A local Mesh Method for Solving PDEs on Point Clouds

Rongjie Lai * Jian Liang † HongKai Zhao ‡

Abstract

In this work, we introduce a numerical method to approximate differential operators and integrals on point clouds sampled from a two dimensional manifold embedded in $\mathbb{R}^n$. Global mesh structure is usually hard to construct in this case. While our method only relies on the local mesh structure at each data point, which is constructed through local triangulation in the tangent space obtained by local principal component analysis (PCA). Once the local mesh is available, we propose numerical schemes to approximate differential operators and define mass matrix and stiffness matrix on point clouds, which can be used to solve partial differential equation (PDE) and variational problem on point clouds. As numerical examples, we use the local mesh method and variational formulation to solve the Laplace-Beltrami eigenproblem and solve the Eikonal equation to compute distance map and geodesics on point clouds.

1 Introduction

In many problems in science and engineering, data is commonly represented as a collection of points, referred as a point cloud, embedding in a certain space. In practice, a common feature for many point cloud data is that those data points sit on a low dimensional manifold $\mathcal{M}$ in an ambient space $\mathbb{R}^n$ possible with a large dimension. Analyzing, processing and characterizing point cloud become important tasks in many applications, such as computer visions, data mining and machine learning [1, 2, 3].

A typical example is 3D modeling in computer graphics and computer vision, where geometric objects, usually 2-dimensional surfaces, are represented as a set of 3D points obtained by a laser scanner. In practice, a triangulated surface or an implicit representation, such as level set function, is constructed to approximate the point cloud data. Based on these representations, partial differential equation (PDE) based methods and variational methods provide powerful tools to extract intrinsic geometric information either locally or globally for the underlying manifolds. Here, we propose a local mesh method that can be applied to solving PDE and variational problem on point clouds without requiring the construction of a global mesh or representation of the point cloud, which may be difficult and computationally expensive in three and higher dimensions.

*Department of mathematics, University of Southern California, CA, US (rongjiel@usc.edu).
†Department of mathematics, University of California, Irvine, CA, US (jianl@uci.edu).
‡Department of mathematics, University of California, Irvine, CA, US (zhao@math.uci.edu).

To solve variational problems or differential equations on point clouds, we first need to approximate integrals and differential operators. Previously, Belkin et al. [4] proposed to approximate the Laplace-Beltrami operator on point clouds through the heat diffusion in the ambient Euclidean space, and then the Laplace-Beltrami eigenproblem (Helmholtz equation) on point clouds is discussed. Construction of a local mesh in the tangent space is needed to approximate integration of the heat kernel. Their method suffers from the low order approximation and is only limited to the approximation of the Laplace-Beltrami operator on point clouds. After that, Luo et al. [5] proposed to approximate integrals on point clouds through computing summation with voronoi weight or principle eigenvector weight. More recently, Liang et al. [6, 7] proposed a more systematic way to approximate differential operators on point clouds. In their method, discrete approximation of differential operators are constructed by local least square approximations of the manifold and hence the metric using K nearest neighbors (KNN). Since the differential operators are intrinsically approximated from more accurate local manifold reconstruction, their method can achieve high order accuracy and enjoy more flexibility since no mesh is needed at all. It can be applied to manifolds with arbitrary dimensions and co-dimensions with or without boundary. Moreover, the computational complexity depends on the true dimension of the manifold rather than the dimension of the embedded space. However, their method can not deal with integrals and variational problems easily due to the lack of meshes.

In this paper, we propose an approach that requires only local mesh in the tangent space and can approximate both differential operators and integrals on point clouds. Hence, this method applies to cases where the tangent space can be locally triangulated. Our method is based on several simple observations as follows. ①. Local mesh construction is much easier than global mesh construction for general point clouds; ②. Definition of differential operators at any point only depends on the local structure of the point cloud; ③. Integration can be represented as an inner product weighted by the mass matrix, which can be obtained from local information of point clouds by choosing a locally supported basis. To clearly represent our approach in this paper, we only restrict our discussion on point clouds sampled from two dimensional manifolds in an ambient space $\mathbb{R}^n$, while all techniques we discussed here can be easily adapted to problems on point clouds sampled from $k-$dimensional manifolds in $\mathbb{R}^n$. We first use local PCA to estimate the tangent space at each point, then the local mesh can be obtained from the Delaunay triangulation in the tangent space. After the local mesh is obtained, differential operators can be approximated. With the constructed local connectivity, we propose a simple numerical approximation of the mass matrix and stiffness matrix similar to those using finite elements on triangulated surfaces. Then integrals, such as area and volume, and variational problem can be approximated numerically. Based on the variation formulation and our local mesh method, we solve the Laplace-Beltrami eigen-problem. Moreover, we also numerically solve the Eikonal equation to compute distance map and geodesics on point clouds based on either the fast sweeping or the fast marching method.

The rest of the paper is organized as follows. In section 2, we introduce the idea of local mesh construction for point clouds. Then differential operators and integrals can be approximate for

given point clouds based on the constructed local mesh. Selected applications to the local mesh method are discussed section 3. We propose an analogue of finite element method to solve Laplace-Beltrami eigenproblems on point clouds. Moreover, we discuss numerical approximation of distance maps and geodesics on point clouds by solving the Eikonal equation based on the fast marching or the fast sweeping methods. After that, section 4 reports all numerical experiments and comparisons to demonstrate the proposed methods. Conclusions are made in section 5.

2 Local Mesh method

Given a point cloud $\mathcal{P} = \{p_i \in \mathbb{R}^n | i = 1, \ldots, N\}$ sampled from a smooth surface/manifold $\mathcal{M}$, it is a challenge to extract its global information without a global mesh or parametrization. For instance, a elementary geometric quantity such as the area of $\mathcal{M}$ is not easy to obtain. One way is to first approximate the manifold $\mathcal{M}$ by constructing a global triangulation or implicit representation. Then, many differential geometry, PDE and variational tools can be applied to $\mathcal{M}$. However, surface reconstruction for point clouds may be difficult and time consuming, especially for a large amount of data in three or higher dimensions with complicated geometry and topology plus possible noise and nonuniform sampling. On the other hand, local geometry and connectivity at each point can be easily obtained. This motivates our method for approximating differential operator, PDEs and variational formulation on point clouds based on local mesh. Moreover, one can obtain useful global information for point clouds by solving PDE problems on point cloud [6].

2.1 Local connectivity construction for point clouds

Let's denote the indices set of K-nearest neighborhood (KNN) of each point $p_i \in \mathcal{P}$ by $\mathcal{I}(i)$. Local PCA on KNN is well-known and widely used for local linear approximation for point clouds [8]. One can determine the local tangent space and normal space using local PCA. For convenience, we we translate the K-nearest neighborhood $\mathcal{N}(i) = \{p_k, k \in \mathcal{I}(i)\}$ to center p_i at the origin. We can estimate local linear structure near p_i using the covariance matrix P_i of $N(i)$, defined by:

$$P_i = \sum_{k \in N(i)} (p_k - c_i)^T (p_k - c_i) \qquad (1)$$

Here, c_i is the local barycenter $c_i = \frac{1}{K} \sum_{k \in N(i)} p_k$. The eigenvectors (e_i^1, e_i^2, e_i^3) of P_i form an orthogonal frame associated with eigenvalues $(\lambda_i^1, \lambda_i^2, \lambda_i^3)$ with $\lambda_i^1 \geq \lambda_i^2 \geq \lambda_i^3 \geq 0$. If the point cloud is sampled from a two dimensional manifold, and the local sampling is dense enough to resolve local feature size, then

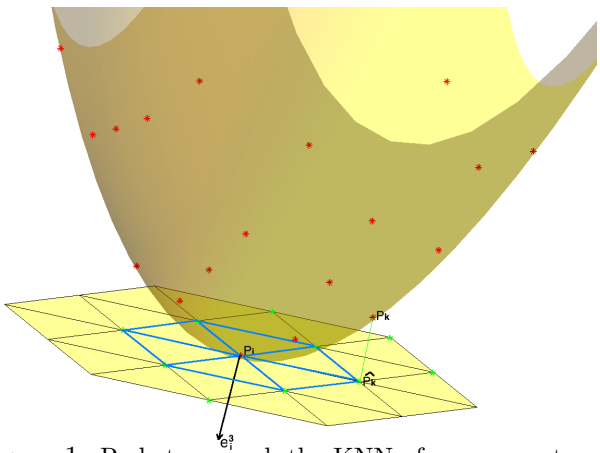


Figure 1: Red stars mark the KNN of p_i, green stars mark the projection of red stars on the tangent plane at p_i, and blue triangles color-code the connectivity of the first ring of p_i.

$e_i^1 \geq e_i^2 \gg e_i^3$. $e_i^1 \geq e_i^2$ provide two orthogonal directions in the tangent plane, and e_i^3 is the normal direction of $\mathcal{P}$ at the point p_i.

To construct the local connectivity and a mesh at the point p_i, we project its K-nearest neighborhood $\mathcal{N}(i)$ on the tangent plane $\mathcal{T}_i$ of p_i spanned by e_i^1, e_i^2. Namely, we have the following construction:

$$\hat{p}_k = Proj_{\mathcal{T}_i}(p_k) = p_k - p_i - \langle p_k - p_i, e_i^3 \rangle e_i^3, \quad k \in \mathcal{I}(i) \tag{2}$$

With this projection, all points in $\{\hat{p}_k, k \in I(i)\}$ belong to the tangent plane $\mathcal{T}_i$. Then, the local mesh structure near p_i can be obtained by the standard Delaunay triangulation. Denote all triangles of the first ring of p_i by $\mathcal{R}(i) = \{T_i^1, \cdots, T_i^{l_i}\}$ and all vertices in the first ring of p_i by $\mathcal{V}(i)$. The local connectivity of p_i is provided by $\mathcal{C}_i = \{p_i; \mathcal{V}(i), \mathcal{R}(i)\}$. (See Figure. 2.1)

2.2 Numerical approximation of differential operators on point clouds

Since differential operators, such as gradient, divergence and Laplacian, are defined locally, we can obtain point-wise numerical approximation of these differential operators using the local mesh constructed in section 2.1. Here, we directly borrow the idea of approximating differential operators on triangle meshes [9, 10, 11, 12, 13] on the first ring of each point of the point cloud $\mathcal{P}$.

To make the paper self-contained, we write down details of the numerical approximation of gradient, divergence and Laplace operators on the local mesh at a point p_i. First, the discretization is defined on on a single triangle by their coordinate invariant definition in differential geometry [14, 15]. Then an average in the first ring is taken weighted by the area of each triangle.

First of all, we show the discretization on a single triangle T with three vertices $\{p_i \in \mathbb{R}^n \mid i = 0, 1, 2\}$. In the discrete case, we have a function $f = \{f(p_0), f(p_1), f(p_2)\}$ and a vector field $\overrightarrow{V} = \{\overrightarrow{V}(p_0), \overrightarrow{V}(p_1), \overrightarrow{V}(p_2)\}$ defined on each vertex respectively. With the barycentric coordinates $\{(u^1, u^2, 1 - u^1 - u^2) \mid 0 \leqslant u^1, u^2, u^1 + u^2 \leqslant 1\}$ of T, any point $p \in T$, the linear interpolation of $f, \overrightarrow{V}$ in T can be given by:

$$\begin{cases} p = u^1(p_1 - p_0) + u^2(p_2 - p_0) + p_0 \\ f(p) = u^1(f(p_1) - f(p_0)) + u^2(f(p_2) - f(p_0)) + f(p_0) \\ \overrightarrow{V}(p) = u^1(\overrightarrow{V}(p_1) - \overrightarrow{V}(p_0)) + u^2(\overrightarrow{V}(p_2) - \overrightarrow{V}(p_0)) + \overrightarrow{V}(p_0) \end{cases} \tag{3}$$

Then we have $\partial_{u^1} = p_1 - p_0, \partial_{u^2} = p_2 - p_0$, and the metric matrix of T would be:

$$g = (g_{i,j})_{i,j=1,2} = \begin{pmatrix} \partial_{u^1} \cdot \partial_{u^1} & \partial_{u^1} \cdot \partial_{u^2} \\ \partial_{u^2} \cdot \partial_{u^1} & \partial_{u^2} \cdot \partial_{u^2} \end{pmatrix} \text{ and } (g^{i,j})_{i,j=1,2} = g^{-1} \tag{4}$$

where $\cdot$ is the dot product in $\mathbb{R}^n$. We then have the following discretization:

$$\nabla_T f(p_0) = \sum_{i,j=1}^{2} g^{ij} \frac{\partial f}{\partial x^j} \partial_{u^i} = (f(p_1) - f(p_0), f(p_2) - f(p_0)) g^{-1} \begin{pmatrix} \partial_{u^1} \\ \partial_{u^2} \end{pmatrix}$$

$$= (f(p_1) - f(p_0), f(p_2) - f(p_0)) \begin{pmatrix} \partial_{u^1} \cdot \partial_{u^1} & \partial_{u^1} \cdot \partial_{u^2} \\ \partial_{u^2} \cdot \partial_{u^1} & \partial_{u^2} \cdot \partial_{u^2} \end{pmatrix}^{-1} \begin{pmatrix} p_1 - p_0 \\ p_2 - p_0 \end{pmatrix} \tag{5}$$

We next discretize the divergence operator on the triangle T. Suppose $\overrightarrow{V} = v_1 \partial_{u^1} + v_2 \partial_{u^2}$ is a vector field on T. Then the coefficients v^1, v^2 are given by:

$$\begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = g^{-1} \begin{pmatrix} \overrightarrow{V} \cdot \partial_{u^1} \\ \overrightarrow{V} \cdot \partial_{u^2} \end{pmatrix} \tag{6}$$

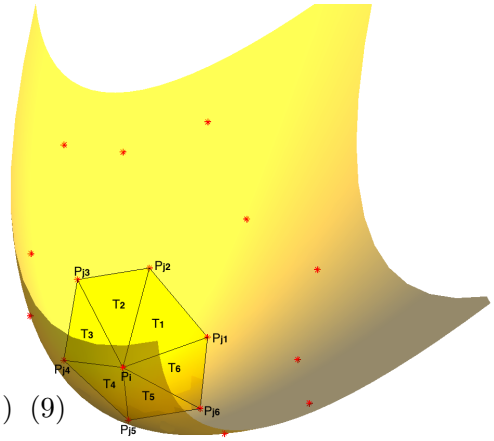
Differentiate both sides of above equality, we have:

$$\begin{pmatrix} \frac{\partial}{\partial u^1} v_1 \\ \frac{\partial}{\partial u^2} v_2 \end{pmatrix} = \begin{pmatrix} g^{11}(\overrightarrow{V}(p_1) - \overrightarrow{V}(p_0)) \cdot \partial_{u^1} + g^{12}(\overrightarrow{V}(p_1) - \overrightarrow{V}(p_0)) \cdot \partial_{u^2} \\ g^{21}(\overrightarrow{V}(p_2) - \overrightarrow{V}(p_0)) \cdot \partial_{u^1} + g^{22}(\overrightarrow{V}(p_2) - \overrightarrow{V}(p_0)) \cdot \partial_{u^2} \end{pmatrix} \tag{7}$$

Since $\sqrt{G}$ is constant on each triangle, we can obtain the discretization of the divergence operator on triangle T as follows:

$$\mathrm{div}_T \overrightarrow{V}(p_0) = \frac{1}{\sqrt{G}} \sum_{i=1}^{2} \frac{\partial}{\partial u^i} (\sqrt{G} v_i) = \frac{\partial}{\partial u^1}(v_1) + \frac{\partial}{\partial u^2}(v_2) \tag{8}$$

Now, we consider the discretization of the gradient and divergence operators on the local mesh at a point. We take a weighted average on the first ring of p_i in terms of triangle areas. Namely, for any function f and vector field $\overrightarrow{V}$ defined on the given point cloud $\mathcal{P}$, the gradient and divergence operators at each point are approximated as follows:



$$\nabla_{\mathcal{P}} f(p_i) = \frac{1}{\sum_{T \in \mathcal{R}(i)} Area(T)} \sum_{T \in \mathcal{R}(i)} Area(T) \nabla_T f(p_i) \tag{9}$$

$$\mathrm{div}_{\mathcal{P}} \overrightarrow{V}(p_i) = \frac{1}{\sum_{T \in \mathcal{R}(i)} Area(T)} \sum_{T \in \mathcal{R}(i)} Area(T) \mathrm{div}_T \overrightarrow{V}(p_i) \tag{10}$$

Moreover, we can also approximate the Laplace-Beltrami operator as follows:

$$\Delta_{\mathcal{P}} f(p_i) \approx \sum_{j \in \mathcal{V}(i)} \omega_{ij}(p_i) \left(f(p_j) - f(p_i) \right), \tag{11}$$

where $\omega_{ij}(p_i) = \dfrac{cot\alpha_{ij}^1(p_i) + cot\alpha_{ij}^2(p_i)}{2}$, α_{ij}^1 and α_{ij}^2 are the two angles opposite to the edge $\overline{p_i p_j}$, $\mathcal{V}(i)$ is the first ring neighborhood of the vertex p_i.

Remark 1 *Here, we only consider approximate the differential operators using the first ring on local mesh obtained from section 2.1. More accurate approximation can be obtained using the second ring structure.*

2.3 Integral approximation

Integration on point clouds, such approximation of surface area, enclosed volume, and variational formula, is an important task in many applications. One way is to reconstruct the surface first, which can be difficult and expensive itself. More direct methods have been proposed before. Li et al. [16] propose to approximate areas of surfaces in $\mathbb{R}^3$ by estimating the intersections of lines in $\mathbb{R}^3$ based on Cauchy-Crofton formula. More recently, Lui et al. [5] propose to estimate area of point clouds in $\mathbb{R}^n$ using voronoi cells of local mesh reconstruction from point clouds. Here we propose a method for integral approximation using local mesh.

First we construct the mass matrix on a point cloud $\mathcal{P}$ through which we can approximate integration on $\mathcal{P}$ as follows. Given a point cloud $\mathcal{P} = \{p_i \in \mathbb{R}^n \mid i = 1, \cdots N\}$ with local connectivity $\mathcal{C} = \{\mathcal{C}_i = \{p_i; \mathcal{V}(i), \mathcal{R}(i)\} \mid i = 1, \cdots, N\}$ constructed in section 2.1, we define the set of nodal basis $\{e_j\}_{j=1}^N$ on $\mathcal{P}$ as:

$$e_j : \mathcal{P} \to \mathbb{R}, \quad e_j(p_i) = \begin{cases} 1, & if \quad i = j \\ 0, & otherwise \end{cases} \qquad i, j = 1, \cdots, N \tag{12}$$

Then, any function $f = (f(p_1), \cdots, f(p_N))^T$ defined on $\mathcal{P}$ can be written as $f = \sum_{i=1}^N f(p_i) e_i$. Given a point $p_i \in \mathcal{P}$ with the first ring structure $\mathcal{R}(i) = \{T_i^1, \cdots, T_i^{l_i}\}$ in $\mathcal{C}_i$, we define:

$$\langle e_i, e_j \rangle = \begin{cases} \dfrac{1}{6} \displaystyle\sum_{k=1}^{l_i} Area(T_i^k), & if \quad i = j \\ \dfrac{1}{12}[Area(T_i^{i_1}) + Area(T_i^{i_2})], & if \quad p_i p_j \text{ is the adjacent edge of } T_i^{i_1} \text{ and } T_i^{i_2} \\ 0, & otherwise \end{cases} \tag{13}$$

Consider the linear interpolation of e_i on its first ring structure, we define the inner product of e_i, e_j on $\mathcal{P}$:

$$\begin{aligned} \int_{\mathcal{P}} e_i e_j &= \frac{1}{2}[\int_{\mathcal{R}(i)} e_i e_j + \int_{\mathcal{R}(j)} e_i e_j] \\ &= \frac{1}{2}[\sum_{k=1}^{l_i} \int_{T_i^k} e_i e_j + \sum_{k=1}^{l_j} \int_{T_j^k} e_i e_j] \\ &= \frac{1}{2}(\langle e_i, e_j \rangle + \langle e_j, e_i \rangle) \end{aligned} \tag{14}$$

We define the $N \times N$ mass matrix $\mathbb{M} = (m_{ij})$ for the point cloud $\mathcal{P}$ by:

$$m_{ij} = \int_{\mathcal{P}} e_i e_j = \frac{1}{2}(\langle e_i, e_j \rangle + \langle e_j, e_i \rangle) \tag{15}$$

Here $\mathbb{M}$ is a sparse and symmetric matrix. Let's denote the constant one function on $\mathcal{P}$ by $\mathbf{1} = (1, \cdots, 1)^T$. For any two functions $f = (f(p_1), \cdots, f(p_N))^T$ and $g = (g(p_1), \cdots, g(p_N))^T$, we can approximate the following integrations on the point cloud $\mathcal{P}$ by:

$$\int_{\mathcal{P}} f = f^T \mathbb{M} \mathbf{1}, \qquad \int_{\mathcal{P}} fg = f^T \mathbb{M} g. \tag{16}$$

For example, we can approximate the area of the underlying surface of $\mathcal{P}$ by:

$$Area(\mathcal{P}) = \int_{\mathcal{P}} \mathbf{1} = \mathbf{1}^T \mathbb{M} \mathbf{1} = \sum_{i,j} m_{ij} \tag{17}$$

If the underlying surface of the given point cloud $\mathcal{P}$ is a closed surface $\mathcal{M}$ in $\mathbb{R}^3$, we can also estimate the 3D volume of the manifold $\mathcal{D}$ bounded by $\mathcal{M}$ using Stokes' theorem. Assume each $p_i = (p_i^1, \cdots, p_i^3)$ is represented as a point in $\mathbb{R}^3$. We define three global coordinates functions of $\mathcal{P}$ by:

$$x^\alpha : \mathcal{P} \to \mathbb{R}, \quad x^\alpha(p_i) = p_i^\alpha, \qquad \alpha = 1, 2, 3. \tag{18}$$

The Stokes' theorem tells us:

$$Volume(\mathcal{D}) = \int_{\mathcal{D}} \mathbf{1} = \int_{\mathcal{D}} \frac{\partial}{\partial x^\alpha}(x^\alpha) = \int_{\mathcal{M}} x^\alpha \nu^\alpha \approx \int_{\mathcal{P}} x^\alpha \nu^\alpha = (x^\alpha)^T \mathbb{M} \nu^\alpha, \qquad \alpha = 1, 2, 3. \tag{19}$$

where $\vec{\nu} = (\nu^1, \nu^2, \nu^3)$ is the outward normal vector of $\mathcal{M}$. Therefore, we approximate the volume of $\mathcal{D}$ as follows:

$$Volume(\mathcal{D}) = \frac{1}{3} \sum_{\alpha=1}^{3} (x^\alpha)^T \mathbb{M} \nu^\alpha = \frac{1}{3} \sum_{\alpha=1}^{3} \left(\sum_{i,j=1}^{N} p_i^\alpha m_{ij} \nu_j^\alpha \right) \tag{20}$$

3 Solving PDEs on point clouds

As long as differential operators and integrals can be approximated on the given point clouds based on the local mesh method, many problems related to differential equations on point cloud can be further studied. As applications, we propose an analogue of the finite element method on point cloud to solve a typical elliptic eigenvalue problem, Laplace-Beltrami eigenvalue problem. After that, we demonstrate how to use the proposed local mesh method to solve the Eikonal equation, a typical nonlinear partial differential equation, on the given point cloud by adapting the fast marching and fast sweeping methods [17, 18, 19, 20]. Using the resulting distance maps, we also discuss the construction of geodesics on point clouds.

3.1 Laplace-Beltrami eigenproblems

As an intrinsic differential operator defined on a surface, the Laplace-Beltrami (LB) operator can be used as a bridge to connect local and global geometry of the surface. Its eigenvalues and eigenfunctions can be used to study and characterize the surface. For example, generic LB eigenfunctions of

surfaces are morse functions [21], which enable LB eigenfunctions as descriptors for the topologies of the surfaces. Moreover, LB eigenfunctions can be viewed as either global or local descriptors to analyze surface geometric structures [22, 23, 24, 25, 26]. Furthermore, the LB operator is also closely related to harmonic maps between two surfaces. Recently a few numerical methods [4, 6, 7] have been proposed to compute LB eigenproblems on point clouds. These methods are all based on pointwise discretization of the LB operator. Here, we design a variational formulation using the local mesh method, which can be used to solve other type of elliptic PDEs on point clouds as well.

Let's consider a Riemannian surface $(\mathcal{M}, g)$ with boundary $\partial\mathcal{M}$ [1]. The LB operator $\Delta_{\mathcal{M}} = \text{div}_{\mathcal{M}} \circ \nabla_{\mathcal{M}}$ is self-adjoint and elliptic, so its spectrum is discrete. The LB eigen-system $\{\lambda_k, \phi_k\}_{k=0}^{\infty}$ of $(\mathcal{M}, g)$ is composed of eigenvalues of $\Delta_{\mathcal{M}}$, $0 = \lambda_0 < \lambda_1 \leq \lambda_2 \leq \cdots$, and the corresponding eigenfunctions $\phi_0, \phi_1, \phi_2, \cdots$ satisfying:

$$\begin{cases} \Delta_{\mathcal{M}}\phi_k = -\lambda_k\phi_k, \\ \nabla_{\mathcal{M}}\phi_k \cdot \vec{n}|_{\partial\mathcal{M}} = 0, \end{cases} \iff \int_{\mathcal{M}} \nabla_{\mathcal{M}}\phi_k \nabla\eta = \lambda_k \int_{\mathcal{M}} \phi_k\eta, \quad \& \quad \phi_k \perp \{\phi_0, \cdots, \phi_{k-1}\}. \tag{21}$$

where $\vec{n}$ is the normal vector of $\partial\mathcal{M}$ in $\mathcal{M}$. The LB eigen-system plays an essential role to undertand intrinsic surface geometry [27, 28]. Computationally, the variational formulation has been well studied based on triangulated surfaces [12, 29] and implicit representations [30, 31] and successfully used in many fields such as computer science [32, 33, 34, 35, 36, 22, 37, 23] and medical image analysis [38, 39, 40, 41, 42, 25, 43, 24, 26, 44].

Given a point cloud $\mathcal{P} = \{p_i \in \mathbb{R}^n \mid i = 1, \cdots N\}$ sampled from a Riemannian surface $\mathcal{M}$ embedded in $\mathbb{R}^n$, a discrete analogue of LB eigen-system on $\mathcal{P}$ can also be studied. Based on our local mesh method, we can use the numerical approximation of the LB operator on $\mathcal{P}$ given by (11) and compute the spectrum of the corresponding matrix. We can also compute LB eigensystem on point clouds based on the variational formulation in (21), as a demonstration of an analogue of finite element method (FEM), on the point cloud $\mathcal{P}$ using its local connectivity $\mathcal{C} = \{\mathcal{C}_i = \{p_i; \mathcal{V}(i), \mathcal{R}(i)\} \mid i = 1, \cdots, N\}$. Numerical comparisons of our method with previous methods will be conducted in section 4.2.

Consider $\{e_i\}_{i=1}^N$ as the set of linear elements defined on $\mathcal{P}$ given in (12), and write $E = Span_{\mathbb{R}}\{e_i\}_{i=1}^N$. Then the discrete version of LB eigensystem on $\mathcal{P}$ can be defined by the following weak formulation:

$$\int_{\mathcal{P}} \nabla_{\mathcal{P}}\phi \ \nabla_{\mathcal{P}}\eta = \lambda \int_{\mathcal{P}} \phi \ \eta, \quad \forall \ \eta \in E. \tag{22}$$

In fact, if we can write $\phi = \sum_i^N v_i e_i$, the essential part of the above problem is a numerical approximation of $\int_{\mathcal{P}} \nabla_{\mathcal{P}} e_i \cdot \nabla_{\mathcal{P}} e_j$, which forms the stiffness matrix $\mathbb{S} = (S_{ij})_{N\times N}$ for the given point cloud $\mathcal{P}$. Ideally, we would like $\mathbb{S}$ to have properties, such as symmetric and nonnegative definite, same as those for the stiffness matrix for a triangulated surface [29]. However, these are global properties which may not be possible to achieve due to the use of local mesh only. To be more

[1] If $\mathcal{M}$ is a closed surface, then $\partial\mathcal{M} = \emptyset$

specific, the first ring structure of p_i is not necessary compatible with the first ring structure of p_j although p_j belongs to the first ring of p_i. To have a numerical approximation of the stiffness matrix, we first define

$$A_{ij} = \sum_{T \in \mathcal{R}(i)} \int_T \nabla_T e_i \cdot \nabla_T e_j = -\frac{1}{2} \left[\cot \alpha_{ij}^1(p_i) + \cot \alpha_{ij}^2(p_i) \right], \quad i \neq j \tag{23}$$

where $\nabla_T e_i$ and $\nabla_T e_j$ are computed by (5) and $\alpha_{ij}^k(p_i), k = 1, 2$ are the angles opposite to the edge connecting points p_i and p_j in the first ring of p_i. This is the approximation of $\int_{\mathcal{P}} \nabla_{\mathcal{P}} e_i \cdot \nabla_{\mathcal{P}} e_j$ based on the first ring structure at p_i. Note that A_{ij} may not be equal to A_{ji} due to the possible incompatibility of the first ring structures of p_i and p_j. One simple symmetrized definition of the stiffness matrix is the following:

$$S_{ij} = \begin{cases} \frac{1}{2} \left(A_{ij} + A_{ji} \right), & \text{if} \quad i \neq j \\ -\sum_{k \neq i} S_{ik}, & \text{if} \quad i = j \end{cases} \tag{24}$$

The above definition of the diagonal element is to enforce the consistence condition, i.e., constant function is an eigenfunction with zero eigenvalue. In particular, if all triangles in the first ring structure are acute, off-diagonal elements are non-positive and diagonal elements $S_{ii} = |\sum_{k \neq i} S_{ik}|$ are positive. Hence all eigenvalues are real and non-negative. When the distribution of points is reasonably uniform, this definition of stiffness matrix works quite well. However, when the points are distributed non-uniformly, the first ring structure of p_i is more likely incompatible with the first ring structure of p_j. Numerical tests suggest that the following definition for off-diagonals elements produces better results (see Figure. 5):

$$S_{ij} = \begin{cases} \max(A_{ij}, A_{ji}) & \text{if} \quad A_{ij} \leq 0 \quad \text{and} \quad A_{ji} \leq 0 \\ \min(A_{ij}, A_{ji}) & \text{if} \quad A_{ij} \geq 0 \quad \text{and} \quad A_{ji} \geq 0 \\ \min(A_{ij}, A_{ji}) & \text{if} \quad A_{ij} \cdot A_{ji} < 0 \\ -\sum_{k \neq i} S_{ik} & \text{if} \quad i = j \end{cases} \tag{25}$$

The intuition for this definition of off-diagonals is that (a) no direct coupling between p_i and p_j unless they are both in each other's first ring, (b) use better triangles in the first ring structures at p_i and p_j to approximate $\int_{\mathcal{P}} \nabla_{\mathcal{P}} e_i \cdot \nabla_{\mathcal{P}} e_j$ when they are both in each other's first rings. The first case in (25) means we use triangles with acute angles α_{ij}^k or α_{ji}^k further away from 0 degree. The second case in (25) means we use triangles with obtuse angles α_{ij}^k or α_{ji}^k further away from 180 degree. The third case in (25) means we use triangles with acute angles α_{ij}^k or α_{ji}^k rather than those with obtuse angles.

Then the solutions to (22) can be computed from the following generalized matrix eigen-problem:

$$\begin{cases} \mathbb{S} v = \lambda \mathbb{M} v, \text{ where } \ v = (v_1, \cdots, v_N)^T \\ \phi = \sum_i^N v_i e_i \end{cases} \tag{26}$$

Note that the stiffness matrix $\mathbb{S}$ and the mass matrix $\mathbb{M}$ are symmetric and sparse matrices of size $N \times N$. Thus, all eigenvalues and eigenfunctions computed from the above method are real. Computationally, there are a variety of softwares, such as Matlab, to solve the above problem (26). We would like to point out that this approach is not necessary restricted to solve LB eigensystem. Similar approach can be used to solve other types of PDE, such as the poisson equation, the heat equation etc., on point clouds.

3.2 Distance maps and geodesics on point clouds

Once local connectivity is available on a point cloud, one can use appropriate local solver on triangular mesh to solve many types of PDEs on the point cloud. Here we specifically demonstrate how to solve a special type of nonlinear hyperbolic PDE, the Eikonal equation (27), to compute distance maps and geodesics on point clouds using local mesh method. Here is the Eikonal equation for the distance map to a given set Γ on a manifold $\mathcal{M}$:

$$\begin{cases} |\nabla_{\mathcal{M}} d(x)| = 1 \\ d(x) = 0, \quad x \in \Gamma \subset \mathcal{M} \end{cases} \tag{27}$$

In particularly, a distance map d_p to a given point p can be computed by setting $\Gamma = p$. Once the distance map d_p is obtained, one can construct the geodesic from any point $q \in \mathcal{M}$ to p by solving the ODE:

$$\begin{cases} \dfrac{dX(s)}{ds} = -\nabla_{\mathcal{M}} d_p(s) \\ X(0) = q \end{cases} \tag{28}$$

where $X(s)$ traces out the geodesic path between two points $p, q \in \mathcal{M}$ and $d_p(\cdot)$ is the distance function with boundary condition $d_p(p) = 0$.

To solve the Eikonal equation (27) on a point cloud $\mathcal{P}$, the first step is to design an appropriate discretization at each point of $\mathcal{P}$. Since we have a local mesh, i.e., the first ring structure, we can directly adopt the monotone upwind discretization for triangular mesh [18, 20], which is equivalent to an approximation of the dynamic programming principle on the first ring. As the result of the discretization, at each point $p_i \in \mathcal{P}$ we have a nonlinear equation that couples the value at p_i with the values of its first ring neighbors. Hence we have to solve a system of nonlinear equations numerically. Fortunately, we have efficient ways to solve it either using the fast marching method [17] or the fast sweeping method [19] which are also generalized to triangular mesh in [18] and [20] respectively, where only first ring structures of the given triangular mesh are needed. Thus, the proposed local mesh method can be directly adapted to either fast marching method or fast sweeping method to solve the Eikonal equation on point clouds. This is a more direct and more accurate approach to obtain distance maps than the previous method proposed in [45], where the Eikonal equation is solved in the tubular neighborhood of the point cloud $\mathcal{P}$ in $\mathbb{R}^n$ to approximate the true Eikonal equation defined on $\mathcal{P}$.

Once we have computed the distance map $d_p(x)$ on the point cloud $\mathcal{P}$, we can find the shortest

path from p to q on $\mathcal{P}$ by solving (28). To have more accurate approximation of geodesics and avoid zigzag paths, we can not require the nodes of discretized geodesic path strictly passing through points of $\mathcal{P}$. The question is how to add new points in the way that the geodesic can truly reflect the geometry of underlying manifold represented by the point cloud. Here, we propose to use the local interpolation method introduced in [6, 7] to add necessary points for tracing geodesics.

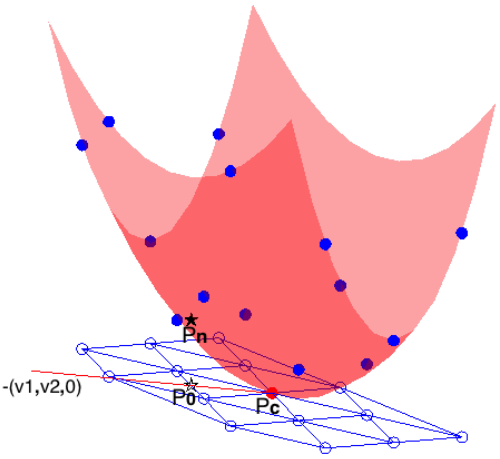


Figure 2: Geodesic path tracing. The current point (p_c), its KNN points and the new point (p_n) are marked as the red solid circle, the blue solid circles and the black solid star respectively, whose projections on the tangle plane at p_c are marked as corresponding hollow markers.The red line has direction $(-v_1, -v_2, 0)$.

Suppose a point p_c (current point on the geodesic) has already been obtained, we intend to find the next point on the geodesic path. Notice that p_c may not be in the point cloud $\mathcal{P}$. Without loss of generality, suppose $p_1, p_2, \cdots, p_K \in \mathcal{P}$ are KNN of p_c in the point cloud $\mathcal{P}$. Using PCA, we can build a local coordinate system centered at p_c, $\langle p_c; e_1, e_2, e_3 \rangle$ and the KNN of p_c has local coordinates (x_i, y_i, z_i). We use moving least squares (MLS) to locally approximate both the surface as $\Gamma = (x, y, z(x, y))$ and the distance function d_p as $d_p = d_p(x, y)$, then we can use the definition of the gradient operator to write $\nabla_{\mathcal{P}} d(p_c)$ as $\vec{v}$ (more details can be found in [6] and [7]). We construct the Delaunay triangulation of the projections $\hat{p_c}, \hat{p_1}, \hat{p_2}, \cdots, \hat{p_K}$ and find the first ring $\mathcal{R} = \{T_c^1, \cdots, T_c^l\}$ of p_c, which is the same as we did in Section 2.1. Suppose $\vec{v}$ has local coordinate (v_1, v_2, v_3) in $\langle p_c; e_1, e_2, e_3 \rangle$. We find the intersection of line segment starting at p_c with the direction $(-v_1, -v_2, 0)$ and the first ring, notice that this computation is done within the tangent space of p_c. Denote the intersection as $p_0 = (x_0, y_0, 0)$, we then project it back to the approximated surface to obtain the next point on the geodesic path $p_n = (x_0, y_0, z(x_0, y_0))$. This process is illustrated in Figure 2.

We start from the target point and set it as the current point on the geodesic path. Using the algorithm introduced above, we trace back the distance field to get the next point on the geodesic path (notice that it may not belong to the point cloud $\mathcal{P}$) and set it to be the current one. We repeat the process until the current point is close enough to the source point. Numerical examples will be shown in Section 4.

4 Numerical Results

In this section, numerical experiments are presented to illustrate applications of the proposed local mesh method to point clouds. First, we report numerical results of the area and volume approximation for point clouds sampled from different surfaces. Second, we demonstrate the proposed method in section 3.1 to solve the Laplace-Beltrami eigenproblem for point cloud data. In addi-

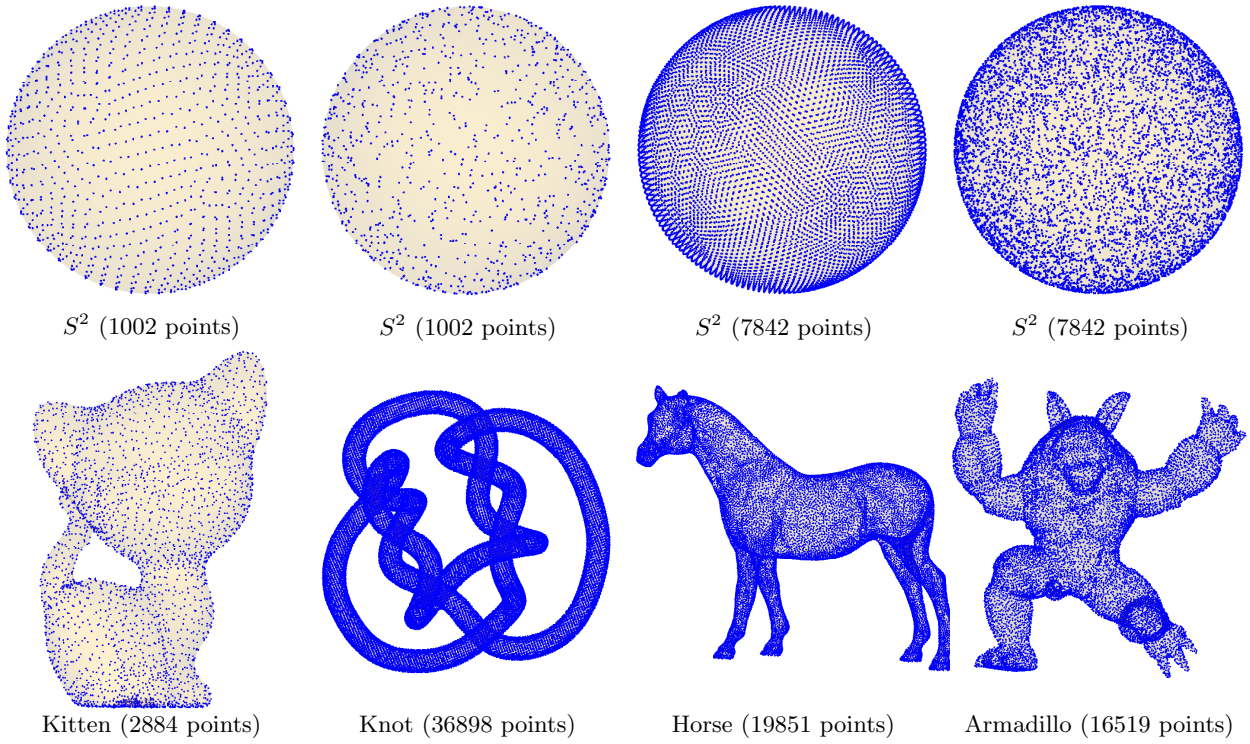


Figure 3: Point cloud data set

tion, numerical comparison with existing methods are also conducted. Finally, we show numerical results of distance maps and geodesic paths for point clouds. All our experiments are implemented by MATLAB in a PC with a 4G RAM and a 2.66 GHz CPU.

4.1 Area and Volume approximation

In general accuracy of the area and volume approximation depends on the size and distribution of the point cloud sampled from the given closed surface. Larger size and better distribution sampling lead to better numerical approximation. To demonstrate this point, we first test our method on different point clouds sampled from the unit sphere S^2 in $\mathbb{R}^3$. We choose two groups of point clouds on S^2. One group of point clouds are uniformly sampled from S^2 with the number of points about geometrically increasing from 1k to 16k, and another group of point clouds are non-uniformly sampled with the same number of points as the first group. The first row of Figure 3 plots two uniformly sampled and two non-uniformly sampled point clouds used in this experiment. As a comparison, we also compute the area approximation using the voronoi weighting scheme introduced in [5]. The numerical results and the corresponding relative errors for this two groups of data are reported in Table. 4.1. According to our results, we can observe that our method provides highly accurate approximation of the area and volume for the unit sphere in the case of uniformly sampled data set, while the approximation results using nonuniformly sampled data are not as good as the ones obtained from uniformly sampled data set. In both cases, our results are more accurate than the results using the voronoi weighting scheme in [5]. In addition, we can also observe that

the relative errors halve as the number of point doubles for uniformly sampled points. However, this behavior can not be observed for nonuniformly sampled points. Similar phenomenons can be observed for point clouds sampled from a flat torus $T = \{(\cos u, \sin u, \cos v, \sin v) \in \mathbb{R}^4 \mid (u, v) \in [0, 2\pi)^2\}$ in $\mathbb{R}^4$.

Next apply our method to a few real data sets illustrated in the second row of Figure 3. Since there is no true area and volume for these surfaces, we use the results computed from their triangular mesh representation as the ground truth. The numerical approximation and the relative errors are reported in the last four rows of Table. 4.1, where we can see that our method works as well as using the triangulated surfaces.

surface	# of points	Area					Volume		
		Our method		Truth	Voronoi weighting scheme [5]		Our method		Truth
		Results	Error		Results	Error	Results	Error	
uniform sampling on S^2	1002	12.5278	0.307 %	$4\pi \approx$ 12.5664	12.5115	0.437 %	4.1605	0.675 %	$\frac{4}{3}\pi \approx$ 4.1888
	1962	12.5466	0.157 %		12.5382	0.224 %	4.1743	0.345 %	
	4002	12.5567	0.077 %		12.5525	0.110 %	4.1817	0.169 %	
	7842	12.5614	0.039 %		12.5593	0.057 %	4.1852	0.086 %	
	16002	12.5640	0.019 %		12.5629	0.028 %	4.1870	0.042 %	
non-uniform sampling on S^2	1002	12.3065	1.638 %	$4\pi \approx$ 12.5664	12.1803	3.072 %	4.0866	2.439 %	$\frac{4}{3}\pi \approx$ 4.1888
	1962	12.4548	0.888 %		12.3123	2.022 %	4.1394	1.180 %	
	4002	12.4814	0.676 %		12.4025	1.304 %	4.1526	0.865 %	
	7842	12.4924	0.589 %		12.4370	1.030 %	4.1675	0.508 %	
	16002	12.5318	0.275 %		12.4937	0.578 %	4.1762	0.301 %	
Flat torus in $\mathbb{R}^4$	1024	39.3517	0.321 %	$4\pi^2 \approx$ 39.4784	38.9658	1.298 %	–	–	–
	2025	39.4143	0.162 %		39.2186	0.658 %	–	–	
	4096	39.4467	0.080 %		39.3498	0.326 %	–	–	
	8100	39.4624	0.041 %		39.4133	0.165 %	–	–	
	16129	39.4704	0.020 %		39.4457	0.083 %	–	–	
Kitten	2884	1.6996	1.6543 %	1.6720	1.6358	2.165 %	0.1234	1.042 %	0.1221
Knot	36898	24816.9033	0.811 %	24617.1913	24557.2129	0.244 %	54928.5678	2.844 %	53409.6952
Horse	19851	10499.4573	0.492 %	10551.3689	10325.4225	2.141 %	42278.6122	0.907 %	42692.7569
Armadillo	16519	37462.2439	2.873 %	36416.1313	34330.6387	5.727 %	235840.0965	1.361 %	239093.6180

Table 1: Area and volume approximation results. Here $Error = |Result - Truth|/Truth$

4.2 Laplace-Beltrami eigenproblems

To test the accuracy of the proposed method for solving LB eigenproblems of point clouds, we would like to first apply our method to point clouds data sampled from the unit sphere S^2. For the unit sphere, its exact value of the n-th LB eigenvalue is given by $\lambda_n = n(n+1)$, with multiplicity $2n+1$. Thus, we can compare the numerical results with these true values. In other words, we compute the relative error:

$$E_{\max,n} = \max_{i=1,\cdots,2n+1} \left\{ \frac{|\tilde{\lambda}_{n,i} - \lambda_n|}{\lambda_n} \right\}$$

where $\tilde{\lambda}_{n,i}$'s are the approximated LB eigenvalues to λ_n computed from the proposed method introduced in section 3.1 , and i runs over λ_n's multiplicity. By the definition, $E_{\max,n}$ represents the worst possible error in computing λ_n.

In our experiments, we apply the two definitions of the stiffness matrix in (24) and (25) to the same two groups of spherical point clouds used in section 4.1. As comparisons, we also compute the LB eigenproblem using other point cloud based methods, such as the MLS (using quadratic approximation) method [6], the PCD method [4], and a mesh based finite element method [11] for the same two groups of data. We show $E_{\max,n}$ for $\lambda = 20$ and 72 of the uniformly and non-uniformly sampled data in Figure 4 and in Figure 5, respectively. For the uniformly sampled data set, the numerical errors of our local mesh method using (24) and (25) are the same up to 10^{-10}, thus we only list one group of results using the local mesh method. In this case, it can be seen that the error of our approach is of the same order as the mesh based method, while they are slightly better that the results obtained by MLS approach and much better than PCD Laplacian method. The results for nonuniformly sampled point clouds are not as accurate as results obtained from the uniformly sampled data, while they are also comparable with the MLS approach and more accurate than PCD Laplacian method. In addition, it is also clear to see that the results using the stiffness matrix defined in (25) are more accurate than those using (24) for the non-uniform sampled data. For some of the non-uniform data examples, a global triangular mesh can not be constructed and hence mesh based finite element method does not work for those examples.

Size	1002	1962	4002	7842	16002
Local Mesh Method using (24) or (25) (LMM)					
$\lambda = 20$	0.0206	0.0105	0.0051	0.0026	0.0013
$\lambda = 72$	0.0721	0.0363	0.0176	0.0090	0.0044
Moving Least Square Laplacian (MLS)					
$\lambda = 20$	0.0516	0.0306	0.0135	0.0046	0.0020
$\lambda = 72$	0.1541	0.0774	0.0476	0.0298	0.0138
PCD Laplacian (PCD)					
$\lambda = 20$	0.0773	0.0487	0.0431	0.0411	0.0403
$\lambda = 72$	0.1391	0.1174	0.1128	0.1108	0.1100
Finite Element Method (FEM)					
$\lambda = 20$	0.0165	0.0085	0.0042	0.0021	0.0010
$\lambda = 72$	0.0660	0.0342	0.0169	0.0087	0.0043

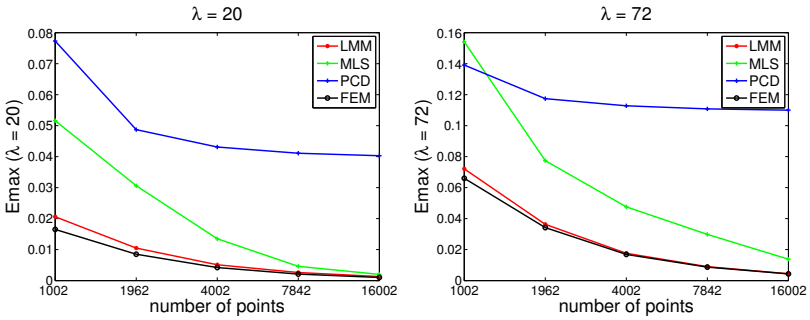


Figure 4: E_{max} errors for uniformly sampled data on unit sphere.

Size	1002	1962	4002	7842	16002
Local Mesh Method using (24) (LMM1)					
$\lambda = 20$	0.0734	0.0454	0.0425	0.0195	0.0329
$\lambda = 72$	0.2129	0.1159	0.0812	0.0377	0.0458
Local Mesh Method using (25) (LMM2)					
$\lambda = 20$	0.0633	0.0375	0.0251	0.0144	0.0092
$\lambda = 72$	0.2130	0.1088	0.0656	0.0317	0.0224
Moving Least Square Laplacian (MLS)					
$\lambda = 20$	0.0907	0.0465	0.0241	0.0079	0.0060
$\lambda = 72$	0.5898	0.3232	0.1277	0.0357	0.0272
PCD Laplacian (PCD)					
$\lambda = 20$	0.2585	0.1436	0.1112	0.0976	0.0947
$\lambda = 72$	0.3830	0.2846	0.2534	0.2395	0.2353

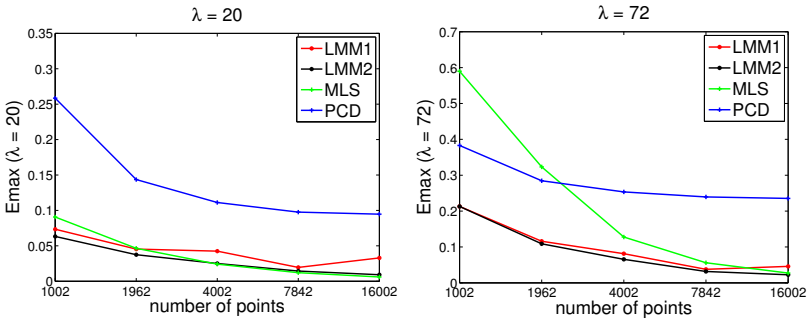


Figure 5: E_{max} errors for non-uniformly sampled data on unit sphere.

To demonstrate the robustness of our method for point clouds sampled from general surfaces,

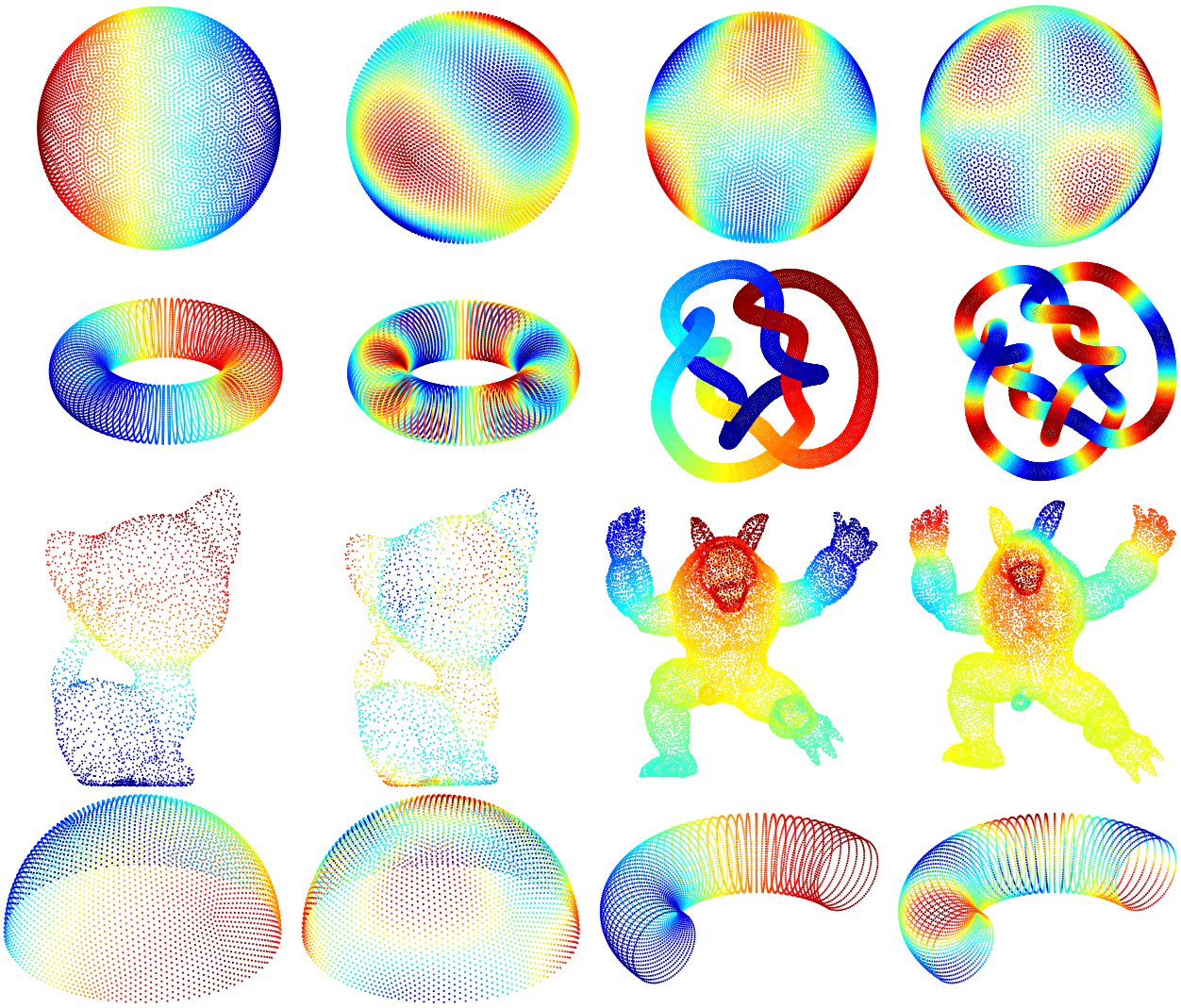


Figure 6: Laplace-Beltrami eigenfunctions are color-coded on point clouds sampled from open and closed surfaces.

either open or closed, we further show results of LB eigenfunctions for different point clouds in Figure 6. The first row in Figure 6 illustrates a few of LB eigenfunctions on the point cloud sampled from the unit sphere. It is clearly to see that the patterns of the computation results are the same as the spherical harmonics which further verifies our proposed method. LB eigenfunctions of point clouds sampled from surfaces with more complicated structures are illustrated in the second and the third rows of Figure 6. In addition, we also report a few results of LB eigenfunctions of point clouds sampled from open surfaces.

4.3 Distance maps and geodesics on point clouds

In our first numerical experiment, we choose a point cloud $\mathcal{P}$ sampled from a hemisphere shown in Figure 7. Distance functions to four different subsets of $\mathcal{P}$ (marked as red stars) are computed on the given point cloud. In the first row of Figure 7, distance functions to the four different given sets

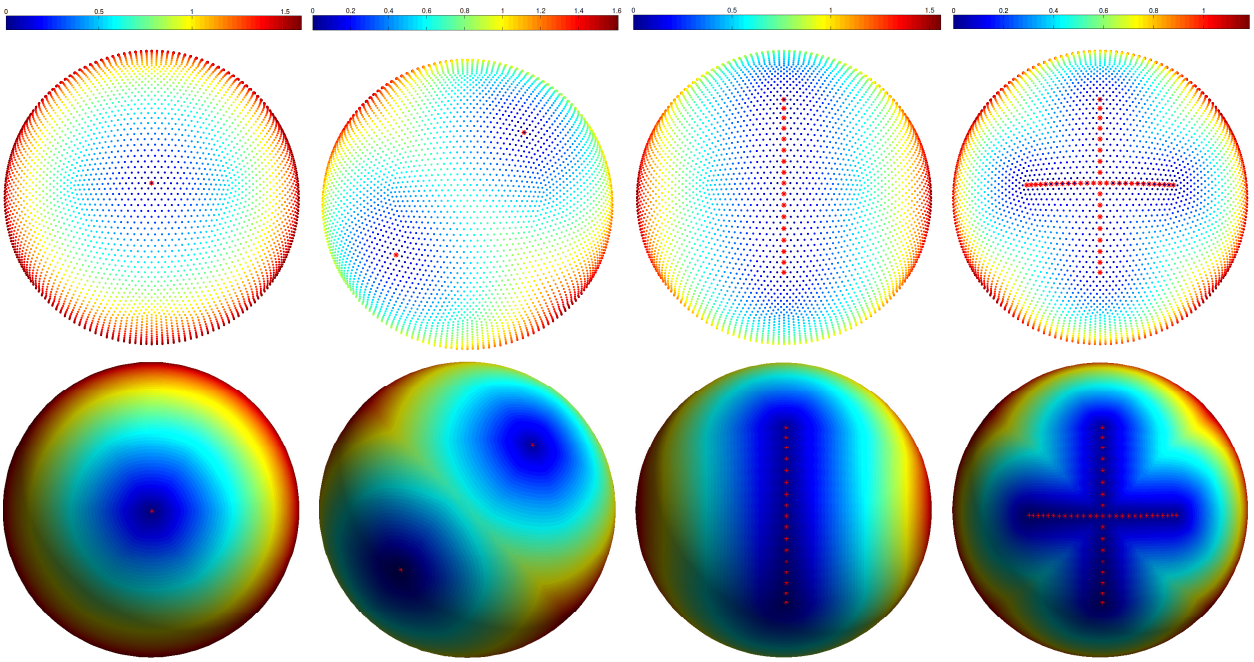


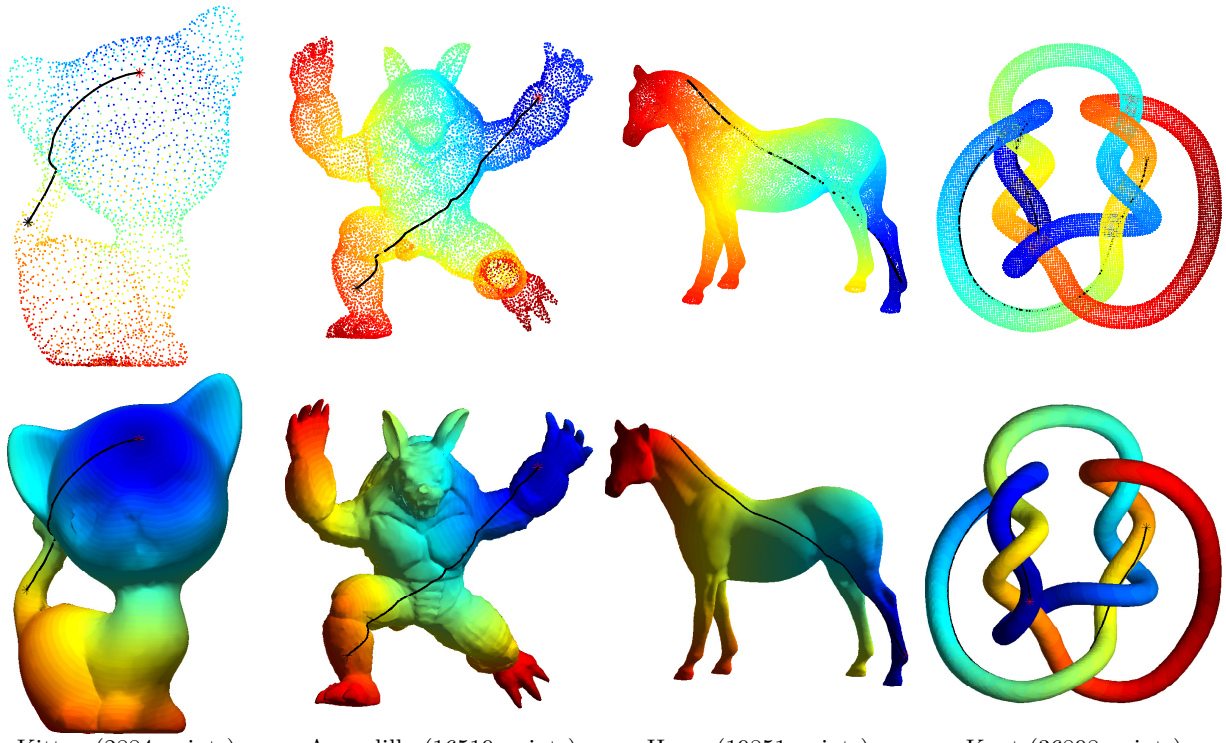
Figure 7: Distance maps on point clouds. The first row: distance maps to the given set Γ (red star points) are color-coded on the point cloud sampled from a hemisphere. The second row: distance maps, using mesh based method, to the same red stars on the hemisphere.

marked as red stars are color-coded on the point cloud. As a comparison, we also plot numerical results obtained by mesh based approach in the second row of Figure 7. The results are consistent.

$S^2 \subset \mathbb{R}^{10}$	uniform samlping					non-uniform samlping				
size	1002	1962	4002	7842	16002	1004	1964	4004	7844	16004
ground truth L_T	$\pi \approx 3.14159265$					$\pi \approx 3.14159265$				
numerical result L	3.140372	3.140995	3.141278	3.141425	3.141503	3.141159	3.143439	3.143498	3.141948	3.142561
relative error $\lvert L - L_T\rvert / L_T$	0.0389%	0.0190%	0.0100%	0.0053 %	0.0029%	0.0138%	0.0588%	0.0606%	0.0113%	0.0308%

Table 2: Geodesic distance from the north pole to the south pole on the point clouds sampled from S^2 isometric embedded in $\mathbb{R}^{10}$.

Our next experiment reports numerical results of geodesic tracing on point clouds using the method discussed in section 3.2. We choose point clouds sampled from the unit sphere $S^2 \subset \mathbb{R}^3$. They can also be viewed as point clouds in $\mathbb{R}^n$ by simply adding the last $n-3$ coordinates as zero. After that, by applying a randomly chosen orthonormal $n \times n$ matrix as a rotation and a randomly chosen vector as a translation, we can obtain point clouds sampled from the isometrically embedded S^2 in $\mathbb{R}^n$. We apply our developed method to compute distance maps and geodesics on these point clouds to demonstrate the capability in handling point clouds in high dimension (n=10). Table. 2 shows our results of distance between the north pole and south pole. It is clear to see that our method provides highly accurate estimation for geodesics on point clouds sampled from the unit sphere S^2 embedded in $\mathbb{R}^{10}$. In addition, we also report numerical results of geodesic tracing on point clouds sampled from more complicated surfaces in the first row of Figure 8. As comparisons,



Kitten (2884 points) Armadillo (16519 points) Horse (19851 points) Knot (36898 points)

Figure 8: Geodesics on point clouds. The first row: Distance maps are color-coded on the point clouds, and geodesics are illustrated by black curves. The second row: distance maps and geodesics obtained by mesh based method,

results obtained by using mesh based method introduced in [18] are plotted in the second row of Figure 8. The results show that our method can compute geodesics accurately and robustly on point clouds.

5 Conclusion

In this work, we propose a local mesh method that can be used to approximate differential operators, integrals, variational formulation and to solve PDEs on point clouds. The local connectivity and mesh at each point is constructed through local approximation of the tangent space, which is much easier and cheaper than the construction of a global mesh, especially in high dimensions. Examples of the area and volume approximation, solving eigenvalue problem of Laplace-Beltrami operator using variational formulation, and computing distance maps and geodesics on point clouds are presented.

Acknowledgement

Rongjie Lai's work is supported by Zumberge Individual Award from USC's James H. Zumberge Faculty Research and Innovation Fund. Jian Liang and Hongkai Zhao's works are partially sup-

ported by NSF DMS 1115698, ONR N00014-11-1-0602, ARO/MURI W911NF-07-1-0185, and NGA NURI HM1582-10-1-0012.

References

[1] F. Camastra and A. Vinciarelli. Estimating the intrinsic dimension of data with a fractal-based method. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(10):1404–1407, 2002.

[2] Mikhail Belkin and Partha Niyogi. Semi-supervised learning on riemannian manifolds. *Machine Learning*, 56(1-3):209–239, 2004.

[3] G. Chen, A.V. Little, M. Maggioni, and L. Rosasco. Some recent advances in multiscale geometric analysis of point clouds. *Wavelets and Multiscale Analysis*, pages 199–225, 2011.

[4] Mikhail Belkin, Jian Sun, and Yusu Wang. Constructing Laplace operator from point clouds in $\mathbb{R}^d$. In *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1031–1040, Philadelphia, PA, USA, 2009.

[5] Chuanjiang Luo, Jian Sun, and Yusu Wang. Integral estimation from point cloud in d-dimensional space: a geometric view. In *Symposium on Computational Geometry*, pages 116–124, 2009.

[6] J. Liang, R. Lai, T. W. Wong, and H. Zhao. Geometric understanding of point clouds using laplace-beltrami operator. *CVPR*, 2012.

[7] Jian Liang and Hongkai Zhao. Solving partial differential equations on point clouds. Technical report, UCLA, CAM Reports 2012.

[8] I. Jolliffe. *Principal component analysis*. Wiley Online Library, 2005.

[9] G. Taubin. Geometric signal processing on polygonal meshes. *EUROGRAPHICS*, 2000.

[10] M. Meyer, M. Desbrun, P. Schröder, and A. H. Barr. Discrete differential-geometry operators for triangulated 2-manifolds. *Visualization and Mathematics III. (H.C. Hege and K. Polthier, Ed.) Springer Verlag*, pages 35–57, 2003.

[11] M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr. Discrete differential geometry operators in nd. *In Proc. VisMath'02 Berlin Germany*, 2002.

[12] Guoliang Xu. Convergent discrete laplace-beltrami operator over triangular surfaces. *Proceedings of Geometric Modelling and Processing*, pages 195–204, 2004.

[13] R. Lai and T.F.Chan. A framework for intrinsic image processing on surfaces. *UCLA CAM 10-25, submitted*, 115(12):1647—1661, 2011.

[14] J. W. Milnor and J. D. Stasherff. *Characteristic Classes*. Princeton University Press, 1974.

[15] J. M. Lee. *Riemannian Manifolds: An Introduction to Curvature*. Springer-Verlag, 1997.

[16] X. Li, W. Wang, R.R. Martin, and A. Bowyer. Using low-discrepancy sequences and the crofton formula to compute surface areas of geometric models. *Computer-Aided Design*, 35(9):771–782, 2003.

[17] J. A. Sethian. A fast marching level set method for monotonically advancing fronts. *Proc. Nat. Acad. Sci. U.S.A.*, pages 1591–1595, 1996.

[18] R. Kimmel and J. A. Sethian. Computing geodesic paths on manifolds. *Proc. Natl. Acad. Sci. USA*, 95(15):8431–8435, 1998.

[19] H. Zhao. A fast sweeping method for eikonal equations. *Mathematics of computation*, 74(250):603–628, 2005.

[20] J. Qian, Y.T. Zhang, and H.K. Zhao. Fast sweeping methods for eikonal equations on triangular meshes. *SIAM Journal on Numerical Analysis*, 45(1):83, 2007.

[21] K. Uhlenbeck. Generic properties of eigenfunctions. *Amer. J. of Math.*, 98(4):1059–1078, 1976.

[22] J. Sun, M. Ovsjanikov, and L. Guibas. A concise and provabley informative multi-scale signature baded on heat diffusion. *Eurographics Symposium on Geometry Processing*, 2009.

[23] M. M. Bronstein and I. Kokkinos. Scale-invariant heat kernel signatures for non-rigid shape recognition. *Proc. Computer Vision and Pattern Recognition (CVPR)*, 2010.

[24] Y. Shi, R. Lai, R. Gill, D. Pelletier, D. Mohr, N. Sicotte, and A. W. Toga. Conformal metric optimization on surface (CMOS) for deformation and mapping in laplace-beltrami embedding space. *MICCAI*, 2011.

[25] R. Lai, Y. Shi, K. Scheibel, S. Fears, R. Woods, A. W. Toga, and T. F. Chan. Metric-induced optimal embedding for intrinsic 3D shape analysis. *CVPR*, 2010.

[26] R. Lai, Y. Shi, N. Sicotte, and A. W. Toga. Automated corpus callosum extraction via laplace-beltrami nodalâ parcellation and intrinsic geodesic curvature flows on surfaces. *ICCV*, 2011.

[27] I. Chavel. *Eigenvalues in Riemannian Geometry*. Academic press. INC, 1984.

[28] J. Jost. Riemannian geometry and geometric analysis. *Springer, 3rd edition*, 2001.

[29] M. Reuter, F.E. Wolter, and N. Peinecke. Laplace-Beltrami spectra as Shape-DNA of surfaces and solids. *Computer-Aided Design*, 38:342–366, 2006.

[30] J. Brandman. A level-set method for computing the eigenvalues of elliptic operators defined on compact hypersurfaces. *Journal of Scientific Computing*, 37(3):282–315, 2008.

[31] W. Gao, R. Lai, Y. Shi, I. Dinov, and A.W. Toga. A narrow-band approach for approximating the laplace-beltrami spectrum of 3d shapes. In *AIP Conference Proceedings*, volume 1281, page 1010, 2010.

[32] B. Levy. Laplace-beltrami eigenfunctions: Towards an algorithm that understands geometry. *IEEE International Conference on Shape Modeling and Applications, invited talk*, 2006.

[33] S. Dong, P-T. Bremer, M. Garland, V. Pascucci, and J. C. Hart. Spectral surface quadrangulation. *ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH*, 2006.

[34] R. M. Rustamov. Laplace-beltrami eigenfunctions for deformation invariant shape representation. *Eurographics Symposium on Geometry Processing*, 2007.

[35] B. Vallet and B. Levyvy. Spectral geometry processing with manifold harmonics. *Computer Graphics Forum (Proceedings Eurographics)*, 2008.

[36] M. Reuter. Hierarchical shape segmentation and registration via topological features of laplace-beltrami eigenfunctions. *International Journal of Computer Vision*, doi:10.1007/s11263-009-0278-1, 2009.

[37] A. M. Bronstein, M. M. Bronstein, R. Kimmel, M. Mahmoudi, and G. Sapiro. A gromov-hausdorff framework with diffusion geometry for topologically-robust non-rigid shape matching. *Int. Jour. Comput. Vis.*, 89(2-3):266–286, 2010.

[38] A. Qiu, D. Bitouk, and M. I. Miller. Smooth functional and structural maps on the neocortex via orthonormal bases of the Laplace-Beltrami operator. *IEEE Trans. Med. Imag.*, 25(10):1296–1306, 2006.

[39] Y. Shi, R. Lai, S. Krishna, N. Sicotte, I. Dinov, and A. W. Toga. Anisotropic Laplace-Beltrami eigenmaps: Bridging Reeb graphs and skeletons. In *Proc. MMBIA*, 2008.

[40] Y. Shi, R. Lai, K. Kern, N. Sicotte, I. Dinov, and A. W. Toga. Harmonic surface mapping with Laplace-Beltrami eigenmaps. In *Proc. MICCAI*, 2008.

[41] R. Lai, Y. Shi, I. Dinov, T. F. Chan, and A. W. Toga. Laplace-beltrami nodal counts: a new signature for 3d shape analysis. In *Proc. ISBI*, pages 694–697, 2009.

[42] Y. Shi, I. Dinov, and A. W. Toga. Cortical shape analysis in the laplace-beltrami feature space. In *Proc. MICCAI*, 2009.

[43] Y. Shi, R. Lai, J. Morra, I. Dinov, P. Thompson, and A. W. Toga. Robust surface reconstruction via laplace-beltrami eigen-projection and boundary deformation. *IEEE Trans. Medical Imaging*, 29(12):2009–2022, 2010.

[44] Y. Shi, R. Lai, and A. Toga. Unified geometry and topology correction for cortical surface reconstruction with intrinsic reeb analysis. *MICCAI*, 2012.

[45] F. Mémoli and G. Sapiro. Distance functions and geodesics on submanifolds of rd and point clouds. *SIAM Journal on Applied Mathematics*, pages 1227–1260, 2005.