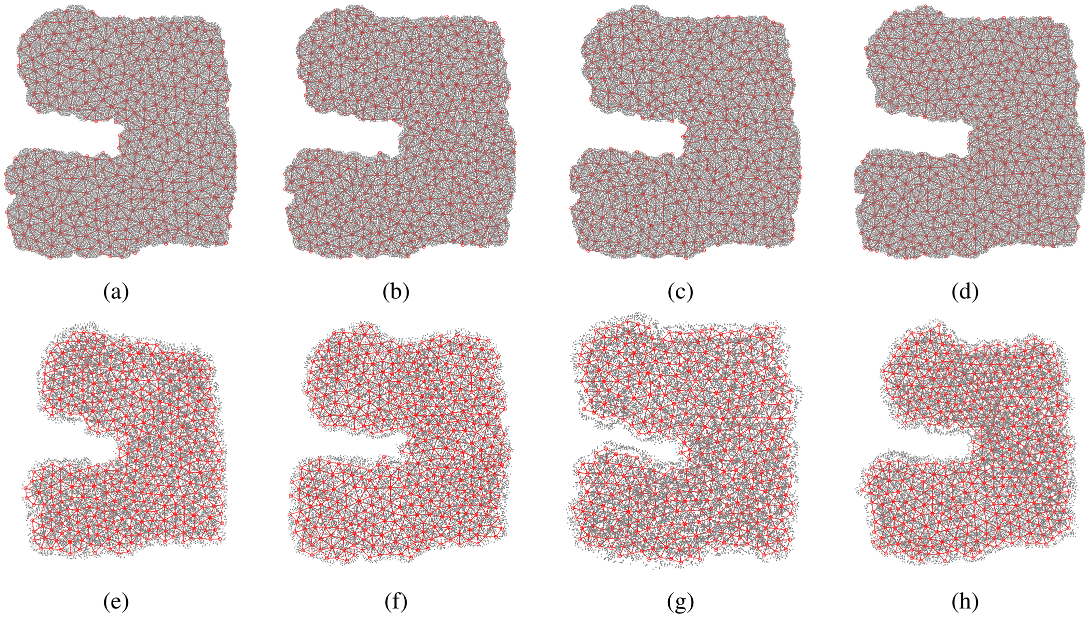


Figure 6. Networks with different transmission models: (a)-(d) the original networks with different transmission models that result in different nodal degrees and constructed triangular meshes; (e)-(h) the embedding results of our algorithm. All the networks have the same number of nodes and the same communication range. (a) UDG model with transmission range 1, and localization error 0.25 in (e); (b) QUASI-UDG model with $\alpha = 0.75$ and $\rho = 0.5$, and localization error 0.34 in (f); (c) Log-Norm model with $\mu = 0.5$ and $\rho = 4$, and localization error 0.42 in (g); (d) Probability model with $p = 0.65$, and localization error 0.43 in (h).

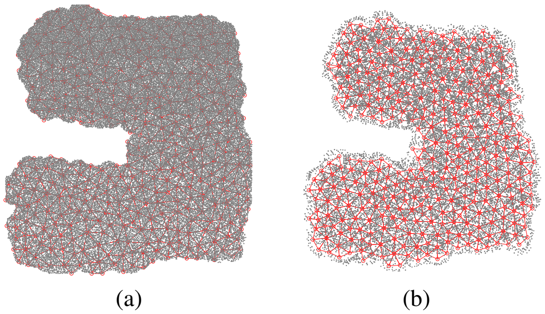


Figure 7. Network with non-uniform node distribution: (a) The original network with non-uniform node distribution. The nodal degrees rang from 11.3 to 18.7. (b) The localization result of our algorithm with the localization error 0.46.

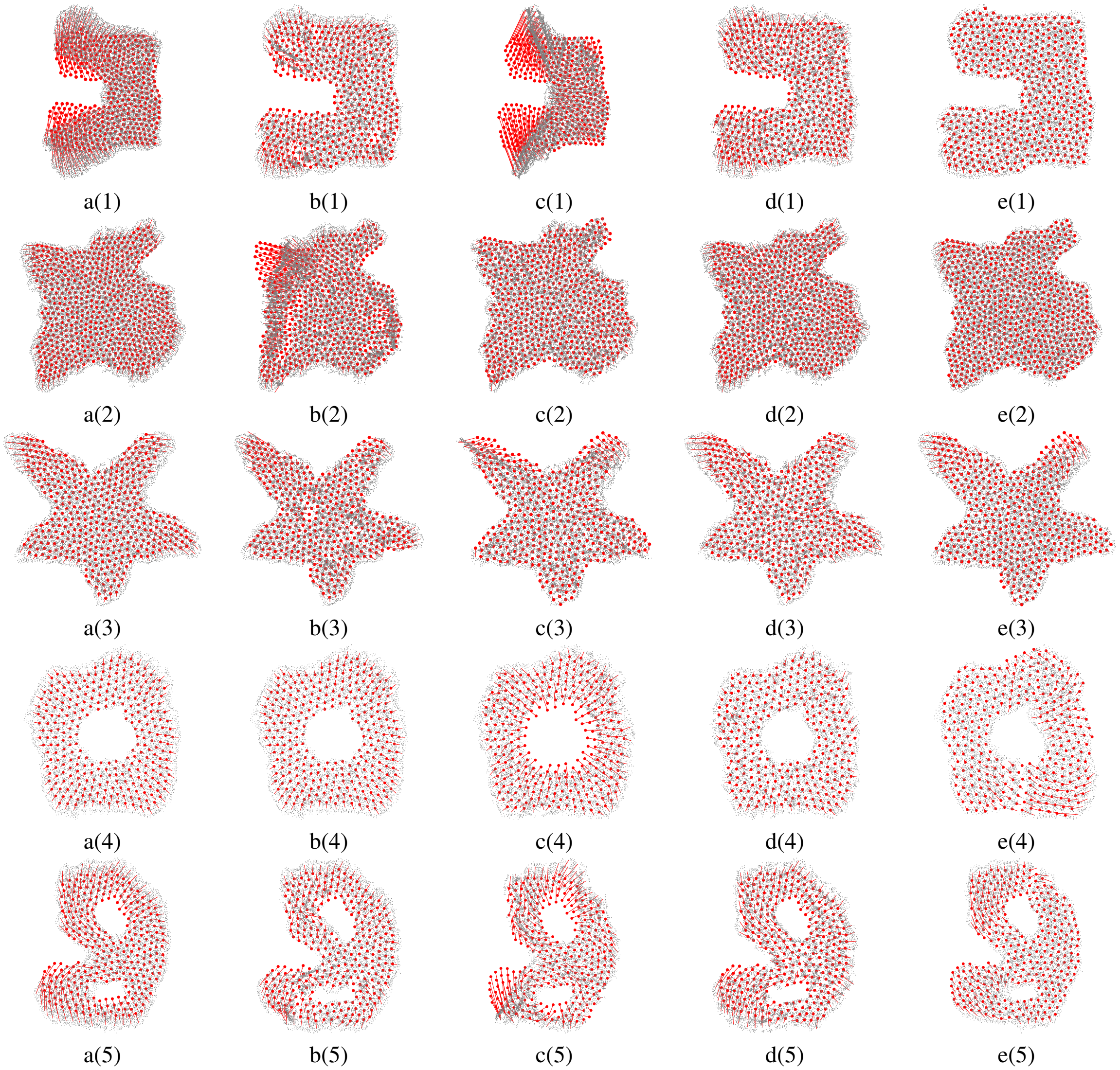
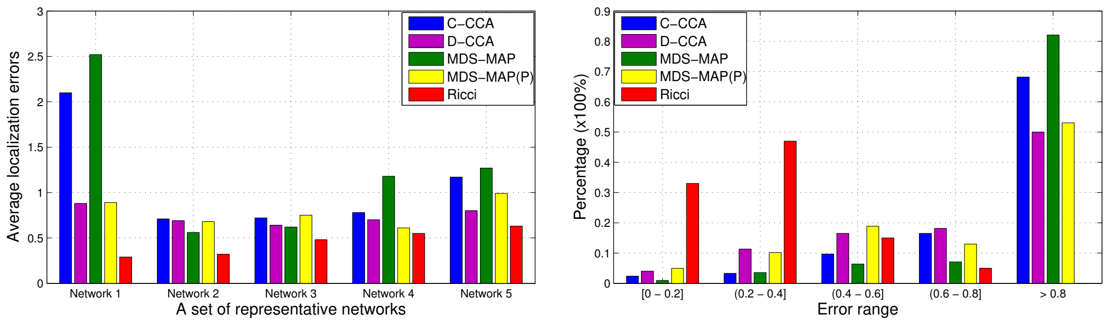


Figure 8. Comparison with other methods on networks with mere connectivity. a(1)-a(5): C-CCA scheme; b(1)-b(5): D-CCA scheme; c(1)-c(5): MDS-MAP scheme; d(1)-d(5): MDS-MAP(P) scheme; e(1)-e(5): Ricci scheme. A red line segment is drawn for each node, starting from the real coordinates marked with red and ending at the computed coordinates marked with grey.



(a)

(b)

Figure 9. (a) Comparison of average localization errors of different methods on networks shown in Figs. 8 with mere connectivity information: Network 1 shown in Fig. 8 a(1)-e(1), Network 2 shown in Fig. 8 a(2)-e(2), Network 3 shown in Fig. 8 a(3)-e(3), Network 4 shown in Fig. 8 a(4)-e(4), and Network 5 shown in Fig. 8 a(5)-e(5). (b) The distribution of connectivity-based localization errors on the reversed C-shape network with different approaches: C-CCA, D-CCA, MDS-MAP, MDS-MAP(P), and Ricci.

For networks with holes, similarly as the reversed C-shape network, MDS-MAP(P) and D-CCA perform better (shown in Figs. 8 b(4) and d(4), b(5) and d(5)) compared with their centralized counterparts MDS-MAP and C-CCA (shown in Figs. 8 a(4) and c(4), a(5) and c(5)). But they have to pay the cost to merge different subnetworks together.

The proposed optimal flat metric based approach, on the contrary, achieves the least overall localization errors in all simulated scenarios as demonstrated in Figs. 8 e(1), e(2), e(3), e(4) and e(5). Fig. 9 (a) summarizes the average localization errors with different approaches on models shown in Figs. 8.

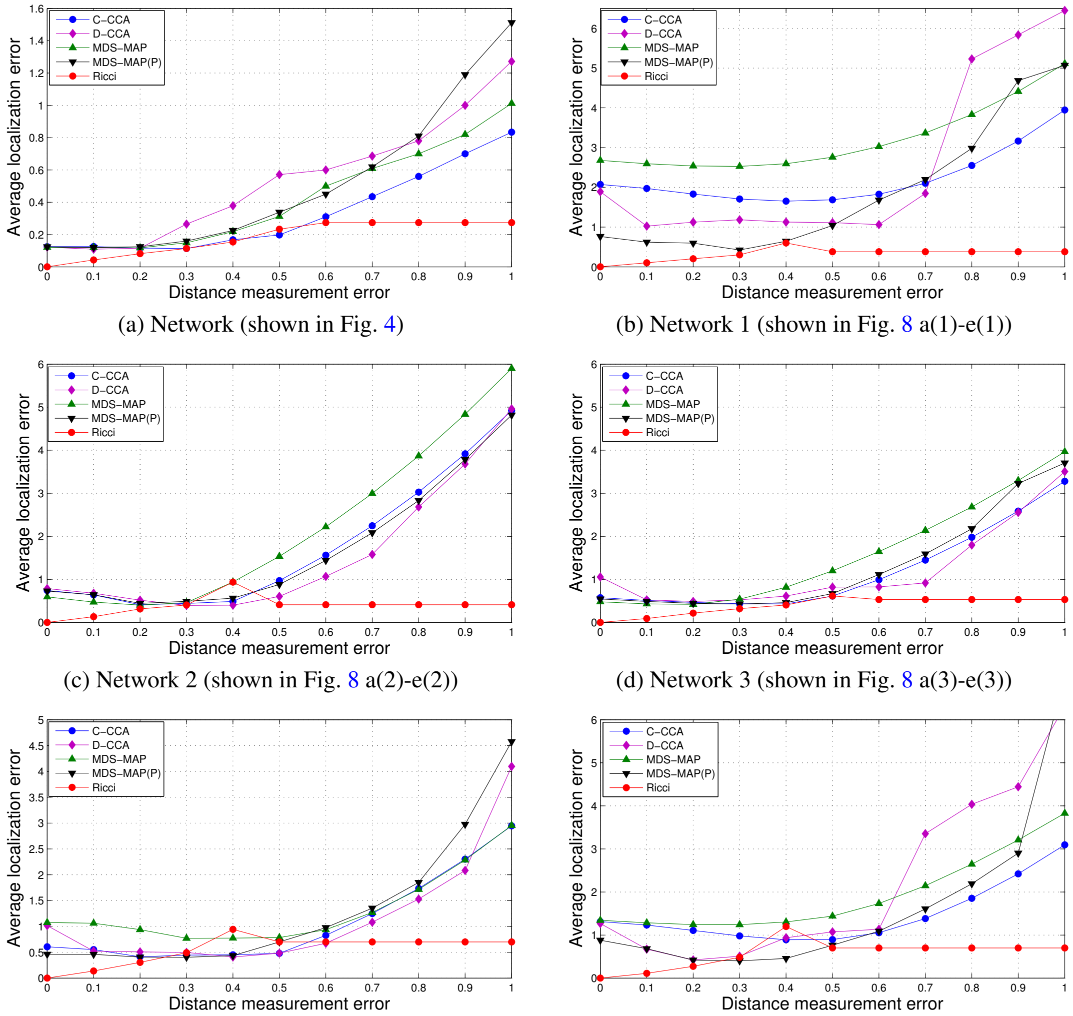
Fig. 9 (b) illustrates the distribution of localization errors on the reversed C-shape network with different localization approaches. Since the results under other networks show similar statistics, we omitted them here. As can be seen, the localization errors of the optimal flat metric based approach are nicely distributed at the lower range.

6.5. Comparison with Other Methods on Networks With Range Distance Measurements

Fig. 10 summarizes the average localization errors of the proposed optimal flat metric based approach and other methods on networks shown in Fig. 4 and Fig. 8 with one-hop communication radio range distance information. Specifically, the one-hop communication radio range distance measurement error increases from 0% to 100% of the actual distance.

Fig. 10 shows that when the measurement error is less than 10%, a localized network based on the proposed algorithm is very close to the original one. When the measurement error is less than 40% shown in Figs. 10 (b) and (d), 30% shown in Figs. 10 (a) and (c), and 25% shown in Figs. 10 (e) and (f), the proposed algorithm achieves the highest localization accuracy compared with other methods. But from the other side, the proposed algorithm is more sensitive to measurement error than other localization methods. Its performance decreases with the increase of the measurement error. When the measurement error is more than 50%, many triangles formed by three sensor nodes within their mutual communication radio range degenerate and the measured one-hop communication radio range distances even don't satisfy triangle inequality. We can't take the measured distances as a legal and initial metric of the constructed mesh any more, so we have to take an average of the measured one-hop communication radio range distance as an initial metric of the mesh, which means each edge is assigned a uniform length. Such averaged distance is robust to random errors, which explains that the performance of the proposed algorithms keeps stable When the measurement error is more than 50%.

C-CCA and MDS-MAP, the centralized neural network based and MDS based approaches, on the contrary, are least sensitive to measurement error. Such random error directly affect the measured distances of pairs of nodes



(a) Network (shown in Fig. 4)

(b) Network 1 (shown in Fig. 8 a(1)-e(1))

(c) Network 2 (shown in Fig. 8 a(2)-e(2))

(d) Network 3 (shown in Fig. 8 a(3)-e(3))

(e) Network 4 (shown in Fig. 8 a(4)-e(4))

(f) Network 5 (shown in Fig. 8 a(5)-e(5))

Figure 10. Comparison of average localization errors of different methods on networks shown in Fig. 4 and Fig. 8 with one-hop communication radio range distance measurement errors increased from 0% to 100%.

within one-hop communication radio range, which is only a small portion of the distance matrix of a network. But for a pair of nodes not within one-hop communication radio range, the longer their distance is, the smaller the random error accumulated along their path is. So we don't observe noticeable performance decline of C-CCA when the measurement error is small. The performance of C-CCA starts to decrease when the measurement error is over 40% shown in Figs. 10 (c) and (d), 50% shown in Figs. 10 (a) and (e), and 70% shown in Figs. 10 (b) and (f). Similarly as C-CCA, there is no noticeable performance decline of MDS-MAP When the measurement error is small. The performance of MDS-MAP starts to decrease when the measurement error is over 30% shown in Figs. 10 (a), (c) and (d), 50% shown in Figs. 10 (b) and (f), and 60% shown in Fig. 10 (e).

D-CCA and MDS-MAP(P), the distributed neural network based and MDS based approaches, can achieve a higher localization accuracy than their centralized counterparts when the measurement error is small. But they are also more sensitive to measurement error than their centralized counterparts.

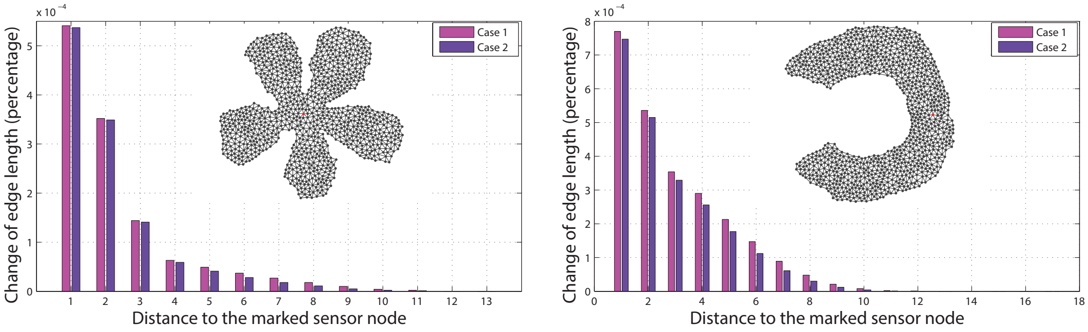
Combining Fig. 8 and Fig. 10, it is obvious that when the measurement error is more than 50% of the actual distance, such measured one-hop communication radio range distance won't help improve localization accuracy for any of the compared localization methods including the proposed one in this paper.

6.6. Error Propagation on Networks With One-hop Communication Radio Range Distance Measurements

We conduct the following experiments to test the propagation rate of measurement error resulting from bad estimation or wrong measurement. Two networks with constructed triangular mesh structure are shown in the right top of Figs. 11 (a) and (b). If the given one-hop communication radio range distance is free of measurement error, the discrete Ricci flow based method localizes the two networks with localization errors 1.4×10^{-5} and 1.5×10^{-5} , respectively. While if the distance measurement around one node is wrong, for example, much slower response of a node to its neighboring nodes' signals will result in a much longer distance approximation. The measurement error around the node will affect not just localization of this node, but also other nodes in the network (e.g., error will propagate). We introduce such measurement error at one selected node (marked with red in the two triangular networks shown in Figs. 11 (a) and (b)) by multiplying some constant K ($K = 2.0$ for case 1, $K = 1.5$ for case 2) with the lengths of its neighboring edges, such that the node is no longer planar based on the wrong measurement. We measure the effect of the "one node measurement error" by comparing the change of the computed flat metric (edge length) with the original one. Figs. 11 (a) and (b) shows the distribution of the average errors of edge lengths of the two testing networks with respect to the distance to the selected node respectively. The closer the two ending nodes of one edge are to the selected node, the bigger the distortion of the computed edge length is. It is obvious that the error propagation decreases dramatically with the increase of the distance to the distorted node. The localization error of the two networks with distorted vertices are 0.0018 and 0.0020 respectively.

6.7. Computing Time

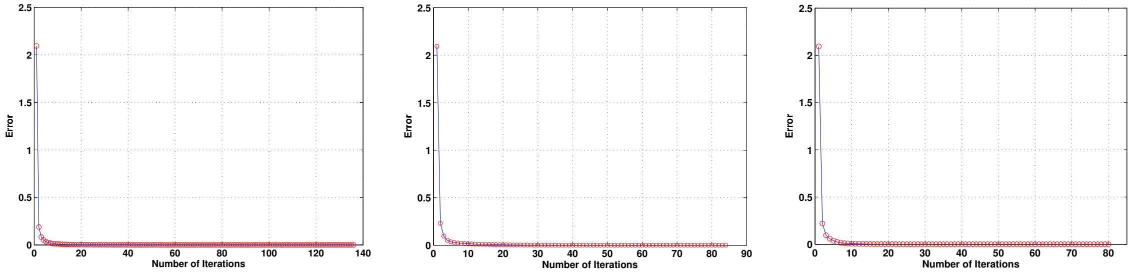
In our experiments, we set the step length of running discrete Ricci flow to 0.1, and the error threshold to 1×10^{-5} . Fig. 12 gives the convergence rate and time using discrete Ricci flow to compute optimal flat metric for part of our simulation networks. Specifically, for the reversed C-shape network with $3k$ sensor nodes and 424 landmarks chosen, the convergence time of discrete Ricci slow is 2 seconds. The curvature error is less than 0.1 and 0.0004 after 5 and 82 iterations, respectively. For the network with one hole and $2k$ sensor nodes and 283 landmarks chosen, the convergence time is 1 second. The curvature error is less than 0.1 and 0.0004 after 4 and 55 iterations, respectively. For the network with two holes and $2k$ sensor nodes and 297 landmarks chosen, the convergence time is 1 second. The curvature error is less than 0.1 and 0.0004 after 4 and 48 iterations, respectively. The time of isometric embedding of each network is no more than a few seconds. So the total computing time of a network including choosing the landmark nodes, building a triangular structure, computing optimal flat metric and isometric embedding the network is no more than 10 seconds on a Dell Latitude e6420 laptop. Note that the computation is fully distributed that each vertex node computes and exchanges information with only its neighboring vertex nodes.



(a)

(b)

Figure 11. The error of the computed edge length decreases dramatically with its distance to the distorted node marked with red.



(a)

(b)

(c)

Figure 12. Convergence rate of discrete surface Ricci flow applied to compute optimal flat metric of networks, with step length 0.1 and error threshold 1×10^{-5} . (a) A reversed C-shape network with 424 landmarks: the convergence time is 2 seconds; (b) A network with one hole and 283 landmarks: the convergence time is 1 second; (c) A network with two holes and 297 landmarks: the convergence time is 1 second.

We can also apply centralized method to compute optimal flat metric using discrete Ricci flow. Each vertex node sends its connectivity information and distance measurement within one-hop transmission range if available to a central server. The central server can apply Newton's method as discussed in our previous paper [26] to compute the optimal flat metric with discrete Ricci flow. Compared with the distributed method introduced in this paper, the centralized method converges in less than 5 iterations with the total computing time less than 1 second for all the testing networks.

7. Conclusion

This work proposes a novel optimal flat metric based localization method suitable for large-scale sensor networks. The method can be proved to generate an optimal flat metric that introduces the least distortion from the initially estimated edge length. The computation is fully distributed and highly scalable with its computation time and communication cost linear to the size of the network. Extensive simulations and comparison with other methods under various representative network settings are carried out, showing superior performance of the proposed algorithms.

References

1. Yi Shang , Wheeler Ruml , Ying Zhang , and Markus P. J. Fromherz. “Localization from Mere Connectivity,” In *Proc. of MobiHoc*, pp. 201–212, 2003.
2. Yi Shang, and Wheeler Rum. “Improved MDS-based Localization,” In *Proc. of INFOCOM*, pp. 2640–2651, 2004.
3. Vijayanth Vivekanandan, and Vincent W.-S. Wong. Ordinal MDS-based “Localization for Wireless Sensor Networks,” *International Journal of Sensor Networks*, 1(3/4), pp. 169-178, 2006.
4. G. Giorgetti , S. Gupta, and G. Manes. “Wireless Localization Using Self-Organizing Maps,” In *Proc. of IPSN*, pp. 293–302, 2007.
5. L. Li, and Thomas Kunz. “Localization Applying An Efficient Neural Network Mapping,” In *Proc. of The 1st International Conference on Autonomic Computing and Communication Systems*, pp. 1–9, 2007.
6. Richard S. Hamilton. “Three Manifolds with Positive Ricci Curvature,” *Journal of Differential Geometry,* 17, pp. 255–306, 1982.
7. H. Lim, and J. Hou. “Distributed Localization for Anisotropic Sensor Networks,” *ACM Transactions on Sensor Networks*, 5(2), pp. 11–37, 2009.
8. Hongyi Wu , Chong Wang, and Nian-Feng Tzeng. “Novel Self-Configurable Positioning Technique for Multi-hop Wireless Networks,” *IEEE/ACM Transactions on Networking*, 13(3), pp. 609–621, 2005.
9. Yue Wang , Sol Lederer, and Jie Gao. “Connectivity-based Sensor Network Localization with Incremental Delaunay Refinement Method,” In *Proc. of INFOCOM*, pp. 2401–2409, 2009.
10. Pratik Biswas, and Yinyu Ye. “Semidefinite programming for ad hoc wireless sensor network localization,” In *Proc. of IPSN*, pp. 46–54, 2004.
11. Anthony M.-C. So, and Yinyu Ye. “Theory of semidefinite programming for sensor network localization,” In *Proc. of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp.405–414, 2005.
12. Fabian Kuhn, Thomas Moscibroda, and Rogert Wattenhofer. “Unit disk graph approximation,” In *Proceedings of the 2004 joint workshop on Foundations of mobile computing*, pp. 17–23, 2004.
13. Thomas Moscibroda, Regina O’Dell, Mirjam Wattenhofer, and Rogert Wattenhofer. “Virtual coordinates for ad hoc and sensor networks,” In *Proceedings of the 2004 joint workshop on Foundations of mobile computing*, pp. 8-16, 2004.
14. Amitabh Basu, Jie Gao, Joseph S. B. Mitchell, and Girishkumar Sabhnani. “Distributed localization using noisy distance and angle information,” In *Proceedings of the 7th ACM international symposium on Mobile ad hoc networking and computing*, pp. 262–273, 2006.
15. Jose Costa, Neal Patwari, and Alfred Hero. “Distributed weighted-multidimensional scaling for node localization in sensor networks,” *ACM Transactions on Sensor Networks*, 2(1), pp. 39–64, 2006.
16. Rik Sarkar and Xiaotian Yin and Jie Gao and Feng Luo and Xianfeng David Gu. “Greedy Routing with Guaranteed Delivery Using Ricci Flows,” In *Proc. of IPSN*, pp. 121–132, 2009.
17. Stefan Funke and Nikola Milosavljevic. “Guaranteed-Delivery Geographic Routing Under Uncertain Node Locations,” In *Proc. of INFOCOM*, pp. 1244–1252, 2007.
18. Stefan Funke and Nikola Milosavljevic. “How Much Geometry Hides in Connectivity–Part II,” In *SODA*, pp. 958–967, 2007.
19. Yao Zhao and Hongyi Wu and Miao Jin and Su Xia. “Localization in 3D Surface Sensor Networks: Challenges and Solutions,” In *Proc. of INFOCOM*, pp. 55-63, 2012.
20. Miao Jin and Junho Kim and Xianfeng David Gu. “Discrete Surface Ricci Flow: Theory and Applications,” *Mathematics of Surfaces XII*, 4647, pp. 209–232, 2007.
21. Hongyu Zhou and Hongyi Wu and Su Xia and Miao Jin and Ning Ding. “A Distributed Triangulation Algorithm for Wireless Sensor Networks on 2D and 3D Surface,” In *Proc. of INFOCOM*, pp. 1053-1061, 2011.
22. Michael Henle. “A Combinatorial Introduction to Topology,” *Dover Publications*, 1994.
23. Bennett Chow, and Feng Luo. “Combinatorial Ricci Flows on Surfaces,” *Journal Differential Geometry*, 63(1), pp. 97–129, 2003.
24. William P. Thurston. “Geometry and Topology of Three-Manifolds,” *Princeton lecture notes*, 1976.

25. Christian Bettstetter, and Christian Hartmann. “Connectivity of wireless multihop networks in a shadow fading environment,” *Wireless Networks*, 11(5), pp. 571–579, 2005.
26. M. Jin , J. Kim , F. Luo, and X. Gu. “Discrete Surface Ricci Flow,” *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 14(5), pp. 1030–1043, 2008.
27. R. S. Hamilton, “The Ricci flow on surfaces,” *Mathematics and general relativity (Santa Cruz, CA, 1986), Contemp. Math. Amer.Math.Soc. Providence, RI*, vol. 71, 1988.
28. B. Chow, “The Ricci flow on the 2-sphere,” *J. Differential Geom.*, vol. 33, no. 2, pp. 325–334, 1991.
29. S. H. Weitraub, *Differential Forms: A Complement to Vector Calculus*. Academic Press, 2007.

Table 1. Table of Notation

Symbol	Explanation
B_0	the longest boundary of M
B_i	the i-th boundary of M
b	the number of boundaries of M
E	edges of M
e_{ij}	an edge belonging to E with two ending vertices v_i and v_j
e_n	the number of edges of M
F	triangle faces of M
f_{ijk}	a triangle face belonging to F with vertices v_i, v_j, and v_k
f_n	the number of triangle faces of M
K_i	the discrete Gaussian curvature of $v_i \in V$
$\bar{K}_i$	the target Gaussian curvature of $v_i \in V$
L_i	a shortest path between the initiator of B_i and B_0
l_{ij}	length of e_{ij}
M	a triangulated surface (or mesh in short) embedded in $\mathbb{R}^3$
∂M	the boundary of M
$r_{f_{ijk}}$	a ratio between the longest and shortest edge lengths of f_{ijk}
t	the time
u_i	the logarithm of γ_i of v_i
V	vertices of M
v_n	the number of vertices of M
v_i	a vertex belonging to V with id i
θ_i^{jk}	the corner angle attached to Vertex v_i in Face f_{ijk}
γ_i	the radius of a circle associated with v_i
Γ	the radius function assigned at M
ϕ_{ij}	weight of e_{ij} (the intersection angle of two circles centered at v_i and v_j
Φ	the weight function assigned at M
(Γ, Φ)	circle packing metric of M
$\chi(M)$	the Euler characteristic number of M
δ	the step length computing optimal flat metric
ε	the threshold of curvature error computing optimal flat metric