Localization of Networks on 3D Terrain Surfaces

Xuan Li, Buri Ban, Yang Yang, and Miao Jin

Abstract—The majority of current research on sensor network localization focuses on wireless sensor networks deployed on two dimensional (2D) plane or in three dimensional (3D) space, very few on the 3D terrain surface. However, many real-world applications require large-scale sensor networks deployed on the surface of a complex 3D terrain. Compared with planar and 3D network localization, terrain surface network localization generates unique and fundamental hardness.

We explore 3D surface network localization with terrain models. A digital terrain model (DTM), available to the public with a variable resolution of up to one meter, is a 3D representation of a terrain's surface. It is commonly built using remote sensing technology or from land surveying and can be easily converted to a triangular mesh. Given a sensor network deployed on the surface of a 3D terrain with one-hop distance information available, we can extract a triangular mesh from the connectivity graph of the network. The constraint that the sensors must be on the known 3D terrain's surface ensures that the triangular meshes of the network and the terrain's surface overlap and approximate the same geometric shape.

The basic idea of the localization algorithms is to map the two triangular meshes extracted from the connectivity graph of a sensor network and the DTM of its deployed terrain surface to the plane. The two meshes mapped to the plane can be easily aligned if the location information of anchor nodes is available. We introduce a fully distributed algorithm to construct a well-aligned mapping between the two triangular meshes in the plane based on anchor nodes information. However, accidents may happen on anchor nodes. We then introduce an anchor-free algorithm to extract feature points with geometric properties intrinsic to surface distances and independent of the embedding of the two meshes in 3D. The matched feature points induce transformations to align the two meshes in the plane. With the aligned triangular meshes of a network and its deployed terrain surface, each sensor node of the network can easily locate reference grid points from the DTM of the terrain to calculate its own geographic location. We carry out extensive simulations under various scenarios to evaluate the overall performance of the proposed algorithms with different factors such as the one-hop distance measurement error, the resolution of a DTM, and the performance of the algorithm in the situation of connectivity only.

Index Terms—Localization, Sensor network, 3D Terrain surface, Digital Terrain Model

1 Introduction

A variety of applications in wireless sensor networks require geographic locations of sensor nodes. Instead of equipping each sensor node with high-cost localization hardware such as a GPS receiver, different localization algorithms and protocols have been proposed that allow the sensor nodes to derive their own locations.

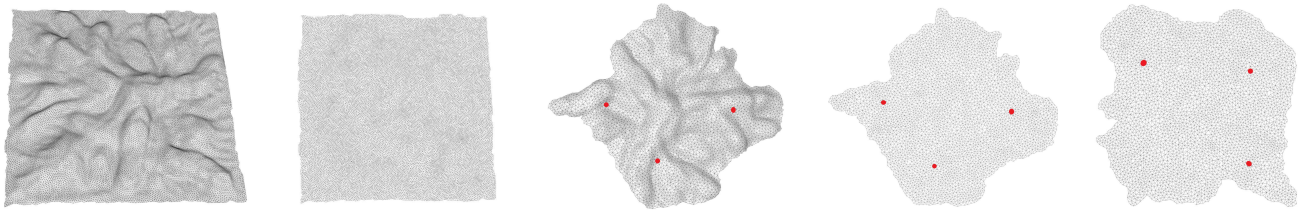
In real-world applications, many large-scale sensor networks are deployed over complex terrains, such as the volcano monitoring project [1] and ZebraNet [2]. Localization of a network deployed over a $3D$ surface, called surface network localization [3], is much more challenging than the well-studied localization of a network deployed in a $2D$ plane or $3D$ volume space. The reason is that distance between two remote sensors deployed over a $3D$ surface can only be approximated by the length of the shortest path along the surface due to limited radio range. Such distance is called surface distance, different from $3D$ Euclidean distance of the two nodes. Zhao et al. in [3] proved that a localization algorithm doesn't exist for a network deployed over a 3D surface with surface distance information only, even if we assume accurate range distance measurement available. An intuitive example given in [3] is that a piece of paper can be rolled to different shapes, but the distance between any pair of points on the paper is well preserved. With pure surface distance information, we can never figure out the current shape of a paper. We can also learn the hardness of localization of a network deployed over a 3D surface from differential geometry. There exists no unique embedding in $3D$ within rigid motions for a general surface with metric (equivalent to surface distance) only [4]. The only exception is proved by Alexandrov that any simplicial complex homeomorphic to a sphere with strictly non-negative Gaussian curvature at each vertex can be isometrically embedded uniquely in 3D as a convex polyhedron [5]. It is rare to find a terrain surface with a perfect convex shape, so the Alexandrov theorem couldn't help the design of algorithms for surface network localization.

Zhao et al. in [3] assume a sensor node can measure not only distances between its neighboring nodes but also its own height information. They require that a sensor network is deployed on a surface with single-value property - any two points on the surface have different projections on the plane. Such property ensures that a network deployed over a $3D$ surface can be projected to $2D$ plane by removing the z coordinate without ambiguity. Any existing $2D$ network localization method can then be applied to the projected one in the plane. The localized x and y coordinates in the plane, and the height information form a complete set of geographic coordinates of each sensor node in $3D$.

Later, a cut-and-sew algorithm is proposed in [6] to generalize the localization algorithm introduced in [3] from single-value surfaces to general ones. Authors in [6] take a divide-and-conquer approach to partition a general $3D$ surface network into a minimal set of single-value patches. The algorithm localizes each single-value patch individually, and then merges all of them into a unified coordinates system.

However, integrating height measurement into every sensor of a network is not always practical and affordable, especially for

- *Xuan Li, Buri Ban, Yang Yang, and Miao Jin are with the Center for Advanced Computer Studies, University of Louisiana, Lafayette, LA 70503.*
  E-mails: xuan.li1@louisiana.edu, banburi.811@gmail.com, yxy6700@cacs.louisiana.edu, miao.jin@louisiana.edu



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

Fig. 1. Anchor-based localization: (a) A triangular mesh of a terrain surface in 3D. (b) The triangular mesh of the terrain surface is conformally mapped to the plane. (c) A triangular mesh is extracted from the connectivity graph of a network deployed over the terrain surface with three randomly deployed anchor nodes marked with red. (d) The triangular mesh of the network is conformally mapped to the plane. (e) The mapped triangular mesh of the network is well-aligned with the one of terrain surface in the plane based on the location information of anchor nodes.

a large-scale sensor network. The motivation of this work is to explore the possibility of localization of a network deployed over surfaces with one-hop distance information only or even just mere connectivity, if we have the information of the deployed terrain surface.

1.1 Our Approaches

A 3D representation of the bare earth (topographic) model of a terrain's surface is called a digital terrain model (DTM). DTMs are commonly built using remote sensing technology or from land surveying. They are available to the public with a variable resolution of up to one meter. For example, the Shuttle Radar Topography Mission (SRTM) [7] is a high-resolution digital topographic database that provides DTM data for North and South America with high accuracy and dense coverage. Note that a DTM does not capture the natural features such as trees or forests and built ones such as buildings on the Earths surface.

A DTM is represented as a grid of squares, where the longitude, latitude, and altitude (i.e., 3D coordinates) of all grid points are known. It is straightforward to convert a grid into a triangulation, e.g., by simply connecting a diagonal of each square. Therefore a triangular mesh of the DTM of a terrain surface can be available before we deploy a sensor network on it.

On the other hand, to provide sufficient sensing coverage and cope with nodes' failure, most sensor networks designed for real applications require sufficient sensor density, especially for those deployed on terrain regions. Such sensor density allows a simple distributed algorithm to extract a fine triangular mesh based on the network connectivity graph and range distance. Vertices of the triangular mesh are the set of sensor nodes. An edge between two neighboring vertices indicates the communication link between the two sensors. The constraint that sensors must be on a known 3D terrain surface ensures that the triangular mesh of the terrain surface overlaps with the one extracted from the network connectivity graph. The question is how the latter can be localized in reference to the former.

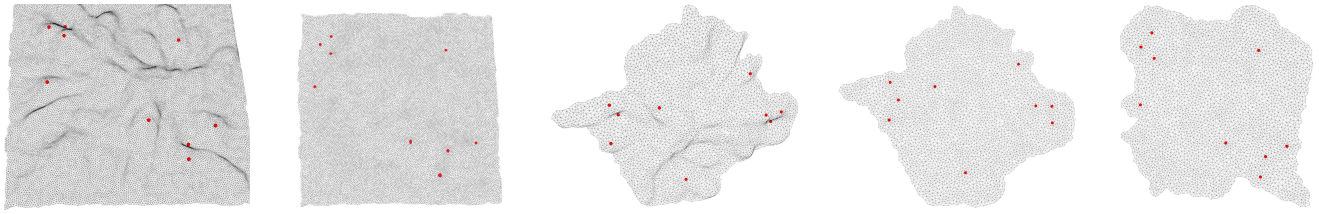
The proposed approach is based on surface conformal structure. The Conformal structure is an intrinsic geometric structure of surfaces, determined by surface distance. The Conformal structure can tolerate a small local deformation of a surface, so the conformal structure of a surface is consistent even if the surface is approximated by different triangulations with various densities. Surfaces sharing the same conformal structure exist conformal mapping between them. A conformal mapping is a one-to-one and continuous mapping which preserves angles and local shape.

The triangular mesh of a terrain surface and the one extracted from the connectivity graph of a network deployed over the terrain approximate the same geometric shape. Theoretically, the two triangular meshes share the same conformal structure. With a well-aligned conformal mapping constructed between them based on the positions of anchor nodes, a sensor node of the network can easily locate reference grid points of the DTM to calculate its own location.

Fig. 1 illustrates the basic idea of the anchor-based localization algorithm. Fig. 1 (a) shows a triangular mesh converted from the DTM of a terrain surface. Fig. 1 (c) shows a triangular mesh extracted from the connectivity graph of a network deployed over the terrain surface. We first compute two conformal mappings, denoted as f_1 and f_2 respectively, to map the two triangular meshes to plane as shown in Figs. 1 (b) and (d), respectively. Such one-to-one and continuous mapping exists based on the Riemann theorem that a topological disk surface can be mapped to plane through a conformal mapping [8]. However, the two mapped triangular meshes are not aligned on the plane. We deploy three anchor nodes equipped with GPS devices with the network. Figs. 1 (c) and (d) show the three anchor nodes marked with red. Based on their positions, We construct another conformal mapping denoted as f_3, to align the mapped network triangular mesh with the terrain one on the plane as shown in Fig. 1 (e). Combining the three mappings, $f_1^{-1} \circ f_3 \circ f_2$, induces a well-aligned conformal mapping between the two triangular meshes in 3D. Then a sensor node of the network, i.e., a vertex of the network triangular mesh, simply locates its nearest grid points, i.e., vertices of the network triangular mesh, to calculate its own geographic location.

However, a localization algorithm fully depending on anchor nodes is not reliable. Accidents happen. For example, anchor nodes are dropped to water regions by airplane. The battery of anchor nodes run out. The GPS devices of anchor nodes are broken. Furthermore, the distribution of anchor nodes in a network affects the localization accuracy as we will discuss in Sec. 5.1.1. In an extreme case, all anchor nodes are dropped to the same spot. The anchor-based localization algorithm fails since the alignment of the triangular meshes of network and terrain surface on the plane up to rotation. Considering all the factors, we propose an anchor-free localization algorithm for networks deployed on a $3D$ terrain surface with DTM available.

Without the location information of anchor nodes, we explore geometric properties to guide the alignment of the two mapped triangular meshes in the plane, i.e., the triangular mesh of a terrain surface and the one extracted from the connectivity graph of a network deployed over the terrain. These geometric properties should be intrinsic to surface distances and independent of the embedding of a surface in 3D. Specifically, conformal factor



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

Fig. 2. Anchor-free localization: (a) Feature points marked with red extracted from a triangular mesh of a terrain surface. (b) The triangular mesh of the terrain surface is conformally mapped to the plane. (c) Feature points marked with red extracted from a triangular mesh of a network deployed over the terrain surface. (d) The triangular mesh of the network is conformally mapped to the plane. (e) The mapped triangular mesh of the network is well-aligned with the one of terrain surface in the plane based on the positions of matched feature points. Note that we only show the matched feature points.

and Gaussian curvature are two such geometric properties. The conformal structure is an intrinsic structure of surfaces. Its induced conformal map preserving local shapes and introducing locally only a scaling, i.e., area distortion. The area distortion of a conformal map is measured by the conformal factor. Gaussian curvature, determining the local shape of a surface, depends only on the local surface distance instead of the embedding according to Gauss's Theorema Egregium.

With surface local distance information, we can compute both the conformal factor and Gaussian curvature at every vertex of the two meshes. The combination of the two properties reveals surface intrinsic geometry information both globally and locally. We then extract feature points of the two meshes corresponding to the ridge, peak, or saddle of mountains based on conformal factors and Gaussian curvatures. With the matched feature points, we compute a set of transformations to align the two meshes in the plane. Each sensor node of the network can then easily locate reference grid points of the DTM to calculate its own location at $3D$.

Fig. 2 illustrates the basic idea of the anchor-free localization algorithm. Fig. 2 (a) shows a set of feature points marked with red extracted from a triangular mesh of a terrain surface. Fig. 2 (c) shows a set of feature points marked with red extracted from a triangular mesh of a network deployed over the terrain surface. Similarly, we compute two conformal mappings to map the two triangular meshes to plane as shown in Figs. 2 (b) and (d), respectively. Based on the mapped positions of matched feature points, we construct mappings to align the mapped network triangular mesh with the terrain one in the plane as shown in Fig. 2 (e). Note that we only show the matched feature points in Fig. 2.

The rest of this paper is organized as follows: Sec. 2 introduces briefly the background knowledge necessary to the proposed surface network localization algorithms. Sec. 3 provides each step of both the anchor-based and anchor-free algorithms in detail with discussions in Sec. 4. Sec. 5 gives simulation results. Sec. 6 concludes the paper.

2 Theoretical Background

Before giving the details of the proposed localization algorithms in Sec. 3, we introduce briefly the background knowledge necessary to the algorithms. Specifically, we introduce the concepts of Gaussian curvature in Sec. 2.1, conformal map and conformal factor in Sec. 2. In Sec. 2.3, we introduce Möbius transformation, a special conformal map from a complex plane to itself. We then introduce Barycentric coordinates, a coordinate system we need later for sensor nodes to interpolate locations in Sec. 2.4.

2.1 Gaussian Curvature

The curvature of a curve measures how much the curve is bent. To define the curvature of a surface at a point p, we can slice the surface by a plane normal to the surface at p. By rotating the plane, the curvature of the slice-curve at p gives the curvature of the surface at p in every direction. The largest and smallest curvatures are called principal curvatures. They occur in orthogonal directions. The product of principal curvatures is called the Gaussian curvature. For a unit sphere, both principal curvatures at every point are 1 and hence the Gaussian curvatures are 1 everywhere. For a unit cylinder, the principal curvatures are 1 and 0 and hence the Gaussian curvatures are 0 everywhere. While for a hyperbolic paraboloid, the principal curvatures are 1 and -1, and the Gaussian curvatures are -1 everywhere. Gausss Theorema Egregium says that Gaussian curvature is intrinsic; independent of the embedding of the surface. For example, we can roll up a piece of paper into a cylinder, the principal curvatures change from $(0,0)$ to $(0,1)$ at every point of the paper, but the Gaussian curvatures remains 0 everywhere [9].

There is an intuitive way to measure the Gaussian curvature of a surface. We can estimate the Gaussian curvature of a point p on the surface by looking at a geodesic circle with an infinitesimal radius r centered at p. If the Gaussian curvature is 0, the circumference of the circle is $2\pi r$. If the Gaussian curvature is positive as on the sphere, the circumference is smaller. If the Gaussian curvature is negative as on the hyperbolic paraboloid, the circumference is greater.

In discrete setting, we denote $M = (V,E,F)$ a connected triangular mesh 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. We can estimate the Gaussian curvature K_i at Vertex $v_i \in M$ in a similar way to continuous setting:

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

where θ_i^{jk} represents the corner angle attached to vertex v_i in the face f_{ijk}, and ∂M represents the boundary of the mesh. The discrete Gaussian curvatures are determined by the discrete metrics only.

2.2 Conformal Map and Conformal Factor

A conformal map is a one-to-one and continuous map. It preserves angles and local shapes of a surface and is complex differentiable in a neighborhood of every point in its domain.

Denote S a continuous surface embedded in R^3. g is a Riemannian metric induced from the Euclidean metric of S. Suppose $u : S \to R$ is a scalar function defined on S. Then $\overline{g} = e^{2u}g$ is also a Riemannian metric on S and is conformal to the original one. It can be easily proved that the new metric preserves angles and locally only differs a scaling with the original one.

For a discrete surface mesh M, its edge lengths are sufficient to define its Riemannian metric:

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

as long as, for each face f_{ijk}, the edge lengths satisfy the triangle inequality: $l_{ij} + l_{jk} > l_{ki}$.

We apply discrete surface Ricci flow introduced in [10] to compute a flat metric, a new set of edge lengths conformal to the original one and isometrically embedding M to plane. Discrete surface Ricci flow continuously deforms the edge lengths of M according to the difference between the current and target Gaussian curvatures in a heat-like diffusion process, and converges when the difference is less than a threshold. The convergence of discrete surface Ricci flow is proved in [11]. The new set of edge lengths satisfies the target Gaussian curvatures and is conformal to the original one.

Conformal map preserves the local shape of a surface up to a scaling factor. Conformal factor measures the scaling factor of area distortion. We can estimate the conformal factor at vertex v_i by the ratio of the triangle areas in $3D$ and mapped in $2D$ plane of all f_{ijk} incident to v_i. Denote $Area_{3D}|f_{ijk}|$ the area of triangle f_{ijk} in $3D$, and $Area_{2D}|f_{ijk}|$ the area of triangle f_{ijk} in plane.

$$\lambda(v_i) = \frac{\sum_{f_{ijk} \in F} Area_{3D}|f_{ijk}|}{\sum_{f_{ijk} \in F} Area_{2D}|f_{ijk}|}. \tag{2}$$

In practice, we compute $\frac{1}{\lambda}$. Then at the extreme points, the inverse of conformal factor is very close to zero.

The following Lemma shows that the conformal factor increases exponentially with the height of a long and narrow region of a surface, such as the ridge of a terrain surface. We provide the proof in the Appendix.

Lemma 1. *The conformal factor of a long tube shape increases exponentially with the height of the tube, independent of the individual conformal map.*

2.3 Möbius Transformation

A complex number $z = a + bi$ defined on a complex plane can be simply considered as a point $p(a,b)$ on plane, where a and b are x and y coordinates of Point p respectively.

Definition 1 (Möbius Transformation)**.** *A Möbius transformation is a conformal map between complex plane to itself, represented as:*

$$f(z) = \frac{az+b}{cz+d}, \tag{3}$$

where a,b,c,d are complex numbers, satisfying $ad - bc = 1$.

If a Möbius transformation maps four distinct complex numbers z_1, z_2, z_3, z_4 to four distinct complex numbers w_1, w_2, w_3, w_4 respectively, i.e., four distinct planar points are mapped to another four distinct planar points, the Möbius transformation keeps their cross-ratio invariant, represented as:

$$\frac{(z_1 - z_3)(z_2 - z_4)}{(z_2 - z_3)(z_1 - z_4)} = \frac{(w_1 - w_3)(w_2 - w_4)}{(w_2 - w_3)(w_1 - w_4)}. \tag{4}$$

Note that all operations in Eqn. 3 and 4 including addition, subtraction, multiplication, and division are all defined on complex numbers.

2.4 Barycentric coordinates

Barycentric coordinates provide a coordinate system in which the location of a point inside a simplex (a triangle, tetrahedron, etc.) can be specified as a barycenter of masses placed at its vertices. The location of the point can also extend outside the simplex, where one or more coordinates become negative [12].

Barycentric coordinates provide a convenient way to interpolate a function on triangles as long as the function's value is known at vertices. Let's consider a function f defined on a triangle f_{ijk} with $f(v_i)$, $f(v_j)$, and $f(v_k)$ known. Denote $Area|f_{ijk}|$ the area of triangle f_{ijk}. The function value of any point p located inside this triangle can be written as a weighted sum of the function value at the three vertices:

$$f(p) = t_i f(v_i) + t_j f(v_j) + t_k f(v_k), \tag{5}$$

where

$$t_i = \frac{Area|f_{pjk}|}{Area|f_{ijk}|}, \qquad t_j = \frac{Area|f_{pki}|}{Area|f_{ijk}|}, \qquad t_k = \frac{Area|f_{pij}|}{Area|f_{ijk}|}.$$

It is obvious that t_i, t_j, and t_k are subject to the constraint $t_i + t_j + t_k = 1$. t_i, t_j, and t_k are called Barycentric Coordinates of Point p in f_{ijk}.

Note that the function value at each point is defined to be the location information in our localization algorithms.

3 Surface Network Localization

Given a wireless sensor network deployed on a terrain surface, we apply the algorithm proposed in [13] to extract a refined triangular mesh from the connectivity graph of the network based on locally measured distances between nodes within one-hop communication range. Vertices of the triangular mesh are the set of sensor nodes. An edge between two neighboring vertices indicates the communication link between the two sensors. Denote M_1 the triangular mesh converted from the DTM of a terrain surface and M_2 the triangular mesh extracted from the connectivity graph of a network deployed on the terrain surface. We propose three-step localization algorithms. We explain each step in detail, specifically, conformal mapping of M_1 and M_2 to plane in Sec. 3.1, anchor-based alignment of M_1 and M_2 on plane in Sec. 3.2, anchor-free alignment of M_1 and M_2 on plane in Sec. 3.3, and localization of M_2 in Sec. 3.4. The algorithms are fully distributed and have no constraint on communication models. Time complexity and communication cost are analyzed in Sec. 3.5.

3.1 Conformal Map to Plane

Given a triangular mesh $M = (V,E,F)$ embedded in $\mathbb{R}^3$, we apply discrete surface Ricci flow [10] to compute the conformal mapping of M to plane.

Considering that the boundary shape of a large-scale sensor network or a terrain surface can be complicated and concave, the mapping result should be independent of the boundary shape. Therefore we apply discrete surface Ricci flow with the following free-boundary condition: we assign the target Gaussian curvatures

of all non-boundary vertices to zero, and discrete surface Ricci flow deforms only the circle radii of non-boundary vertices. Discrete surface Ricci flow converges when the target Gaussian curvatures of non-boundary vertices equal zero. i.e., flat. Note that boundary vertices are ending vertices of boundary edges. Boundary edges of M can be easily detected since they are shared by only one triangle face. Denote the mapping $f : M \to D \in \mathbb{R}^2$. The mapping result is stored at each v_i as a complex number (i.e., $z = x + yi$), and (x,y) serves as the planar coordinates of v_i.

Algorithm 1 provides the detailed steps of computing conformal map using discrete surface Ricci flow under a free-boundary condition. Algorithm 1 is fully distributed. Each node only needs to exchange information with its one-range neighbors. Note that we can pre-compute the conformal mapping of M_1 under a free-boundary condition to plane and then pre-load the mapping result to sensor nodes before their deployment.

3.2 Anchor-based Alignment

Denote f_1 and f_2 the mappings that conformally map M_1 and M_2 to planar regions D_1 and D_2, respectively. We need to construct another mapping that aligns D_2 with D_1 on plane.

Eqn. 4 provides a natural alignment of two planar regions based on three pairs of anchor points. Denote f a Möbius transformation that maps the planar region D_1 with three distinct points z_1, z_2, z_3 to the planar region D_2 with three distinct points w_1, w_2, w_3. Particularly, z_1, z_2, z_3 are mapped to w_1, w_2, w_3, respectively. We use complex numbers to represent points on plane. Assume we use z_{ij} to denote $z_i - z_j$, and w_{ij} to denote $w_i - w_j$, f can be represented in a closed form from Eqn. 4,

$$f(z) = \frac{w_2(z-z_1)z_{23}w_{12} - (z-z_2)z_{13}w_{23}w_1}{(z-z_1)z_{23}w_{12} - (z-z_2)z_{13}w_{23}}. \quad (6)$$

Note that all the operations in Eqn. 6 are defined on complex numbers.

Assume three anchor nodes - sensor nodes equipped with GPS - are randomly deployed with other sensors. Each anchor node is assigned a set of planar coordinates, e.g., mapped to plane by f_2. Denote the planar point of an anchor node mapped by f_2 with a complex numbers $z_i (1 \le i \le 3)$.

Each anchor node then checks its stored M_1 or simply sends a request with its known geographic position to a server to locate three nearest grid points of the DTM, denoted as v_i, v_j, and v_k. Since M_1 and M_2 are not perfectly overlap in general, the anchor node does not necessarily locate inside $f_{ijk} \in M_1$. We compute the projection point of the anchor node to f_{ijk}. The projection point is the closest point of M_1 to the anchor node. Since f_1 is a continuous and one-to-one mapping, we can compute the planar coordinates of the projection point mapped by f_1 based on the planar coordinates of v_i, v_j, and v_k. Specifically, denote (t_1, t_2, t_3) the Barycentric Coordinates of the projection point on f_{ijk}, $f_1(v_i)$, $f_1(v_j)$, and $f_1(v_k)$ the planar coordinates of v_i, v_j, and v_k mapped by f_1, specifically, the planar coordinates of the projection point mapped by f_1 is: $t_1 f_1(v_i) + t_2 f_1(v_j) + t_3 f_1(v_k)$. Denote the planar coordinates of the projection point mapped by f_1 with a complex number $w_i (1 \le i \le 3)$.

Each anchor node conducts flooding to send out its z_i and w_i to the whole network. When receiving the three pairs of planar coordinates, a non-anchor node $v_i \in M_2$ simply plugs them and its planar coordinates by f_2 into Eqn. 6. The computed one is the aligned planar coordinates of the sensor node.

Algorithm 1 Algorithm of Conformal Map under Free-boundary Condition

Input: Triangular mesh M
Output: Planar coordinates of $v_i \in V$

for all $v_i \in V$ **do**
  $u_i = 0$. {initialize a scale function defined at v_i}
end for
non-stop = true
while non-stop **do**
  for all $e_{ij} \in E$ **do**
    Calculate edge length l_{ij}
    $l_{ij} = e^{u_i} * e^{u_j} * d_{ij}$
    {d_{ij} is the measured distance between v_i and v_j}
  end for
  for all $v_i \in V$ **do**
    for all $\theta_i^{jk} | f_{ijk} \in F$ **do**
      Compute corner angle θ_i^{jk}

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

      {Inverse cos law}
    end for
    Compute Gaussian curvature K_i at v_i (Eqn. 1)
    if $v_i \notin \partial M$ **then**
      Update u_i: $u_i = u_i - \delta K_i$
      {v_i non-boundary vertex}
      { δ is the step length set to 0.05 in our simulations}
    end if
  end for
  $K_{max} = 0$
  for all $v_i \notin \partial M$ **do**
    if $|K_i| > K_{max}$ **then**
      $K_{max} = K_i$
    end if
  end for
  if $K_{max} < \varepsilon$ **then**
    non-stop = false
  end if
end while
v_0 marks itself and sets its planar coordinates $p_0^{2D} = (0,0)$
{v_0 can be a random vertex or one with the smallest node ID}
v_0 marks one of its direct neighbors e.g., v_1, and sets $p_1^{2D} = (0, l_{01})$
for all $v_i \in V$ **do**
  if v_i not marked **then**
    for all $\theta_i^{jk} | f_{ijk} \in F$ **do**
      if both v_j and v_k marked **then**
        Compute the intersection points of two circles centered at p_j^{2D} and p_k^{2D} with radii l_{ij} and l_{ik}, respectively
        p_i^{2D} is set as one intersection point s.t. p_i^{2D}, p_j^{2D}, and p_k^{2D} follow the right-hand rule
      end if
    end for
  end if
end for

3.3 Anchor-Free Alignment

The anchor-free alignment algorithm extracts feature points of the triangular meshes of the network and the terrain surface in Sec. 3.3.1, matching the two sets of feature points in Sec. 3.3.2, aligning the mapped network mesh to the terrain one in the plane based on the matched feature points in Sec. 3.3.3.

3.3.1 Feature Points Extraction

For both the network and terrain meshes, we compute the conformal factor and Gaussian curvature at each vertex based on Eqns. 1 and 2, respectively. Considering that a triangular mesh with noise may affect the approximation accuracy of discrete Gaussian curvature, we modify Eqn. 1 to compute K_i at each v_i by averaging Gaussian curvatures within one-hop region of v_i:

$$\bar{K}_i = \frac{K_i + \sum_{j=1}^{m} K_j}{1+m} \tag{7}$$

where m is the number of one-hop neighbors of Vertex v_i.

We then select feature points based on the conformal factor and discrete Gaussian curvature at each vertex and classify them into three groups. Specifically, Group I are the feature points with high conformal factors and the highest positive Gaussian curvatures. Vertices belonging to group I lie on the ridge or peak of a terrain. Group II are the feature points with high conformal factors and zero Gaussian curvatures. Vertices in group II distribute along the ridge of a mountain. Group III are feature points with high conformal factors and the lowest negative Gaussian curvatures. Vertices in group III are saddle points between two neighboring mountains. Vertexes with the same group ID and connecting with each other form a cluster. To reduce the computational complexity of later feature points match, our algorithm samples vertices within a cluster. Algorithm 2 introduces the details to extract feature points from the network and terrain meshes.

Algorithm 2 Algorithm to Extract Feature Points

Input: Triangular mesh M, $v_i \in V$ with p_i^{3D} and p_i^{2D}
Output: Feature points and their group and cluster IDs

for all $v_i \in V$ **do**
  Calculate the conformal factor (Eqn. 2)
  Calculate the Gaussian curvature (Eqn. 7)
end for
Sort vertices by conformal factors in descending order
Denote λ_+ the set of vertices ranked in top 5% of the list
Sort vertices by Gaussian curvatures in descending order
Denote K_+ the set of vertices with positive Gaussian curvatures and ranked in top 5% of the list
Denote K_0 the set of vertices with Gaussian curvatures less than ε
Denote K_- the set of vertices with negative Gaussian curvatures and ranked in bottom 5% of the list
$\lambda_+ \cap K_+$: vertices in group I
$\lambda_+ \cap K_0$: vertices in group II
$\lambda_+ \cap K_-$: vertices in group III
Vertexes with same group ID and connecting with each other form a cluster
for Each cluster **do**
  if Size ≥ 5 **then**
    Sort vertexes by conformal factors in descending order
    Insert vertices ranked in top 20% into the list of feature points.
    Update $\lambda_{\min}$
    {$\lambda_{\min}$ stores the smallest conformal factor in the list of feature points}
  end if
end for
for Each cluster **do**
  if Size < 5 **then**
    Pick v_i with the highest conformal factor
    if $\lambda(v_i) > \lambda_{\min}$ **then**
      Insert v_i into the list of feature points.
    end if
  end if
end for
return A list of feature points with each feature point associated with group and cluster IDs.

3.3.2 Feature Points Matching

We modify the point pattern matching method with fuzzy relaxation in [14] to match feature points in the same group. We represent feature points extracted from the network and terrain meshes as $A_K = \{a_1, ..., a_n\}$ and $B_K = \{b_1, ..., b_m\}$, respectively where $K = \{I, II, III\}$. Note that a_i in group K may correspond to several b_j with the same group ID, or not any b_j.

A probability function p_{ij} is defined to represent the probability that a_i matches b_j.

A compatibility function $c(a_i, b_j, a_h, b_k)$ is defined to measure how much the position of a_h relative to a_i differs from the position of b_k relative to b_j. Specifically, the compatibility function $c(a_i, b_j, a_h, b_k)$ is defined as:

$$c(a_i, b_j, a_h, b_k) = \frac{1}{1+\delta^2}, \tag{8}$$

where

$$\delta = \frac{d_{jk} - \alpha * d_{ih}}{d_{jk}}.$$

The distance function d is measured by the shortest surface distance, i.e., the sum of edge lengths, between two feature points on a triangular surface. Notice that α is a scaling factor defined as the ratio of the average distances of feature points in DTM and feature points in a network model. We require that $i \neq h$ and $j \neq k$.

The relaxation formula at the r^{th} iteration is defined as:

$$p_{ij}^{(r+1)} = \frac{1}{n} \sum_{h=1}^{n} \left[\max_{k=1}^{m} c(a_i, b_j, a_h, b_k) p_{hk}^{(r)} \right]. \tag{9}$$

The initial value of $p_{ij}^{(0)}$ is set to 1. When p_{ij} convergences, the algorithm returns the probability of A_i matches B_j.

Algorithm 3 gives the steps to find the matched feature points. When the algorithm convergences, for feature points in each cluster of the network mesh, we choose one with the largest probability matched to the terrain one. If two feature points in different clusters of the network mesh are matched to the same cluster in the terrain one, we select the one with a larger probability. We then sort the matched pairs of feature points based on the probability and choose the top pairs for later alignment.

3.3.3 Alignment

With the matched feature points denoted as $A = \{a_1, ..., a_n\}$ and $B = \{b_1, ..., b_n\}$ of the network and terrain meshes, respectively, we need to find a translation vector denoted as T and a rotation matrix denoted as R to well align them in plane. Formally speaking, we

Algorithm 3 Algorithm to Match Feature Points

Input: α, $A_k, B_k, k = I, II, III$
Output: p_{ij}

for all a_i **do**
  for all b_j **do**
    if a_i and b_j in the same group **then**
      $p_{ij} = 1$.
    end if
  end for
end for
while p_{ij} convergence **do**
  for all a_i **do**
    for all b_j **do**
      if a_i and b_j in the same group **then**
        for all a_h & $h \neq i$ **do**
          for all b_k & $k \neq j$ **do**
            $c(i, j, h, k) = 0$
            if a_h and b_k in the same group **then**
              Calculate the $c(i, j, h, k)$ by (8).
            end if
          end for
        end for
      end if
    end for
    Calculate the $p_{ij}^{(r+1)}$ by (9).
  end for
end while
For each a_i, find b_j with the highest p_{ij}.
return p_{ij}

need to find a 2-vector T and a $2x2$ matrix R such that:

$$\sum_{i=1}^{n} ||(Ra_i + T) - b_i||^2$$

is minimized.

Denote $\bar{a}$ and $\bar{b}$ the centroids of A and B, respectively, where

$$\bar{a} = \frac{1}{n}\sum_{i=1}^{n} a_i, \qquad \bar{b} = \frac{1}{n}\sum_{i=1}^{n} b_i.$$

T is computed as $T = \bar{a} - \bar{b}$. Denote $X = x_1, ..., x_n$ and $Y = y_1, ..., y_n$ where $x_i = a_i - \bar{a}$ and $y_i = b_i - \bar{b}$, respectively. Denote $S = XY^T$. We have the singular value decomposition (SVD) of S: $S = U\sigma V^T$. R is computed as $R = VU^T$. The proof is provided in Appendix.

The computed optimal T and R are then broadcasted by feature points to the whole network. Each sensor node simply plugs its planar coordinates p_i^{2D} computed in Sec. 3.1 into the following equation:

$$Rp_i^{2D} + T.$$

3.4 Localization

With the aligned planar coordinates to the terrain one, each sensor node locates three nearest grid points of the terrain DTM on plane. Denote v_i, v_j, and v_k the three nearest grid points to a sensor node. To compute the $3D$ geographic location, the node computes the Barycentric Coordinates using its own planar coordinates with respect to v_i, v_j, and v_k mapped on plane (Eqn. 5). Denote (t_i, t_j, t_k) the Barycentric Coordinates. The sensor node finds its $3D$ geographic coordinates as

$$t_i p_i^{3D} + t_j p_j^{3D} + t_k p_k^{3D},$$

where p_i^{3D}, p_j^{3D}, and p_k^{3D} are the 3D geographic coordinates of v_j, v_k, and v_l respectively.

3.5 Time Complexity and Communication Cost

Assume we measure the communication cost by the number of exchanged messages. Both the time complexity and communication cost of the proposed localization algorithms are dominated by the step to compute conformal mapping of M_1 and M_2 to plane. The time complexity of discrete surface Ricci flow is measured by the number of iterations, given by $-C\frac{log\varepsilon}{\lambda}$, where C is a constant, ε is a threshold of curvature error, and λ is the step length of each iteration (we set to 0.05 in our experiments) [11]. Since each vertex only needs to exchange u values with its direct neighbors, the communication cost is given by $O(-C\frac{log\varepsilon}{\lambda}ng)$, where g is the average vertex degree of M, and n is the size of M. Note that g is six for a triangular mesh. The time complexity and communication cost of planar embedding based on computed edge lengths by discrete surface Ricci flow are linear to n.

A special note is that we don't need to compute the conformal mapping of M_1 each time. We only need to compute it once before we start to deploy a network, and then pre-load only the mapping data related to the FoI (Field of Interest) to sensor nodes if they have sufficient storage. Otherwise, a server may be designated to keep the DTM database.

4 Discussions

4.1 The Size of Anchor Nodes

Theoretically speaking, the proposed localization algorithm requires only three anchor nodes to align two triangular meshes on a plane. If there are more than three anchor nodes deployed with the network, we can apply the least-square conformal mapping method in [15] instead of Möbius transformation to incorporate more anchor nodes into the alignment to improve the localization accuracy.

Fig. 3 shows one example. For a network with size 2.6k deployed on a 3D surface as shown in Fig. 4(a), the localization error of the network decreases with the increased number of anchor nodes. Compared with Möbius transformation based alignment introduced in Sec. 3.2, least-square conformal mapping based alignment is more flexible to take anchor nodes into alignment. But from the other side, theleast-square conformal mapping method introduced in [15] is centralized with high computational complexity.

4.2 Connectivity Only

When range distance measurement is not available, we can still extract a sparse triangular mesh from a network connectivity graph. A simple landmark-based algorithm discussed in [16], [17] uniformly selects a subset of nodes in a distributed way and denotes them as landmarks, such that any two neighboring landmarks are approximately a fixed K hops away ($K \geq 6$). The dual of a discrete Voronoi diagram with generators the set of landmarks forms a triangulation. Vertices of the triangulation are the set of landmarks. The edge between two neighboring vertices is the shortest path between the two landmarks. We simply assume the edge length of the triangulation a unit one, and then apply the same anchor-based localization algorithm as discussed in Sec. 3 to localize landmark nodes.

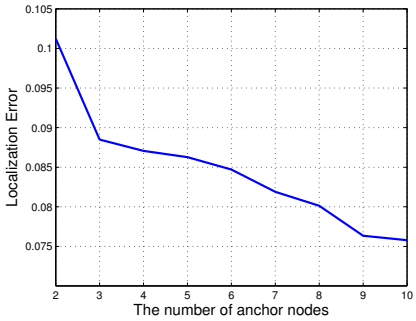


Fig. 3. Localization error decreases with the increased number of anchor nodes. Note that localization error is computed as a ratio of the average node distance error (all sensor nodes of a network) and the average node transmission range.

A non-landmark node, denoted as n_i, finds its three nearest landmarks, denoted as v_1, v_2, v_3 with computed 3D coordinates $p(v_1)$,$p(v_2)$, and $p(v_3)$ respectively. Denote d_1, d_2, and d_3 the shortest distances (hop counts) of node n_i to the three landmarks v_1, v_2, v_3 respectively. Then node n_i computes its 3D coordinates $p(n_i)$ simply by minimizing the mean square error among the distances:

$$\sum_{j=1}^{3}(|p(n_i)-p(v_j)|-d_j)^2. \quad (10)$$

4.3 Network Density

The algorithm in [13] to extract a triangular mesh from the connectivity graph of a network assumes that a triangular graph is a sub-graph of the connectivity graph of the network. Such an assumption is true only when the node density of the network is not too low. In our simulations, the average node degree of the connectivity graph of a network is around or above 8.

4.4 DTM Resolution and Availability

For a large-sized field of interest covered with a sensor network and a corresponding DTM with high resolution, e.g., a grid of size less than $1m^2$, it is unnecessary to apply such a high-density DTM to locate sensor nodes with both the transmission range and average pair of sensor node distance much higher than the grid size. A realistic solution to speed up the total computing time without reducing the localization accuracy is to downsample the DTM such that its resolution is aligned with the density of a sensor network

In case DTM is not available, a sensor network deployed on a 3D terrain surface can be localized with a high cost by deploying a set of anchor nodes in a structured way on the terrain, simulating a sparse version of DTM.

5 Simulations

We pick a set of representative terrain surfaces with their digital terrain models (DTMs) available on the web shown in the first column of Fig. 4. Wireless sensor nodes are randomly deployed on the surfaces, as shown in the second column of Fig. 4. The sizes of sensor networks deployed on terrain surfaces I, II, III. VI, and V are 2.6k, 3.6k, 5.1k, 8.2k, and 9k, respectively. For both the terrain surfaces and the networks, a convex shape is not a necessary condition. The third column of Fig. 4 shows the localized sensor networks based on the proposed localization algorithms.

TABLE 1
The distribution of Anchor-based Localization Errors under Different Sets of Anchor Nodes

Terrain		I	II	III	IV	V
Error	μ	0.1356	0.2098	0.0951	0.6680	0.3613
	$\tilde{x}$	0.1343	0.1512	0.0956	0.3399	0.1785
	σ	0.1717	0.0352	0.0158	0.6268	0.3106

We carry out extensive simulations under various scenarios to evaluate the overall performance of the proposed algorithms with different factors such as the one-hop distance measurement error, the resolution of a DTM, and the performance of the algorithm in the situation of connectivity only. Note that there is no simple alternative to localize a surface network as we discussed in Sec. 1, so there is no comparison with the existing method.

5.1 Localization Error

We compute localization error as a ratio of the average node distance error (all sensor nodes of a network) and the average node transmission range. We assume the accuracy of distance measurement between nodes within the communication range in this subsection.

5.1.1 Anchor-based Localization

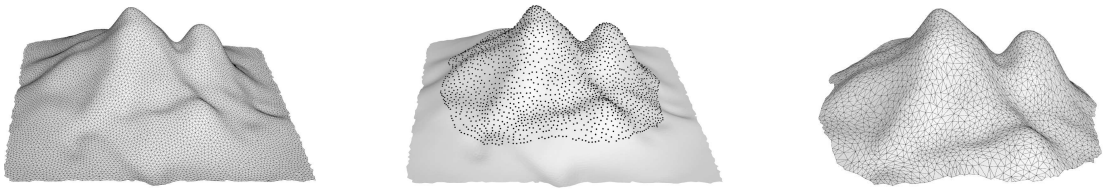
We assume sensor nodes with accurate one-hop distance measurement and DTMs with high resolutions. For each network, we randomly deploy three anchor nodes and calculate the localization errors of the network based on the anchor-based algorithm in Sec. 3. We repeat eight times for each network. Denote x_i the i^{th} localization error. We compute the arithmetic mean $\mu = \frac{1}{8}\sum_{i=1}^{8} x_i$ and the standard deviation $\sigma = \sqrt{\frac{1}{7}\sum_{i=1}^{8}(x_i-\mu)^2}$. Table 1 shows the mean (μ), the median ($\tilde{x}$), and the standard deviation (σ) of localization errors under different sets of anchor nodes. The positions of anchor nodes affect the performance of the anchor-based localization algorithm. In general, the more scattered we deploy the three anchor nodes in a network, the lower the localization error is. In an extreme case, all anchor nodes are dropped to the same spot. Since the aligned triangular meshes of a network and terrain surface on a plane differ a rotation, the anchor-based localization algorithm fails.

5.1.2 Anchor-Free Localization

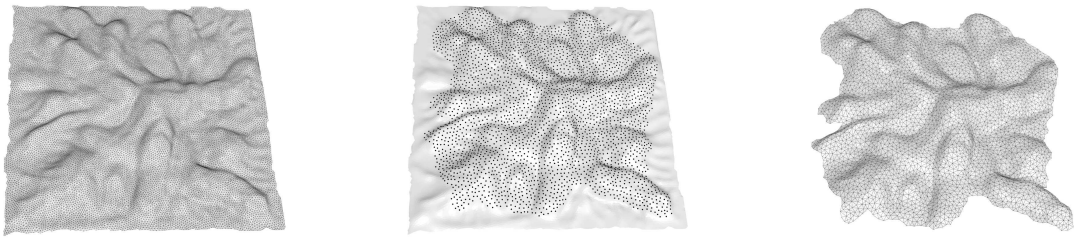
Table 2 gives the localization error of the anchor-free algorithm. The proposed anchor-free algorithm achieves a reasonably good localization accuracy except for a sensor network deployed on terrain surface V. We will discuss and analyze the failure case in Sec. 5.6.

TABLE 2
Anchor-free Localization Error

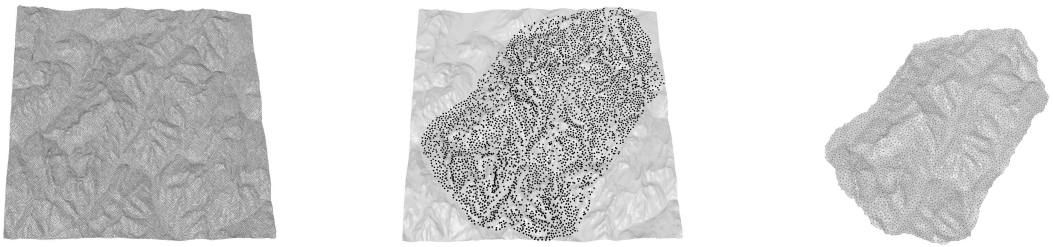
Terrain	I	II	III	VI	V
Error	0.1543	0.4851	0.3304	0.4828	6.6471



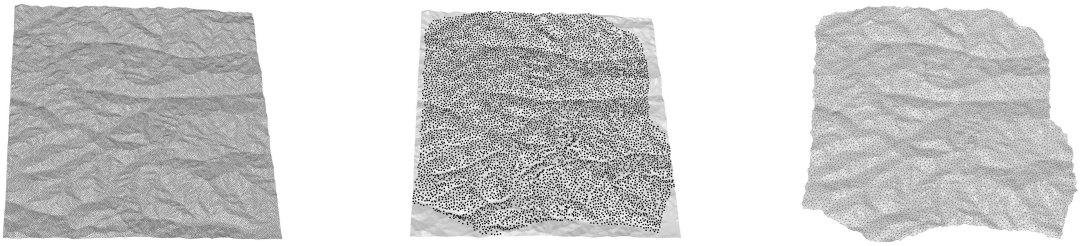
(a) WSN deployed on terrain surface I



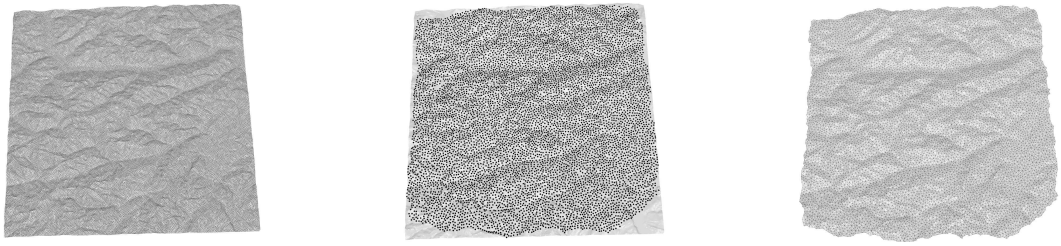
(b) WSN deployed on terrain surface II



(c) WSN deployed on terrain surface III



(d) WSN deployed on terrain surface IV



(e) WSN deployed on terrain surface V

Fig. 4. The first column shows the triangular meshes converted from the DTMs of a set of terrain surfaces. The second column shows large-scale wireless sensor nodes randomly deployed on the terrain surfaces. The third column shows the localized sensor networks.

5.1.3 Comparison of Anchor-based and Anchor-free Localizations

Fig. 5 compares the performances of the anchor-based and anchor-free localization algorithms. It is obvious that the performance of the anchor-based algorithm is better than the anchor-free one, although the two algorithms both achieve a reasonably good localization accuracy except for the anchor-free one on terrain surface V. Note that for each network, we choose the set of anchor nodes that gives a median localization error based on the repeated tests in Sec. 5.1.1.

5.2 Distance Measurement Error

We test the performances of our algorithms with distance measurement error.

5.2.1 Anchor-based Localization

For each network, we choose the set of anchor nodes that gives the median localization error based on the repeated tests in Sec. 5.1.1. The third column of Table 3 gives the localization errors of the anchor-based algorithm with distance measurement error. The results show that the anchor-based algorithm tolerates a small measurement error, but the performance drops with a relatively large one. A feasible solution for a network with potentially

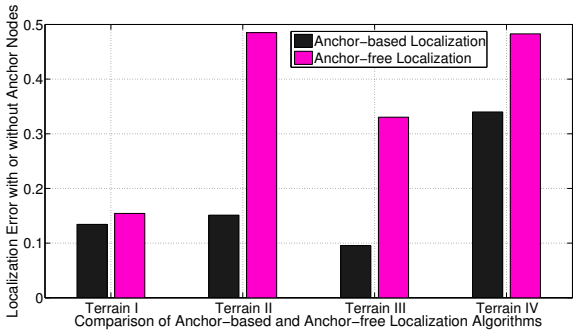


Fig. 5. Comparison of Anchor-based and Anchor-free Localization Algorithms. Note that localization error is computed as a ratio of the average node distance error (all sensor nodes of a network) and the average node transmission range.

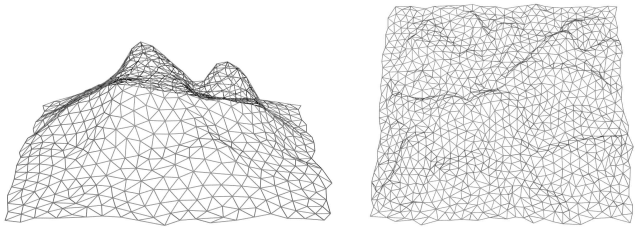
TABLE 3
Localization Error with Distance Measurement Error

Terrain	Distance measurement error (%)	Anchor-based localization algorithm	Anchor-free localization algorithm
I	0	0.1343	0.1543
	5	0.1886	0.3121
	10	0.3816	0.4333
	15	0.5658	−−
	20	0.7250	−−
II	0	0.1512	0.4851
	5	0.1495	0.7778
	10	0.1519	1.0486
	15	0.2400	−−
	20	0.3400	−−
III	0	0.0956	0.3304
	5	0.1433	0.4479
	10	0.1971	0.6702
	15	0.2540	−−
	20	0.3076	−−
IV	0	0.3399	0.4828
	5	0.3738	0.2027
	10	0.4037	0.9164
	15	0.4472	−−
	20	0.4690	−−
V	0	0.1785	6.6471
	5	0.2072	−−
	10	0.2210	−−
	15	0.2380	−−
	20	0.2642	−−

large measurement errors is to select uniformly a set of landmark nodes such that each landmark node has a one-hop distance to its landmark neighbors, i.e., a Voronoi diagram with small and constant cell size. A triangular mesh can be constructed from the chosen landmark nodes with edge length approximately the averaged transmission range. Similar to connectivity-based surface localization discussed in Sec. 4.2, we localize the landmark nodes first and then other non-landmark nodes.

5.2.2 Anchor-Free Localization

The fourth column of Table 3 gives the localization errors of the anchor-free algorithm with distance measurement error. The results show that the anchor-free algorithm is sensitive to measurement error. The reason is that the error of edge lengths decrease the accuracy of the computed conformal factors and Gaussian curvatures of a network mesh. Therefore, the algorithm may extract wrong feature points from a network mesh to match



(a) Terrain I (b) Terrain II

Fig. 6. Low resolution terrain surfaces with only 5% of their original resolutions shown in Fig. 4(a) and (b).

TABLE 4
Localization Error with Different Resolutions of DTM

Terrain	Percentage of its original resolution	Anchor-based Localization Algorithm	Anchor-free Localization Algorithm
I	1	0.1343	0.1543
	20%	0.1347	0.3597
	10%	0.1650	0.6929
	5%	0.3400	0.5840
II	1	0.1512	0.4851
	20%	0.2150	0.5213
	10%	0.2200	0.5765
	5%	0.2700	0.5994

those extracted from the terrain one, generating a misalignment of the network and the terrain meshes in a plane. A sensor node thus picks wrong grid points for the reference to compute its coordinates in $3D$.

5.3 Terrain Models with Different Resolutions

To evaluate the impact of the resolution of a DTM, we compute the localization errors of a network deployed on a terrain surface with different resolutions of its DTM. We pick the terrain models shown in Fig. 4(a) and (b) as the testing ones. The original DTM shown in Fig. 4(a) and (b) has the highest resolution. We lower the density of the original one to 20%, 10%, and 5%, respectively such that the highest density of a DTM is twenty times the lowest one. Fig. 6(a) and (b) show the two DTMs with only 5% resolutions of their original ones.

5.3.1 Anchor-based Localization

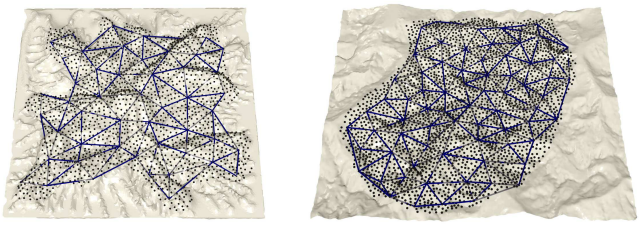
The third column of Table 4 gives the localization errors of the anchor-based algorithm. The results show that the resolution of a DTM has a small impact on the performance of the anchor-based algorithm unless it is extremely low. Note that for each network, we choose the set of anchor nodes that gives a median localization error based on the repeated tests in Sec. 5.1.1.

5.3.2 Anchor-Free Localization

The fourth column of Table 4 gives the localization errors of the anchor-free algorithm. The results show that the resolution of a DTM has some impact on the performance of the anchor-free algorithm. It is difficult to extract feature points from a triangular mesh with very low density.

5.4 Networks with Connectivity Information Only

As we discussed in Sec. 4.2, we uniformly select a subset of nodes marked as landmark nodes for a network with mere connectivity.



(a) Terrain II (b) Terrain III

Fig. 7. Sparse Triangulations Extracted from Networks with Connectivity Information Only.

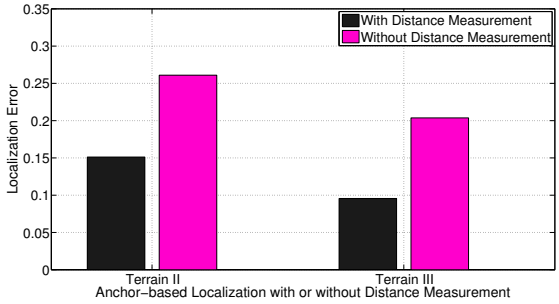


Fig. 8. Anchor-based Localization with or without Distance Measurement. Note that localization error is computed as a ratio of the average node distance error (all sensor nodes of a network) and the average node transmission range.

We then build a sparse triangulation with each vertex a landmark node and each edge approximated by a constant length, i.e., a fixed hop count. Fig. 7 shows the sparse triangular meshes extracted from two networks with size $2k$ and $3k$ deployed on terrain surfaces II and III, respectively. The localization errors for landmark nodes of the two networks are 0.2610 and 0.2037 respectively.

Fig. 8 compares the performances of the anchor-based localization algorithms with and without distance measurement within a one-hop communication range. It is obvious that the algorithm performs better with distance measurement. However, the localization accuracy of the algorithm without distance measurement is still reasonably good.

5.5 The Convergence Time

We carry out experiments to test the number of iterations of discrete surface Ricci flow required for convergence. Fig. 9 shows the convergence rates of discrete surface Ricci flow on two networks with size $2k$ and $3k$ deployed on terrain surfaces II and III, respectively. For all network models in simulations, discrete surface Ricci flow converges in less than hundreds of iterations. For a large-size terrain surface mesh, we can pre-compute its free-boundary conformal map and then load the result to an individual sensor node before deployment. We can also apply Newton's numerical method to compute the solution of discrete surface Ricci flow. The computation of the centralized method is efficient with less than ten iterations in a few seconds for a triangular mesh with $10k$ size.

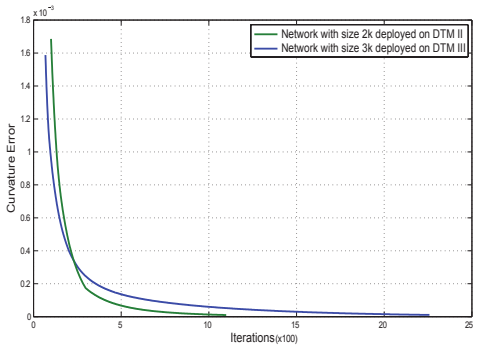


Fig. 9. The convergence rate of the discrete surface Ricci flow algorithm.

5.6 Limitation of Anchor-free Localization

The anchor-free algorithm fails to localize the network deployed on terrain surface V since the localization error is much larger than the transmission range as given in Table. 2. One factor is that terrain surface V is quite flat with fewer features such as ridge or valley. The proposed algorithm can extract only a limited number of feature points from the triangular mesh of the terrain surface. The other one is that the sensor network deployed on terrain surface V is sparse. The extracted triangular mesh from the connectivity graph of the network contains fewer features.

The difference of conformal factors: $\max\lambda - \min\lambda$ indicates how sharp the features of a mesh are. Therefore, it is an effective indicator to pre-determine whether we can apply the anchor-free localization algorithm for a given surface network. Denote the differences of conformal factors as $\Delta\lambda_a$ and $\Delta\lambda_b$ for the triangular meshes of a network and a terrain surface, respectively. We then compute a relative value defined as the following:

$$E = \frac{\Delta\lambda_b - \Delta\lambda_a}{\Delta\lambda_b}. \tag{11}$$

The relative value indicates the percentage of sharp features of a network mesh has lost compared with its deployed terrain one.

Table 5 gives the three values of networks deployed on terrain surfaces shown in Fig. 4. It is obvious that the shape of terrain surface V is the flattest one with the smallest value of $\Delta\lambda_b$ among all the terrain meshes. Meantime, the relative value of the network deployed on terrain surface V is the largest one, more than 50%. It shows that the sensor network deployed on terrain surface V is sparse with a flat triangular mesh extracted from its connectivity graph and many features have been smoothed out.

TABLE 5
The relative value of different terrain model

Terrain	I	II	III	IV	V
Network $\Delta\lambda_a$	5.0832	1.5471	0.7202	0.8538	0.3684
DTM $\Delta\lambda_b$	5.6327	1.7468	1.3361	1.5278	0.9220
E	9.76%	11.43%	46.10%	44.12%	60.04%

6 Conclusion

We have introduced two localization algorithms with and without anchor nodes for sensor networks deployed on the surfaces of 3D terrains. The basic idea of the two algorithms is to construct

a well-aligned mapping between two triangular meshes. One is converted from the DTM of a terrain surface and the other is extracted from the connectivity graph of a network deployed on the terrain surface. Based on the mapping, each sensor node of the network can easily locate reference grid points from the DTM to calculate its own geographic location. It is much more challenging to construct an alignment between the two meshes without the location information of an anchor node. We extract feature points with geometric properties intrinsic to surface distances and independent of the embedding of a surface in 3D from the two meshes. The matched feature points guide the alignment of the two meshes.

We have carried out extensive simulations under various scenarios to evaluate the overall performance of the proposed algorithms with different factors. We have discussed the impact of distance measurement error on the localization accuracy and the limitation of the proposed anchor-free localization algorithm. We have also discussed the possibility of 3D surface network localization with mere connectivity only.

Acknowledgments

Xuan Li and Miao Jin are partially supported by NSF CNS-1320931.

References

[1] G. Werner-Allen, K. Lorincz, J. Johnson, J. Lees, and M. Welsh, "Fidelity and yield in a volcano monitoring sensor network," in *Proceedings of the 7th symposium on Operating systems design and implementation*, OSDI '06, pp. 381–396, 2006.

[2] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. S. Peh, and D. Rubenstein, "Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with zebranet," *SIGARCH Comput. Archit. News*, vol. 30, no. 5, pp. 96–107, 2002.

[3] Y. Zhao, H. Wu, M. Jin, and S. Xia, "Localization in 3d surface sensor networks: Challenges and solutions," in *Proc. of the 31st Annual IEEE Conference on Computer Communications (INFOCOM'12)*, pp. 55–63, 2012.

[4] A. Pressley, *Elementary Differential Geometry*. Springer, 2010.

[5] S. Ray, W. A. Miller, P. M. Alsing, and S.-T. Yau, "Adiabatic isometric mapping algorithm for embedding 2-surfaces in euclidean 3-space," *Classical and Quantum Gravity*, vol. 32, no. 23, 2015.

[6] Y. Zhao, H. Wu, M. Jin, Y. Yang, H. Zhou, and S. Xia, "Cut-and-sew: A distributed autonomous localization algorithm for 3d surface wireless sensor networks," in *Proc. of the 14th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'13)*, 2013.

[7] Shuttle Radar Topography Mission (SRTM). http://www2.jpl.nasa.gov/srtm/.

[8] R. Remmert, *Classical topics in complex function theory*. Springer-Verlag, 1998.

[9] E. of Notices of the AMS, "What is gauss curvature?," *Notices of the American Mathematical Society*, vol. 63, no. 2, pp. 144–145, 2016.

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

[11] B. Chow and F. Luo, "Combinatorial Ricci Flows on Surfaces," *Journal Differential Geometry*, vol. 63, no. 1, pp. 97–129, 2003.

[12] E. Hille, *Analytic Function Theory, Volume I*. Chelsea Publishing Company, 1982.

[13] H. Zhou, H. Wu, S. Xia, M. Jin, and N. Ding, "A distributed triangulation algorithm for wireless sensor networks on 2d and 3d surface," in *Proc. of INFOCOM*, pp. 1053–1061, 2011.

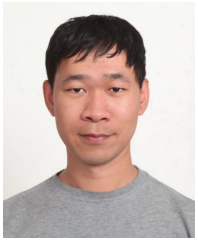
[14] A. Rosenfeld and A. C. Kak, *Digital picture processing*. Academic Press, New York, 1982.

[15] B. Lévy, S. Petitjean, N. Ray, and J. Maillot, "Least squares conformal maps for automatic texture atlas generation," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 362–371, 2002.

[16] S. Funke and N. Milosavljevic, "Guaranteed-delivery geographic routing under uncertain node locations," in *Proc. of INFOCOM*, pp. 1244–1252, 2007.

[17] R. Sarkar, X. Yin, J. Gao, F. Luo, and X. D. Gu, "Greedy routing with guaranteed delivery using ricci flows," in *Proc. of IPSN*, pp. 121–132, 2009.

Xuan Li Xuan Li is currently working toward the Ph.D. degree in computer science at the Center for Advanced Computer Studies (CACS), University of Louisiana at Lafayette. She earned her bachelor's degree and master's degree from Beijing Jiaotong University in 2012 and 2015, respectively. Her research focuses on geometric algorithm design and localization in large scale wireless sensor networks.



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

Yang Yang received the B.S. degree in computer science from Northwestern Polytechnical University, Xian, China in 2009, and the M.S. and Ph.D. degrees in computer sciences from the University of Louisiana at Lafayette, Lafayette, LA, USA, in 2011 and 2014, respectively. His research focuses on designing geometric algorithms for wireless sensor networks in both areas of in-network information processing and localization.

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

APPENDIX

Lemma 2. *The conformal factor of a long tube shape increases exponentially with the height of the tube, independent of individual conformal map.*

Proof. Suppose we have a long thin cylinder and we plan to conformally parameterize it. If we use polar coordinates (ρ,θ) with The center of the top mapped to the origin, the conformal factor is a function dependent only on ρ because of symmetry. The Gaussian curvature of the cylinder is zero, and

$$k(\rho,\theta) = \frac{1}{\lambda^2}\Delta\log\lambda = 0. \tag{12}$$

We can deduce $\lambda(\rho) = e^{a\rho+b}$, where a,b are constants. No matter what kind of conformal map we choose, the stretching is exponential. □

Lemma 3. *Given two sets of matched points denoted as $A = \{a_1,...,a_n\}$ and $B = \{b_1,...,b_n\}$. A translation vector T and rotation matrix R that minimize:*

$$\sum_{i=1}^{n} ||(Ra_i+T)-b_i||^2,$$

satisfy:

Denote $\bar{a}$ and $\bar{b}$ the centroids of A and B, respectively. T satisfies $T = \bar{a}-\bar{b}$.

Denote $X = \{x_1,...,x_n\}$ and $Y = \{y_1,...,y_n\}$ where $x_i = a_i - \bar{a}$ and $y_i = b_i - \bar{b}$, respectively. Denote $S = XY^T$. We have the singular value decomposition (SVD) of S: $S = U\sigma V^T$. R satisfies $R = VU^T$.

Proof. Assume R is fixed and denote

$$F(T) = \sum_{i=1}^{n} ||(Ra_i+T)-b_i||^2.$$

By taking the derivative of F w.r.t. T, we have:

$$\begin{aligned}\frac{\partial F(T)}{T} &= \sum_{i=1}^{n} 2(Ra_i+T-b_i)\\ &= 2R\sum_{i=1}^{n} a_i + 2Tn - 2\sum_{i=1}^{n} b_i.\end{aligned}$$

Denote:

$$\begin{aligned}\bar{a} &= \frac{1}{n}\sum_{i=1}^{n} a_i,\\ \bar{b} &= \frac{1}{n}\sum_{i=1}^{n} b_i.\end{aligned}$$

$$\begin{aligned}\frac{\partial F(T)}{T} &= 2Rn\bar{a} + 2Tn - 2n\bar{b}\\ &= 0.\end{aligned}$$

We have:

$$T = \bar{a} - R\bar{b},$$

where T transforms the centroid of A to the centroid of B.

Plug $T = \bar{a} - R\bar{b}$ into the objective function:

$$\begin{aligned}\sum_{i=1}^{n} ||(Ra_i+T)-b_i||^2 &= \sum_{i=1}^{n} ||(Ra_i+\bar{b}-R\bar{a})-b_i||^2\\ &= \sum_{i=1}^{n} ||R(a_i-\bar{a})-(b_i-\bar{b})||^2\end{aligned}$$

Denote:

$$\begin{aligned}x_i &= a_i - \bar{a},\\ y_i &= b_i - \bar{b}.\end{aligned}$$

$$\begin{aligned}\sum_{i=1}^{n} ||(Ra_i+T)-b_i||^2 &= \sum_{i=1}^{n} ||Rx_i-y_i||^2\\ &= \sum_{i=1}^{n} (Rx_i-y_i)^T(Rx_i-y_i)\\ &= \sum_{i=1}^{n} (x_i^Tx_i - y_i^TRx_i - x_i^TR^Ty_i + y_i^Ty_i)\end{aligned}$$

Considering

$$min\sum_{i=1}^{n}(-y_i^TRx_i - x_i^TR^Ty_i) = max\sum_{i=1}^{n}(y_i^TRx_i + x_i^TR^Ty_i)$$

and

$$x_i^TR^Ty_i == y_i^TRx_i$$

We need to find R such that

$$max\sum_{i=1}^{n}(y_i^TRx_i)$$

Denote:

$$\begin{aligned}Y &= \sum_{i=1}^{n} y_i,\\ X &= \sum_{i=1}^{n} x_i.\end{aligned}$$

$$\begin{aligned}\sum_{i=1}^{n}(y_i^TRx_i) &= tr(Y^TRX)\\ &= tr(RXY^T).\end{aligned}$$

Compute singular value decomposition of $XY^T = U\sigma V^T$

$$\begin{aligned}\sum_{i=1}^{n}(y_i^TRx_i) &= tr(RU\sigma V^T)\\ &= tr(\sigma(V^TRU)).\end{aligned}$$

Since V, R, and U are all orthogonal matrices, $M = V^TRU$ is also an orthogonal matrix with all entries m_{ij} are smaller than 1 in magnitude

$$\begin{aligned}tr(\sigma(V^TRU)) &= \sum_{i=1}^{2}\sigma_i m_{ii}\\ &\le \sum_{i=1}^{2}\sigma_i.\end{aligned}$$

When $R = VU^T$, we have $V^TRU = I$.

□