

Geometric Approaches for 3D Shape Denoising and Retrieval



Anis Kacem

A Thesis
in
The Department
of
Electrical and Computer Engineering
Concordia University

Presented in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy at
Concordia University
Montreal, QC, Canada

April 2013



© **Anis Kacem, 2013**

**CONCORDIA UNIVERSITY
SCHOOL OF GRADUATE STUDIES**

This is to certify that the thesis prepared

By: Anis Kacem

Entitled: Geometric Approaches for 3D Shape Denoising and Retrieval

and submitted in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY (Electrical and Computer Engineering)

complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

_____Chair
Dr. N. Bhuiyan

_____External Examiner
Dr. H. Krim

_____External to Program.
Dr. C. Alecsandru

_____Examiner
Dr. A. Youssef

_____Examiner
Dr. N. Bouguila

_____Thesis Supervisor
Dr. A. Ben Hamza

Approved by _____
_____Chair of Department or Graduate Program Director
Dr. A.R. Sebak , Graduate Program Director

June 6, 2013

_____Dr. Robin A.L. Drew, Dean
Faculty of Engineering & Computer Science

Abstract

Geometric Approaches for 3D Shape Denoising and Retrieval

Anis Kacem

A key issue in developing an accurate 3D shape recognition system is to design an efficient shape descriptor for which an index can be built, and similarity queries can be answered efficiently. While the overwhelming majority of prior work on 3D shape analysis has concentrated primarily on rigid shape retrieval, many real objects such as articulated motions of humans are nonrigid and hence can exhibit a variety of poses and deformations.

Motivated by the recent surge of interest in content-based analysis of 3D objects in computer-aided design and multimedia computing, we develop in this thesis a unified theoretical and computational framework for 3D shape denoising and retrieval by incorporating insights gained from algebraic graph theory and spectral geometry. We first present a regularized kernel diffusion for 3D shape denoising by solving partial differential equations in the weighted graph-theoretic framework. Then, we introduce a computationally fast approach for surface denoising using the vertex-centered finite volume method coupled with the mesh covariance fractional anisotropy. Additionally, we propose a spectral-geometric shape skeleton for 3D object recognition based on the second eigenfunction of the Laplace-Beltrami operator in a bid to capture the global and local geometry of 3D shapes. To further enhance the 3D shape retrieval accuracy, we introduce a graph matching approach by assigning geometric features to each endpoint of the shape skeleton. Extensive experiments are carried out on two 3D shape benchmarks to assess the performance of the proposed shape retrieval framework in comparison with state-of-the-art methods. The experimental results show that the proposed shape descriptor delivers best-in-class shape retrieval performance.

Table of Contents

Title Page	i
Table of Contents	iv
List of Tables	vii
List of Figures	viii
1 Introduction	1
1.1 Framework and Motivation	1
1.2 Problem Statement	2
1.2.1 Surface Denoising	5
1.2.2 3D Shape Recognition	5
1.3 Literature Review	6
1.3.1 Surface Denoising	6
1.3.2 3D Shape Recognition	12
1.4 Thesis Overview and Contributions	15
2 Regularized Kernel Weighted Diffusion	16
2.1 Introduction	16
2.2 Regularized Diffusion on Weighted Graphs	18
2.2.1 Differential Operators on Weighted Graphs	18
2.2.2 Problem Statement	20
2.2.3 Illustrative Case	22
2.2.4 Choice of Weights	23

2.2.5	Kernel Weighted Diffusion	24
2.3	Experimental Results	27
2.3.1	Output Results without Fitting Term	28
2.3.2	Output Results with Fitting Term	28
2.3.3	Quantitative evaluation	32
3	Vertex-Centered Finite Volume Diffusion	34
3.1	Introduction	34
3.2	Proposed Method	36
3.2.1	Eigenanalysis of Mesh Covariance Matrix	36
3.2.2	Finite Volumes of a Dual Mesh	39
3.2.3	Proposed Mesh Denoising Flow	40
3.3	Experimental Results	43
3.3.1	Qualitative evaluation of the proposed method	45
3.3.2	Quantitative evaluation of the proposed method	45
4	Spectral Geometric Shape Retrieval	54
4.1	Introduction	54
4.2	Spectral Reeb Graph Framework	57
4.2.1	Laplace-Beltrami Operator	57
4.2.2	Morse theory for Topological Modeling	58
4.2.3	Spectral Geometric Analysis of the Laplace-Beltrami Operator	60
4.2.4	Discrete Laplace-Beltrami Operator	62
4.2.5	Spectral Skeleton	63
4.3	Spectral Reeb Graph Matching	68
4.3.1	Indexing	71
4.3.2	Endpoint correspondence	71
4.3.3	Matching endpoints using skeleton paths	74
4.4	Experimental Results	75
5	Conclusions and Future Work	82

5.1	Contributions of the Thesis	82
5.1.1	Kernel Weighted Diffusion	82
5.1.2	Finite Volume Diffusion	83
5.1.3	Spectral Geometric Shape Recognition	83
5.2	Future Research Directions	83
5.2.1	Partial Matching of 3D Shapes	83
5.2.2	Unifying Topology and Geometry	83
5.2.3	From Image Processing to Geometry Processing	84
	References	85

List of Tables

4.1	Matching results using the proposed approach. Each database object is matched against all the other objects in the database. Each cell shows the dissimilarity measure $\mathcal{D}(G, \tilde{G})$ between two objects selected from the database. The smallest value corresponds to the correct match.	79
4.2	Retrieval results using the McGill Shape Benchmark. The query shapes are shown in the second row. The top ten retrieved objects (top-to-bottom) using spherical harmonics (SH), Reeb graph path dissimilarity (RGPD), and our proposed approach (SRG) are shown in rows 5 to 14.	80
4.3	Retrieval results using Princeton 3D Benchmark. The query shapes are shown in the second row. The top five retrieved objects (top-to-bottom) of our proposed approach.	81

List of Figures

1.1	Triangular mesh representation.	2
1.2	Vertex neighborhood v_i^* (left). Illustration of $\text{area}(t)$ and normal vector $n(t)$ (right).	3
1.3	Teapot model (left). Right: vertex normals.	4
1.4	(a) Illustration of vertex gradient on a 3D face model; (b) close-up view.	4
1.5	3D object recognition diagram.	6
1.6	3D object matching problem.	7
1.7	Oversmoothing effect of Laplacian smoothing.	8
1.8	Illustration of the mesh mean filter algorithm.	9
1.9	Illustration of the mesh median filter algorithm.	9
1.10	Illustration of the angles α_{ij} and β_{ij}.	10
1.11	Cauchy weight function with $c = 2.3849$.	12
1.12	Illustration of two neighboring rings.	13
2.1	3D level surface of the Gaussian kernel function $K(x)$.	24
2.2	(a) 3D object with $m = 3403$ vertices and its (b) mesh neighborhood weighting kernel matrix.	26
2.3	(a) 3D object; (b) Mesh KDE; (c) Horizontal slices of the mesh KDE.	26
2.4	Different cases of diffusion: Spherical diffusion (left); planar diffusion (center); linear diffusion (right).	28
2.5	Output results without fitting term: (a) Original horse model; (b) Noisy Model; (c) $\kappa = 0.3$; (d) $\kappa = 0.4$; (e) $\kappa = 0.5$; (f) $\kappa = 0.6$.	29

2.6	Output results without fitting term: (a) Original Model; (b) Noisy Model; (c) Uniform weighted diffusion; (d) Kernel weighted diffusion with $\kappa = 0.45$; (e) Laplacian filter; (f) Mean filter; (g) Angle median filter; (h) Bilateral filter.	30
2.7	Output results with fitting term: (a) Original Model; (b) Noisy Model; (c) Uniform weighted diffusion; (d) Kernel weighted diffusion with $\kappa = 0.45$; (e) Laplacian filter; (f) Mean filter; (g) Angle median filter; (h) Bilateral filter.	31
2.8	Quantitative visual errors without fitting term.	33
2.9	Quantitative visual errors with fitting term	33
3.1	Illustration of the mesh covariance matrix around the neighborhood of a vertex v_i.	37
3.2	Left: 3D face model and the eigenvectors of its mesh covariance matrix. Right: close-up view.	38
3.3	3D face model colored by the eigenvalues of its mesh covariance matrix.	39
3.4	cell-centered vs. vertex-centered control volumes.	40
3.5	The dual mesh is shown in solid thin gray lines. Thick blue lines show the original mesh (also referred to as primal mesh).	41
3.6	(a) 3D face model; (b) dual mesh.	41
3.7	(a) Noisy 3D bunny model; (b) denoised model using our method; (c) output result of our method colored by fractional anisotropy; (d) output result of our method colored by mean curvature. The number of iterations is set to 1.	44
3.8	Comparison of denoising results on a 3D hand model. (a)-(b) Noisy model (back and front views); (c)-(d) multiscale anisotropic Laplacian; (e)-(f) our method. The number of iterations is set to 1.	46
3.9	Output results of our method colored by: (a)-(b) fractional anisotropy; (c)-(d) mean curvature. The number of iterations is set to 1.	47
3.10	Comparison of denoising results on a 3D horse model. (a) Noisy model; (b) multiscale anisotropic Laplacian; (c) our method; (d) output result of our method colored by mean curvature. The number of iterations is set to 1.	48

3.11	Comparison of denoising results on a 3D foot bone model. (a) Noisy model; (b) multiscale anisotropic Laplacian; (c) our method; (d) output result of our method colored by mean curvature. The number of iterations is set to 1.	49
3.12	Comparison of denoising results on a 3D bunny model (enlarged view). Noisy model (top); (b) multiscale anisotropic Laplacian (middle); (c) our method (bottom). The number of iterations is set to 1.	50
3.13	Output result of our method colored by mean curvature. The number of iterations is set to 1.	51
3.14	Plots of L_2 face-normal error vs. iteration number for the hand model.	51
3.15	Plots of L_2 face-normal error vs. iteration number for the horse model.	52
3.16	Plots of L_2 face-normal error vs. iteration number for the fan disk model.	52
3.17	Top row (from left to right): Noisy rabbit model and output results using our method with iteration numbers 1, 5, 10, and 15. Bottom row: mean curvature visualization of the models shown in top row.	53
4.1	Parametric representation of a surface.	58
4.2	Illustration of: (a) Level curve L_a, (b) Subsurface \mathbb{M}_a, (c) Subsurface and Level curve.	59
4.3	Evolution of \mathbb{M}_a as a changes.	60
4.4	Cotangent weight scheme: illustration of the angles α_{ij} and β_{ij}.	62
4.5	3D elephant model and sparsity pattern plot of the associated cotangent matrix C.	64
4.6	(a)-(c) 3D horse model colored by $\varphi_2, \varphi_3, \varphi_4$. (d)-(f) level sets of $\varphi_2, \varphi_3, \varphi_4$.	65
4.7	(a)-(b) 3D hand model colored by φ_2 (front and back views). (c)-(d) Level sets of φ_2 (front and back views).	66
4.8	(a) Isocontours are invariant under both global and local deformation. (b) Proportionality correspondence of pairwise nonrigid shapes with varied topological structure. (c) Isocontours are consistent among different classes of shapes.	67
4.9	(a) 3D horse model colored by φ_2; (b) level sets of φ_2; (c) spectral Reeb graph.	68
4.10	Spectral Reeb graph of 3D Octopus model and its skeleton endpoints displayed in blue color.	70

4.11 (a) Horse’s spectral Reeb graph. (b) Shortest paths between pairs of endpoints on the skeleton.	72
4.12 Shortest paths between the mesh centroid and an endpoint on the skeleton.	73
4.13 Robustness of spectral Reeb graph to noise.	76
4.14 Sample shapes from McGill’s Articulated Shape Database. Only two shapes for each of the 10 classes are shown.	77
4.15 Precision vs. Recall curves for Reeb graph path dissimilarity, spherical harmonics, medial surfaces, Reeb graph path dissimilarity, and proposed approach using the McGill Shape Benchmark 4.14.	78

List of Acronyms

PDE	Partial Differential Equations
LB	Laplace-Beltrami
SRG	Spectral Reeb Graph
GPS	Global Point Signature
HKS	Heat Kernel Signature
ADF	Auto Diffusion Function
SH	Spherical Harmonics
MS	Medial Surfaces
RGPD	Reeb Graph Path Dissimilarity

Introduction

In this chapter, we present the framework and motivation behind this work, followed by the problem statement, literature review and thesis contributions.

1.1 Framework and Motivation

With the increasing use of 3D scanners to create 3D models, there is a rising need for robust mesh denoising to remove inevitable noise in the measurements. Even with high-fidelity scanners, the acquired models are invariably noisy, and therefore require filtering. The great challenge in image processing and computer graphics is to devise computationally efficient and optimal algorithms for recovering images and 3D models contaminated by noise and preserving their geometrical structure. With the increasing use of scanners to create 3D models which are usually represented as triangle meshes in computer graphics and geometric-aided design, there is a rising need for robust and efficient 3D mesh denoising techniques to remove undesirable noise from the data.

In the same vein, the importance of 3D shape recognition is irrupting due to the difficulty in processing information expeditiously without its recognition. With the increasing use of 3D scanners and as a result of emerging multimedia computing technologies, vast databases of 3D models are distributed freely or commercially on the Internet. The availability and widespread usage of such large databases coupled with the need to explore 3D models in depth as well as in breadth has

sparked the need to organize and search these vast data collections, retrieve the most relevant selections, and permit them to be effectively reused. 3D objects consist of geometric and topological information, and their compact representation is an important step towards a variety of computer vision applications, particularly matching and retrieval in a database of 3D models. The first step in 3D object matching usually involves finding a reliable shape descriptor or skeletal graph which will encode efficiently the 3D shape information.

1.2 Problem Statement

In computer graphics and geometric-aided design, triangle meshes have become the de facto standard representation of 3D objects. A triangle mesh \mathbb{M} may be defined as $\mathbb{M} = (\mathcal{V}, \mathcal{E})$ or $\mathbb{M} = (\mathcal{V}, \mathcal{T})$, where $\mathcal{V} = \{v_1, \dots, v_m\}$ is the set of vertices, $\mathcal{E} = \{e_{ij}\}$ is the set of edges, and $\mathcal{T} = \{t_1, \dots, t_n\}$ is the set of triangles, as depicted in Figure 1.1.

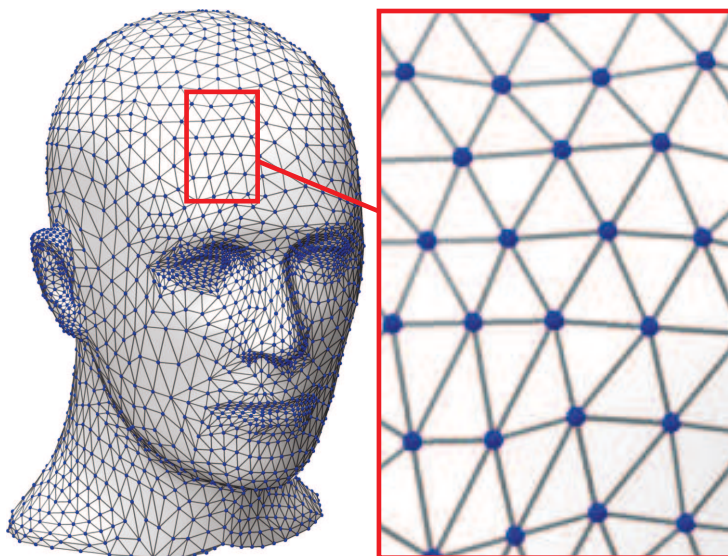


FIGURE 1.1: Triangular mesh representation.

Each edge $e_{ij} = [v_i, v_j]$ of the triangle mesh connects a pair of vertices $\{v_i, v_j\}$. Two distinct vertices $v_i, v_j \in \mathcal{V}$ are adjacent (denoted by $v_i \sim v_j$ or simply $i \sim j$) if they are connected by an edge, i.e. $e_{ij} \in \mathcal{E}$. The neighborhood of a vertex v_i is the set $v_i^* = \{v_j \in \mathcal{V} : v_i \sim v_j\}$. The degree d_i of a vertex v_i is simply the cardinality of v_i^* . We denote by $\mathcal{T}(v_i^*)$ the set of triangles

of the ring v_i^* . Figure 1.2(left) depicts an example of a neighborhood v_i^* , where the degree of the vertex v_i is $d_i = 6$, and the number of triangles of the set $\mathcal{T}(v_i^*)$ is also equal to 6.

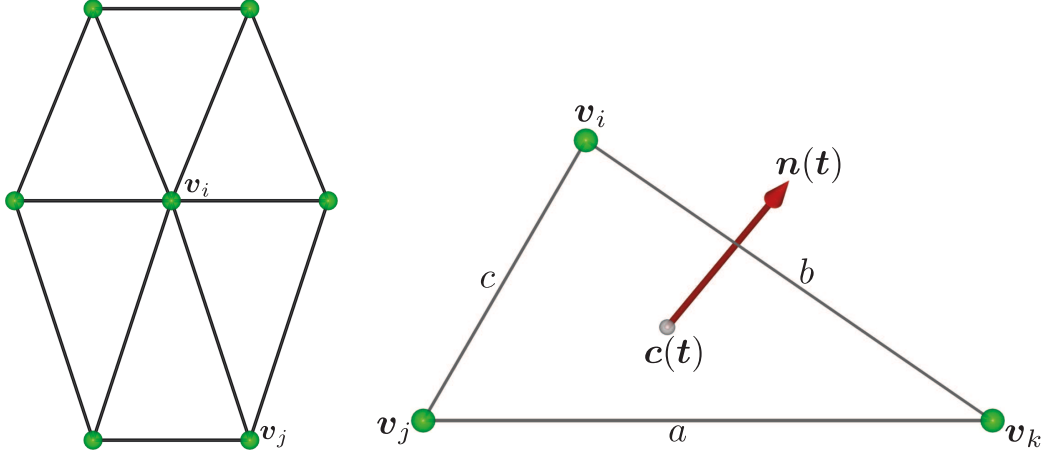


FIGURE 1.2: Vertex neighborhood v_i^* (left). Illustration of $\text{area}(t)$ and normal vector $n(t)$ (right).

Consider a triangle $t \in \mathcal{T}$ with vertices v_i, v_j and v_k , and side lengths a, b and c as illustrated in Fig. 1.2 (right). The triangle normal $n(t)$ can be calculated as the vector cross product of two edges of the triangle, and according to Heron's formula, $\text{area}(t)$ is equal to

$$\frac{1}{4} \sqrt{(a + (b + c))(a + (b - c))(c + (a - b))(c - (a - b))}, \quad (1.1)$$

where a, b and c are arranged such that $a \geq b \geq c$.

The centroid $c(t)$ of the triangle t is obtained by averaging its vertices

$$c(t) = \frac{v_i + v_j + v_k}{3}. \quad (1.2)$$

The normal vector n_i at a vertex v_i of the mesh is obtained by averaging the normals of its neighboring triangles

$$n_i = \frac{1}{d_i} \sum_{t_j \in \mathcal{T}(v_i^*)} n(t_j). \quad (1.3)$$

Figure 1.3 depicts a 3D teapot model (left) and its vertex normals (right).

The mean edge length $\bar{\ell}$ of the mesh \mathbb{M} is given by

$$\bar{\ell} = \frac{1}{|\mathcal{E}|} \sum_{e_{ij} \in \mathcal{E}} \|e_{ij}\|, \quad (1.4)$$

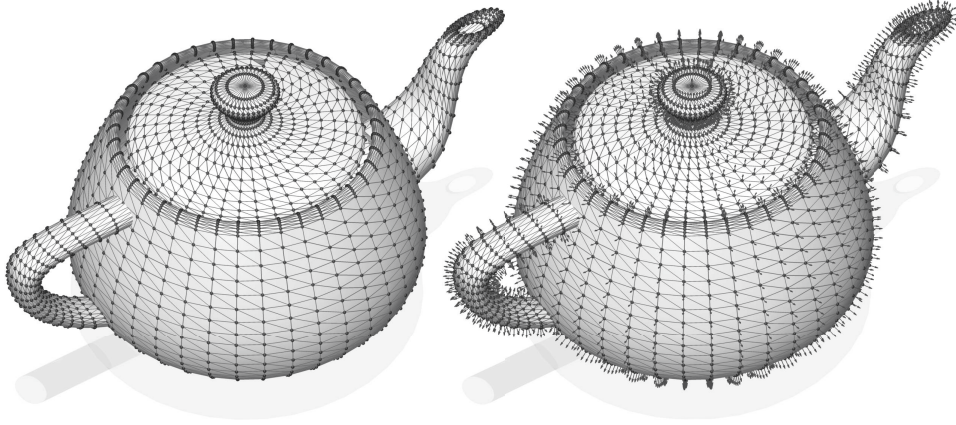


FIGURE 1.3: Teapot model (left). Right: vertex normals.

where $\|e_{ij}\| = \|\mathbf{v}_i - \mathbf{v}_j\|$ if $\mathbf{v}_i \sim \mathbf{v}_j$, and $\|e_{ij}\| = 0$ otherwise.

We define the vertex gradient operator $\nabla \mathbf{v}_i$ (see Figure 1.4) as

$$\nabla \mathbf{v}_i = \left\{ \frac{\mathbf{v}_j}{\sqrt{d_j}} - \frac{\mathbf{v}_i}{\sqrt{d_i}} : \mathbf{v}_j \in \mathbf{v}_i^* \right\}. \quad (1.5)$$

We also define the vertex Laplace operator as

$$\Delta \mathbf{v}_i = \sum_{j \sim i} \frac{1}{\sqrt{d_i}} \left(\frac{\mathbf{v}_j}{\sqrt{d_j}} - \frac{\mathbf{v}_i}{\sqrt{d_i}} \right). \quad (1.6)$$

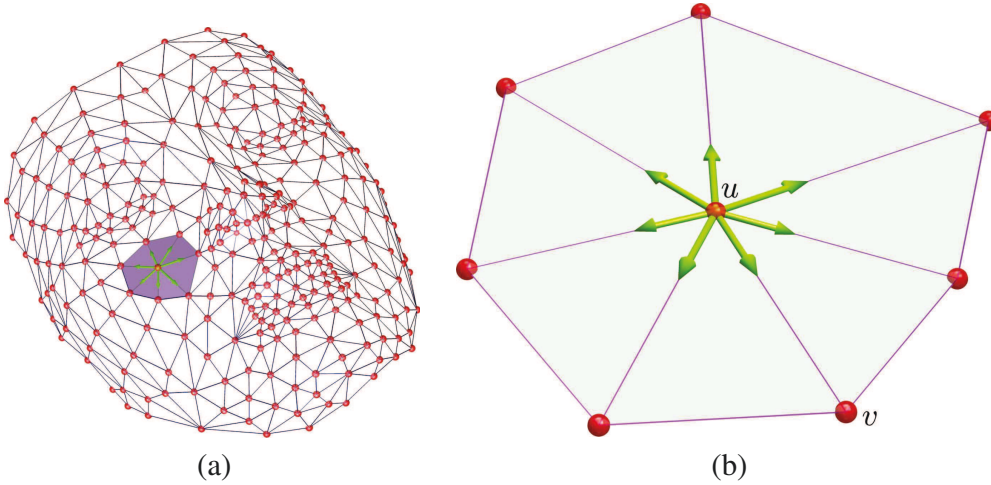


FIGURE 1.4: (a) Illustration of vertex gradient on a 3D face model; (b) close-up view.

1.2.1 Surface Denoising

In all real applications, measurements are usually perturbed by noise. In the course of acquiring, transmitting or processing a 3D model for example, the noise-induced degradation often yields a resulting vertex observation model, and the most commonly used is the additive one,

$$\mathbf{v} = \mathbf{u} + \boldsymbol{\eta}, \quad (1.7)$$

where the observed vertex \mathbf{v} includes the original vertex \mathbf{u} , and the random noise process $\boldsymbol{\eta}$ which is usually assumed to be Gaussian with zero mean and standard deviation σ .

Surface denoising refers to the process of recovering a 3D model contaminated by noise. The challenge of the problem of interest lies in recovering the vertex \mathbf{u} from the observed vertex \mathbf{v} , and furthering the estimation by making use of any prior knowledge/assumptions about the noise process $\boldsymbol{\eta}$.

Generally, surface denoising methods may be classified into two major categories: isotropic and anisotropic. The former techniques filter the noisy data independently of direction, while the latter methods modify the diffusion equation to make it nonlinear or anisotropic in order to preserve the sharp features of a 3D mesh surface. Most of these nonlinear methods were inspired by anisotropic-type diffusions in the image processing literature [1–8].

1.2.2 3D Shape Recognition

Roughly speaking, 3D object recognition techniques may be classified into two major categories: feature-based and global methods, as illustrated in Figure 1.5. Most 3D shape matching techniques proposed in the literature of computer graphics, computer vision and computer-aided design are based on geometric representations which represent the features of an object in such a way that the shape dissimilarity problem reduces to the problem of comparing two such object representations. Feature-based methods require that features be extracted and described before two objects can be compared. An alternative to feature-based representations is global methods. The idea here is to represent an object by a global measure or shape distribution defined on the surface of the object. The shape matching problem is then performed by computing a dissimilarity measure between the shape distributions of two arbitrary objects.

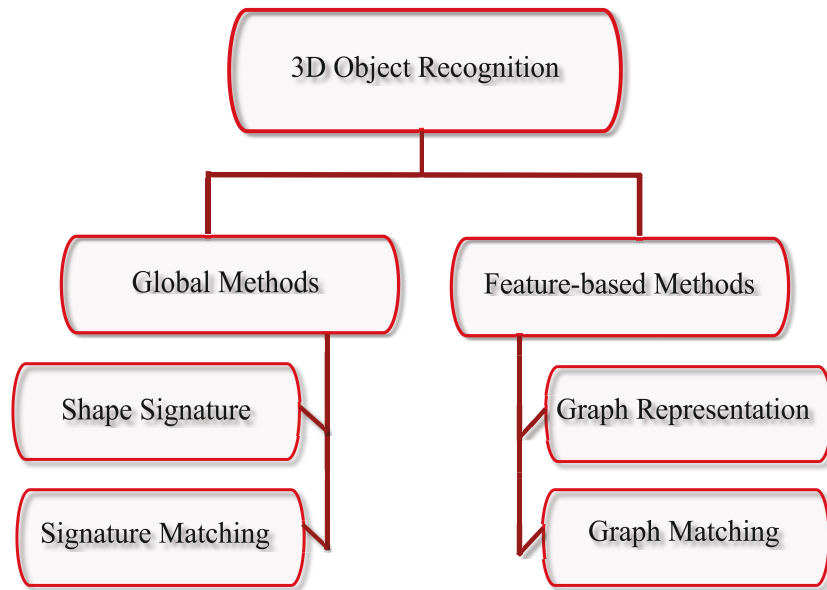


FIGURE 1.5: 3D object recognition diagram.

The goal of 3D object matching may be described as follows: Given two 3D objects \mathbb{M}_1 and \mathbb{M}_2 to be matched, find their respective global measures or shape descriptors s_1 and s_2 , and calculate how dissimilar these objects are using a dissimilarity measure $\mathcal{D}(s_1, s_2)$ that has to be quantified. The basic idea behind the shape descriptor is to characterize a 3D object with a skeleton graph that will help discriminate between objects in a database of 3D models. The 3D object matching problem is depicted in Figure 1.6.

1.3 Literature Review

This section provides an brief overview of related work.

1.3.1 Surface Denoising

In this subsection, we review some representative methods for 3D surface denoising that are closely related to our proposed approaches, and we briefly show their mathematical foundations and algorithmic methodologies as well as their limitations.

The partial differential equation (PDE)-based smoothing approach is commonly formulated in a continuous domain which enjoys a large arsenal of analytical tools, and hence offers a greater

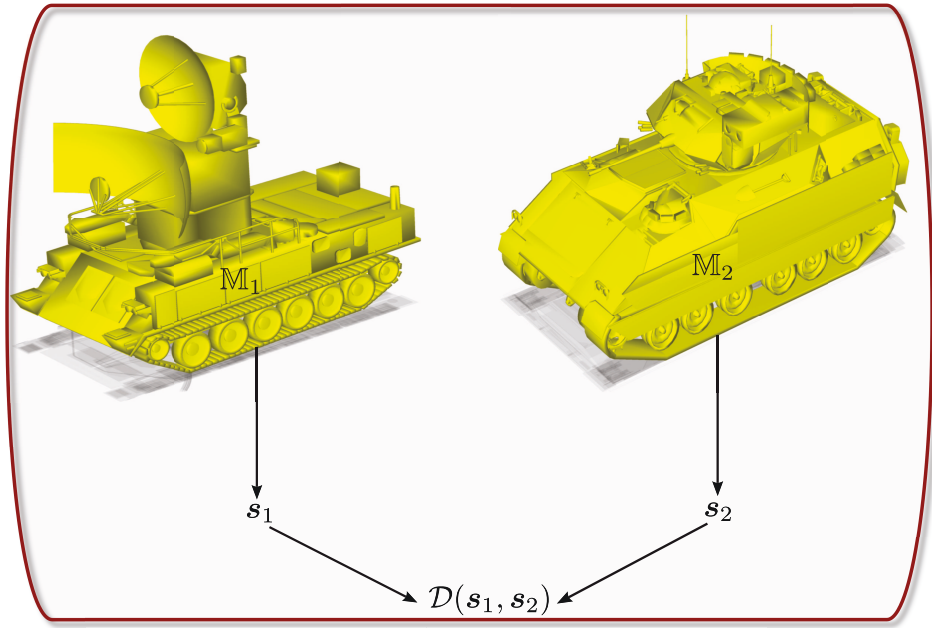


FIGURE 1.6: 3D object matching problem.

flexibility. Laplacian smoothing is the most commonly used mesh smoothing method which repeatedly and simultaneously adjusts the location of each mesh vertex to the geometric center of its neighboring vertices using the following update rule

$$\mathbf{v}_i \leftarrow \mathbf{v}_i + \sum_{j \sim i} \left(\frac{\mathbf{v}_j - \mathbf{v}_i}{d_i} \right). \quad (1.8)$$

It is worth pointing out that the Laplacian flow given by Eq. (1.8) is the discrete form of the isotropic heat equation $\mathbf{v}_t = \Delta \mathbf{v}$ applied to each vertex of the triangle mesh, where we assume that all vertices have the same degree. Although the Laplacian smoothing flow is simple and fast, it tends, however, to produce a shrinking effect and an oversmoothing result, as shown in Figure 1.7.

A. Mean Filter for Averaging Face Normals:

The mean filter procedure is depicted in Figure 1.8 and is applied in three successive steps [6]:

Step 1 : compute the area weighted average face normal $\mathbf{m}(\mathbf{t}_i)$ for each mesh triangle \mathbf{t}_i :

$$\mathbf{m}(\mathbf{t}_i) = \frac{1}{\sum_{\mathbf{t}_j \in \mathbf{t}_i^*} A(\mathbf{t}_j)} \sum_{\mathbf{t}_j \in \mathbf{t}_i^*} A(\mathbf{t}_j) \mathbf{n}(\mathbf{t}_j). \quad (1.9)$$

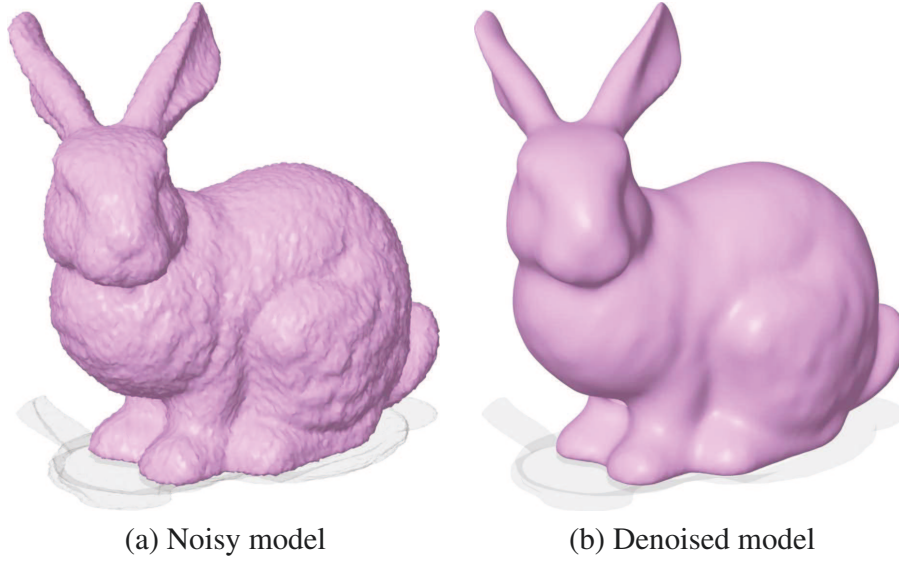


FIGURE 1.7: Oversmoothing effect of Laplacian smoothing.

Step 2 : normalize the averaged normal $\mathbf{m}(\mathbf{t}_i)$:

$$\mathbf{m}(\mathbf{t}_i) \leftarrow \frac{\mathbf{m}(\mathbf{t}_i)}{\|\mathbf{m}(\mathbf{t}_i)\|} \quad (1.10)$$

Step 3 : update each vertex \mathbf{v}_i in the mesh as follows:

$$\mathbf{v}_i \leftarrow \mathbf{v}_i + \frac{1}{\sum_{\mathbf{t}_j \in \mathcal{T}(\mathbf{v}_i^*)} A(\mathbf{t}_j)} \sum_{\mathbf{t}_j \in \mathcal{T}(\mathbf{v}_i^*)} A(\mathbf{t}_j) \boldsymbol{\pi}_i(\mathbf{t}_j) \quad (1.11)$$

where $\boldsymbol{\pi}_i(\mathbf{t}_j) = \langle \mathbf{e}_{ij}, \mathbf{m}(\mathbf{t}_j) \rangle \mathbf{m}(\mathbf{t}_j)$, and $\mathbf{e}_{ij} = \mathbf{c}_j - \mathbf{v}_i$ is the vector from vertex \mathbf{v}_i to the centroid \mathbf{c}_j of the triangle \mathbf{t}_j . Note that by definition of the inner product, the vector $\boldsymbol{\pi}_i(\mathbf{t}_j)$ is the projection of the vector \mathbf{e}_{ij} onto the direction of the normal \mathbf{t}_j .

B. Angle Median Filtering for Face Normals:

For each triangle $\mathbf{t}_i \in \mathcal{T}$, denote by $\Theta_i = \{\theta_{ij} = \angle(\mathbf{n}(\mathbf{t}_i), \mathbf{n}(\mathbf{t}_j)) : \mathbf{t}_j \in \mathbf{t}_i^*\}$ the set of angles between $\mathbf{n}(\mathbf{t}_i)$ and $\mathbf{n}(\mathbf{t}_j)$, where $\mathbf{n}(\mathbf{t}_i)$ is the normal of \mathbf{t}_i and $\mathbf{n}(\mathbf{t}_j)$ is the normal of \mathbf{t}_j . As illustrated in Figure 1.9, instead of computing the average face normal in Step 1 of the mean filter, in the angle median filtering method [6] we first compute the median angle $\hat{\theta}_i = \text{median}(\Theta_i) = \angle(\mathbf{n}(\mathbf{t}_i), \mathbf{n}(\hat{\mathbf{t}}_j))$ where $\hat{\mathbf{t}}_j$ is the triangle where the median angle is achieved, and then we replace the weighted average normal $\mathbf{m}(\mathbf{t}_i)$ by $\mathbf{n}(\hat{\mathbf{t}}_j)/\|\mathbf{n}(\hat{\mathbf{t}}_j)\|$.

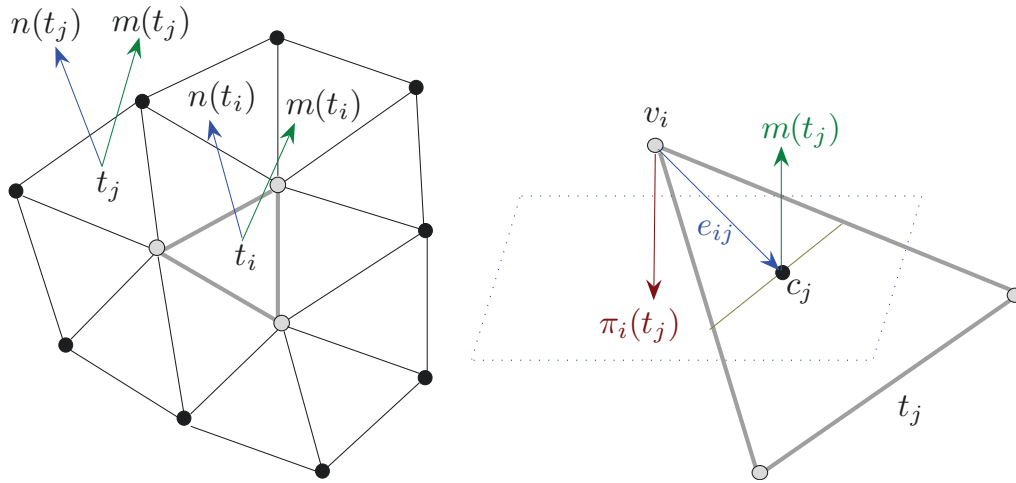


FIGURE 1.8: Illustration of the mesh mean filter algorithm.

The mean and median filtering methods show better performance than the Laplacian flow. These two methods, however, require a large number of iterations to reach stable results.

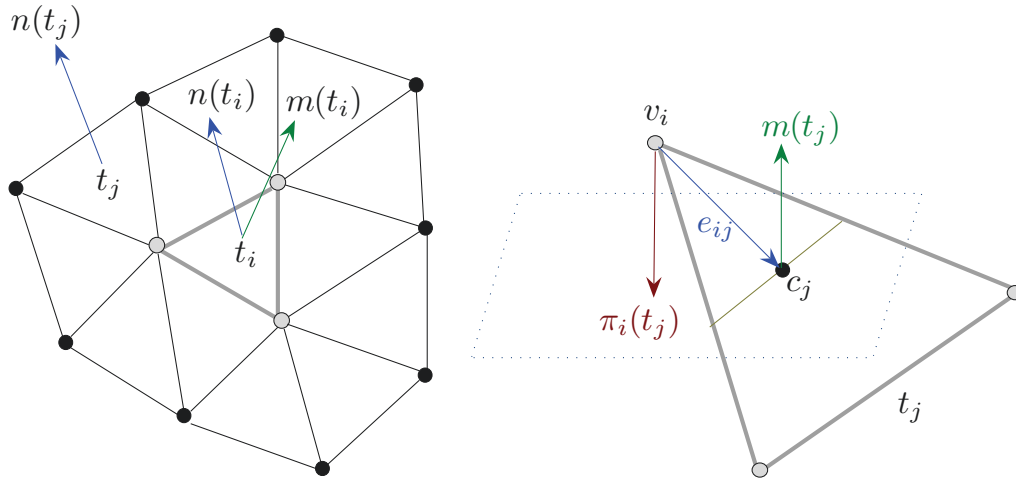


FIGURE 1.9: Illustration of the mesh median filter algorithm.

C. Weighted Laplacian Filter:

Instead of using unit edge costs, the weighted Laplacian smoothing method [9] chooses edge weights based on the approximation to the curvature normal. The edge weights w_{ij} are given by $w_{ij} = \cot \alpha_{ij} + \cot \beta_{ij}$, where α_{ij} and β_{ij} are the angles $\angle v_i v_{j-1} v_j$ and $\angle v_i v_{j+1} v_j$ depicted in

Figure 1.10. Then, the update rule of the weighted Laplacian smoothing procedure is given by

$$\mathbf{v}_i \leftarrow \mathbf{v}_i + \frac{1}{\sum_{j \sim i} w_{ij}} \sum_{j \sim i} w_{ij} (\mathbf{v}_j - \mathbf{v}_i). \quad (1.12)$$

The improved edge weights are used to compensate for the irregularities of the triangle mesh and to help avoid the edge equalization.

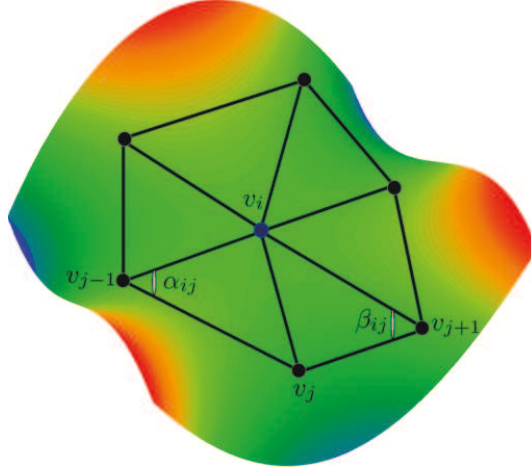


FIGURE 1.10: Illustration of the angles α_{ij} and β_{ij} .

D. Bilateral Mesh Denoising:

Similar to the mean and angle median filters, the bilateral 3D mesh denoising method [7] was also adopted from the bilateral filtering technique used in image denoising. This algorithm filters each vertex \mathbf{v}_i of the mesh in the normal direction using local neighborhoods according the following update rule:

$$\mathbf{v}_i \leftarrow \mathbf{v}_i + \mathbf{n}_i \left(\frac{\sum_{j \sim i} (w_{ij}^c w_{ij}^s) \langle \mathbf{n}_i, \mathbf{v}_i - \mathbf{v}_j \rangle}{\sum_{j \sim i} w_{ij}^c w_{ij}^s} \right), \quad (1.13)$$

where \mathbf{n}_i is the vertex normal, w_{ij}^c is the standard Gaussian filter

$$w_{ij}^c = e^{-\|\mathbf{v}_i - \mathbf{v}_j\|^2 / 2\sigma_c^2}$$

with parameter σ_c , and w_{ij}^s is feature-preserving weight function

$$w_{ij}^s = e^{-\langle \mathbf{n}_i, \mathbf{v}_i - \mathbf{v}_j \rangle^2 / 2\sigma_s^2}$$

with parameter σ_s . Bilateral mesh denoising algorithm is parameter-dependent and requires the user to assign the two parameters σ_c and σ_s interactively. The lack of object information, however, might affect the smoothing result.

E. Vertex-based Anisotropic Diffusion:

The vertex-based anisotropic diffusion [10] is given by the discrete partial differential equation

$$\mathbf{v}_t = \text{div}(g(|\nabla \mathbf{v}|)\nabla \mathbf{v}), \quad (1.14)$$

where g is Cauchy weight function given by

$$g(x) = \frac{1}{1 + x^2/c^2}, \quad (1.15)$$

and c is a constant tuning parameter that needs to be estimated. Intuitively, the smoothing effect of the vertex-based anisotropic diffusion may be explained as follows: in flat regions of a 3D mesh where the vertex gradient magnitudes are relatively small, Eq. (1.14) is reduced to the heat equation which tends to smooth more but the smoothing effect is unnoticeable. And around the sharp features of the 3D mesh where the vertex gradient magnitudes are large, the diffusion flow given by Eq. (1.14) tends to smooth less and hence leads to a much better preservation of the mesh geometric structures. It can be shown (see [11]) that the 95% asymptotic efficiency on the standard Gaussian distribution is obtained with the tuning constant $c = 2.3849$.

In discrete form, the vertex-based anisotropic diffusion flow is reduced to the following update rule

$$\mathbf{v}_i \leftarrow \mathbf{v}_i + \sum_{j \sim i} \frac{1}{\sqrt{d_i}} \left(\frac{\mathbf{v}_j}{\sqrt{d_j}} - \frac{\mathbf{v}_i}{\sqrt{d_i}} \right) \left(g(|\nabla \mathbf{v}_i|) + g(|\nabla \mathbf{v}_j|) \right), \quad (1.16)$$

where the gradient magnitudes are given by

$$|\nabla \mathbf{v}_i| = \left(\sum_{j \sim i} \left\| \frac{\mathbf{v}_i}{\sqrt{d_i}} - \frac{\mathbf{v}_j}{\sqrt{d_j}} \right\|^2 \right)^{1/2}, \quad (1.17)$$

and

$$|\nabla \mathbf{v}_j| = \left(\sum_{\mathbf{v}_k \in \mathbf{v}_j^*} \left\| \frac{\mathbf{v}_j}{\sqrt{d_j}} - \frac{\mathbf{v}_k}{\sqrt{d_k}} \right\|^2 \right)^{1/2}. \quad (1.18)$$

Note that the update rule of the proposed method requires the use of two neighboring rings as depicted in Figure 1.12.

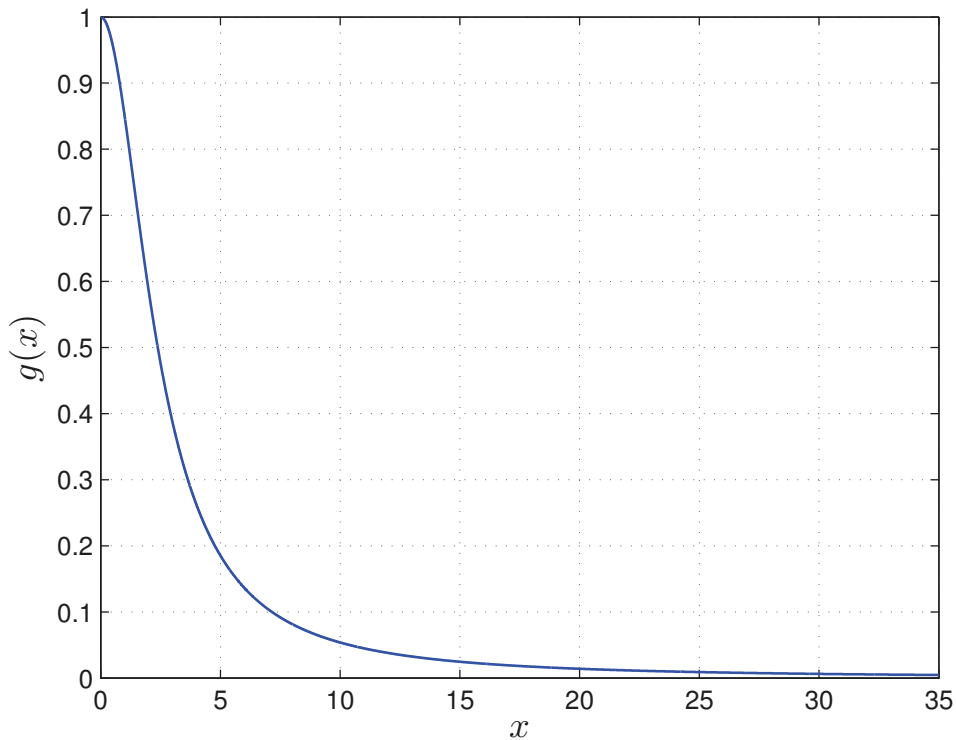


FIGURE 1.11: Cauchy weight function with $c = 2.3849$.

1.3.2 3D Shape Recognition

Recent advances in 3D imaging and processing, graphics hardware and networks have led to a whopping increase in geometry models available freely or commercially on the Internet. As a result, the task of efficiently measuring the 3D object similarity to find and retrieve relevant objects for a given query and categorize an object into one of a set of classes has become of paramount importance in a wide range applications, including computer-aided design, video gaming, special effects and film production, medicine, and archaeology. The main challenge in 3D object retrieval algorithms is to compute an invariant shape descriptor that captures well the geometric and topological properties of a shape [12–16].

The vast majority of 3D shape representation techniques proposed in the literature of computer graphics and computer vision are primarily based on geometric and topological representations which represent the features of an object [17–19]. For example, Siddiqi *et al.* [18] introduced a shock detection approach based on singularity theory to generate a skeletal shape model. Also, Siddiqi *et al.* [20] proposed a directed acyclic graph representation for 3D retrieval using medial

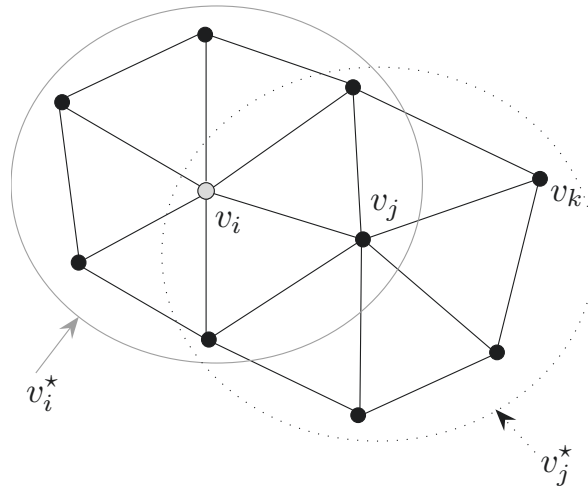


FIGURE 1.12: Illustration of two neighboring rings.

surfaces. This approach uses the geometric information associated with each graph node along with an eigenvalue labeling of the adjacency matrix of the subgraph rooted at that node. Cornea *et al.* [21] devised a 3D matching framework for 3D volumetric objects using a many-to-many matching algorithm. This algorithm is based on establishing correspondences among two skeletal representations via distribution-based matching in metric spaces. Hassouna *et al.* [22] proposed a level set based framework for robust centerline extraction of 2D shapes and 3D volumetric objects. This approach is based on the gradient vector flow and uses a wave propagation technique, which identifies the curve skeletons as the wave points of maximum positive curvatures. Tagliasacchi *et al.* [23] introduced a curve skeleton extraction algorithm from imperfect point clouds. A major drawback of curve skeletons is that they cannot capture general shape features such as surface ridges, and are essentially restricted to objects which resemble connected tubular forms.

An alternative to feature-based representations are global methods, which represent a 3D object by a global measure or shape distribution defined on the surface of the object [24–26]. Ankerst *et al.* [24] used shape histograms to analyze the similarity of 3D molecular surfaces. These histograms are built from uniformly distributed surface points taken from the molecular surfaces, and are defined on concentric shells and sectors around the centroid of the surface. Osada *et al.* [25] proposed a global approach for computing shape signatures of arbitrary 3D models. The key idea is to represent an object by a global histogram based on the Euclidean distance defined on the surface of an object. More recently, Ion *et al.* [27] presented an articulation-insensitive shape match-

ing approach by constructing histograms from the eccentricity transform using geodesic distances. Kazhdan *et al.* [26] proposed a rotation invariant spherical harmonic representation that transforms rotation dependent shape descriptors into rotation independent ones. Chen *et al.* [28] presented a lightfield descriptor for 3D object retrieval by comparing ten silhouettes of the 3D shape obtained from ten viewing angles distributed uniformly on the viewing enclosing sphere. The dissimilarity between two shapes is computed as the minimal distance obtained by rotating the viewing sphere of one lightfield descriptor relative to the other lightfield descriptor. The computation of this descriptor is, however, significantly time consuming compared to spherical harmonics [29].

The recently emerging field of diffusion geometry provides a generic framework for many methods in the analysis of geometric shapes [30]. It formulates the heat diffusion processes on manifolds. Spectral shape analysis is a methodology that relies on the eigensystem (eigenvalues and/or eigenfunctions) of the Laplace-Beltrami operator to compare and analyze geometric shapes. The Laplace-Beltrami operator on manifolds is the analogous of the Laplace operator on Euclidean spaces.

1.4 Thesis Overview and Contributions

The thesis contributions may be summarized as follows:

- In Chapter 2, we propose a regularized kernel diffusion filter for 3D mesh denoising in the weighted graph framework. The proposed approach is able to reduce the over-smoothing effect and effectively remove undesirable noise while preserving prominent geometric features of a 3D mesh such as curved surface regions, sharp edges, and fine details. Illustrating experimental results are presented to show the effectiveness of the proposed approach.
- In Chapter 3, we introduce a surface denoising scheme using the vertex-centered finite volume method coupled with the mesh covariance fractional anisotropy. The approach is computationally fast and able to effectively remove undesirable noise while preserving prominent geometric features of a 3D mesh surface such as curved surface regions, sharp edges, and fine details. Extensive experimental results on various 3D models demonstrate the effectiveness of the proposed iterative algorithm, which yields satisfactory output results in just one single iteration.
- In Chapter 4, we propose a spectral skeleton-based approach for deformable 3D object matching and retrieval. This skeleton is constructed from the second eigenfunction of the Laplace-Beltrami operator defined on the surface of the 3D object, and thus it is invariant to isometric transformations. Moreover, our shape descriptor is both compact and efficient to compute. We also present a robust matching framework by comparing the shortest paths between the skeleton endpoints. The experimental results demonstrate the feasibility of the proposed approach in object matching and retrieval on two 3D shape benchmarks.
- In Chapter 5, we summarize the contributions of this thesis, and propose several future research directions that are directly or indirectly related to the ideas developed therein.

Regularized Kernel Weighted Diffusion

2.1 Introduction

Recent advances in computer and information technology have increased the use of 3D models in many fields including medicine, the media, art and entertainment. The great challenge in image processing and computer graphics is to devise computationally efficient and optimal algorithms for recovering images and 3D models contaminated by noise and preserving their geometrical structure. With the increasing use of scanners to create 3D models which are usually represented as triangle meshes in computer graphics and geometric-aided design, there is a rising need for robust and efficient 3D mesh denoising techniques to remove undesirable noise from the data.

Numerous partial differential equations (PDE)-based methods have been proposed in the literature to tackle the problem of 2D image denoising with a good preservation of features [31–36]. Much of the appeal of PDE-based methods lies in the availability of a vast arsenal of mathematical tools which at the very least act as a key guide in achieving numerical accuracy as well as stability. Partial differential equations or gradient descent flows are generally a result of variational problems [37]. The 3D mesh denoising problem, however, has received much less attention [1–5]. The widely used mesh smoothing method is the so-called Laplacian flow, which repeatedly and simultaneously adjusts the location of each mesh vertex to the geometric center of its neighboring

vertices [1]. Despite its simplicity, the Laplacian smoothing flow produces, however, the shrinking effect and an oversmoothing result. More recent mesh denoising techniques include the mean, median, and bilateral filters [6–8] which are all adopted from the image processing literature. Also, a number of anisotropic diffusion methods for triangle meshes and implicit surfaces have been proposed recently. The authors in [9, 38] introduced a weighted Laplacian smoothing technique by choosing new edge weights based on curvature flow operators. This denoising method avoids the undesirable edge equalization from Laplacian flow and helps to preserve curvature for constant curvature areas. However, re-computing new edge weights after each iteration results in more expensive computational cost. In [39], Clarenz *et al.* proposed a multiscale surface smoothing method based on the anisotropic curvature evolution problem. By discretizing nonlinear partial differential equations, this method aims to detect and preserve sharp edges by two user defined parameters which are a regularization parameter for filtering out high frequency noisy and a threshold for edge detection. This multiscale method was also extended to the texture mapped surfaces [40] in order to enhance edge type features of the texture maps. Different regularization parameters and edge detection threshold values, however, need to be defined by users onto noisy surfaces and textures respectively before the smoothing process. Bajaj *et al.* [41] presented a unified anisotropic diffusion for 3D mesh smoothing by treating discrete surface data as a discretized version of a 2-D Riemannian manifold and establishing a PDE diffusion model for such a manifold. This method helps enhancing sharp features while filtering out noise by considering 3-ring neighbors of each vertex to achieve non-linear approach of smoothing process. Tasdizen *et al.* [42, 43] introduced a two-step surface smoothing method by solving a set of coupled second-order PDEs on level set surface models. Instead of filtering the positions of points on a mesh, this method operates on the normal map of a surface and manipulates the surface to fit the processed normals. All the surfaces normals are processed by solving second-order equations using implicit surfaces. In [44], Hildebrandt *et al.* presented a mesh smoothing method by using a prescribed mean curvature flow for simplicial surfaces. This method develops an improved anisotropic diffusion algorithm by defining a discrete shape operator and principal curvatures of simplicial surfaces.

In this chapter, we propose a regularized weighted diffusion for 3D shape smoothing using the statistical concept of kernel density estimation. Kernel density estimates are output as smooth curves with the amount of smoothing governed by a bandwidth value used during calculation [45].

The rest of this chapter is organized as follows. In Section 2.2, a regularized kernel weighted diffusion is introduced. In Section 2.3, we provide experimental results to demonstrate a much improved performance of the proposed method in 3D mesh denoising.

2.2 Regularized Diffusion on Weighted Graphs

Inspired by continuous regularization of images, we present a general framework based on a discrete regularization on weighted graphs of arbitrary topology.

2.2.1 Differential Operators on Weighted Graphs

A weighted graph $G = (V, E, w)$ is defined by a set of vertices V , a set of edges $E = \{e\} \subseteq V \times V$, and a weight function $w : E \rightarrow \mathbb{R}_+$ that assigns nonnegative weights to the edges. Each edge $e = [u, v] \in E$ connects a pair of vertices $\{u, v\}$. Two distinct vertices $u, v \in V$ are adjacent or neighbors (written $u \sim v$) if they are connected by an edge, i.e. $[u, v] \in E$. The weight function is symmetric, that is $w(u, v) = w(v, u)$ for all $[u, v] \in E$. If two vertices u and v are not connected, then $w(u, v) = 0$. The degree function is defined as $d : V \rightarrow \mathbb{R}_+$ such that $d(u) = \sum_{v \sim u} w(u, v) = \sum_{v \in V} w(u, v)$ for all $u \in V$.

Denote by $\mathcal{H}(V)$ the Hilbert-space of real-valued functions equipped with the inner product

$$\langle \varphi_1, \varphi_2 \rangle_{\mathcal{H}(V)} = \sum_{v \in V} \varphi_1(v) \varphi_2(v), \quad (2.1)$$

where $\varphi_1, \varphi_2 : V \rightarrow \mathbb{R}$. The L^2 -norm of a function $\varphi : V \rightarrow \mathbb{R}$ is $\|\varphi\|_2^2 = \sum_{v \in V} |\varphi(v)|^2 d(v)$

Similarly, we denote by $\mathcal{H}(E)$ the Hilbert-space of real-valued functions equipped with the inner product

$$\langle \psi_1, \psi_2 \rangle_{\mathcal{H}(E)} = \sum_{(u,v) \in E} \psi_1(u, v) \psi_2(u, v) = \sum_{u \in V} \sum_{v \sim u} \psi_1(u, v) \psi_2(u, v), \quad (2.2)$$

where the vectors $\psi_1 = \psi_1(u, v) \in E$ and $\psi_2 = \psi_2(u, v) \in E$, and $\psi_1, \psi_2 : E \subseteq V \times V \rightarrow \mathbb{R}$.

Definition 2.2.1 *The directional derivative (or edge derivative), denoted by $(\nabla_w \varphi)(u, v)$ or $\partial_v \varphi(u)$, of a function $\varphi \in \mathcal{H}(V)$ at a vertex u , along an edge $e = (u, v)$, is defined as*

$$(\nabla_w \varphi)(u, v) = \sqrt{w(u, v)} (\varphi(v) - \varphi(u)), \quad (2.3)$$

The weighted graph gradient $\nabla_w \varphi(u) : V \rightarrow E$ at a vertex u is defined as the vector of all directional derivatives $(\nabla_w \varphi)(u, v)$ at u , that is

$$\nabla_w \varphi(u) = \{(\nabla_w \varphi)(u, v) : v \sim u\} = \{(\nabla_w \varphi)(u, v) : v \in V\}. \quad (2.4)$$

Definition 2.2.2 The graph divergence $\text{div}_w \psi(u) : E \rightarrow V$ is defined as the adjoint of the graph gradient

$$\langle \nabla_w \varphi, \psi \rangle_{\mathcal{H}(E)} = \langle \varphi, -\text{div}_w \psi \rangle_{\mathcal{H}(V)}, \quad \forall \varphi \in \mathcal{H}(V), \psi \in \mathcal{H}(E) \quad (2.5)$$

and it is given by

$$\text{div}_w \psi(u) = \sum_{v \sim u} \sqrt{w(u, v)} (\psi(u, v) - \psi(v, u)) = \sum_{v \in V} \sqrt{w(u, v)} (\psi(u, v) - \psi(v, u)). \quad (2.6)$$

Note that if $\varphi(u)$ is a constant for all $u \in V$, then Eq. (2.5) yields $\langle \varphi, -\text{div}_w \psi \rangle_{\mathcal{H}(V)} = 0$, that is

$$\sum_{u \in V} \text{div}_w \psi(u) = 0, \quad \forall \psi \in \mathcal{H}(E) \quad (2.7)$$

Definition 2.2.3 The weighted graph Laplacian is an operator $\Delta : \mathcal{H}(V) \rightarrow \mathcal{H}(E)$ given by

$$\Delta_w \varphi(u) = \frac{1}{2} \text{div}_w (\nabla_w \varphi(u)) = \sum_{v \in V} w(u, v) (\varphi(v) - \varphi(u)) \quad (2.8)$$

Note that a factor $1/2$ is used to be consistent with the definition of the standard Laplace operator.

The Laplacian is self-adjoint

$$\langle \Delta_w \varphi(u), \varphi(u) \rangle = \langle \varphi(u), \Delta_w \varphi(u) \rangle \quad (2.9)$$

and negative semidefinite

$$\langle \Delta_w \varphi(u), \varphi(u) \rangle = -\langle \nabla_w \varphi(u), \nabla_w \varphi(u) \rangle \quad (2.10)$$

The local H^1 norm and the total variational semi-norm of φ at u are defined as follows:

$$\begin{aligned} J_{LH^1}^w(\varphi) &= |\nabla_w \varphi(u)|^2 = \sum_{v \in V} w(u, v) (\varphi(v) - \varphi(u))^2 \\ J_{LTV}^w(\varphi) &= |\nabla_w \varphi(u)| = \sqrt{\sum_{v \in V} w(u, v) (\varphi(v) - \varphi(u))^2} \end{aligned} \quad (2.11)$$

The nonlocal H^1 norm and the total variation semi-norm of φ at u are defined as follows:

$$\begin{aligned} J_{NLH^1}^w(\varphi) &= \sum_{u \in V} |\nabla_w \varphi(u)|^2 = \sum_{u \in V} \sum_{v \in V} w(u, v) (\varphi(v) - \varphi(u))^2 \\ J_{NLTV}^w(\varphi) &= \sum_{u \in V} |\nabla_w \varphi(u)| = \sum_{u \in V} \left(\sum_{v \in V} w(u, v) (\varphi(v) - \varphi(u))^2 \right)^{1/2} \end{aligned} \quad (2.12)$$

The nonlocal TV semi-norm is analogous to the classical total variation in image processing.

2.2.2 Problem Statement

In all real applications, measurements are perturbed by noise. In the course of acquiring, transmitting or processing a 3D mesh for example, the noise-induced degradation may be dependent or independent of data. The noise is usually described by its probabilistic model, e.g., Gaussian noise is characterized by two moments. Application-dependent, a degradation often yields a resulting mesh observation model, and the most commonly used is the additive one,

$$\varphi_0 = \varphi + \eta, \quad (2.13)$$

where the observed data $\varphi_0 \in \mathcal{H}(V)$ includes the noise-free data $\varphi \in \mathcal{H}(V)$ and the independent and identically distributed (i.i.d) random noise process $\eta \in \mathcal{H}(V)$.

Mesh denoising refers to the process of recovering a mesh data contaminated by noise. Unknown prevailing statistics or underlying mesh/noise models often make a “target” desired performance quantitatively less well defined. Specifically, it may be qualitative in nature (e.g., preserve high gradients in a geometric setting, or determine a worst case noise distribution in a statistical estimation setting with a number of interpretations), and may not necessarily be tractably assessed by an objective and optimal performance measure. The formulation of such qualitative goals, is typically carried out by way of adapted functionals which upon being optimized, achieve the stated goal. This approach is the so-called *variational* approach, which enjoys a large arsenal of analytical tools and offers a greater flexibility. Generally, the construction of an energy functional is based on some characteristic quantity specified by the task at hand. This energy functional is oftentimes coupled to a regularizing force/energy in order to rule out a great number of solutions and to also avoid any degenerate solution.

When considering the signal model (2.13), our goal may be succinctly stated as one of estimating the underlying data φ based on an observation φ_0 and/or any potential knowledge of the

noise statistics to further regularize the solution. This yields the following fidelity-constrained optimization problem

$$\begin{aligned} \min_{\varphi} \quad & \mathcal{F}(\varphi) \\ \text{s.t.} \quad & \|\varphi - \varphi_0\|^2 = \sigma^2 \end{aligned} \quad (2.14)$$

where \mathcal{F} is a given functional which often defines, as noted above, the particular emphasis on the features of the achievable solution. In other words, we want to find an optimal solution that yields the smallest value of the objective functional among all solutions that satisfy the constraints. Using Lagrange's theorem, the minimizer of (2.14) is given by

$$\hat{\varphi} = \arg \min_{\varphi} \left\{ \mathcal{F}(\varphi) + \frac{\lambda}{2} \|\varphi - \varphi_0\|^2 \right\}, \quad (2.15)$$

where λ is a nonnegative regularization parameter chosen so that the constraint $\|\varphi - \varphi_0\|^2 = \sigma^2$ is satisfied. In practice, the parameter λ is often estimated or chosen *a priori*. Note that the right hand side of Eq. (2.15) consists of two terms: a smoothness term and a fitting term. The smoothness term seeks a function that is smooth, whereas the fitting term is used to find a smoothed function close enough to initial function φ_0 . Unlike, the smoothness term or regularizer consists in seeking for a function which is smooth. The regularization parameter λ specifies the trade-off between these two competing terms.

A critical issue, however, is the choice of the functional \mathcal{F} , which is often driven by geometric arguments. A generalization of the nonlocal H^1 and total variation is given by the functional

$$\mathcal{F}(\varphi) = \sum_{u \in V} F(|\nabla_w \varphi(u)|) = \sum_{u \in V} F \left(\left(\sum_{v \in V} w(u, v) (\varphi(v) - \varphi(u))^2 \right)^{1/2} \right) \quad (2.16)$$

where $F : \mathbb{R}^+ \rightarrow \mathbb{R}$ is a given smooth function. Using (2.20), we hence define a functional

$$\begin{aligned} \mathcal{L}(\varphi) &= \mathcal{F}(\varphi) + \frac{\lambda}{2} \|\varphi - \varphi_0\|^2 \\ &= \sum_{u \in V} F(|\nabla_w \varphi(u)|) + \frac{\lambda}{2} \|\varphi - \varphi_0\|^2, \end{aligned} \quad (2.17)$$

which by the formulation in (2.15) becomes

$$\hat{\varphi} = \arg \min_{\varphi} \mathcal{L}(\varphi). \quad (2.18)$$

To solve the optimization problem (2.18), a variety of iterative methods such as gradient descent or fixed point method may be applied. The first-order necessary condition to be satisfied by any

minimizer of the functional \mathcal{L} given by (2.17) is that its first variation vanishes. The first variation of $\mathcal{L}(\varphi)$ with respect to φ is given by

$$\partial_\varphi \mathcal{L}(\varphi) = \partial_\varphi \mathcal{F}(\varphi) + \lambda(\varphi(u) - \varphi_0(u))$$

where

$$\partial_\varphi \mathcal{F}(\varphi) = -\operatorname{div}_w \left(\frac{F'(|\nabla_w \varphi(u)|)}{|\nabla_w \varphi(u)|} \nabla_w \varphi(u) \right)$$

Thus, a minimizer of $\mathcal{L}(\varphi)$ may be interpreted as the steady state solution to the following regularized gradient descent flow

$$\begin{aligned} \frac{\partial \varphi(u)}{\partial t} &= \operatorname{div}_w \left(g(|\nabla_w \varphi(u)|) \nabla_w \varphi(u) \right) + \lambda(\varphi_0(u) - \varphi(u)) \\ &= \sum_{v \in V} \sqrt{w(u, v)} (\varphi(v) - \varphi(u)) \left(g(|\nabla_w \varphi(u)|) + g(|\nabla_w \varphi(v)|) \right) + \lambda(\varphi_0(u) - \varphi(u)) \end{aligned} \quad (2.19)$$

where $g(z) = F'(z)/z$, with $z > 0$.

The solution to the regularized gradient descent flow given by Eq. (2.19) leads to a family of nonlinear filters, parameterized by the weight function and the regularization parameter. There are basically two main advantages behind using this framework. Firstly, the regularization is formulated directly in discrete setting. Secondly, filters are computed by simple and efficient iterative algorithms, without solving any PDEs. Since the proposed approach is general, any discrete data set can be transformed into a weighted graph, by using a similarity measure between data. Thus, we can consider any function defined on the data as a function defined on the set of vertices of the weighted graph.

2.2.3 Illustrative Case

An interesting functional is the nonlocal p -Dirichlet given by

$$\mathcal{F}(\varphi) = \frac{1}{p} \sum_{u \in V} |\nabla_w \varphi(u)|^p = \frac{1}{p} \sum_{u \in V} \left(\sum_{v \in V} w(u, v) (\varphi(v) - \varphi(u))^2 \right)^{p/2} \quad (2.20)$$

which includes as special cases the nonlocal H^1 norm and the total variation semi-norm when $p = 2$ and $p = 1$, respectively [46]. Thus, the regularized gradient descent flow associated to the nonlocal p -Dirichlet is given by

$$\frac{\partial \varphi(u)}{\partial t} = \Delta_w^p \varphi(u) + \lambda(\varphi_0(u) - \varphi(u)) \quad (2.21)$$

where $\Delta_w^p \varphi(u)$ denotes the weighted p -Laplacian given by

$$\Delta_w^p \varphi(u) = \sum_{v \in V} \sqrt{w(u, v)} (\varphi(v) - \varphi(u)) \left(|\nabla_w \varphi(u)|^{p-2} + |\nabla_w \varphi(v)|^{p-2} \right) \quad (2.22)$$

and the parameter $p \in (0, +\infty)$ represents the smoothness degree.

Note that $|\nabla_w \varphi|$ is not differentiable when $\nabla_w \varphi = 0$ (e.g. when $p < 2$). To overcome the resulting numerical difficulties, we use the following slight modification

$$|\nabla_w \varphi|_\epsilon = \sqrt{|\nabla_w \varphi|^2 + \epsilon},$$

where ϵ is positive sufficiently small.

2.2.4 Choice of Weights

A. General weight function:

Let $\varphi \in \mathcal{H}(V)$ be a function defined on each vertex of the set of mesh vertices V . Similarities between data points are estimated by comparing their features. Features generally depend on the function φ and the vertex set V . Each vertex $v \in V$ is assigned a feature vector, denoted by $F_\varphi(v) \in \mathbb{R}^q$, where q is the dimension of the feature vector.

$$\omega(u, v) = \exp \left(-\frac{\|F_\varphi(v) - F_\varphi(u)\|^2}{2\sigma_d^2} \right), \quad \forall [u, v] \in E, \quad (2.23)$$

where σ_d is a parameter that depends on the variations of $\|u - v\|$ and $\|F_\varphi(v) - F_\varphi(u)\|$ over the weighted graph. The graph structure, associated with one of the above weight functions, describes a general family of filters. This family is linked to several filters defined in the context of image and mesh processing. Consider for example the case of p -Laplacian: when $p = 2$, the filter associated with the weight function given by Eq. (2.23) is equivalent to the bilateral filter, introduced in the context of image denoising [47, 48]. It is a nonlinear filter that combines geometric and range filtering. Bilateral filtering is also used to denoise meshes [7]. It is obtained by using the scalar feature $F_\varphi(v) = \varphi(v)$ for all $v \in V$. Using the same parameters, the filter can also be considered as a discrete nonlocal mean filter, introduced in the context of images [49].

B. Composed weight function:

The weight function $\omega(u, v)$ associated to a weighted graph provides a measure of distance between its vertices that can simply incorporate local, semi-local or nonlocal features according to the

topology of the graph and the mesh. Consider the following nonlocal weight function given by

$$\omega^L(u, v) = \omega(u, v) \exp\left(-\frac{\|F(\varphi_0, v) - F(\varphi_0, u)\|^2}{h^2}\right), \quad \forall [u, v] \in E, \quad (2.24)$$

In addition to the difference between values, $\omega^L(u, v)$ includes a similarity estimation of the compared features by measuring a L^2 -distance between the patches around the vertices u and v . The functions $F(\varphi_0, u)$ and $F(\varphi_0, v)$ can represent for example the area or fractional anisotropy values of u and v , respectively.

2.2.5 Kernel Weighted Diffusion

Kernel density estimates are output as smooth curves with the amount of smoothing governed by a bandwidth value used during calculation [45]. Densities are calculated by placing kernels over the distribution of data points. Kernels that overlap one another increase density values in shared areas of the distribution. For univariate and multivariate data, the Gaussian kernel is the most commonly used one. In particular, for 3D data, the standardized Gaussian kernel function (see Figure 2.1) is given by

$$K(\mathbf{x}) = \frac{1}{(2\pi)^{\frac{3}{2}}} \exp\left(-\frac{\|\mathbf{x}\|^2}{2}\right), \quad \forall \mathbf{x} \in \mathbb{R}^3. \quad (2.25)$$

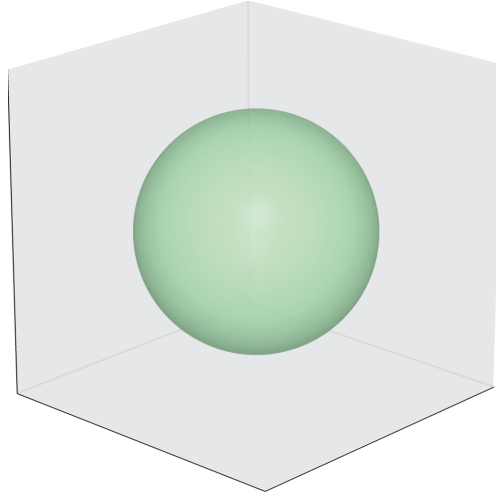


FIGURE 2.1: 3D level surface of the Gaussian kernel function $K(\mathbf{x})$.

Motivated by kernel density estimation as an important data analytic tool that provides a very effective way of showing structure in a set of a data [45, 50], we propose a kernel weight function

given by

$$\omega_{ij} = \mathcal{K}_{H_i}(\mathbf{v}_i - \mathbf{v}_j), \quad (2.26)$$

where

$$\mathcal{K}_{H_i}(\mathbf{v}_i - \mathbf{v}_j) = \det(H_i)^{-1/2} K(H_i^{-1/2}(\mathbf{v}_i - \mathbf{v}_j)) \quad (2.27)$$

and H_i is a symmetric positive semi-definite matrix. This matrix defines a covariance matrix around the neighborhood of vertex \mathbf{v}_i , and it is given by

$$H_i = \sum_{j \sim i} (\mathbf{v}_j - \mathbf{c}_i)(\mathbf{v}_j - \mathbf{c}_i)^T, \text{ where } \mathbf{c}_i = \frac{1}{d_i} \sum_{j \sim i} \mathbf{v}_j. \quad (2.28)$$

It is worth pointing out that H_i is also called the bandwidth matrix in the context of kernel smoothing and it measures the amount of smoothing. Also, note that the choice of the kernel function appears to have very little effect on the quality of the proposed denoising approach. However, the selection of the bandwidth matrix is widely recognized to have more effect on the performance of kernel density estimation. In this proposed method, we use the trivariate Gaussian density as a kernel function, and the data-driven neighborhood covariance as a bandwidth matrix.

The neighborhood weighting kernel \mathcal{K}_{H_i} may be expressed in matrix form as $\mathcal{K} = (\omega_{ij})$, which will be referred to as mesh neighborhood weighting kernel matrix. Each element ω_{ij} of this $m \times m$ sparse matrix is given by the right-hand side of Eq. (2.27). Thus, the mesh neighborhood weighting kernel matrix may be written as

$$\mathcal{K} = (\omega_{ij}) = \begin{cases} \det(H_i)^{-1/2} (2\pi)^{-3/2} & \text{if } i = j \\ \det(H_i)^{-1/2} K(H_i^{-1/2}(\mathbf{v}_i - \mathbf{v}_j)) & \text{if } i \sim j \\ 0 & \text{o.w.} \end{cases} \quad (2.29)$$

Note that the value of the the first row of Eq. (2.29) results directly from Eq. (2.25) when $\mathbf{x} = 0$, that is $K(\mathbf{0}) = (2\pi)^{-3/2}$. Figure 2.2 displays a 3D airplane and its mesh neighborhood weighting kernel matrix.

Next we show how kernel density estimation can be used for 3D mesh reconstruction. Given m mesh vertices \mathbf{v}_i , let \mathbf{v} be a 3D vector whose i -th realization is \mathbf{v}_i . Thus, the mesh kernel density estimate (KDE) may be written in the general form

$$\hat{f}(\mathbf{v}) = \frac{1}{m} \sum_{i=1}^m \det(H)^{-1/2} K(H^{-1/2}(\mathbf{v} - \mathbf{v}_i)) = \frac{1}{m} \sum_{i=1}^m \mathcal{K}_H(\mathbf{v} - \mathbf{v}_i), \quad (2.30)$$

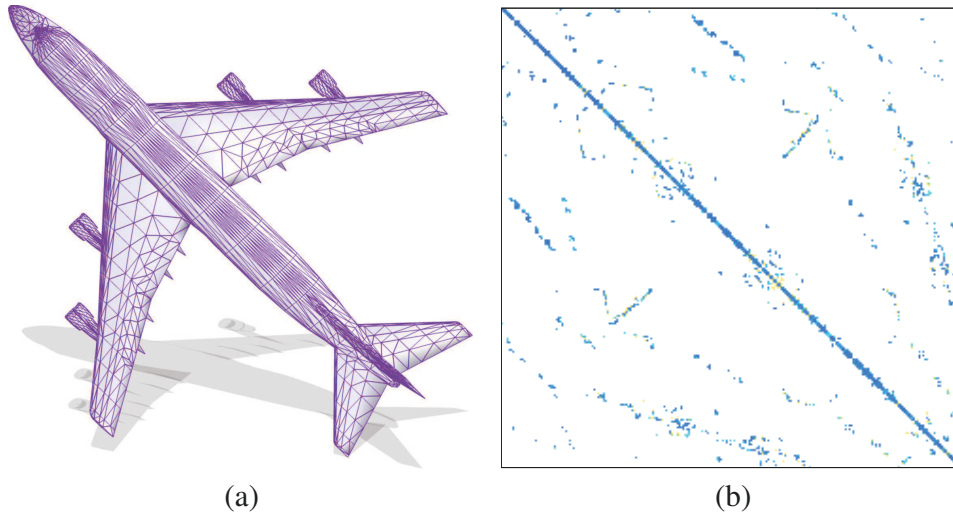


FIGURE 2.2: (a) 3D object with $m = 3403$ vertices and its (b) mesh neighborhood weighting kernel matrix.

where $H = \sum_{i=1}^m (\mathbf{v}_i - \mathbf{c})(\mathbf{v}_i - \mathbf{c})^T$ is the mesh covariance matrix which controls the smoothness of the resulting density estimate, and $\mathbf{c} = (1/m) \sum_{i=1}^m \mathbf{v}_i$ is the mesh centroid. The mesh KDE is a trivariate volumetric function which may be graphically visualized by plotting a level surface (also called implicit surface or isosurface) of \hat{f} as shown in Figure 2.3(b). This figure displays an isosurface of the mesh KDE using the vertices of the 3D object shown in Figure 2.3(a). The horizontal slices of the mesh KDE are also depicted in Figure 2.3(c).

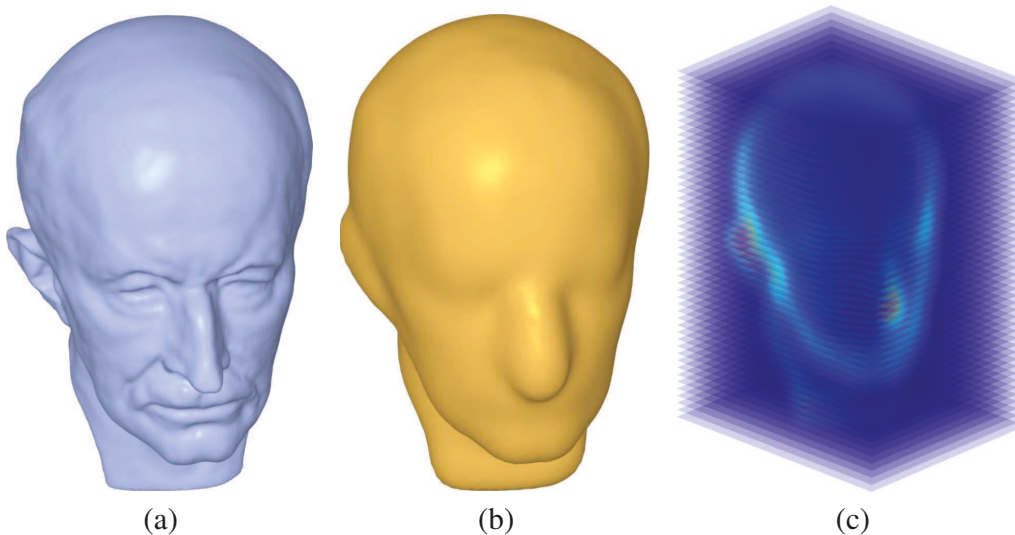


FIGURE 2.3: (a) 3D object; (b) Mesh KDE; (c) Horizontal slices of the mesh KDE.

2.3 Experimental Results

In this section, experimental results are provided to show the effectiveness of the regularized kernel diffusion in 3D surface denoising. The standard deviation of the noise was set to 40% of the mean edge length, that is $\sigma = 0.4 \bar{\ell}$. More precisely, a vertex \mathbf{v}_i of a noisy mesh is given by the additive random noise model:

$$\mathbf{v}_i = \mathbf{u}_i + \sigma(\boldsymbol{\eta}_i \circ \mathbf{n}_i) \quad (2.31)$$

where $\boldsymbol{\eta}_i$ are i.i.d. Gaussian random vectors (i.e. $\boldsymbol{\eta}_i$ is a 3-dimensional vector containing pseudo-random values drawn from the standard normal distribution $N(0, 1)$), \mathbf{n}_i is the unit normal vector at the noise-free vertex \mathbf{u}_i , and \circ denotes the Hadamard product between two vectors (i.e. the elements of the vector $\boldsymbol{\eta}_i \circ \mathbf{n}_i$ are obtained via element-by-element multiplication of the vectors $\boldsymbol{\eta}_i$ and \mathbf{n}_i).

In practical applications, the covariance matrix H_i given by Eq. (2.28) may become singular. To circumvent this singularity problem and also to ensure the stability of the proposed algorithm, we use a regularized covariance matrix as follows

$$\tilde{H}_i = H_i + \kappa I \quad (2.32)$$

where I is a 3×3 identity matrix and κ is a positive fidelity parameter. The parameter κ is often chosen such that $\kappa > \min(\lambda_1, \lambda_2, \lambda_3)$, where λ_1, λ_2 , and λ_3 are the eigenvalues of the regularized covariance matrix \tilde{H}_i at each mesh vertex \mathbf{v}_i . The eigenvalues of \tilde{H}_i can be broadly classified into three possible cases as depicted in Figure 2.4:

- $\lambda_1 \approx \lambda_2 \approx \lambda_3$: degenerate case of a sphere.
- $\lambda_1 \approx \lambda_2 \gg \lambda_3$: oblate ellipsoid.
- $\lambda_1 \gg \lambda_2 \approx \lambda_3$: prolate ellipsoid.

Figure 2.5 illustrates the denoising results on a 3D horse model using different values of the fidelity parameter κ . Notice that the fidelity parameter should be tuned to be small enough to capture the intrinsic of 3D object and large enough not to recapture noise.

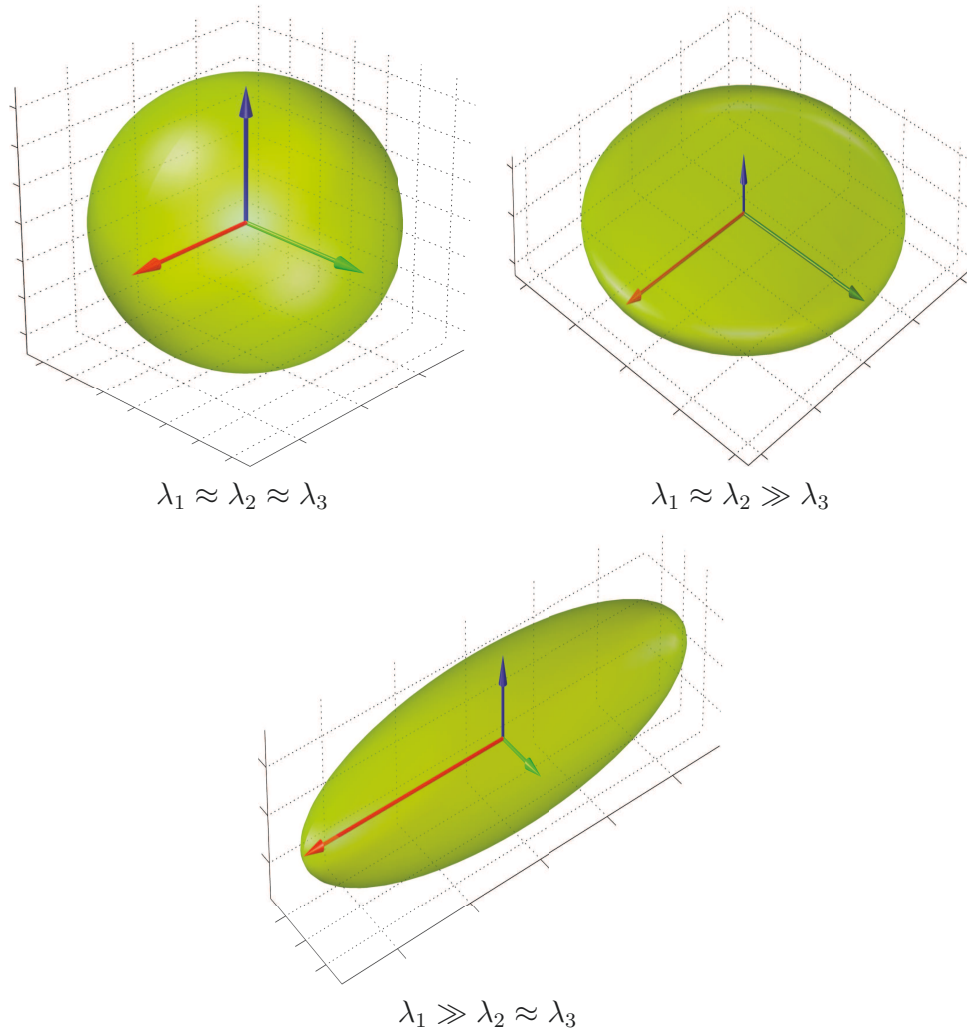


FIGURE 2.4: Different cases of diffusion: Spherical diffusion (left); planar diffusion (center); linear diffusion (right).

2.3.1 Output Results without Fitting Term

Figure 2.6 shows that the best smoothed model is obtained by the kernel weighted diffusion with $\kappa = 0.45$. For fair comparison, the number of iterations is set to 5 and the noise standard deviation is set to 1.5.

2.3.2 Output Results with Fitting Term

As shown in Figure 2.7, adding the fitting term further improve the better performance of kernel weighted diffusion with $\kappa = 0.45$ in comparison with the mean, angle median, Laplacian filter,

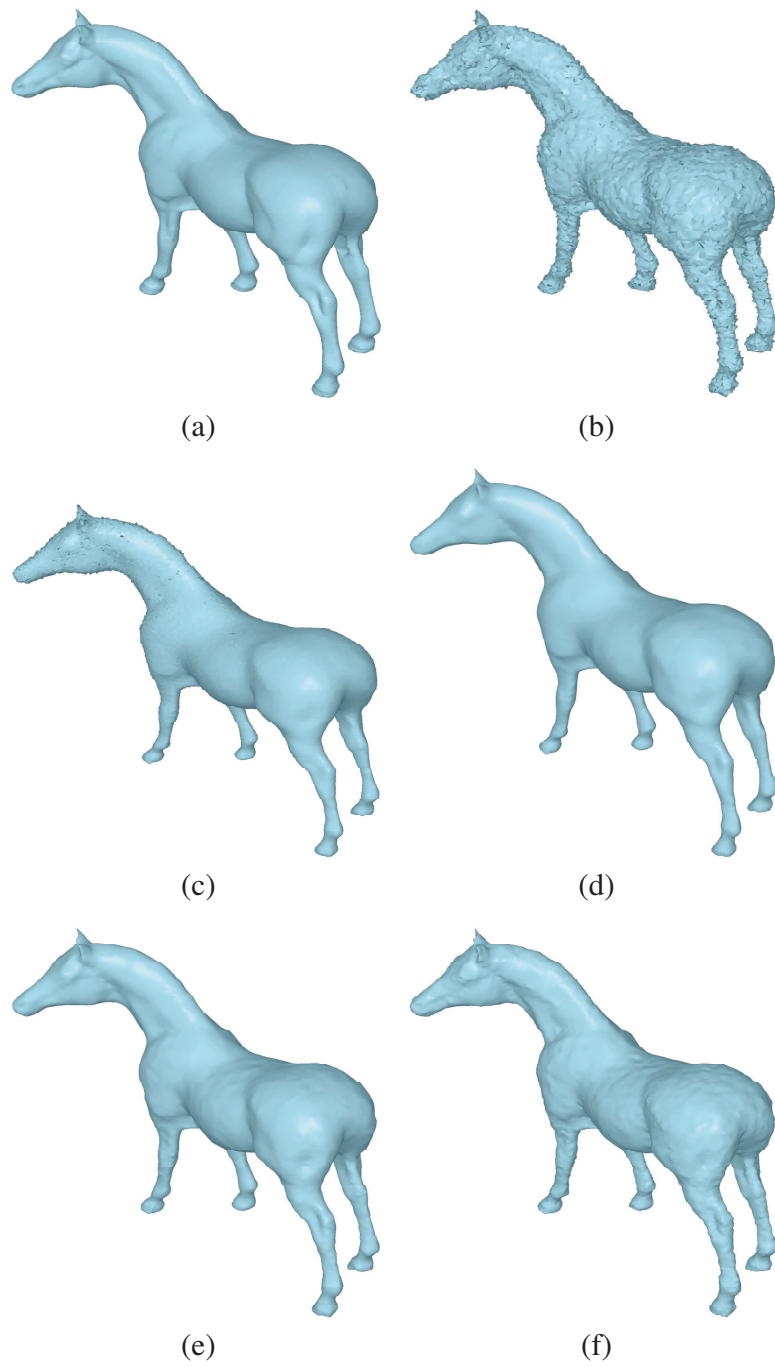


FIGURE 2.5: Output results without fitting term: (a) Original horse model; (b) Noisy Model; (c) $\kappa = 0.3$; (d) $\kappa = 0.4$; (e) $\kappa = 0.5$; (f) $\kappa = 0.6$.

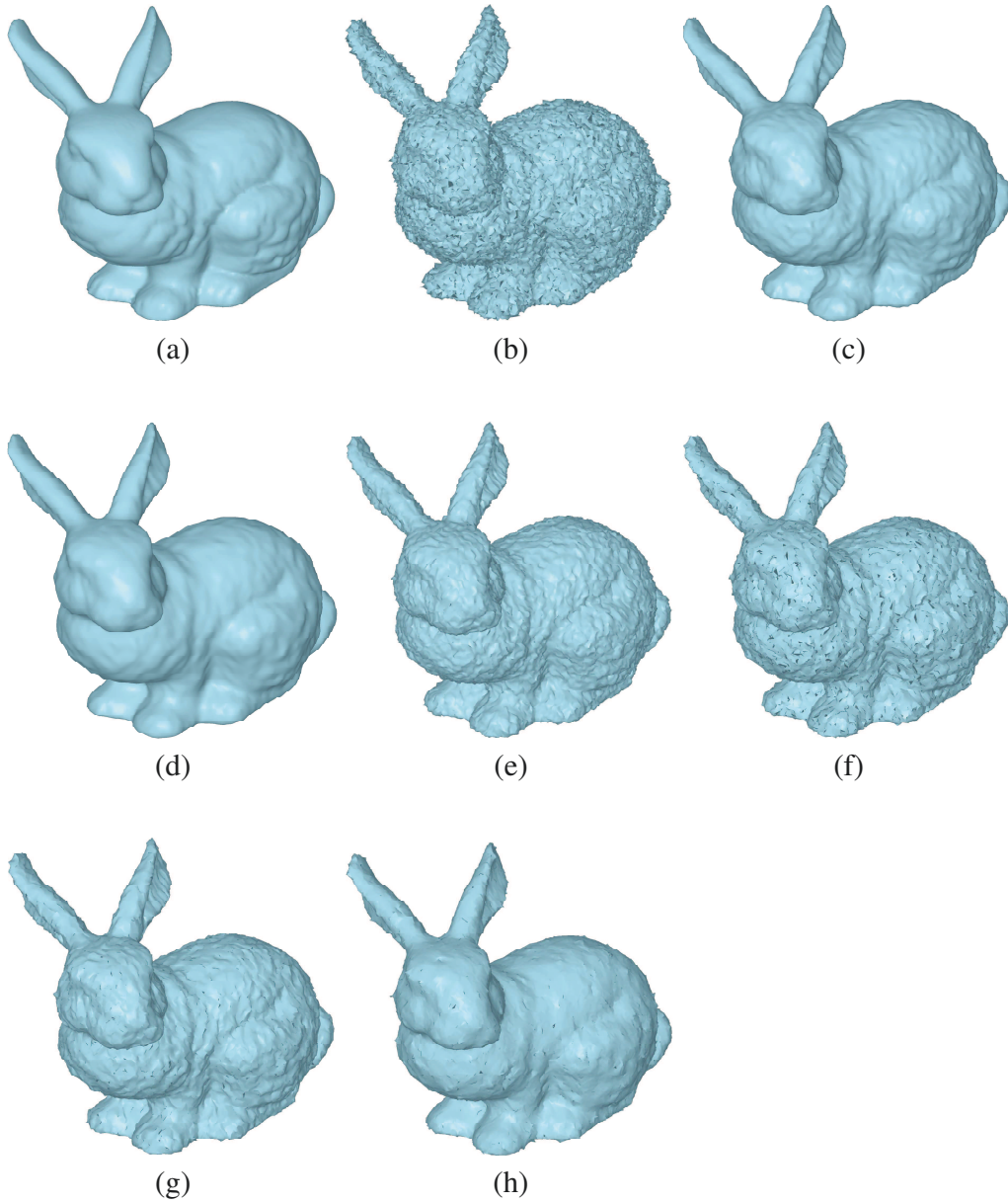


FIGURE 2.6: Output results without fitting term: (a) Original Model; (b) Noisy Model; (c) Uniform weighted diffusion; (d) Kernel weighted diffusion with $\kappa = 0.45$; (e) Laplacian filter; (f) Mean filter; (g) Angle median filter; (h) Bilateral filter.

uniform weighted diffusion and bilateral filter. As in the previous experiment, the number of iterations is also set to 5 and the noise standard deviation is set to 1.5.

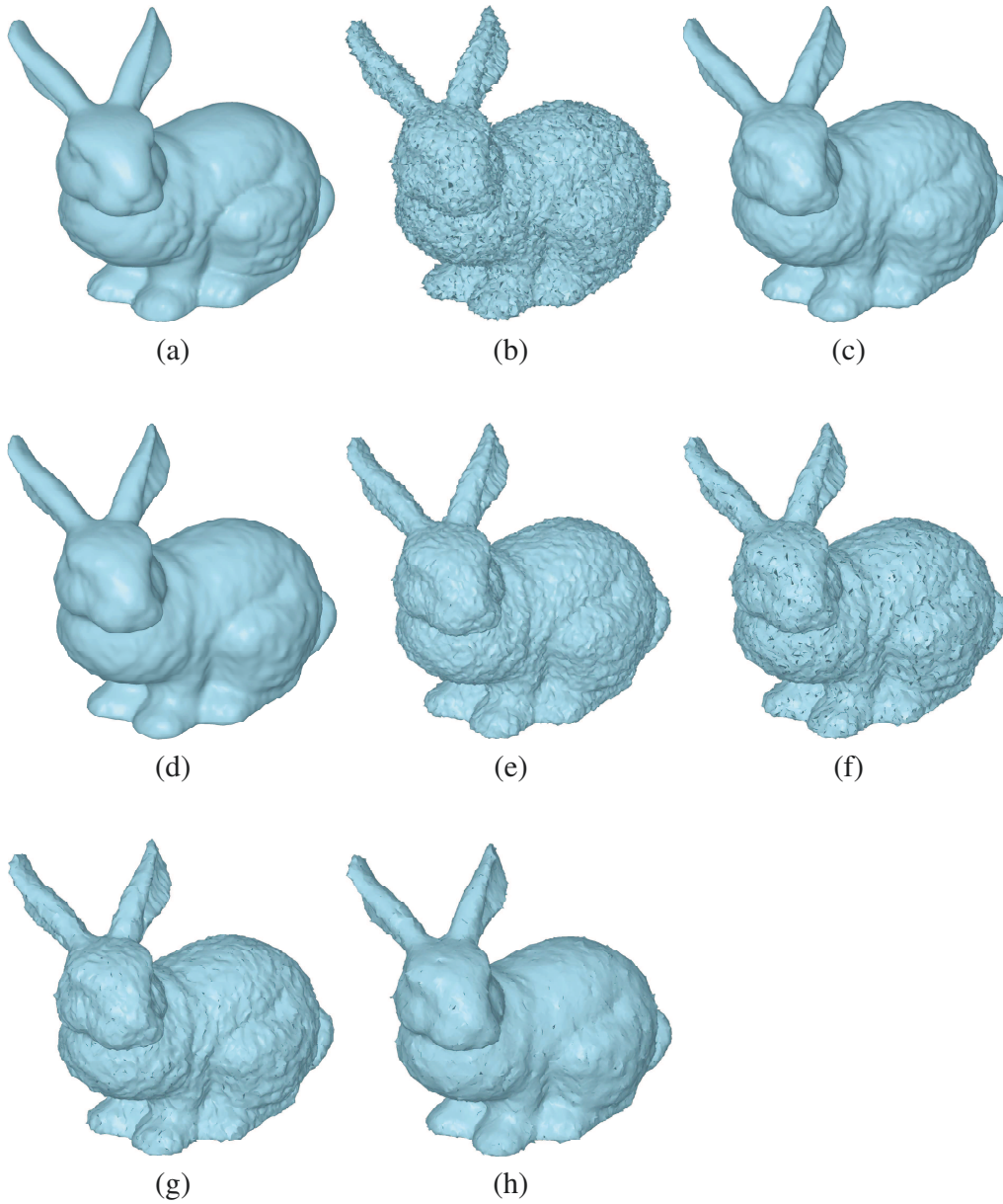


FIGURE 2.7: Output results with fitting term: (a) Original Model; (b) Noisy Model; (c) Uniform weighted diffusion; (d) Kernel weighted diffusion with $\kappa = 0.45$; (e) Laplacian filter; (f) Mean filter; (g) Angle median filter; (h) Bilateral filter.

2.3.3 Quantitative evaluation

Let $\mathbb{M} = (\mathcal{V}, \mathcal{T})$ and $\widehat{\mathbb{M}} = (\widehat{\mathcal{V}}, \widehat{\mathcal{T}})$ be the original model and the smoothing result model with vertices $\mathcal{V} = \{\mathbf{v}_i\}_{i=1}^m$ and $\widehat{\mathcal{V}} = \{\hat{\mathbf{v}}_i\}_{i=1}^m$. To quantify the performance of the improved approach in comparison with Laplacian, mean, angle median and bilateral filters methods, we computed the visual error metric [51] given by

$$E_2 = \frac{1}{2m} \left(\sum_{i=1}^m \|\mathbf{v}_i - \hat{\mathbf{v}}_i\|^2 + \sum_{i=1}^m \|\mathcal{I}(\mathbf{v}_i) - \mathcal{I}(\hat{\mathbf{v}}_i)\|^2 \right), \quad (2.33)$$

where \mathcal{I} is the geometric Laplacian operator defined as

$$\mathcal{I}(\mathbf{v}_i) = \mathbf{v}_i - \frac{1}{d_i} \sum_{j \sim i} \mathbf{v}_j. \quad (2.34)$$

Intuitively, the visual error captures the visual difference between the original model and the denoised one by taking into account geometric closeness and local smoothness difference. More specifically, the first term of this visual metric measures how close the vertices in both models are, whereas the second term captures the object smoothness which basically represents the visual properties of the human eye.

The values of visual error metric for some experiments are depicted in Figure 2.8 and Figure 2.9 which show that the proposed method gives the best results, indicating the consistency with the subjective comparison.

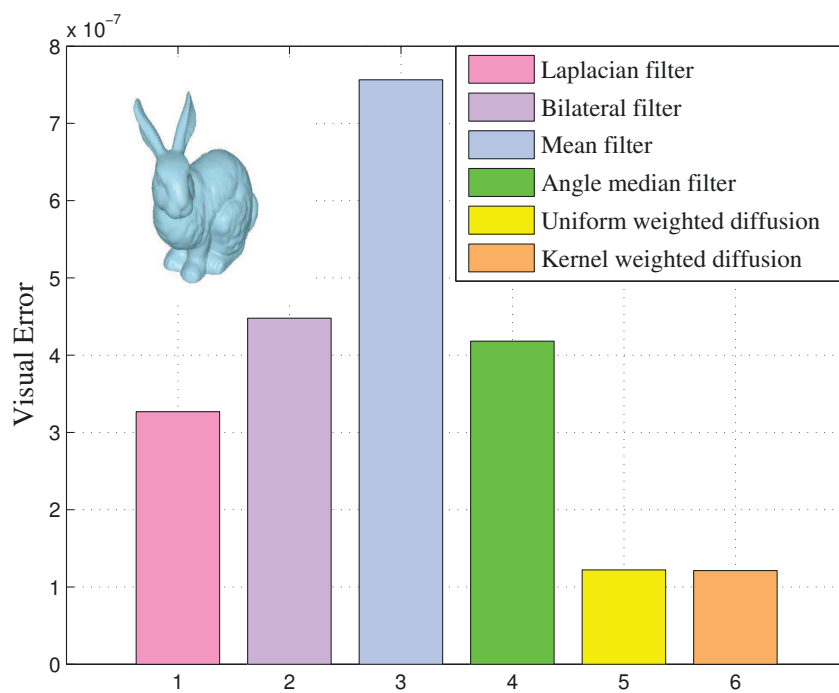


FIGURE 2.8: Quantitative visual errors without fitting term.

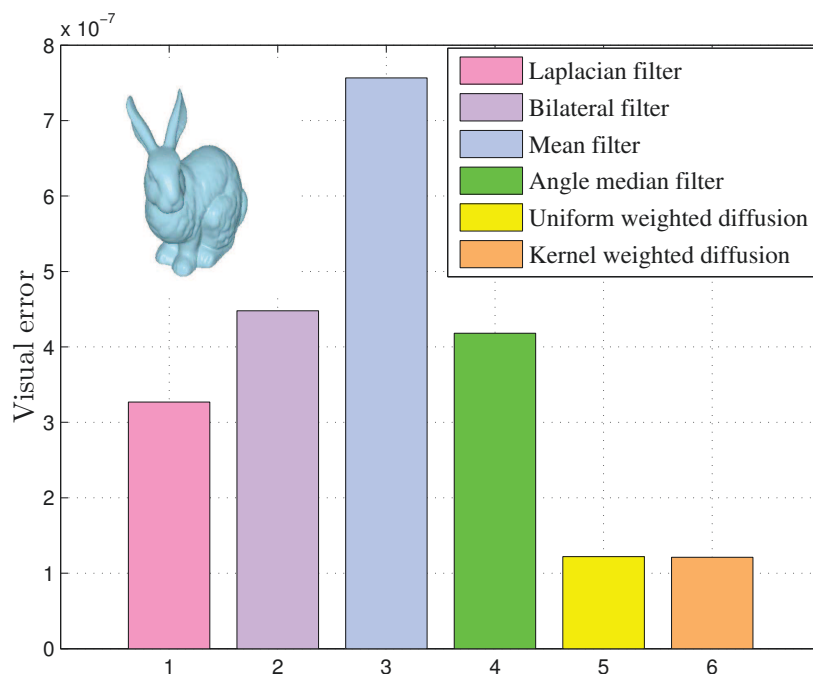


FIGURE 2.9: Quantitative visual errors with fitting term

Vertex-Centered Finite Volume Diffusion

3.1 Introduction

In light of the latest software, hardware and computing advancements, 3D technology has grown beyond being a buzzword. Today, 3D technology has become an essential part of the modern lifestyle and is gaining momentum rapidly, from consumer demand for in-home 3D television experiences to far-reaching positive implications for healthcare through the use of advanced 3D medical imaging systems aimed at improving patient outcomes and expanding their clinical practice. With the increasing use of 3D scanners to create 3D models, which are usually represented as triangle meshes, there is a rising need for robust mesh denoising techniques to remove inevitable noise in the measurements. Even with high-fidelity scanners, the acquired 3D models are usually contaminated by noise, and therefore a reliable mesh denoising technique is often required.

In recent years, a multitude of partial differential equations (PDEs)-based techniques have been proposed to tackle the 3D mesh denoising problem [1–5]. The most commonly used mesh denoising method is the Laplacian flow, which repeatedly and simultaneously adjusts the location of each mesh vertex to the geometric center of its neighboring vertices [1]. Although the Laplacian smoothing flow is simple and fast, it produces, however, a shrinking effect and an oversmoothing result. The most recent mesh denoising techniques include the mean, median, and bilateral filters [6–8]

which are all adopted from the image processing literature. Also, a number of anisotropic diffusion methods for triangle meshes and implicit surfaces have been proposed. Desbrun *et al.* [9, 38] introduced a weighted Laplacian smoothing technique by choosing new edge weights based on curvature flow operators. This mesh denoising method avoids the undesirable edge equalization from Laplacian flow and helps preserve curvature for constant curvature areas. Re-computing new edge weights after each iteration, however, results in a more expensive computational cost. Clarenz *et al.* [39] proposed a multiscale surface smoothing method based on the anisotropic curvature evolution problem. By discretizing nonlinear partial differential equations, this method aims at detecting and preserving sharp edges by two user-defined parameters which are a regularization parameter for filtering out high frequency noisy and a threshold for edge detection. This multiscale method was also extended to texture-mapped surfaces [40] in order to enhance edge-type features of the texture maps. Different regularization parameters and edge detection threshold values need, however, to be defined by the users onto noisy surfaces and textures respectively before the smoothing process. Bajaj *et al.* [41] presented a unified anisotropic diffusion for 3D mesh smoothing by treating discrete surface data as a discretized version of a 2D Riemannian manifold and establishing a partial differential equation diffusion model for such a manifold. This method helps enhance sharp features while filtering out noise by considering 3-ring neighbors of each vertex to achieve a non-linear approach of the smoothing process. Tasdizen *et al.* [42, 43] introduced a two-step surface smoothing method by solving a set of coupled second-order PDEs on level set surface models. Instead of filtering the positions of points on a mesh, this method operates on the normal map of a surface and manipulates the surface to fit the processed normals. All the surface normals are processed by solving second-order equations using implicit surfaces. Also, Hildebrandt *et al.* [44] proposed a mesh smoothing method by using a prescribed mean curvature flow for simplicial surfaces. This method develops an improved anisotropic diffusion algorithm by defining a discrete shape operator and principal curvatures of simplicial surfaces.

Roughly speaking, mesh denoising techniques can be defined as the requirement to adjust vertex positions without changing the connectivity of the 3D mesh, and may be broadly classified into two main categories: one-step and two-step approaches. The one-step approaches directly update vertex positions using the original vertex coordinates and a neighborhood around the current vertex, and also sometimes the face normals. The two-step approaches, on the other hand, first adjust face

normals and then update vertex positions using some error minimization criterion based on the adjusted normals. In many cases, a single pass of a one-step or two-step approach does not yield a satisfactory result, and therefore iterated operations are performed.

In this chapter, we present a 3D mesh denoising method based on the finite volume method combined with the mesh covariance fractional anisotropy. The finite volume method consists of associating a finite volume to each vertex of the mesh and applying the integral conservation law to this local volume. By a finite control volume, we mean a small volume or cell surrounding each mesh vertex. That is, the discretized space in the finite volume method is formed by a set of small cells, one cell being associated to one mesh vertex [52].

The technique proposed in this paper falls into the category of one-step approaches. The key idea of our mesh denoising approach is to use the dual mesh to iteratively update a mesh vertex according to a geometric flow defined in terms of the control volume centered around the vertex and weighted by the fractional anisotropy in order to remove the noise effectively while preserving the nonlinear features of the 3D mesh such as curved surface regions, sharp edges, and fine geometric details.

The rest of this chapter is organized as follows. In section 3.2, a surface denoising flow in the finite volume framework is introduced. In Section 3.3, we provide experimental results to demonstrate the denoising performance of our method on various 3D models.

3.2 Proposed Method

In this section we present a finite volume scheme for surface denoising.

3.2.1 Eigenanalysis of Mesh Covariance Matrix

Let \mathbf{v}_i^* be the neighborhood of a vertex \mathbf{v}_i , and denote by $\boldsymbol{\mu}_i$ the centroid of \mathbf{v}_i^* as shown in Figure 3.1, that is

$$\boldsymbol{\mu}_i = \frac{\sum_{j \sim i} a_j \mathbf{v}_j}{\sum_{j \sim i} a_j}, \quad (3.1)$$

where $a_j = \sum_{k \sim j} \text{area}(\mathbf{t}_k)$, $\mathbf{t}_k \in \mathcal{T}(\mathbf{v}_j^*)$.

We define the mesh covariance matrix C_i around the neighborhood of a vertex v_i as follows

$$C_i = \sum_{j \sim i} (\mathbf{v}_j - \boldsymbol{\mu}_i)(\mathbf{v}_j - \boldsymbol{\mu}_i)^T \quad (3.2)$$

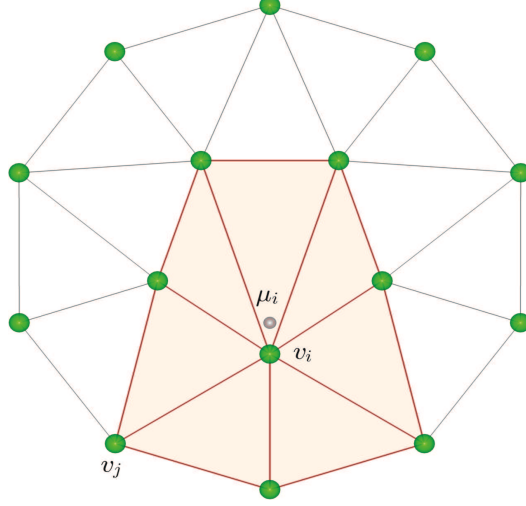


FIGURE 3.1: Illustration of the mesh covariance matrix around the neighborhood of a vertex v_i .

which is a symmetric and nonnegative definite matrix.

Using eigendecomposition, the mesh covariance matrix can be diagonalized to estimate the major, medium, and minor eigenvalues, λ_1 , λ_2 , and λ_3 , respectively; that is $\lambda_1 \geq \lambda_2 \geq \lambda_3$. The corresponding directions of these nonnegative eigenvalues are the major, medium, and minor orthonormal eigenvectors, \mathbf{e}_1 , \mathbf{e}_2 , and \mathbf{e}_3 , respectively. Thus, the spectral decomposition of covariance matrix is given by

$$C_i = \lambda_1 \mathbf{e}_1 \mathbf{e}_1^T + \lambda_2 \mathbf{e}_2 \mathbf{e}_2^T + \lambda_3 \mathbf{e}_3 \mathbf{e}_3^T$$

Note that for the sake of notational simplicity, we drop the index i from the eigenvalues and eigenvectors of C_i .

The local frame of the tangent space is given by $(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3)$, where \mathbf{e}_3 corresponds to the vertex normal \mathbf{n}_i , as shown in Figure 3.2, where the basis vectors of the local frame at each vertex of a 3D face model are displayed. The tangent plane at each mesh vertex is given by the equation: $(\mathbf{v}_i - \boldsymbol{\mu}_i) \cdot \mathbf{n}_i = 0$, where \mathbf{n}_i is the vertex normal. The sum of the eigenvalues, called total variation, is given by

$$\sum_{j \sim i} \|\mathbf{v}_j - \boldsymbol{\mu}_i\|^2 = \lambda_1 + \lambda_2 + \lambda_3 = \text{trace}(C_i), \quad i = 1, \dots, m$$

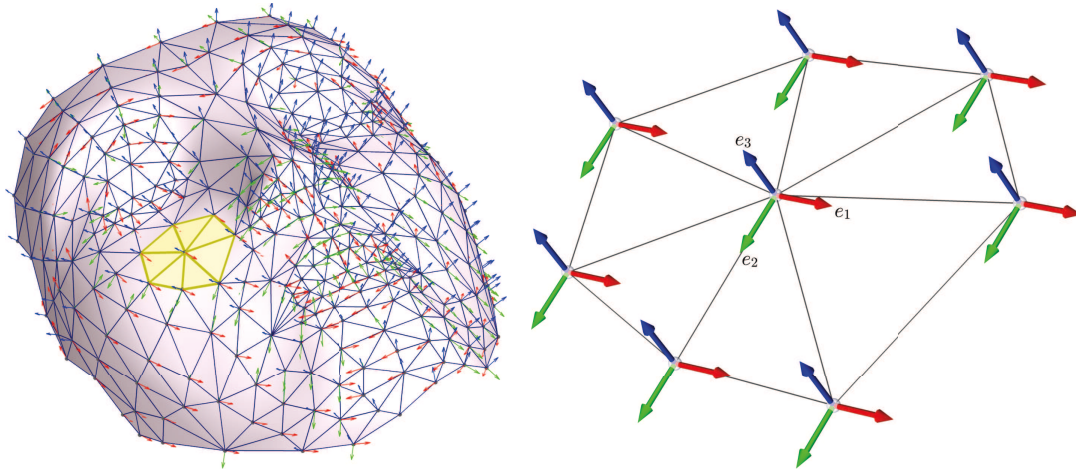


FIGURE 3.2: Left: 3D face model and the eigenvectors of its mesh covariance matrix. Right: close-up view.

The eigenvalues determine the shape of the neighborhood v_i^* of a vertex v_i . That is, the local neighborhood in which we estimate the covariance matrix C_i can be classified into three possible cases:

- *flat*: one eigenvalue λ_3 is very small or equal to zero, and the other two eigenvalues λ_1 and λ_2 have similar finite values, but significantly greater than λ_3 . The vertices in v_i^* are almost coplanar, and the eigenvectors e_1 and e_2 form a plane that fits the points. The vertex v_i belongs to a flat region.
- *edge*: two eigenvalues λ_2 and λ_3 are very small or equal to zero, and λ_1 has a finite greater value. The principal eigenvector e_1 identifies the direction along which the vertices in v_i^* are distributed. The vertex v_i is close to, or on, an edge and the eigenvector e_1 is the orientation of the edge.
- *corner*: all three eigenvalues have finite, nearly equal values. The point set within v_i^* is likely to be a corner.

Figure 3.3 shows the 3D face model colored by the eigenvalues of its mesh covariance matrix, where each vertex is assigned a true color $[\lambda_1, \lambda_2, \lambda_3]$.

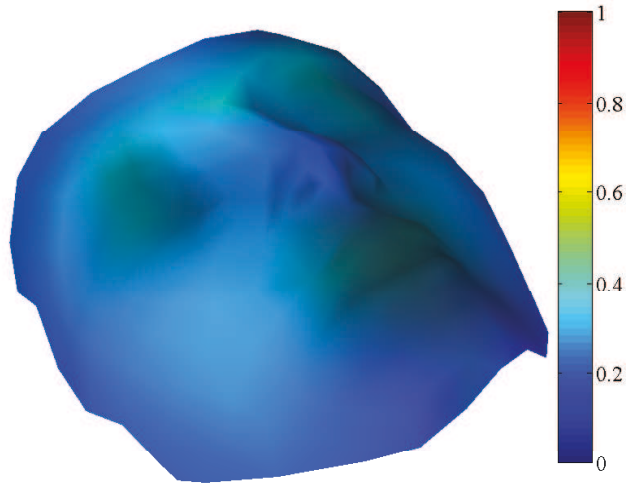


FIGURE 3.3: 3D face model colored by the eigenvalues of its mesh covariance matrix.

3.2.2 Finite Volumes of a Dual Mesh

Finite volume methods are discretization techniques that are well suited for the numerical simulation of various types of conservation laws, and have received considerable attention in various engineering fields, particularly in computational fluid dynamics. The surge in popularity of finite volumes is largely credited to their generality, simplicity, flexibility, and ease of implementation for arbitrary meshes. In addition, finite volume methods are close to the physics of the flow system due to their reliance on the direct discretization on the integral form of the conservation laws [52].

Finite volume methods can be broadly divided into two main classes: cell-centered and vertex-centered methods, as illustrated in Figure 3.4. In the cell-centered methods, the unknowns are associated with the control volumes, which are the mesh triangles. For example, any control volume (triangle) corresponds to a function value at some interior point such as the centroid of the triangle in the mesh as shown in Figure 3.5. In the vertex-centered methods, the unknowns are located at the vertices of the control volumes, which are the polygonal cells shown in Figure 3.5.

Our proposed mesh denoising approach is based on the vertex-centered finite volume methods. Thus, in order to associate to each vertex of a triangle mesh \mathbb{M} a control volume, we may need to construct a dual mesh, denoted by \mathbb{M}^* . The dual mesh is constructed by placing a vertex (typically

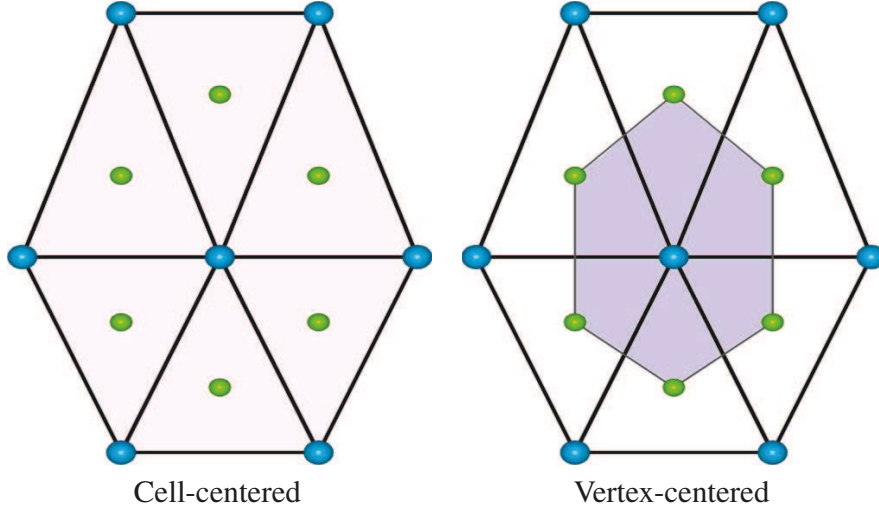


FIGURE 3.4: cell-centered vs. vertex-centered control volumes.

centroid) in each triangle t_k of \mathbb{M} , as shown in Figure 3.5. The centroid c_k is the average of the vertices of the triangle t_k . Notice that the vertices (resp. faces) of the dual mesh correspond to the triangles (resp. vertices) of \mathbb{M} . In other words, for any triangle mesh \mathbb{M} , there is a dual mesh \mathbb{M}^* , obtained by replacing each face (triangle) by a vertex and each vertex by a face, keeping the number of edges the same. Thus, each vertex in the mesh \mathbb{M} corresponds to a face in the dual; similarly, each triangle in \mathbb{M} corresponds to a vertex in the dual. The number of edges in both is the same, resulting in the same Euler characteristic $\chi(\mathbb{M}) = \chi(\mathbb{M}^*)$, which is a topological invariant that describes a topological space's shape or structure regardless of the way it is bent. An example of a 3D mesh and its dual mesh are shown in Figure 3.6.

3.2.3 Proposed Mesh Denoising Flow

In the sequel, we adopt the vertex-centered finite volume approach, where the unknowns are defined at the vertices of the (primal) mesh, and each cell of the dual mesh is a control volume associated to a vertex of the (primal) mesh. Referring to Figure 3.5, the control volume associated to the mesh vertex v_i consists of seven triangular cells having vertex v_i in common.

Let $t_j \in \mathcal{T}(v_i^*)$, the set of triangles of the ring v_i^* with cardinality d_i (i.e. $j = 1, \dots, d_i$), and denote by c_j the centroid of each triangle t_j , as illustrated in Figure 3.5. Denote by $\mathbf{r}_i = (r_{ij} : j \sim i)$ a vector of length d_i , where $r_{ij} = (\mathbf{v}_i - \mathbf{c}_j) \cdot \mathbf{e}_3$ is the scalar projection of $\mathbf{v}_i - \mathbf{c}_j$ onto the minor eigenvector \mathbf{e}_3 (i.e. onto the normal vector \mathbf{n}_i). A vector projection of $(\mathbf{v}_i - \mathbf{c}_j)$ onto \mathbf{n}_i is given by

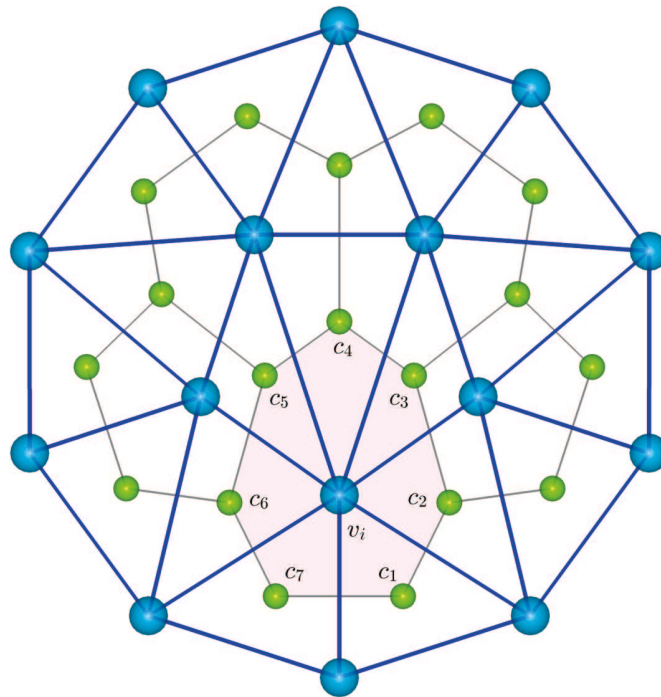
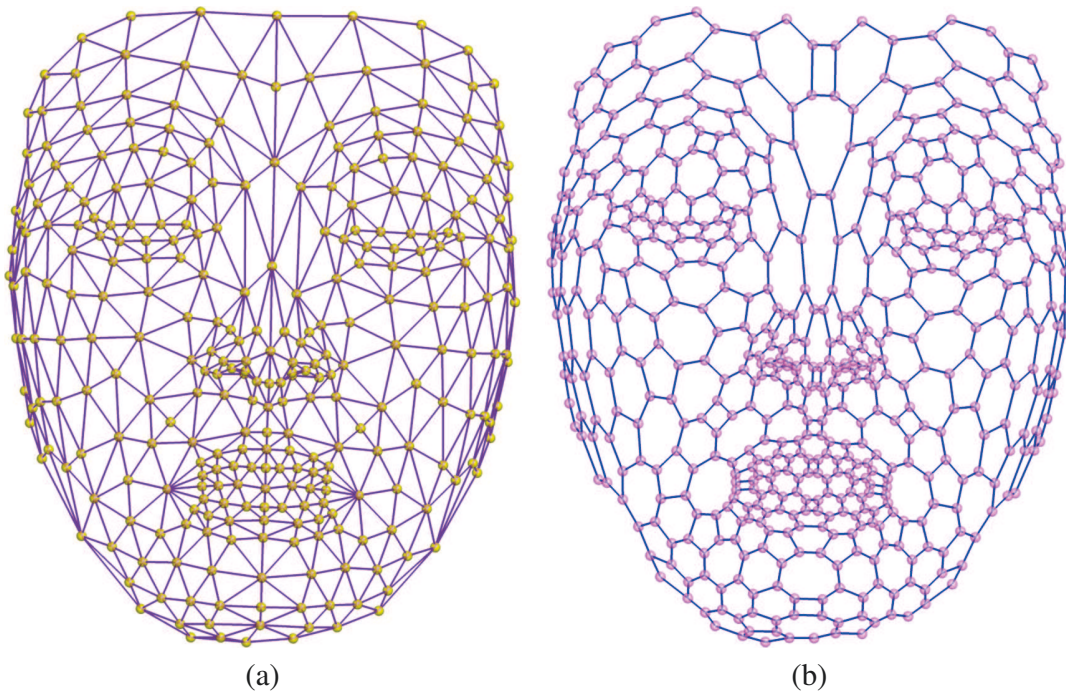


FIGURE 3.5: The dual mesh is shown in solid thin gray lines. Thick blue lines show the original mesh (also referred to as primal mesh).



(a)

(b)

FIGURE 3.6: (a) 3D face model; (b) dual mesh.

$((\mathbf{v}_i - \mathbf{c}_j) \cdot \mathbf{n}_i) \mathbf{n}_i = r_{ij} \mathbf{n}_i$, which measures the amount of $(\mathbf{v}_i - \mathbf{c}_j)$ in the direction of the minor eigenvector \mathbf{n}_i .

Motivated by the finite volume method which has a direct connection to the physical flow properties [53], we propose a 3D mesh denoising flow that updates iteratively each mesh vertex $\mathbf{v}_i, i = 1, \dots, m$, according to the following rule

$$\mathbf{v}_i \leftarrow \mathbf{v}_i - \frac{\sum_{j \sim i} \exp\left(-\frac{\|\mathbf{v}_i - \mathbf{c}_j\|^2}{2\hat{\sigma}_i^2}\right) (\text{FA})_i r_{ij} \mathbf{n}_i}{\sum_{j \sim i} \exp\left(-\frac{\|\mathbf{v}_i - \mathbf{c}_j\|^2}{2\hat{\sigma}_i^2}\right)} \quad (3.3)$$

where $(\text{FA})_i$ is the fractional anisotropy (usually denoted by FA) at each vertex \mathbf{v}_i defined by

$$(\text{FA})_i = \frac{\sqrt{(\lambda_1 - \lambda_2)^2 + (\lambda_2 - \lambda_3)^2 + (\lambda_1 - \lambda_3)^2}}{\sqrt{2(\lambda_1^2 + \lambda_2^2 + \lambda_3^2)}} \quad (3.4)$$

and the parameter $\hat{\sigma}_i$ is estimated using the concept of mean absolute deviation (MAD) from robust statistics [54] as

$$\begin{aligned} \hat{\sigma}_i &= 1.4826 \text{MAD}(\mathbf{r}_i) \\ &= 1.4826 \text{median}(\|\mathbf{r}_i - \text{median}(\|\mathbf{r}_i\|)\|) \end{aligned} \quad (3.5)$$

The fractional anisotropy (FA) is a measure of the variance of the eigenvalues and it is often used in the relatively recent magnetic resonance imaging modality called diffusion diffusion imaging (DTI) [55]. In DTI, each voxel is assigned a tensor (3×3 symmetric matrix) that describes local water diffusion. In other words, DTI measures the direction of local water diffusion in brain tissue (water diffuses more rapidly in the direction of the brain cell fibres). The value of the statistic FA is a scalar between zero and one that describes the degree of anisotropy of a diffusion process. A value of zero means that diffusion is isotropic, i.e. it is unrestricted (or equally restricted) in all directions. A value of one means that diffusion occurs only along one axis and is fully restricted along all other directions. It is worth pointing out that FA may be viewed as a normalized standard deviation of the eigenvalues. An essential advantage of using FA is that it can be computed without first explicitly computing the eigenvalues. In fact, FA can be expressed in terms of the Frobenius norm and the trace of the mesh covariance matrix as follows

$$(\text{FA})_i = \frac{\sqrt{3} \left\| C_i - \frac{1}{3} \text{trace}(C_i) I \right\|}{\sqrt{2} \text{trace}(C_i)} \quad (3.6)$$

where I is the 3×3 identity matrix. By definition, the Frobenius norm of a symmetric matrix is the square root of the sum of its squared elements which equals the square root of the sum of its squared eigenvalues, and the trace is the sum of the diagonal elements which equals the sum of the eigenvalues.

The eigenvectors and eigenvalues of C_i represent the principal axes of the ellipsoid and their corresponding principal diffusion coefficients, respectively. Therefore, the ellipsoid axes are oriented according to the eigenvectors, and their lengths depend on the associated eigenvalues.

Figure 3.7(a)-(b) show a noisy 3D bunny model and the output result of our method. The denoising results of our method colored by fractional anisotropy and mean curvature are also shown in Figure 3.7(c)-(d). Note that a considerable amount of noise has been removed in just one iteration.

3.3 Experimental Results

This section presents experimental results where the multiscale anisotropic Laplacian flow [56], and the proposed method are applied to noisy 3D models obtained by adding a random Gaussian noise to the original 3D models, along the normal vector at each vertex. The multiscale anisotropic Laplacian employs the anisotropic Laplacian operator combined with a roughness scale, and yields significantly better results than the anisotropic Laplacian and the bilateral filter. The experiments were performed on an iMac desktop computer with an Intel Core 2 Duo running at 2.93 GHz and 4 GB RAM; and the proposed mesh denoising algorithm was implemented in MATLAB 7.12 (R2011b).

The standard deviation of the noise was set to 40% of the mean edge length, that is $\sigma = 0.4 \bar{\ell}$, where $\bar{\ell}$ is the mean edge length. More precisely, a vertex \mathbf{v}_i of a noisy mesh is given by the additive random noise model:

$$\mathbf{v}_i = \mathbf{u}_i + \sigma(\boldsymbol{\eta}_i \circ \mathbf{n}_i), \quad (3.7)$$

where $\boldsymbol{\eta}_i$ are i.i.d. Gaussian random vectors (i.e. $\boldsymbol{\eta}_i$ is a 3-dimensional vector containing pseudo-random values drawn from the standard normal distribution $N(0, 1)$), \mathbf{n}_i is the unit normal vector at the noise-free vertex \mathbf{u}_i , and \circ denotes the Hadamard product between two vectors (i.e. the elements of the vector $\boldsymbol{\eta}_i \circ \mathbf{n}_i$ are obtained via element-by-element multiplication of the vectors $\boldsymbol{\eta}_i$ and \mathbf{n}_i).

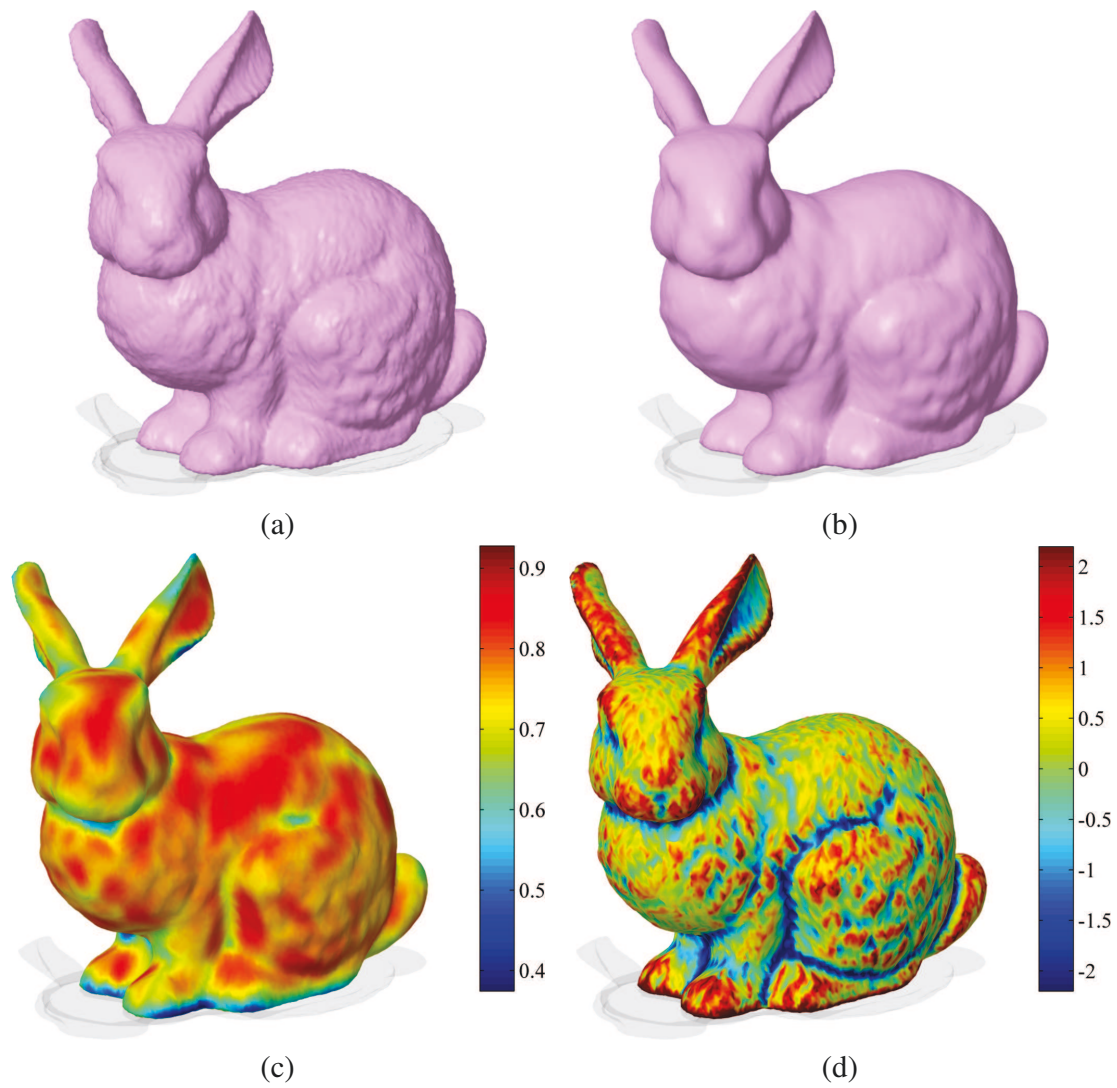


FIGURE 3.7: (a) Noisy 3D bunny model; (b) denoised model using our method; (c) output result of our method colored by fractional anisotropy; (d) output result of our method colored by mean curvature. The number of iterations is set to 1.

3.3.1 Qualitative evaluation of the proposed method

Figure 3.8 compares the denoising results obtained via multiscale anisotropic Laplacian flow and our method on a 3D hand model. Both the back and the front views of the hand model are depicted. These visual comparison results show that our method outperforms the multiscale anisotropic Laplacian approach not only in effectively removing noise but also in preserving well the fine details of the model.

The denoising results of our method colored by fractional anisotropy and mean curvature are also shown in Figure 3.9. This better performance is in fact consistent with a variety of 3D models used for experimentation as shown in Figure 3.10 and Figure 3.11.

Figure 3.12 depicts the output results of our algorithm compared to the multiscale anisotropic Laplacian flow on an enlarged view of a noisy 3D bunny model's head shown in Figure 3.12(a). Note that the geometric structures and the fine details around the eyes, nose, and ears of the denoised bunny model are very well preserved. The output result of our method colored by mean curvature is displayed in Figure 3.13.

3.3.2 Quantitative evaluation of the proposed method

Let $\mathbb{M} = (\mathcal{V}, \mathcal{T})$ and $\widehat{\mathbb{M}} = (\widehat{\mathcal{V}}, \widehat{\mathcal{T}})$ be the original model and the denoised model with vertices $\mathcal{V} = \{\mathbf{v}_i\}_{i=1}^m$ and $\widehat{\mathcal{V}} = \{\widehat{\mathbf{v}}_i\}_{i=1}^m$, and triangles $\mathcal{T} = \{\mathbf{t}_i\}_{i=1}^n$ and $\widehat{\mathcal{T}} = \{\widehat{\mathbf{t}}_i\}_{i=1}^n$, respectively. To quantify the performance of the proposed approach, we computed the L^2 face-normal error metric [6] given by

$$E(\mathbb{M}, \widehat{\mathbb{M}}) = \frac{1}{A(\widehat{\mathbb{M}})} \sum_{\widehat{\mathbf{t}}_i \in \widehat{\mathcal{T}}} A(\widehat{\mathbf{t}}_i) \|\mathbf{n}(\mathbf{t}_i) - \mathbf{n}(\widehat{\mathbf{t}}_i)\|, \quad (3.8)$$

where $\mathbf{n}(\mathbf{t}_i)$ and $\mathbf{n}(\widehat{\mathbf{t}}_i)$ are the unit normals of \mathbf{t}_i and $\widehat{\mathbf{t}}_i$ respectively, and $A(\widehat{\mathbf{t}}_i)$ is the area of $\widehat{\mathbf{t}}_i$.

The values of L^2 face-normal error are displayed in Figure 3.14 through Figure 3.16 for three models (hand, horse, and fan disk) used for experimentation. These plots show the error results between the original and the denoised models using our method at iteration numbers 1, 5, 10, 15, and 20. It should be noted that the error values remain almost unchanged at higher iteration numbers, indicating that 1 iteration suffices to denoise 3D models using our method. This observational evidence is also strongly supported by the visual denoising results shown in Figure 3.17, where 1 iteration appears to yield satisfactory output results.

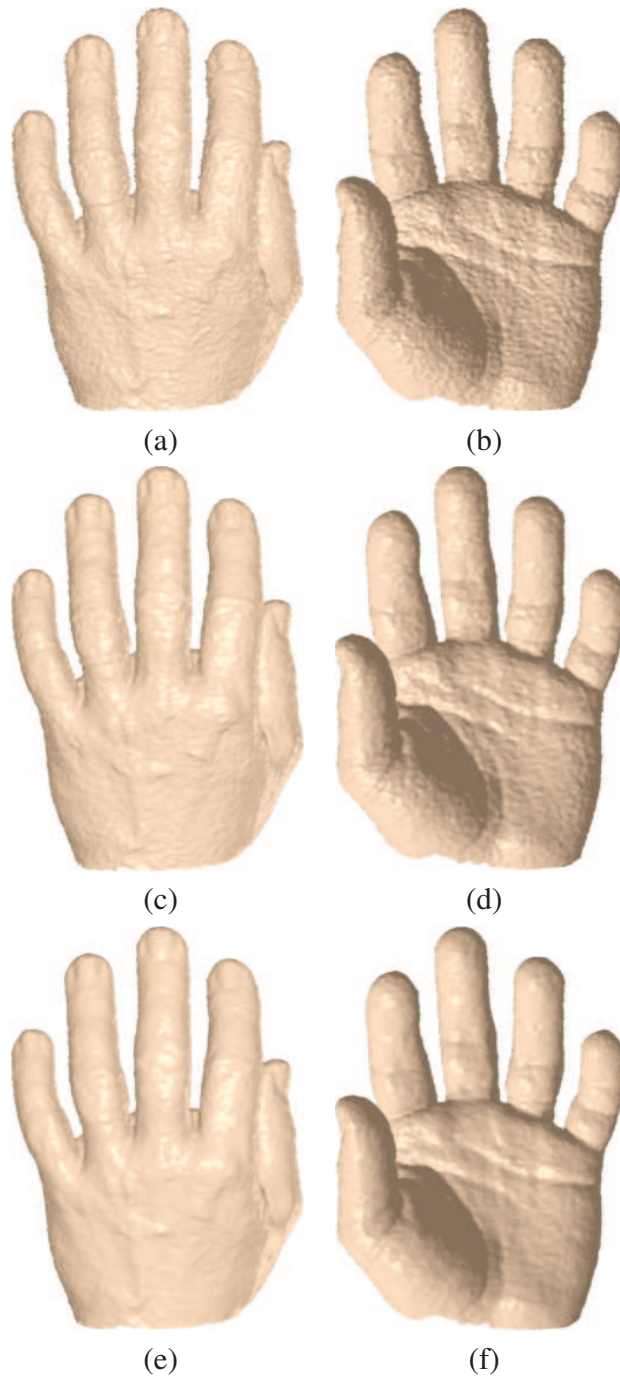


FIGURE 3.8: Comparison of denoising results on a 3D hand model. (a)-(b) Noisy model (back and front views); (c)-(d) multiscale anisotropic Laplacian; (e)-(f) our method. The number of iterations is set to 1.

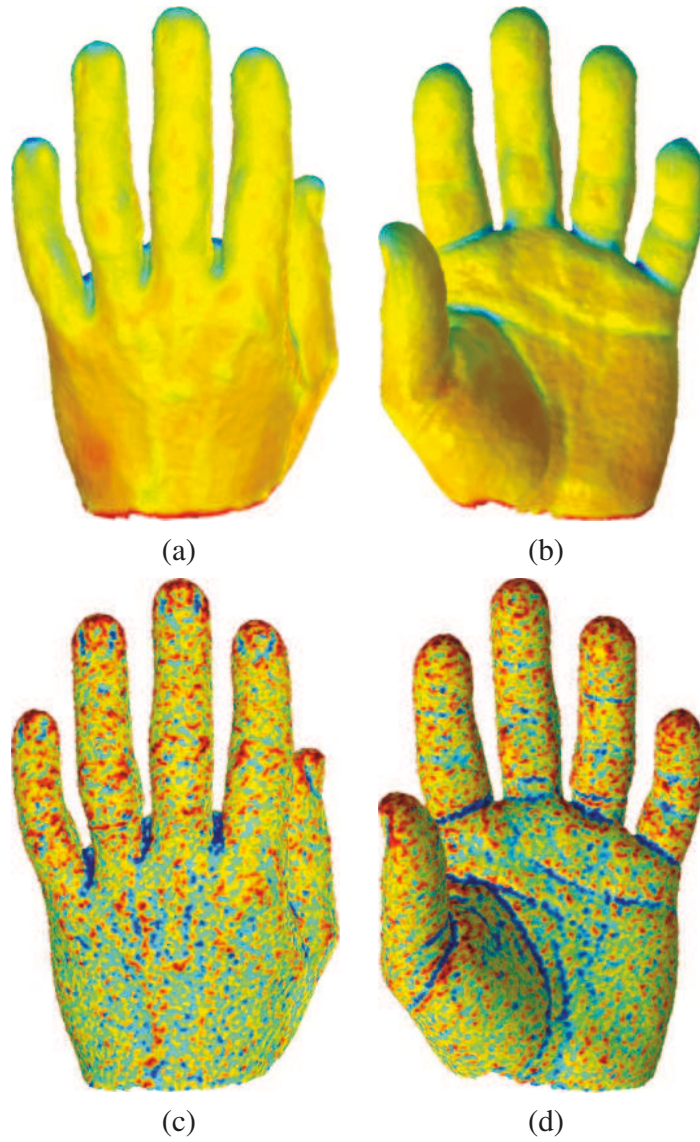


FIGURE 3.9: Output results of our method colored by: (a)-(b) fractional anisotropy; (c)-(d) mean curvature. The number of iterations is set to 1.

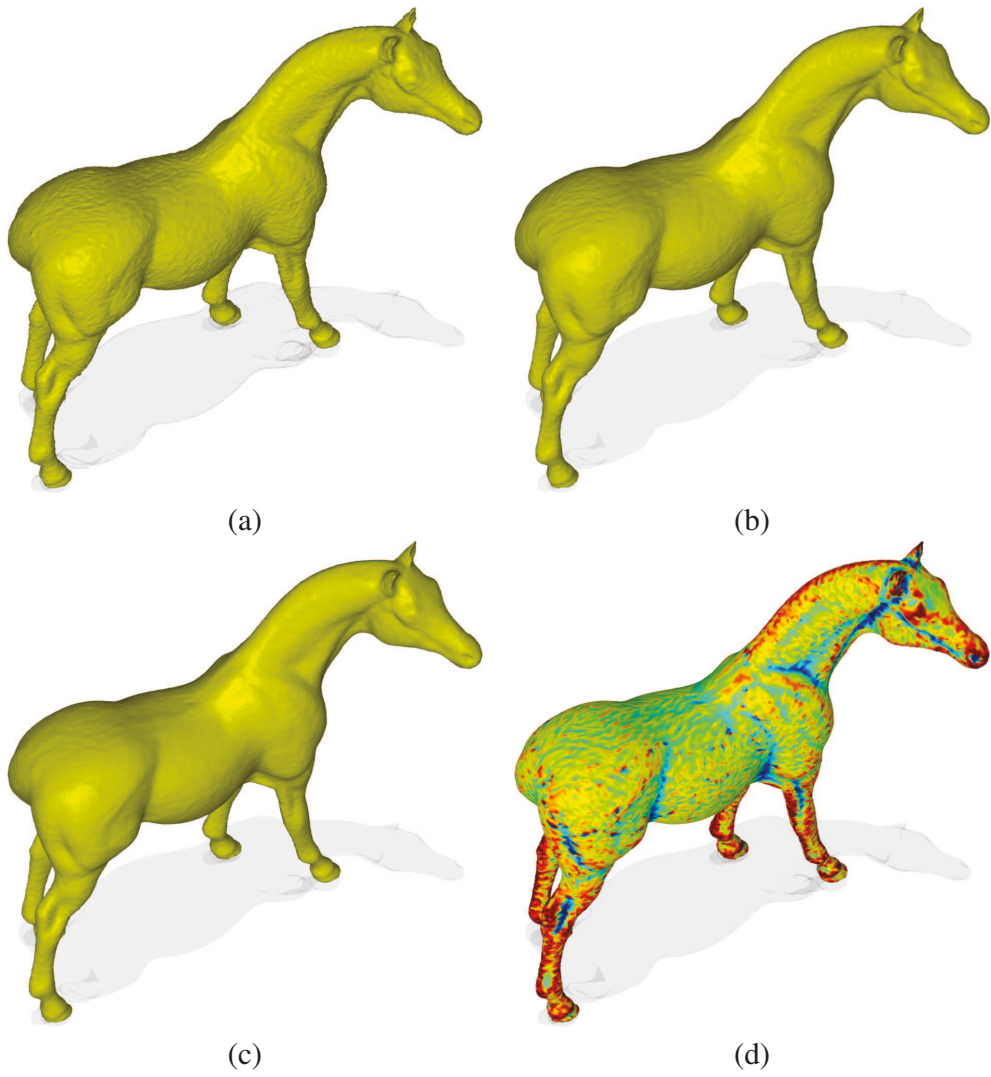


FIGURE 3.10: Comparison of denoising results on a 3D horse model. (a) Noisy model; (b) multiscale anisotropic Laplacian; (c) our method; (d) output result of our method colored by mean curvature. The number of iterations is set to 1.

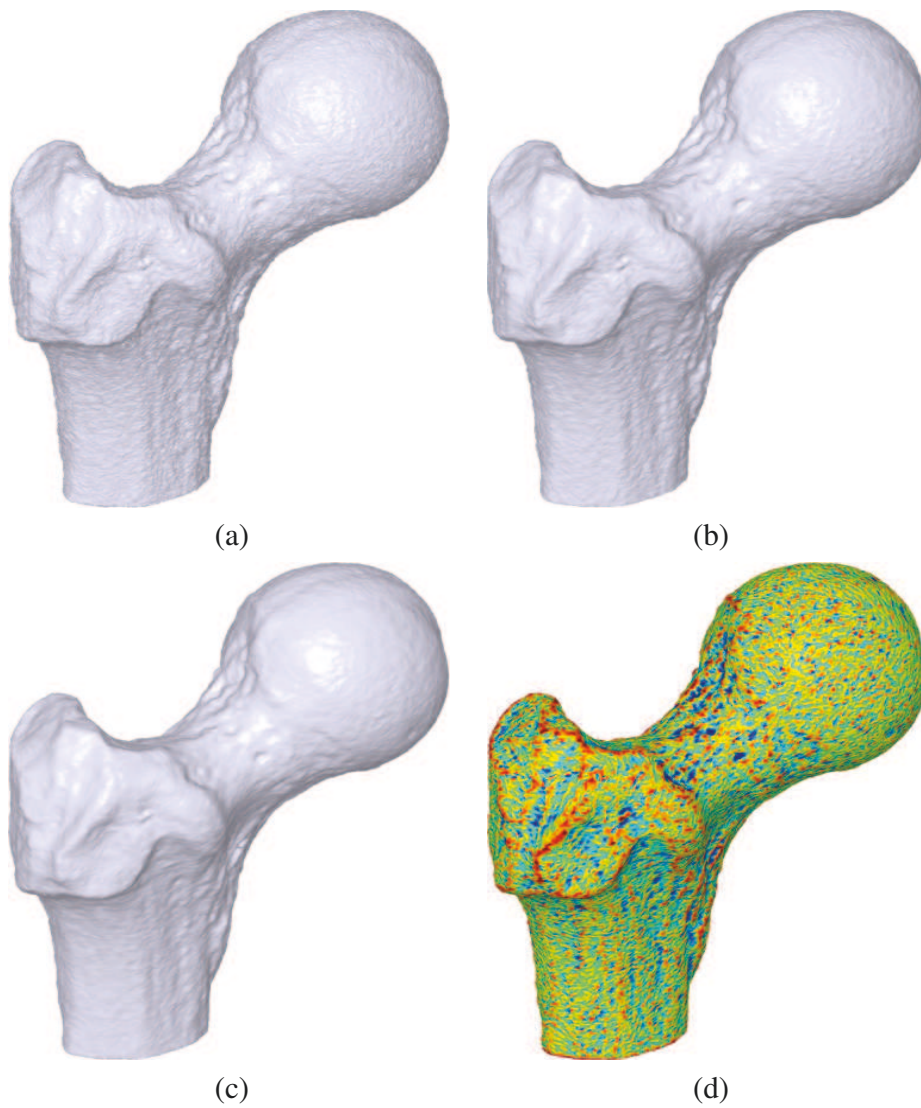


FIGURE 3.11: Comparison of denoising results on a 3D foot bone model. (a) Noisy model; (b) multiscale anisotropic Laplacian; (c) our method; (d) output result of our method colored by mean curvature. The number of iterations is set to 1.

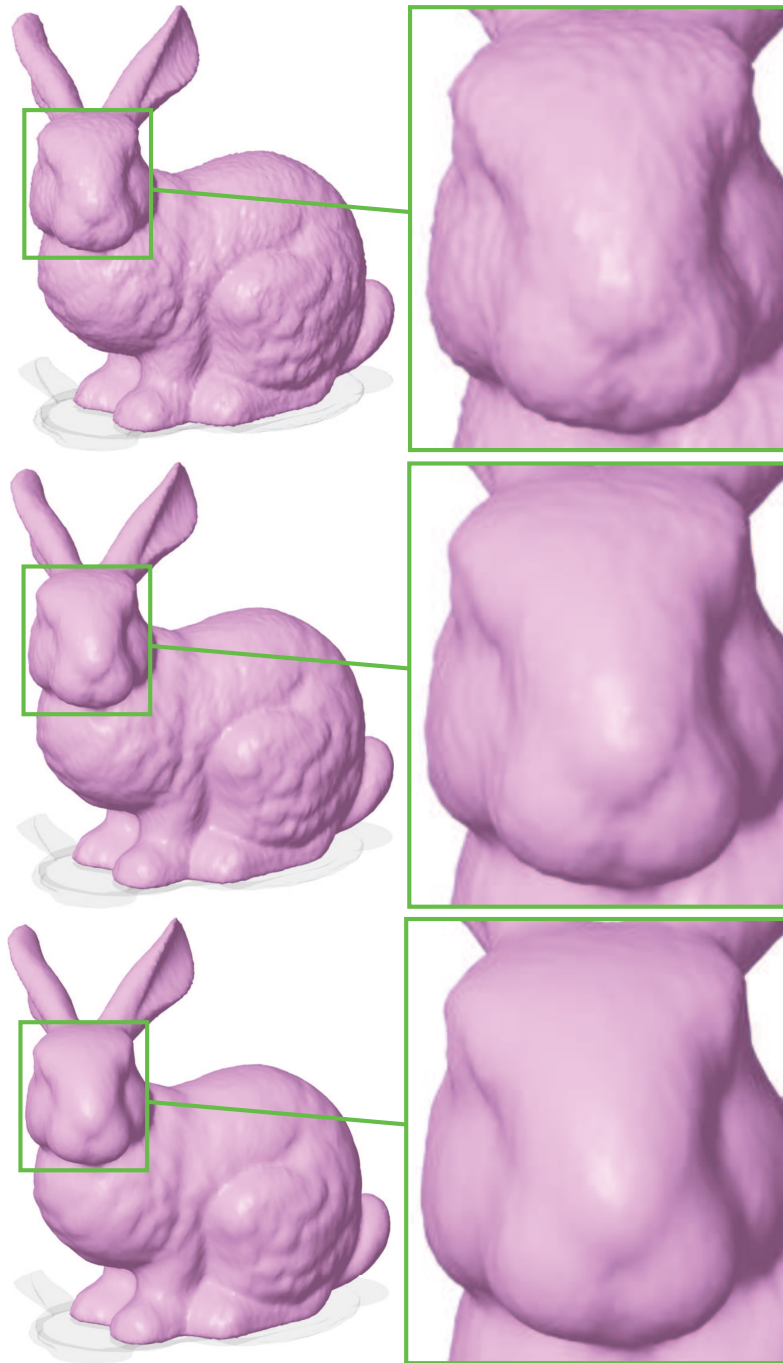


FIGURE 3.12: Comparison of denoising results on a 3D bunny model (enlarged view). Noisy model (top); (b) multiscale anisotropic Laplacian (middle); (c) our method (bottom). The number of iterations is set to 1.

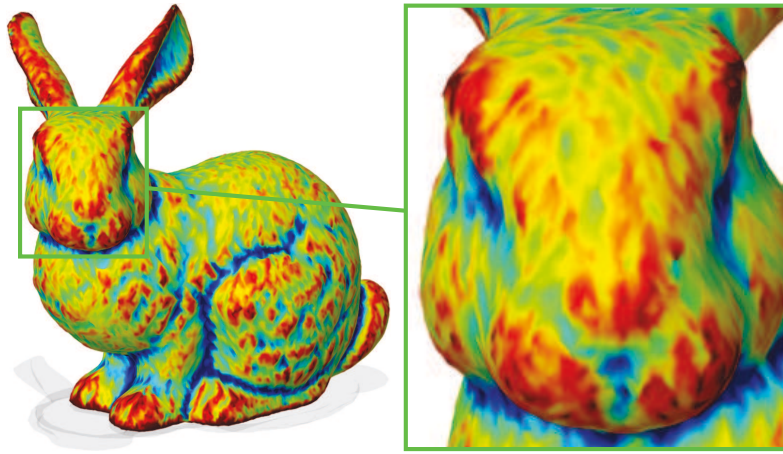


FIGURE 3.13: Output result of our method colored by mean curvature. The number of iterations is set to 1.

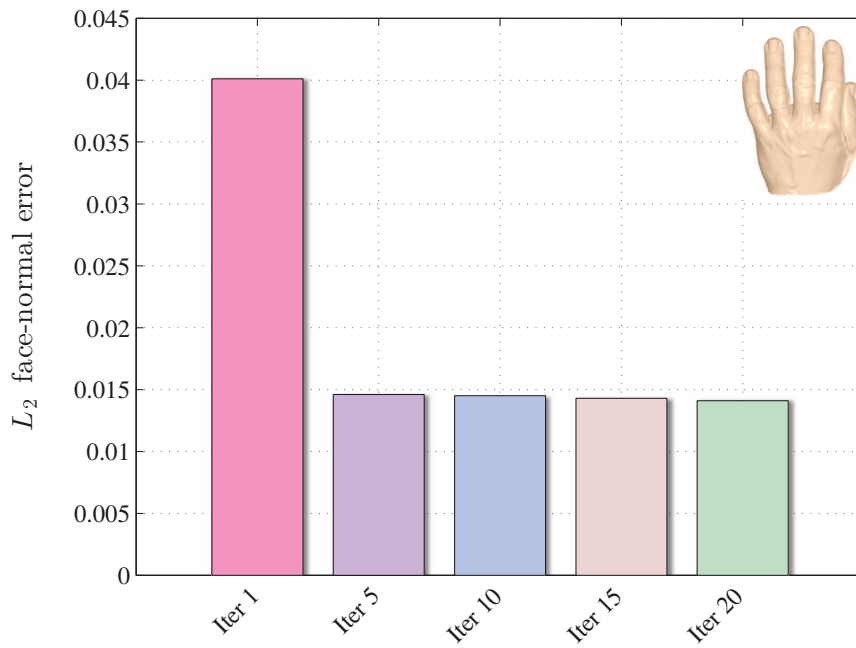


FIGURE 3.14: Plots of L_2 face-normal error vs. iteration number for the hand model.

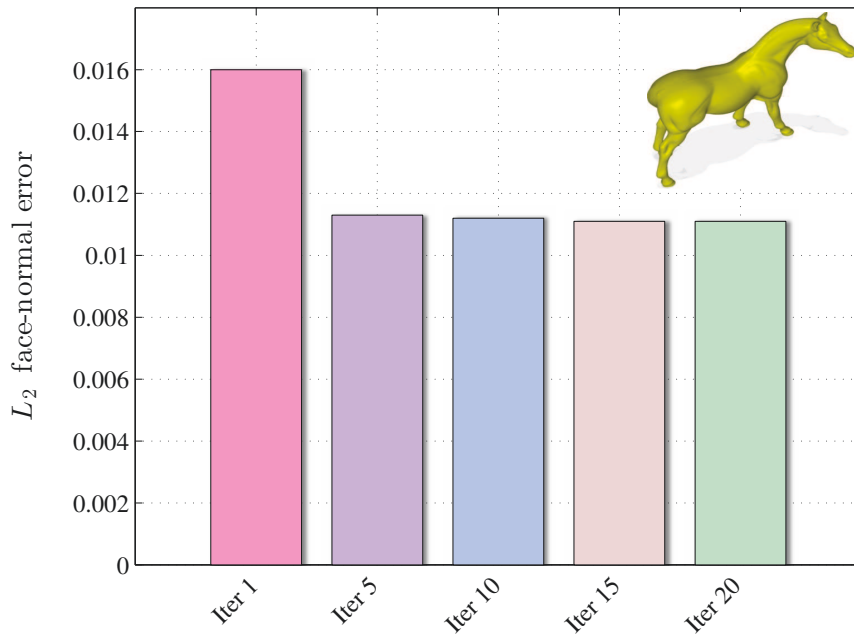


FIGURE 3.15: Plots of L_2 face-normal error vs. iteration number for the horse model.

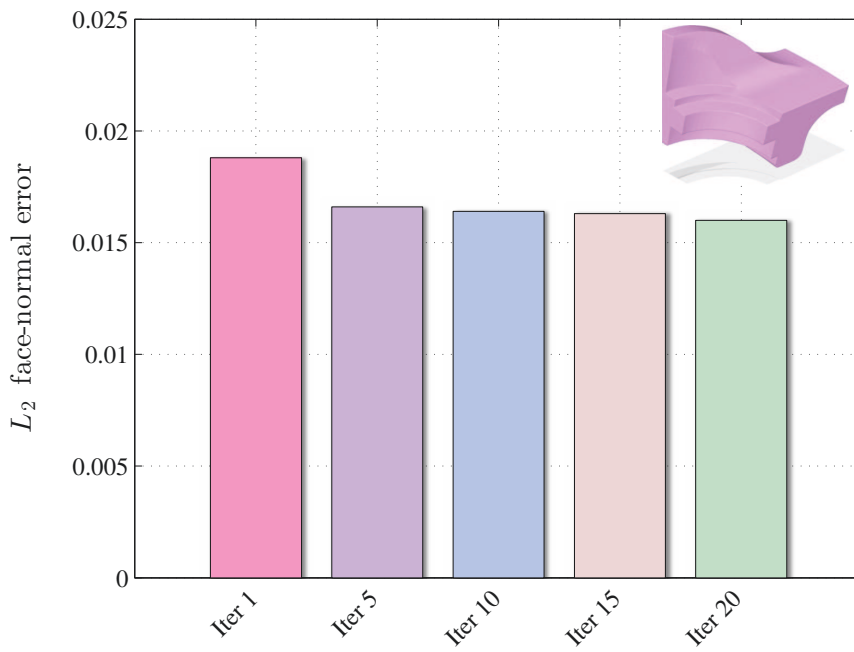


FIGURE 3.16: Plots of L_2 face-normal error vs. iteration number for the fan disk model.

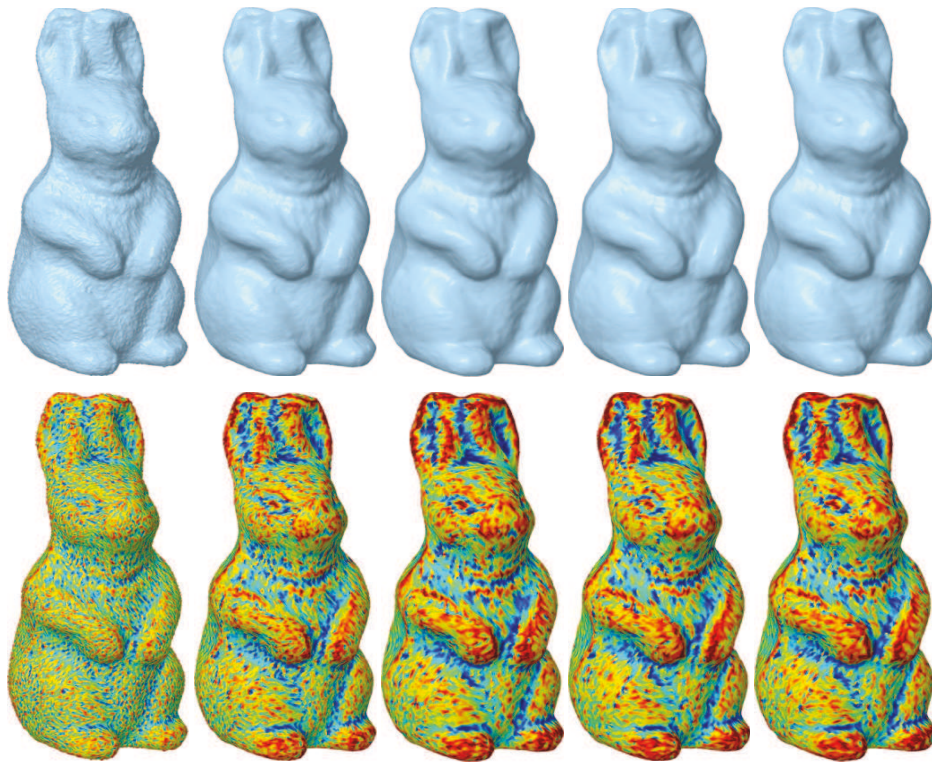


FIGURE 3.17: Top row (from left to right): Noisy rabbit model and output results using our method with iteration numbers 1, 5, 10, and 15. Bottom row: mean curvature visualization of the models shown in top row.

Spectral Geometric Shape Retrieval

4.1 Introduction

The importance of 3D shape recognition is irrupting due to the difficulty in processing information expeditiously without its recognition. With the increasing use of 3D scanners and as a result of emerging multimedia computing technologies, vast databases of 3D models are distributed freely or commercially on the World Wide Web. The availability and widespread usage of such large databases, coupled with the need to explore 3D models in depth as well as in breadth, has sparked the need to organize and search these vast data collections, retrieve the most relevant selections, and permit them to be effectively reused. 3D objects consist of geometric and topological information, and their compact representation is an important step towards a variety of computer vision applications, particularly matching and retrieval in a database of 3D models. The first step in 3D object matching usually involves finding a reliable shape descriptor or skeletal graph, which will encode efficiently the 3D shape information. Skeletonization aims at reducing the dimensionality of a 3D shape while preserving its topology [57, 58]. Unlike text documents, 3D models are not easily retrieved due largely to the variability of their shapes. Attempting to retrieve a 3D model using textual annotation and a conventional text-based search engine would not work properly in many cases [59]. The annotations added by users depend on various factors, including language,

culture, age, and gender. In contrast, content-based 3D shape retrieval methods, which typically use the shape properties of the query shape to search for similar models, perform better than text-based approaches [59].

In this chapter, we describe a spectral skeletonization approach that aims at representing 3D objects with topological coding, which we refer to as *Spectral Reeb Graph (SRG)*. Topology represents the connectedness of a shape and enables parts of shapes, which are connected, to be mapped and drawn equivalently. One of the key mathematical tools used to study the topology of spaces is Morse theory, which is the study of the relationship between functions on a space and the shape of the space. Morse theory studies the properties of a Morse function which has only nondegenerate singular points [57,60], and it describes the topology changes of the level sets of this function at those singularities. Regular or noncritical points do not affect the number or genus of the components of the level sets. It can be shown that Morse functions are dense and stable in the set of all smooth functions, that is the structure of nondegenerate singularities does not change under small perturbations [57,60]. A Morse theoretic representation that captures topological properties of objects is the so-called *Reeb graph* proposed in [17], which is based on the Morse height function. The vertices of the Reeb graph are the singular points of a Morse function defined on the surface of a 3D object [17,57]. The height function-based approach may lead to the extraction of an unbounded number of critical points, except in the case of triangle meshes where the number of critical points is bounded by the number of mesh vertices. This limitation has been addressed in [61] by introducing a fair Morse function that produces the least possible number of critical points. Since the level sets of the height function are horizontal planes perpendicular to the height axis, the main weakness of such Reeb graphs is that they are not invariant to rotation. Hilaga *et al.* [19] used the geodesic distance from point to point on a surface to overcome the problem of automatic extraction of the source point. The geodesic integral is, however, computed using a selected (typically small) random subset of points on the surface, which may lead to inaccuracies in terms of effectively capturing the topological structure of the surface. Moreover, another disadvantage of using the geodesic distance is its sensitivity to topological changes. That is, modifying the shape connectivity may significantly alter the shortest paths between feature points, resulting in significant changes of the geodesic distance. Tierny *et al.* [62] presented a structural oriented Reeb graph based method for partial 3D shape retrieval. Partial similarity between two shapes is then

evaluated by computing a variant of their maximum common sub-graph. Aouada *et al.* [63] proposed a topological Reeb graph using an intrinsic global geodesic function defined on the surface of a 3D object. This approach decomposes a shape into primitives, and then detailed geometric information is added by tracking the evolution of Morse’s function level curves along each primitive. A detailed overview of the mathematical properties of Reeb graphs and their applications to shape analysis is presented in [64]. Pascucci *et al.* [65] introduced a robust method for fast Reeb graph computation that is able to handle non-manifold meshes. Also, Patane *et al.* [66] proposed an efficient Reeb graph computation algorithm by studying the evolution of the level sets only at the saddle points of a Morse function.

More recently, there has been a surge of interest in the spectral analysis of the Laplace-Beltrami (LB) operator, resulting in a slew of applications to manifold learning [67], object recognition and shape analysis [68–71]. It is worth pointing out that spherical harmonics [26] are nothing but the LB eigenfunctions on the sphere. Reuter [69] introduced a Morse-theoretic method for shape segmentation and registration using the topological features of the LB eigenfunctions. These eigenfunctions are computed via a cubic finite element method on triangular meshes, and are arranged in increasing order of their associated eigenvalues. Rustamov [70] proposed a feature descriptor referred to as the global point signature (GPS), which is a vector whose components are scaled eigenfunctions of the LB operator evaluated at each surface point. GPS is invariant under isometric deformations of the shape, but it suffers from the problem of eigenfunctions switching whenever the associated eigenvalues are close to each other. Bronstein *et al.* [71] proposed a non-rigid shape retrieval approach using bags of features based on the heat kernel signature (HKS) [72]. HKS is a temporal shape descriptor, which is defined as an exponentially-weighted combination of the eigenfunctions of the LB operator on a manifold. HKS is a local shape descriptor that has a number of desirable properties, including robustness to small perturbations of the shape, efficiency, and invariance to isometric transformations. However, HKS depends on the time parameter, which needs to be set a priori. Also, the discrete heat kernel requires finding a few hundred eigenvalues and eigenfunctions of a typically large LB matrix, which is often a computationally expensive process. In addition, the choice of the vocabulary size, the time parameter, and the number of eigenvalues/eigenfunctions can have an impact on the performance of the HKS-based retrieval algorithm. The idea of HKS was independently proposed by Gębal *et al.* [73] for 3D shape skeletonization

and segmentation under the name of *auto diffusion function (ADF)*. As the name suggests, ADF describes the diffusion from a surface point to itself. Shi *et al.* [74] used the level curves of the second eigenfunction of the LB operator to construct the spectral skeleton of 3D neuroanatomical structures. In addition to having a nice geometric property of following the pattern of the overall shape of a 3D object, the second eigenfunction of the LB operator can capture the intrinsic structure of elongated shapes (e.g. hippocampus) and it is also invariant to isometric transformations. Moreover, the spectral skeleton is invariant to the pose of the shape [74].

Motivated by the aforementioned invariance properties of the second eigenfunction of the LB operator, we propose to use the spectral Reeb graph framework to construct the shape skeleton of a 3D object. The key idea is to identify and encode regions of topological interest of a 3D object in the Morse-theoretic framework. That is, the level sets (isocontours) of the second eigenfunction are computed (identified), then each level set is encoded as a skeleton node representing the centroid of the isocontour.

The rest of this chapter is organized as follows. In Section 4.2, we start with a review on the LB operator and Morse-theoretic Reeb graph for topological modeling of 3D shapes. This is followed by a brief spectral geometric analysis of the LB operator. We also explore the connection between Morse theory and the spectrum of the LB operator. Then, we delineate algorithmic steps for computing the spectral Reeb graph of a 3D object based on the second eigenfunction of the LB operator. Section 4.3 introduces the path dissimilarity skeleton graph matching method by comparing the relative shortest paths between the skeleton endpoints. In Section 4.4, we present experimental results for topological coding using the spectral Reeb graph and we demonstrate the feasibility of this skeletal graph as a shape descriptor for 3D object matching and retrieval.

4.2 Spectral Reeb Graph Framework

4.2.1 Laplace-Beltrami Operator

Let \mathbb{M} be a smooth orientable 2-manifold (surface) embedded in \mathbb{R}^3 . A global parametric representation (embedding) of \mathbb{M} is a smooth vector-valued map \mathbf{x} defined from a connected open set (parametrization domain) $U \subset \mathbb{R}^2$ to $\mathbb{M} \subset \mathbb{R}^3$ such that $\mathbf{x}(\mathbf{u}) = (x^1(\mathbf{u}), x^2(\mathbf{u}), x^3(\mathbf{u}))$, where $\mathbf{u} = (u^1, u^2) \in U$, as illustrated in Figure 4.1. Note that the components of \mathbf{x} and \mathbf{u} are denoted

by superscripts in place of subscripts. This superscript convention stems from the use of tensor notation which greatly simplifies the formalism of the theory of surfaces [75, 76].

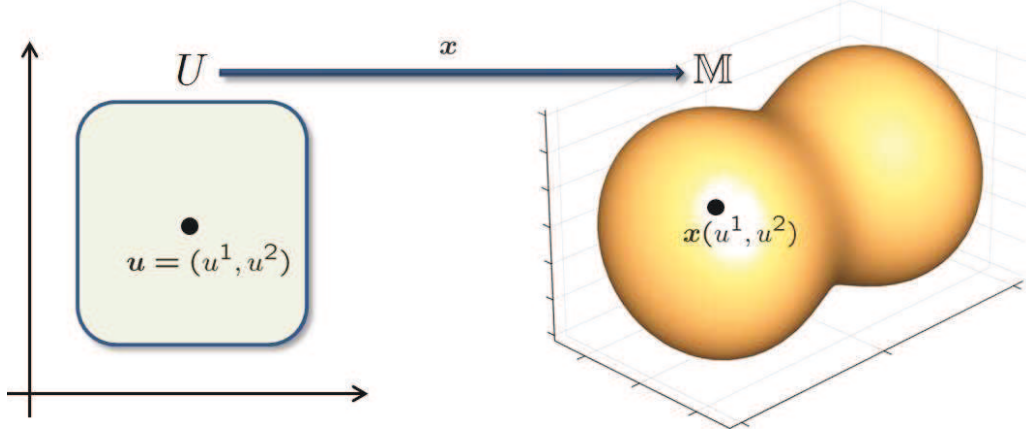


FIGURE 4.1: Parametric representation of a surface.

Given a twice-differentiable function $f : \mathbb{M} \rightarrow \mathbb{R}$, the Laplace-Beltrami (LB) operator is defined as

$$\Delta_{\mathbb{M}} f = -\frac{1}{\sqrt{\det g}} \sum_{i,j=1}^2 \frac{\partial}{\partial u^j} \left(\sqrt{\det g} g^{ij} \frac{\partial f}{\partial u^i} \right), \quad (4.1)$$

where the matrix $g = (g_{ij})$ is the Riemannian metric tensor on \mathbb{M} , and g^{ij} denote the elements of the inverse of g . The metric tensor g is an intrinsic quantity in the sense that it relates to measurements inside the surface [76, 77].

The LB operator is a second-order partial differential operator acting on the space of real-valued functions on a manifold. Let $L^2(\mathbb{M})$ be the space of square integrable functions on the manifold \mathbb{M} . The space $L^2(\mathbb{M})$ is endowed with inner product

$$\langle f_1, f_2 \rangle = \int_{\mathbb{M}} f_1(\mathbf{x}) f_2(\mathbf{x}) da. \quad (4.2)$$

where da is the area element of the surface \mathbb{M} .

4.2.2 Morse theory for Topological Modeling

Morse theory explains the presence and the stability of singular points in terms of the topology of the underlying smooth manifold. The basic principle is that the topology of a manifold is very closely related to the singular points of a smooth function defined on that manifold [60]. A smooth

function $f : \mathbb{M} \rightarrow \mathbb{R}$ on a smooth manifold \mathbb{M} is called a *Morse function* if all its singular points are nondegenerate, i.e. the Hessian matrix is nonsingular at every singular point. A point \mathbf{x} is called a *regular point* of f if the differential $df : T_{\mathbf{x}}\mathbb{M} \rightarrow \mathbb{R}$ is *surjective*, that is, the Jacobian matrix (3×1 in the case of a 2-manifold) has rank equal to $\dim(\mathbb{R}) = 1$. Otherwise, the point \mathbf{x} is called a *critical point*. Nondegenerate singularities are isolated, that is, there cannot be a sequence of nondegenerate singularities converging to a nondegenerate singularity $\mathbf{x} \in \mathbb{M}$. A *level set* $f^{-1}(a)$ of f at a value $a \in \mathbb{R}$ may be composed of one or many connected components. The Morse deformation lemma states that if no critical points exist between two level sets of f , then the two level sets are topologically equivalent and can be deformed onto one another [57]. In particular, they consist of the same number of connected components. Furthermore, Morse theory implies that topological changes on the level sets occur only at critical points. This property can be illustrated by considering the sub-surface \mathbb{M}_a consisting of all points at which f takes values less than or equal to a real number a

$$\mathbb{M}_a = \{\mathbf{x} \in \mathbb{M} : f(\mathbf{x}) \leq a\}. \quad (4.3)$$

Denote by L_a the set of points where the value of f is exactly a , that is $L_a = f^{-1}(a)$. Note that when a is a regular value, the set L_a is a smooth curve of \mathbb{M} and it is the boundary of \mathbb{M}_a as illustrated in Figure 4.2.

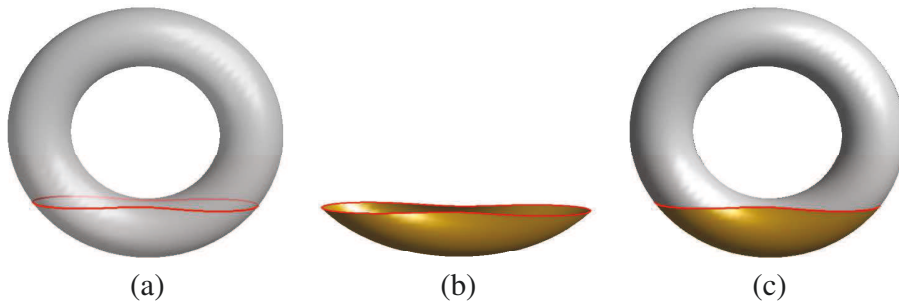


FIGURE 4.2: Illustration of: (a) Level curve L_a , (b) Subsurface \mathbb{M}_a , (c) Sub-surface and Level curve.

Figure 4.3 shows the evolution of the subsurface \mathbb{M}_a as a changes, when f is a height function. If $a < \min_{\mathbf{x} \in \mathbb{M}}\{f(\mathbf{x})\}$, then $\mathbb{M}_a = \emptyset$. And as we increase the parameter a , the subsurface \mathbb{M}_a changes until it covers the entire surface \mathbb{M} .

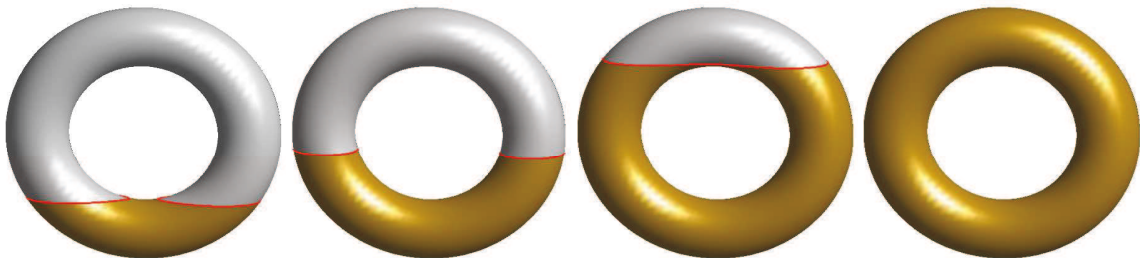


FIGURE 4.3: Evolution of \mathbb{M}_a as a changes.

An interesting concept related to Morse theory and very useful to analyze a surface topology is the Reeb graph. The latter is defined as a quotient space \mathbb{M}/\sim with the equivalence relation given by $x \sim y$ if and only if $f(x) = f(y)$ and x, y belong to the same connected component of $f^{-1}(f(x))$. An equivalence class is defined as $[x] = \{y \in \mathbb{M} : x \sim y\}$. Intuitively, \mathbb{M}/\sim is a space created by taking the space \mathbb{M} and gluing x to any y that satisfies $y \sim x$. The classes $[x]$ are the connected components for the Reeb graph, and being in the same component is an equivalence relation:

$$y \sim x \iff f(y) = f(x) \text{ and } x, y \in \mathcal{C}, \quad (4.4)$$

where \mathcal{C} denotes the connected component of $f^{-1}(f(x))$.

4.2.3 Spectral Geometric Analysis of the Laplace-Beltrami Operator

Spectral geometry is concerned with the eigenvalue spectrum of the LB operator on a compact Riemannian manifold, and aims at describing the relationships between such a spectrum and the geometric structure of the manifold. Spectral geometric problems can be broadly divided into two main categories: direct problems and inverse problems. A direct problem attempts to infer information about the eigenvalues and eigenfunctions of the LB operator from knowledge of the geometry of the manifold. In the inverse problem, however, the goal is to investigate what geometric and topological information of the manifold can be recovered from the spectrum of the LB operator. The inverse spectral problem has been translated over the years into the conversational question “Can one hear the shape of a manifold?” by several mathematicians, most notably by Kac [78] because of its analogy with the wave equation, which models the transverse vibrations of ideal stretched objects. The answer to this question is demonstrably untrue: one cannot hear

the shape of a manifold. In other words, the spectrum cannot completely determine the geometry and topology of a manifold as there exist non-isometric manifolds with the same spectrum [79]. However, some geometric and topological information about the manifold can be inferred, such as the dimension, volume, curvature and the Euler characteristic.

Eq. (4.1) shows that the LB operator is determined by the Riemannian metric, indicating that the spectral theory of the LB operator is intimately connected with the geometry of the Riemannian manifold (\mathbb{M}, g) . An eigenfunction f of the LB operator satisfies $\Delta_{\mathbb{M}}f = \lambda f$, where λ is the associated eigenvalue. Moreover, the eigenfunctions of the LB operator are the critical points (vectors) of the Rayleigh-Ritz quotient, which is an energy functional defined as

$$\mathcal{R}(f) = \frac{\int_{\mathbb{M}} \|\nabla_{\mathbb{M}}f\|^2 da}{\int_{\mathbb{M}} \|f\|^2 da} \quad (4.5)$$

and the eigenvalues are the values of the functional \mathcal{R} at such critical points. Obviously, the infimum value $\lambda_1 = 0$ of $\mathcal{R}(f)$ is achieved for a constant function $f = \varphi_1$. Since $\Delta_{\mathbb{M}}$ is a Hermitian operator, the set of eigenvalues (spectrum) $\{\lambda_i, i = 1, 2, \dots, \infty\}$ of $\Delta_{\mathbb{M}}$ is an infinite discrete subset of \mathbb{R}^+ . Assuming \mathbb{M} is compact, these eigenvalues may be written in increasing order as $0 = \lambda_1 < \lambda_2 \leq \lambda_3 \leq \dots$, with associated eigenfunctions $\{\varphi_i, i = 1, \dots, \infty\}$, such that $\varphi_1 = 1/\sqrt{\text{area}(\mathbb{M})} = \text{const}$. Moreover, the eigenfunctions of the LB operator form an orthogonal basis for the the space $L^2(\mathbb{M})$. That is, $\langle \varphi_i, \varphi_j \rangle$ for $i \neq j$.

The second eigenvalue of the LB operator is given by

$$\lambda_2 = \inf_{f \perp \varphi_1} \mathcal{R}(f) \quad (4.6)$$

and φ_2 is its associated eigenfunction. Note that since φ_1 is a constant function, $f \perp \varphi_1$ implies $\langle f, \varphi_1 \rangle = 0$, which yields $\int_{\mathbb{M}} f da = 0$.

The eigenvalues and eigenfunctions have a nice physical interpretation: the square roots of the eigenvalues $\sqrt{\lambda_i}$ are the eigenfrequencies of the membrane, and $\varphi_i(x)$ are the corresponding amplitudes at x . In particular, the second eigenvalue corresponds to the sound we hear the best [80].

4.2.4 Discrete Laplace-Beltrami Operator

Using a mixed finite element/finite volume method on triangle meshes [?], the value of $\Delta_{\mathbb{M}}f$ at a vertex \mathbf{v}_i can be approximated using the cotangent weight scheme as follows

$$\Delta_{\mathbb{M}}f(\mathbf{v}_i) \approx \frac{1}{a_i} \sum_{j \sim i} \frac{\cot \alpha_{ij} + \cot \beta_{ij}}{2} (f(\mathbf{v}_j) - f(\mathbf{v}_i)), \quad (4.7)$$

where α_{ij} and β_{ij} are the angles $\angle(\mathbf{v}_i \mathbf{v}_{k_1} \mathbf{v}_j)$ and $\angle(\mathbf{v}_i \mathbf{v}_{k_2} \mathbf{v}_j)$ of two faces $\mathbf{t}^\alpha = \{\mathbf{v}_i, \mathbf{v}_j, \mathbf{v}_{k_1}\}$ and $\mathbf{t}^\beta = \{\mathbf{v}_i, \mathbf{v}_j, \mathbf{v}_{k_2}\}$ that are adjacent to the edge $[i, j]$, and a_i is the area of the voronoi cell (shaded polygon), as shown in Figure 4.4. It should be noted that the cotangent weight scheme is numerically consistent and preserves several important properties of the continuous LB operator, including symmetry and positive-definiteness [81].

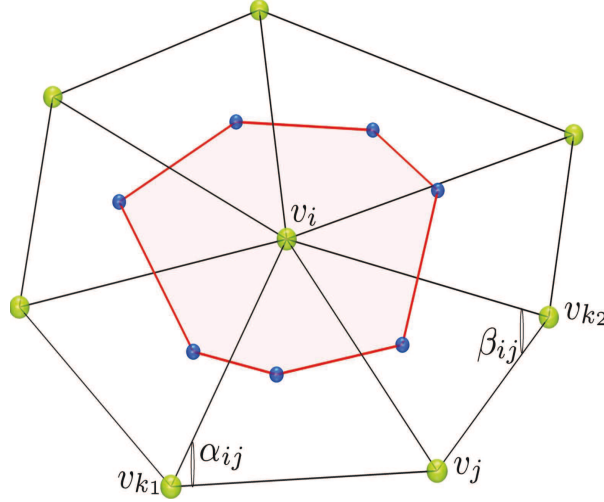


FIGURE 4.4: Cotangent weight scheme: illustration of the angles α_{ij} and β_{ij} .

Define the weight function $\omega : \mathcal{V} \times \mathcal{V} \rightarrow \mathbb{R}$ as

$$\omega_{ij} = \begin{cases} \frac{\cot \alpha_{ij} + \cot \beta_{ij}}{2a_i} & \text{if } i \sim j \\ 0 & \text{o.w.} \end{cases} \quad (4.8)$$

Then, for a function $f : \mathcal{V} \rightarrow \mathbb{R}$ that assigns to each vertex $\mathbf{v}_i \in \mathcal{V}$ a real value $f(\mathbf{v}_i)$ (we can view f as a column vector of length m), we may write the LB operator given by Eq. (4.7) as

$$Lf(\mathbf{v}_i) = \sum_{j \sim i} \omega_{ij} (f(\mathbf{v}_i) - f(\mathbf{v}_j)), \quad (4.9)$$

where the matrix L is given by

$$L = \begin{cases} d_j & \text{if } i = j \\ -\omega_{ij} & \text{if } i \sim j \\ 0 & \text{o.w.} \end{cases} \quad (4.10)$$

and $d_j = \sum_{i=1}^m \omega_{ij}$ is the weighted degree of the vertex \mathbf{v}_i . Note that $\omega_{ij} \neq \omega_{ji}$ implies L is not a symmetric matrix. Thus, the spectrum (set of eigenvalues) of the eigenvalue problem $L\varphi_i = \lambda_i\varphi_i$ may not be real [70]. Noting that $\omega_{ij} = c_{ij}/a_i$, where

$$c_{ij} = \begin{cases} \frac{\cot \alpha_{ij} + \cot \beta_{ij}}{2} & \text{if } i \sim j \\ 0 & \text{o.w.} \end{cases} \quad (4.11)$$

we may factorize the matrix L as $L = R^{-1}C$, where $R = \text{diag}(a_i)$ is a positive-definite diagonal matrix and C is a sparse symmetric matrix given by

$$C = \begin{cases} \sum_{i=1}^m c_{ij} & \text{if } i = j \\ -c_{ij} & \text{if } i \sim j \\ 0 & \text{o.w.} \end{cases} \quad (4.12)$$

Therefore, we may write the eigenvalue problem $L\varphi_i = \lambda_i\varphi_i$ as a generalized eigenvalue problem $C\varphi_i = \lambda_i R\varphi_i$, which can be solved efficiently using the Arnoldi method of ARPACK. Figure 4.5 shows a 3D elephant model and the sparsity pattern of the cotangent matrix C . Recall that the sparsity pattern (or support) of a matrix $A = (a_{ij})$ is the set of indices ij with $a_{ij} \neq 0$.

4.2.5 Spectral Skeleton

The eigenvalues λ_i and associated eigenfunctions φ_i of the LB operator can be computed by solving the following generalized eigenvalue problem:

$$C\varphi_i = \lambda_i R\varphi_i, \quad i = 1, 2, \dots, m \quad (4.13)$$

where φ_i is the unknown eigenfunction evaluated at m mesh vertices. That is, φ_i is an m -dimensional vector.

We may sort the eigenvalues in ascending order as $0 = \lambda_1 < \lambda_2 \leq \dots \leq \lambda_m$ with associated eigenfunctions as $\varphi_1, \varphi_2, \dots, \varphi_m$, where each eigenfunction $\varphi_i = (\varphi_i(\mathbf{v}_1), \dots, \varphi_i(\mathbf{v}_m))'$ is an m -dimensional vector. Moreover, these eigenfunctions are orthogonal $\langle \varphi_i, \varphi_j \rangle_R = 0, \forall i \neq j$,

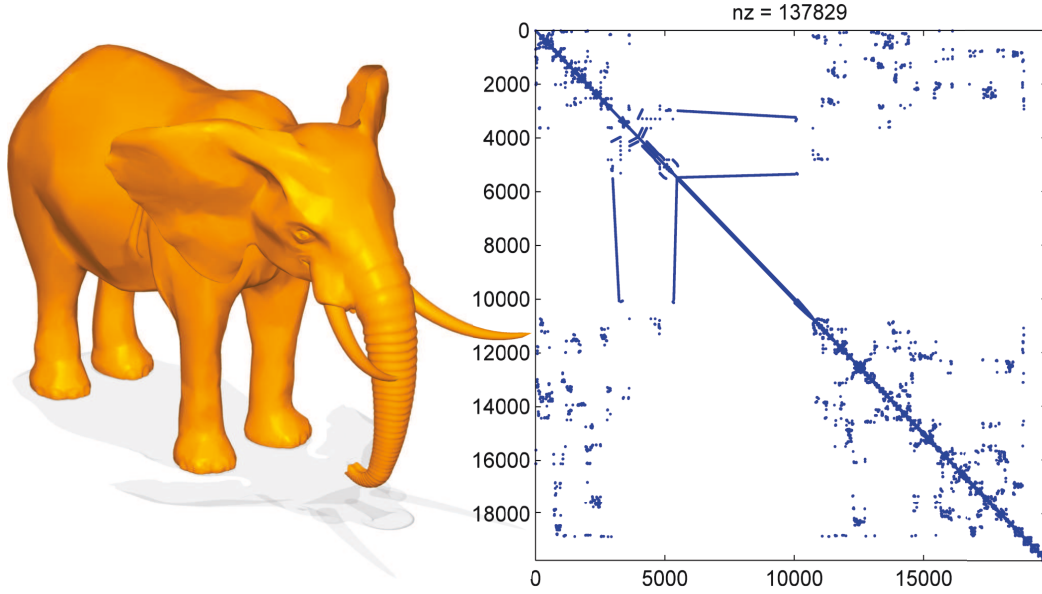


FIGURE 4.5: 3D elephant model and sparsity pattern plot of the associated cotangent matrix C .

where the orthogonality is defined in terms of the R -inner product. That is, $\langle \varphi_i, \varphi_j \rangle_R = \varphi_i' R \varphi_j$. Additionally, any function $f : \mathcal{V} \rightarrow \mathbb{R}$ (viewed as a column-vector of length m) on the triangle mesh \mathbb{M} can be written in terms of the eigenfunctions as follows

$$f = \sum_{i=1}^m \alpha_i \varphi_i, \quad \text{where } \alpha_i = \langle f, \varphi_i \rangle. \quad (4.14)$$

Note that since the sum of each row in the matrix C equals zero, the first eigenvalue λ_1 is zero and the corresponding eigenfunction φ_1 is a constant m -dimensional vector. The top row of Figure 4.6 shows a 3D horse model colored by the second, third and fourth eigenfunctions, while the bottom row displays the isocontours of these eigenfunctions.

We can use the variational characterizations of the eigenvalues in terms of the Rayleigh-Ritz quotient. That is, the second eigenvalue is given by

$$\lambda_2 = \inf_{f \perp \varphi_1} \frac{f' C f}{f' R f} = \inf_{f \perp \varphi_1} \frac{\sum_{i \sim j} c_{ij} (f(\mathbf{v}_i) - f(\mathbf{v}_j))^2}{\sum_i f(\mathbf{v}_i)^2 a_i} \quad (4.15)$$

and $\varphi_2 = (\varphi_2(\mathbf{v}_1), \dots, \varphi_2(\mathbf{v}_m))'$ is its corresponding eigenvector.

The eigenfunction φ_2 is displayed in Figure 4.7(a)-(b), where each vertex \mathbf{v}_i is colored by $\varphi_2(\mathbf{v}_i)$. The level curves of φ_2 are shown in Figure 4.7(c)-(d).

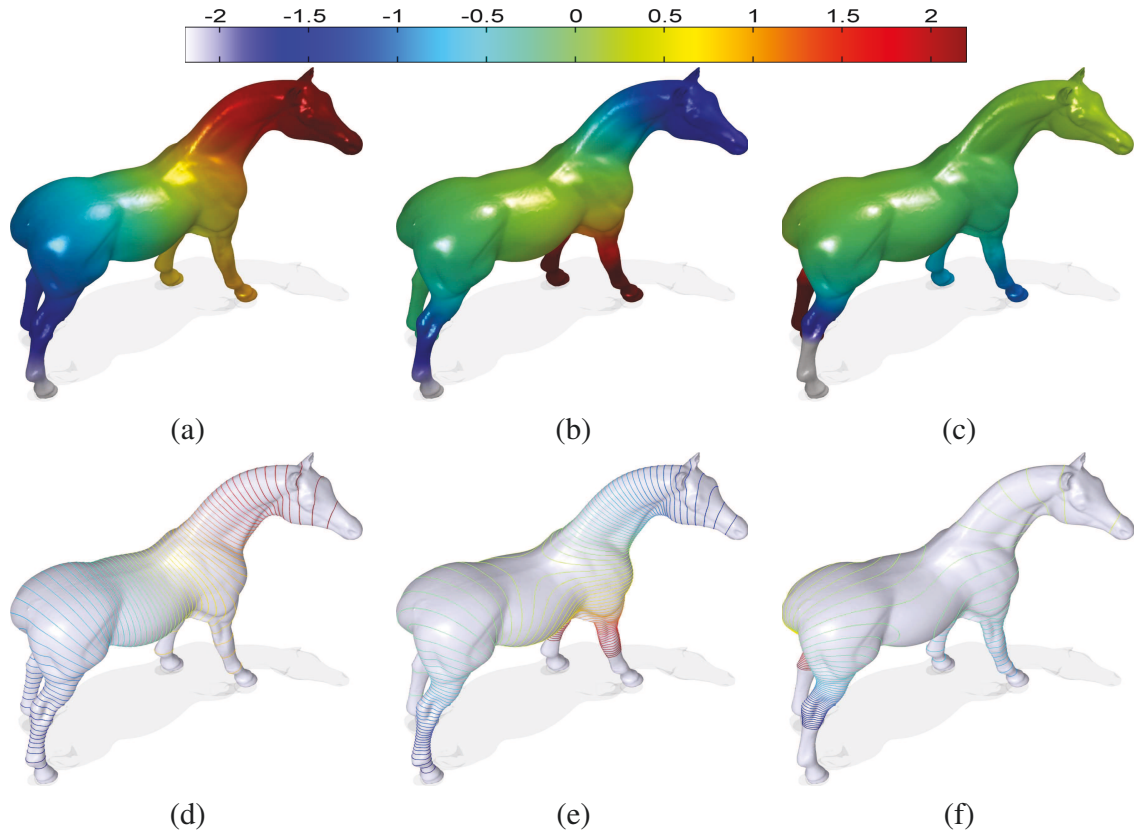


FIGURE 4.6: (a)-(c) 3D horse model colored by $\varphi_2, \varphi_3, \varphi_4$. (d)-(f) level sets of $\varphi_2, \varphi_3, \varphi_4$.

In Figure 4.8(a) we can observe that the isocontours are consistent with global large deformation (first column), local small bend (second column), and among the shapes from different classes, but share similar topological structure (third column). The correspondence of isocontours on the shapes from the same class are displayed in Figure 4.8(b), which shows models that include various topological structures. Finally, the consistency of isocontours on the shapes from the different class are displayed in Figure 4.8(c). Although the shapes are explicitly different, their isocontours can capture their intrinsic correspondence well.

On the other hand, Uhlenbeck [82] showed that the eigenfunctions of the LB operator are Morse functions on the interior of the domain of the operator. Consequently, this generic property of the eigenfunctions gives rise to constructing their associated Reeb graphs.

Analogous to Fourier harmonics for functions on a circle, the LB eigenfunctions associated to lower eigenvalues correspond to low frequency modes, whereas those associated to higher eigen-

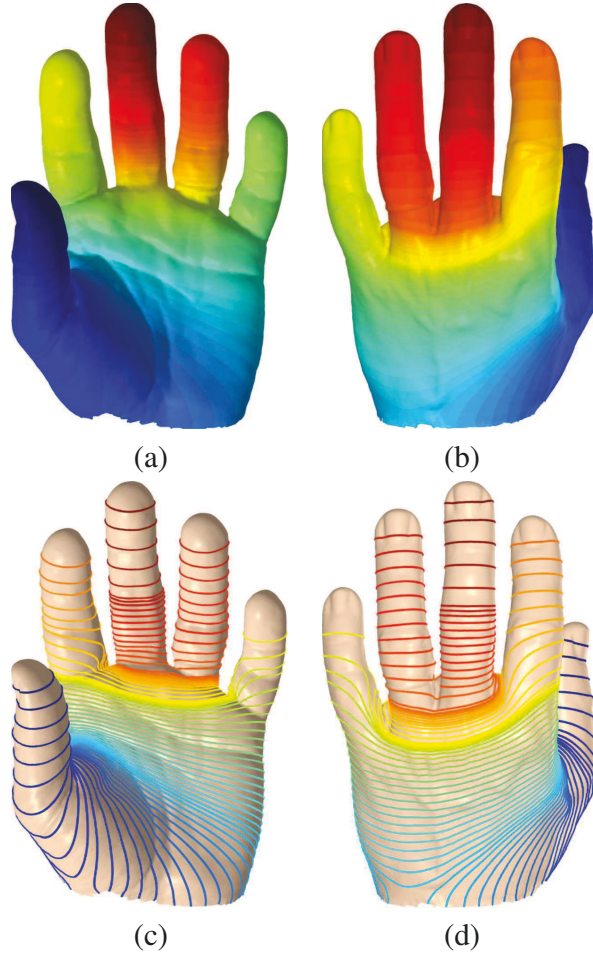


FIGURE 4.7: (a)-(b) 3D hand model colored by φ_2 (front and back views). (c)-(d) Level sets of φ_2 (front and back views).

values correspond to high frequency modes that describe the details on the mesh. As shown in Figure 4.6, Figure 4.7, and Figure 4.9(a), the second eigenfunction of the LB operator captures well the overall shape of 3D objects. Motivated by the isometric invariance property of the second eigenfunction of the LB operator and also by its generic property as a Morse function as well as by the fact that intuitively the second eigenvalue corresponds to the sound we hear the best, we propose to use the spectral Reeb graph to construct the shape skeleton of a 3D object as follows: First, the level sets (isocontours) of the second eigenfunction are computed (identified), as depicted in Figure 4.9(b); then each level set is encoded as a skeleton node representing the centroid of the isocurve, as shown in Figure 4.9(c). The main algorithmic steps for computing the spectral Reeb graph are described in detail in Algorithm 1.

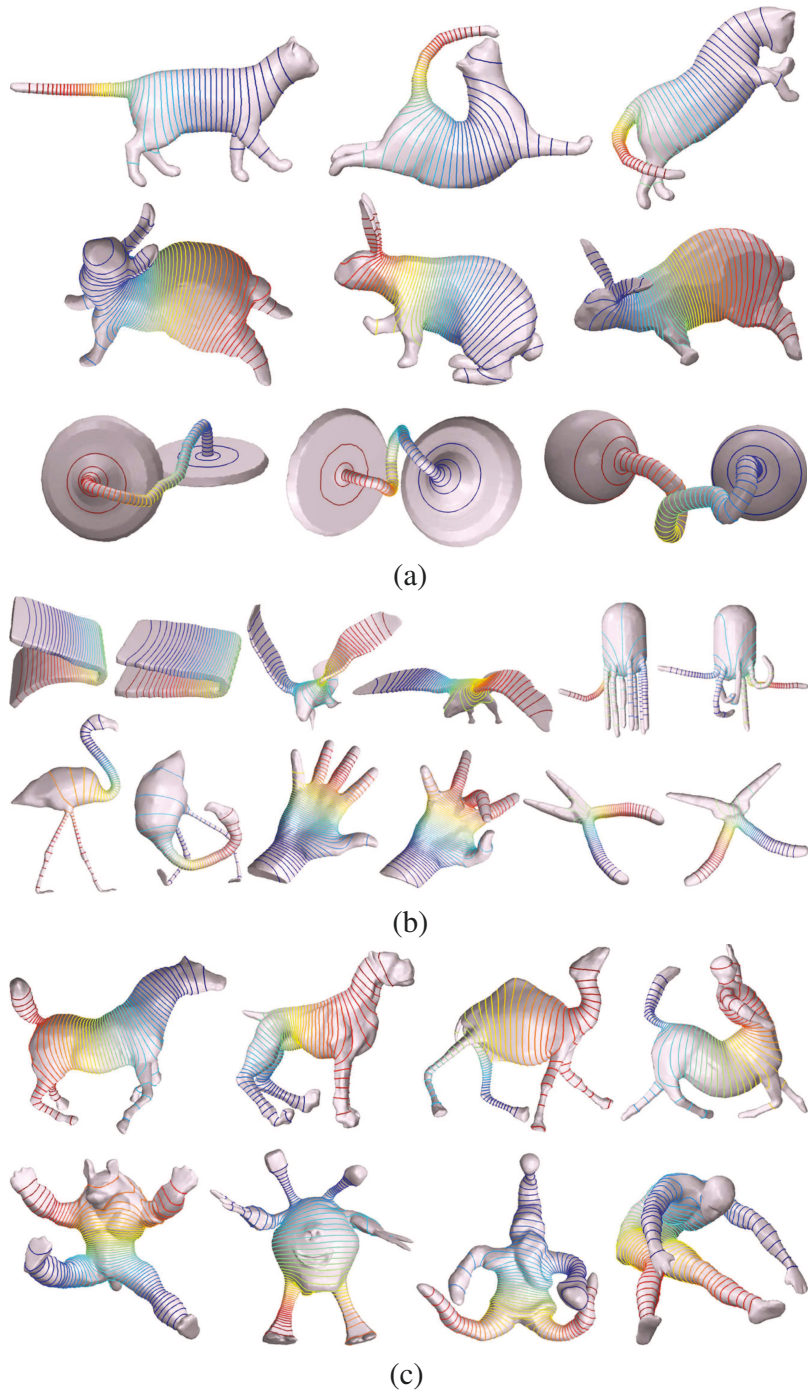


FIGURE 4.8: (a) Isocontours are invariant under both global and local deformation. (b) Proportionality correspondence of pairwise nonrigid shapes with varied topological structure. (c) Isocontours are consistent among different classes of shapes.

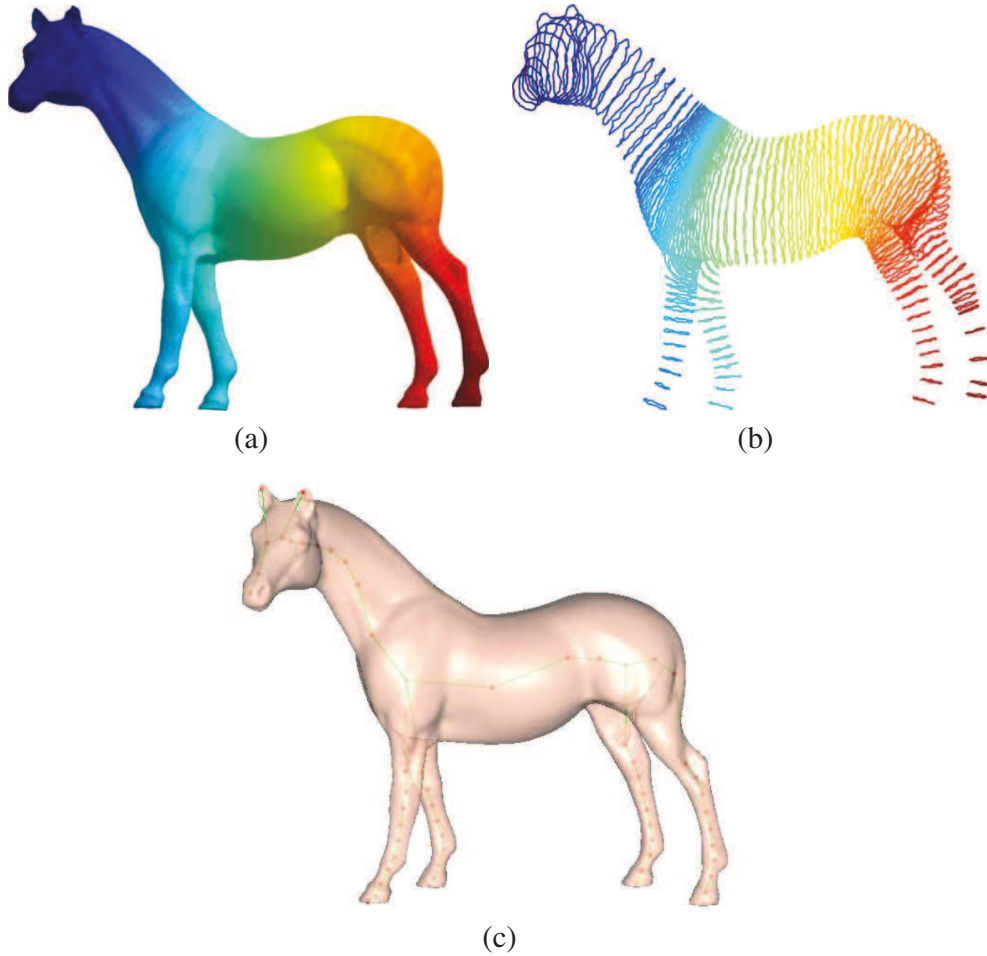


FIGURE 4.9: (a) 3D horse model colored by φ_2 ; (b) level sets of φ_2 ; (c) spectral Reeb graph.

4.3 Spectral Reeb Graph Matching

In this section, we outline a path similarity skeleton graph matching approach by comparing the relative shortest paths between the spectral skeleton endpoints [83]. The proposed skeleton graph matching is based on the dissimilarity of the shortest paths between the endpoints of the skeletal Reeb graph. A skeleton endpoint refers to the skeleton node that is connected by only one edge as shown in Figure 4.10. It is important to point out that endpoints are the salient points of the skeleton and can be seen as visual parts of the original 3D shape [83]. In the same vein as [84], considering only the shortest skeletal paths between endpoints would help avoid the instability problem of the skeleton junction points (i.e. points having three or more adjacent points) and also

Algorithm 1 Proposed skeletonization approach

```
1: Compute the second eigenfunction  $\varphi_2$  of the LB operator by solving the sparse generalized
   eigenvalue problem  $C\varphi_i = \lambda_i R\varphi_i$ .
2: Compute  $N$  level sets  $L_k$  ( $k = 1, \dots, N$ ) of  $\varphi_2$ 
3: for each level set  $L_k$  ( $k = 1$  to  $N$ )
4:  $\text{VerticesSet}_p[0,1] = \text{setIntersect}(\mathbb{M}, 1)$ ;  $\Leftarrow$  Find vertices (subset of the 3D mesh  $\mathbb{M}$ ) of level
   set  $L_k$ 
5:  $\text{NodeSet}_p = \text{centroid}(\text{VerticesSet}_p[0,1](n))$ ;  $\Leftarrow$  Assign a node to each connected component
   at its centroid.
6: for  $k = 2$  to  $N$  do
7:    $\text{VerticesSet}_c[k-1, k] = \text{setIntersect}(\mathbb{M}, k-1, k)$ ;  $\Leftarrow$  Find intersection of  $\mathbb{M}$  from region
    $L_{k-1}$  to  $L_k$ 
8:   for each component  $\text{VerticesSet}_c[k-1, k](n)$  do
9:      $\text{NodeSet}_c = \text{centroid}(\text{VerticesSet}_c[k-1, k](n))$ 
10:    for each connected portion do
11:      Connect  $\text{NodeSet}_c$  and  $\text{NodeSet}_p$ 
12:    end for
13:  end for
14:   $\text{NodeSet}_p = \text{NodeSet}_c$ 
15:   $\text{VerticesSet}_p = \text{VerticesSet}_c$ 
16: end for
```

to make our proposed method more robust to shape deformation. The shortest path between each endpoint and all other endpoints of the skeleton provides an important endpoint feature that will be incorporated into our matching dissimilarity measure.

Our proposed skeleton graph matching approach is based on the assumption that similar skeletons have a similar structure of their endpoints. It is common that the skeletons of similar 3D shapes may have different structures of junction nodes. One of the major advantages of the proposed method is that it does not require that the graphs be converted to trees prior to finding the correspondence, as this conversion may result in the loss of important structural information and, consequently, negatively influence the 3D object recognition result.

In contrast to existing methods for skeleton matching, our proposed approach focuses on the dissimilarity between the shortest paths connecting the skeleton endpoints. We use the shortest paths between endpoints to establish a correspondence relation of the endpoints in different skeletal Reeb graphs. It is worth noting that the idea of using the shortest paths in skeletal graph matching and classification has been previously explored in the literature. For example, Demirci *et al.* [85] proposed transforming the graphs into points in a low-dimensional geometric space using low-

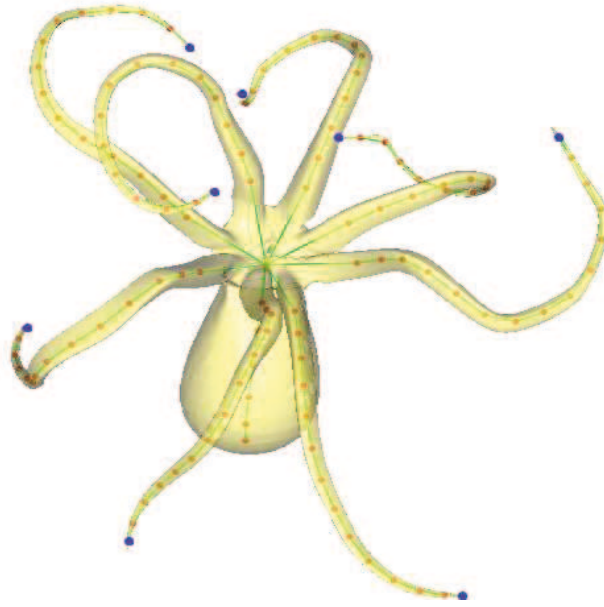


FIGURE 4.10: Spectral Reeb graph of 3D Octopus model and its skeleton endpoints displayed in blue color.

distortion graph embedding techniques. Each point in the embedding space corresponds to a node in the original graphs. The distance in the embedding space reflects the shortest-path distance in the original graphs in order to keep topological relations. Ling *et al.* [86] proposed using the inner-distance to build shape descriptors that are robust to articulation and capture part structure. The inner-distance is defined as the length of the shortest path between landmark points within the shape silhouette.

After generating the 3D shape skeleton, our next step is to develop a robust approach for skeletal graph matching. To this end, we match any two Reeb graphs by establishing a correspondence of their endpoints. Then, we apply a pruning algorithm [87] to remove non-salient nodes from the skeleton graph. The proposed matching method consists of two main steps. The first step, which we refer to as indexing, reduces the number of skeletons to be compared with. In the second step, we match the Reeb graphs by applying a dissimilarity measure to retrieve the closest 3D model. These two steps are explained in more details in the following subsections.

4.3.1 Indexing

A linear search through a database of 3D models is inefficient for large databases, as it requires comparing the query object to each model in the database and selecting the closest one [20]. Therefore, the goal is to apply an efficient indexing mechanism to narrow the search scope in a small set of objects that are most probably similar to the query object. Using our skeletonization algorithm, we may formulate the indexing problem as finding skeletons of which the topological structures are similar to the query skeleton. It is important to note that similar shapes will have the same skeleton even if they are subject to some deformation or transformation. Moreover, these skeletons will have the same number of endpoints.

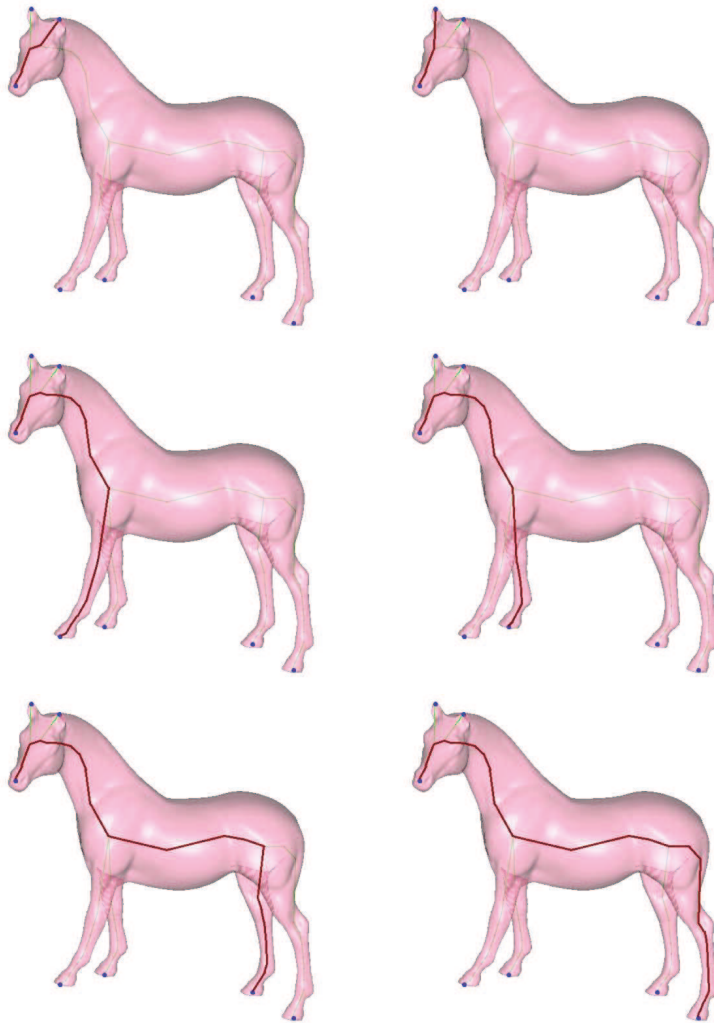
Thus, in our indexing mechanism we use the number of skeleton endpoints as the base for indexing, with an error rate of 2 or 3 nodes, meaning that for two skeletons to be in the same index group they should have the same number of endpoints. However, due to noise there might be a difference of 1 or 2 nodes at most, as a result of the pruning process.

4.3.2 Endpoint correspondence

After applying the indexing mechanism, the next step is to match the skeletons. Our proposed matching method considers both topological and geometrical features of the matched 3D models. We assign to each endpoint in the Reeb graph (query or model) some features that may help identify the closet endpoint in the other skeletal graph. Thus, our skeleton graph matching problem may be reduced to finding the best correspondence between the endpoints in the query and the endpoints in the model. This can be achieved by minimum weight matching of the two sets of endpoints. A dissimilarity measure between the set of endpoints in both query and model skeletons is used. Therefore, the matching problem aims at finding the best correspondence between the query skeleton endpoints and the database skeletons endpoints. Two endpoints are said to be in close correspondence if the dissimilarity measure between their endpoints has a smaller value. In other words, the matching problem is now reduced to finding the maximum correspondence, minimum weight matching of the two sets of endpoints. The endpoint correspondence process is shown in Algorithm 2.



(a)



(b)

FIGURE 4.11: (a) Horse's spectral Reeb graph. (b) Shortest paths between pairs of endpoints on the skeleton.

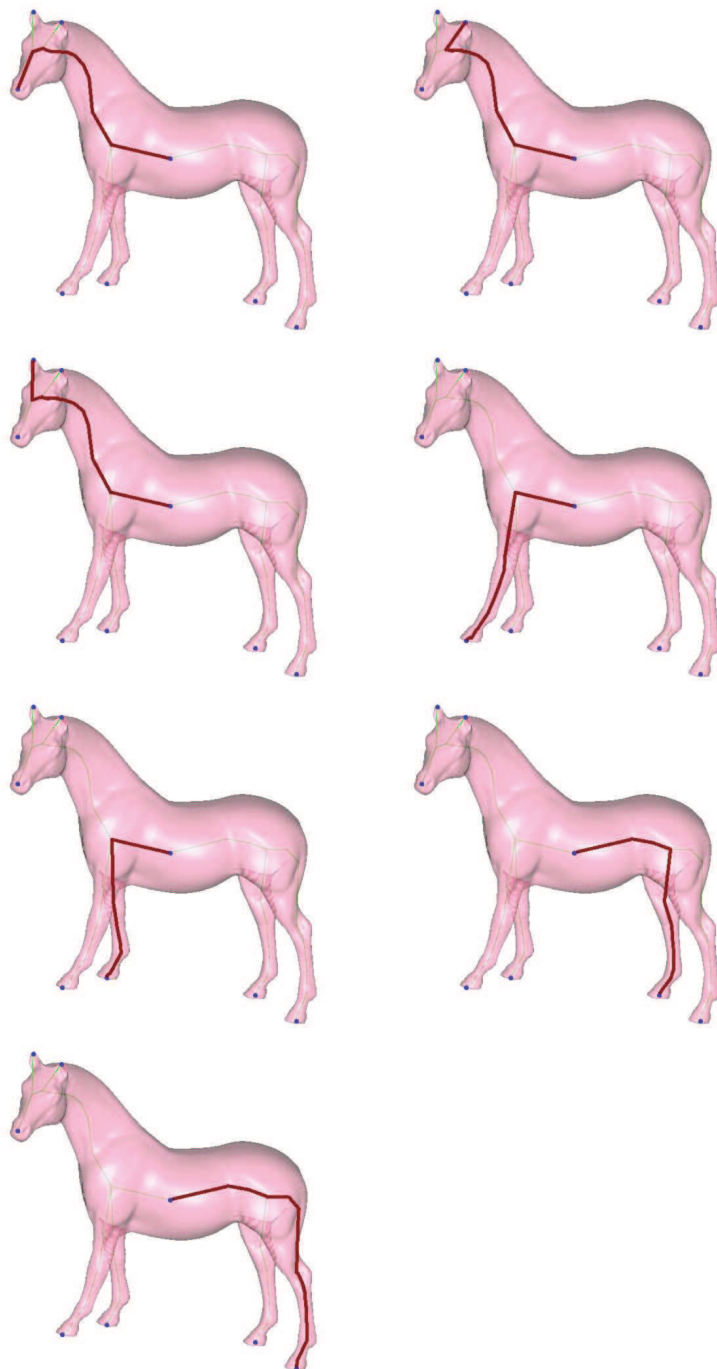


FIGURE 4.12: Shortest paths between the mesh centroid and an endpoint on the skeleton.

Algorithm 2 Endpoint correspondence

Let $E = (\mathbf{v}_i)_{i=1,\dots,n_1}$ and $\tilde{E} = (\tilde{\mathbf{v}}_j)_{j=1,\dots,n_2}$ be two sets of endpoints.

For each endpoint $\mathbf{v}_i \in E$:

- 1: Compute a dissimilarity measure between \mathbf{v}_i and all the nodes in \tilde{E}
- 2: Find the node $\tilde{\mathbf{v}}_j$ with the minimum dissimilarity and assign its correspondence to \mathbf{v}_i
- 3: Delete \mathbf{v}_i and $\tilde{\mathbf{v}}_j$ from the list of nodes in E and \tilde{E} , respectively

Repeat steps 1-3 for all nodes in E until one of the node sets E or \tilde{E} is empty

4.3.3 Matching endpoints using skeleton paths

Endpoint Features: When generating the skeletal Reeb graph of a 3D shape we assign three features to each endpoint of the skeleton. The first feature is the relative node area, which is equal to the area of the neighboring triangles of the endpoint divided by the total area of the 3D model. This feature provides important information about the endpoint as sometimes the skeletons of two models may look similar, albeit their shapes are completely different. Thus, adding this feature to an endpoint will help discriminate between endpoints based on the original 3D shape and not just its skeleton. The reason behind using the relative area is its invariance to scaling. The second feature assigned to an endpoint is the relative node path, which is equal to the sum of shortest path distances from each endpoint to all other endpoints of the skeleton (see Figure 4.11(b)) divided by the sum of the shortest paths from the mesh centroid (root node) to each endpoint. And the third feature is the relative centroid path, which is the shortest path distance from the mesh centroid to each endpoint (see Figure 4.12), divided by the sum of the shortest paths from the mesh centroid to all endpoints.

Endpoints Dissimilarity: Let G and \tilde{G} the spectral Reeb graphs of two 3D shapes \mathbb{M} and $\tilde{\mathbb{M}}$, respectively. The skeleton endpoints sets of G and \tilde{G} are denoted by $E = \{\mathbf{v}_i\}_{i=1,\dots,n_1}$ and $\tilde{E} = \{\tilde{\mathbf{v}}_j\}_{j=1,\dots,n_2}$, respectively. We define the local dissimilarity measure between two endpoints \mathbf{v}_i and $\tilde{\mathbf{v}}_j$ as the Euclidean distance

$$\Phi(\mathbf{v}_i, \tilde{\mathbf{v}}_j) = \|\boldsymbol{\omega}_i - \tilde{\boldsymbol{\omega}}_j\|, \quad (4.16)$$

between the 3D vectors $\boldsymbol{\omega}_i = (a_i, d\mathbf{v}_i, d\mathbf{c}_i)^T$ and $\tilde{\boldsymbol{\omega}}_j = (\tilde{a}_j, d\tilde{\mathbf{v}}_j, d\tilde{\mathbf{c}}_j)^T$, whose components are defined by:

- a_i and \tilde{a}_j are the relative node areas of \mathbf{v}_i and $\tilde{\mathbf{v}}_j$

- $d\mathbf{v}_i = \sum_{k=1}^{n_1} \delta(\mathbf{v}_i, \mathbf{v}_k) / \sum_{k=1}^{n_1} \delta(\mathbf{c}, \mathbf{v}_k)$ and $d\tilde{\mathbf{v}}_j = \sum_{k=1}^{n_2} \delta(\tilde{\mathbf{v}}_j, \tilde{\mathbf{v}}_k) / \sum_{k=1}^{n_2} \delta(\tilde{\mathbf{c}}, \tilde{\mathbf{v}}_k)$ are the relative node paths of \mathbf{v}_i and $\tilde{\mathbf{v}}_j$
- $d\mathbf{c}_i = \delta(\mathbf{c}, \mathbf{v}_i) / \sum_{k=1}^{n_1} \delta(\mathbf{c}, \mathbf{v}_k)$ and $d\tilde{\mathbf{c}}_j = \delta(\tilde{\mathbf{c}}, \tilde{\mathbf{v}}_j) / \sum_{k=1}^{n_2} \delta(\tilde{\mathbf{c}}, \tilde{\mathbf{v}}_k)$ are the relative centroid paths of \mathbf{v}_i and $\tilde{\mathbf{v}}_j$
- \mathbf{c} and $\tilde{\mathbf{c}}$ are the centroids of \mathbb{M} and $\tilde{\mathbb{M}}$, respectively
- $\delta(\cdot, \cdot)$ is the Dijkstra's shortest path distance.

Therefore, the geometric dissimilarity between two spectral Reeb graphs is given by the distance:

$$\mathcal{D}(G, \tilde{G}) = \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \Phi(\mathbf{v}_i, \tilde{\mathbf{v}}_j) = \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \|\boldsymbol{\omega}_i - \tilde{\boldsymbol{\omega}}_j\|. \quad (4.17)$$

The main algorithmic steps of the proposed graph matching approach are described in more details in Algorithm 3.

Algorithm 3 Proposed graph matching approach

Given two 3D objects \mathbb{M} and $\tilde{\mathbb{M}}$

- 1: Generate the skeletal Reeb graphs G and \tilde{G} of \mathbb{M} and $\tilde{\mathbb{M}}$, respectively
 - 2: Apply graph pruning to remove non-salient nodes
 - 3: Find the skeleton endpoints sets $E = (\mathbf{v}_i)_{i=1, \dots, n_1}$ and $\tilde{E} = (\tilde{\mathbf{v}}_j)_{j=1, \dots, n_2}$ of G and \tilde{G} , respectively
 - 4: **for** all endpoints (\mathbf{v}_i) and $(\tilde{\mathbf{v}}_j)$ **do**
 - 5: Compute the relative node areas a_i and \tilde{a}_j of \mathbf{v}_i and $\tilde{\mathbf{v}}_j$, respectively
 - 6: Compute the relative node paths $d\mathbf{v}_i$ and $d\tilde{\mathbf{v}}_j$
 - 7: Compute the relative centroid paths $d\mathbf{c}_i$ and $d\tilde{\mathbf{c}}_j$
 - 8: **end for**
 - 9: Apply Algorithm 2 to find the correspondence between G and \tilde{G}
 - 10: Compute the dissimilarity $\mathcal{D}(G, \tilde{G})$ given by Eq. (4.17).
-

4.4 Experimental Results

In this section we present the results of the proposed framework. We implemented our algorithms using C++, OpenGL, and MATLAB. The experiments were performed on an iMac desktop computer with an Intel Core i5-2400S running at 2.50 GHz and 8 GB RAM.

An important property of any shape descriptor is the ability to match similar shapes even in the presence of noise. To that end, we start by demonstrating the robustness of the proposed skeletonization algorithm to noise. Figure 4.13 depicts the extracted spectral Reeb graphs of a noise-free model and its noisy version using the proposed algorithm. It is evident that the spectral skeletonization algorithm shows a good preservation of the mesh topological structure. Note that the difference between the skeletons of the original model and the noisy model is minor and not readily noticeable.

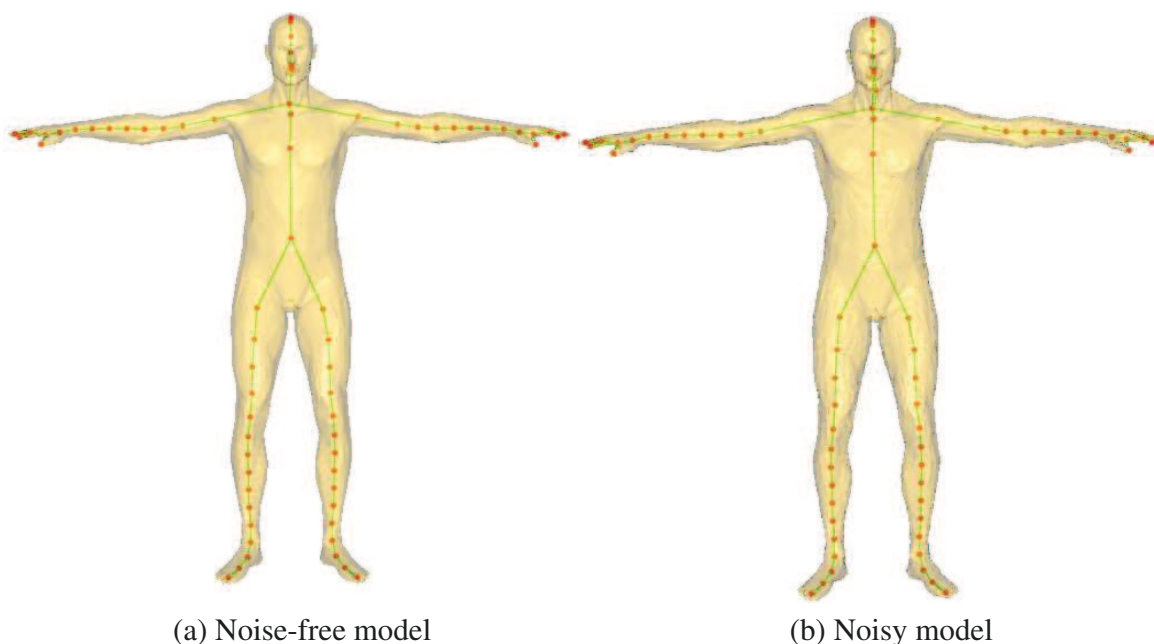


FIGURE 4.13: Robustness of spectral Reeb graph to noise.

We tested the performance of the proposed matching algorithm using the McGill Shape Benchmark [20]. This publicly available database provides a 3D shape repository, which contains 255 objects that are divided into ten categories, namely, ‘Ants’, ‘Crabs’, ‘Spectacles’, ‘Hands’, ‘Humans’, ‘Octopuses’, ‘Pliers’, ‘Snakes’, ‘Spiders’, and ‘Teddy Bears’. Sample models from this database are shown in Figure 4.14.

The McGill’s database objects are represented by voxel grids as well as by triangle meshes. Table 4.1 shows that the proposed approach yields correct matching results, where a low value (displayed in boldface with a colored box around it for emphasis) of the dissimilarity measure indicates that the objects are more similar.

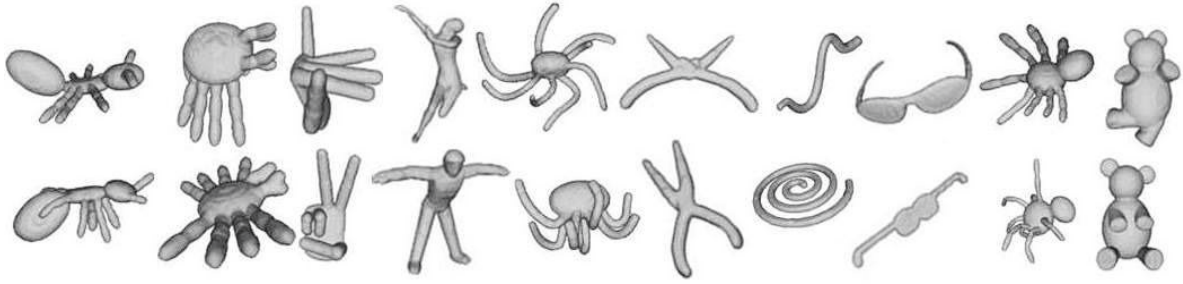


FIGURE 4.14: Sample shapes from McGill's Articulated Shape Database. Only two shapes for each of the 10 classes are shown.

We also compared our approach with spherical harmonics (SH) [26], medial surfaces (MS) [20], and Reeb graph patch dissimilarity (RGPD) [83]. The results show that our method achieves better retrieval results than the spherical harmonics as shown in Table 4.2, where the top ten retrieved 3D objects are displayed (top-to-bottom). As can be seen in Table 4.2, the proposed approach returns correct results whereas the spherical harmonics method yields poor retrieval results (columns 2, 4, and 6). Also, the proposed algorithm performs slightly better than the RGPD approach.

To carry out comparison experiments on the entire benchmark of articulated 3D objects, we evaluated the retrieval performance of the proposed approach using the standard information retrieval evaluation measure of precision *versus* recall curve, where

$$\text{precision} = \frac{\text{No. relevant objects retrieved}}{\text{Total No. objects retrieved}} \quad (4.18)$$

and

$$\text{recall} = \frac{\text{No. relevant objects retrieved}}{\text{Total No. relevant objects in the collection}}. \quad (4.19)$$

A precision-recall curve that is shifted upwards and to the right indicates superior performance. A perfect retrieval result produces a horizontal curve (at precision = 1.0), indicating that all the shapes within the query objects class are returned as the top ranked matches. It is evident from Figure 4.15 that our method significantly outperforms spherical harmonics, medial surfaces, and the Reeb graph path dissimilarity approach.

Finally, we tested the performance of the proposed approach on the Princeton Shape Benchmark [29]. The Princeton Shape Benchmark is a publicly available database of 3D polygonal models collected from the Word Wide Web, along with a set of software tools that are widely

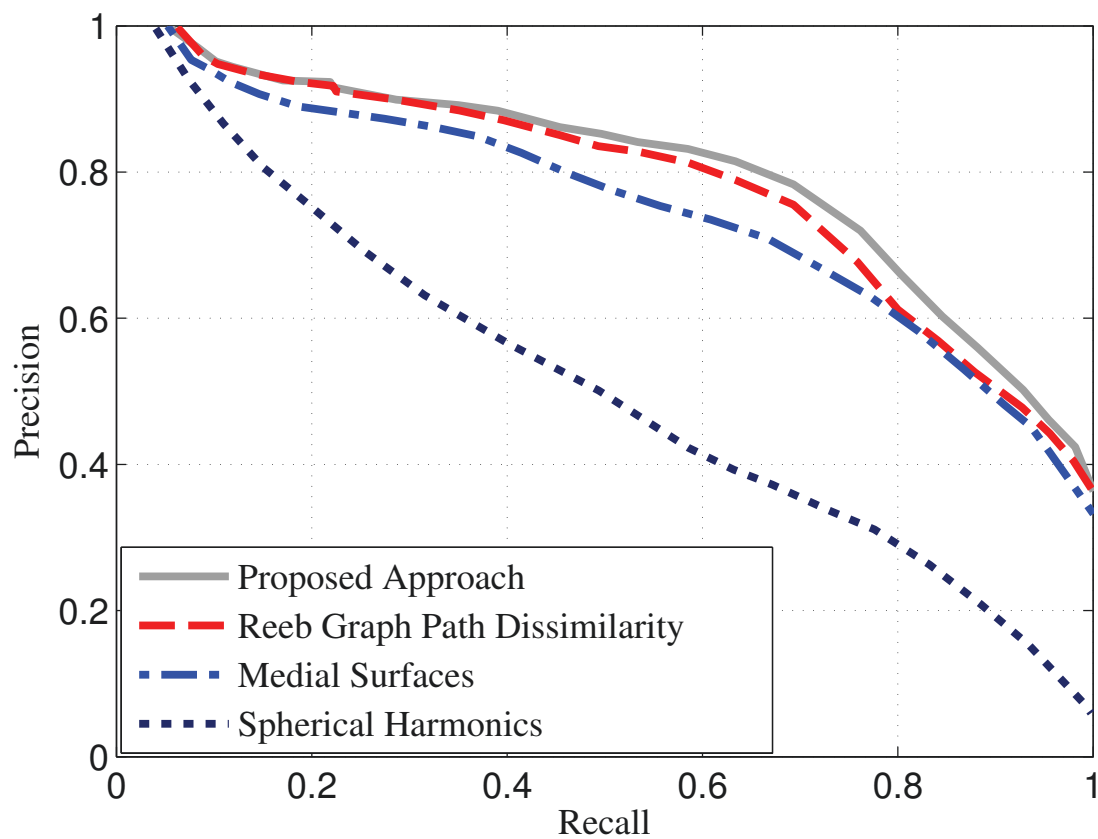


FIGURE 4.15: Precision vs. Recall curves for Reeb graph path dissimilarity, spherical harmonics, medial surfaces, Reeb graph path dissimilarity, and proposed approach using the McGill Shape Benchmark 4.14.

used by researchers to report their shape matching and retrieval results and compare them to the results of existing algorithms. As can be seen in Table 4.3, the proposed approach shows superior performance over spherical harmonics, where the top five retrieved 3D objects are displayed (top-to-bottom).

TABLE 4.1: Matching results using the proposed approach. Each database object is matched against all the other objects in the database. Each cell shows the dissimilarity measure $\mathcal{D}(G, \tilde{G})$ between two objects selected from the database. The smallest value corresponds to the correct match.

							
	0.0124	0.1127	0.1216	0.1258	0.1131	0.1344	0.1257
	0.1116	0.0073	0.1136	0.1297	0.1227	0.1124	0.1131
	0.1311	0.1142	0.0653	0.1356	0.1315	0.1171	0.1137
	0.1146	0.1329	0.1113	0.0055	0.1332	0.1621	0.1552
	0.1193	0.1248	0.1342	0.1421	0.1131	0.1572	0.1592
	0.1327	0.1109	0.1152	0.1474	0.11719	0.1021	0.1116
	0.1223	0.1128	0.1175	0.1453	0.1623	0.1121	0.0042

TABLE 4.2: Retrieval results using the McGill Shape Benchmark. The query shapes are shown in the second row. The top ten retrieved objects (top-to-bottom) using spherical harmonics (SH), Reeb graph path dissimilarity (RGPD), and our proposed approach (SRG) are shown in rows 5 to 14.






















































































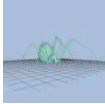


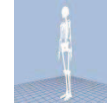

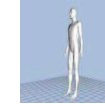

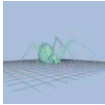


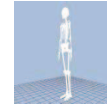
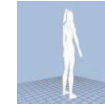
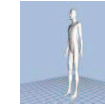
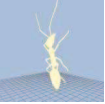







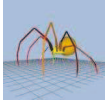



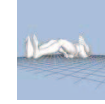


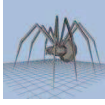

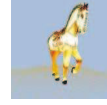



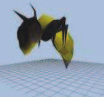



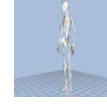


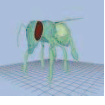
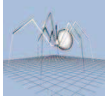


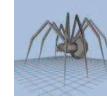


Query								
								
Retrieved Objects								
SRG	RGPD	SH	SRG	RGPD	SH	SRG	RGPD	SH
								
								
								
								
								
								
								
								
								

TABLE 4.3: Retrieval results using Princeton 3D Benchmark. The query shapes are shown in the second row. The top five retrieved objects (top-to-bottom) of our proposed approach.

Query						
						
Retrieved Objects						
						
						
						
						
						
						

Conclusions and Future Work

This thesis has presented two 3D mesh denoising techniques, namely the regularized weighted graph based kernel diffusion and the vertex-centered finite volume diffusion. We also proposed a spectral geometric approach for nonrigid 3D shape retrieval using the second eigenfunction of the Laplace-Beltrami operator on manifolds. We have demonstrated through extensive experiments the much better performance of the proposed methods in comparison with existing techniques in the literature.

In Section 5.1, the contributions made in each of the previous chapters and the concluding results drawn from the associated research work are presented. Suggestions for future research directions related to this thesis are also provided in Section 5.2.

5.1 Contributions of the Thesis

5.1.1 Kernel Weighted Diffusion

In Chapter 2, we introduced a simple and fast 3D shape smoothing technique using the concept of multivariate kernel density estimation [88]. The main idea behind our proposed approach is to use a regularized weighted kernel diffusion in a bid to avoid over-smoothing and to preserve the geometric structure of the 3D mesh data, while effectively removing undesirable noise. The experimental

results showed that our proposed technique is robust and outperforms existing approaches.

5.1.2 Finite Volume Diffusion

In Chapter 3, we proposed a simple and fast 3D mesh denoising method in the finite volume framework [89]. The main idea behind our approach is to use a vertex-centered finite volume scheme in conjunction with local weights defined in terms of the mesh covariance fractional anisotropy. The approach shows a good performance in preserving the geometric structure of the 3D mesh data, while effectively removing undesirable noise. The experimental results showed that our method outperforms the multiscale anisotropic Laplacian technique.

5.1.3 Spectral Geometric Shape Recognition

In Chapter 4, we proposed the use of shortest path distance matching algorithm for shape skeletons constructed from the second eigenfunction of the Laplace-Beltrami operator [90, 91]. The better performance of the proposed framework was demonstrated on McGill’s articulated shape database compared to spherical harmonics, medial surfaces, and Reeb graph path dissimilarity. Additionally, we also use our algorithm on the Princeton’s shape benchmark and we showed that the proposed approach gives also satisfactory results for non-articulated shape models.

5.2 Future Research Directions

Several interesting research directions, motivated by this thesis, are discussed below:

5.2.1 Partial Matching of 3D Shapes

Our ongoing research efforts are currently focused on further improving the results by appropriately choosing more discriminatory endpoint features. We also intend to extend our approach to partial matching of 3D shapes.

5.2.2 Unifying Topology and Geometry

Viewed from the Morse-theoretic perspective, the eigenfunctions of LB operator capture the topological features of shapes. Integrating the surface geometry into the shape skeleton would provide

a unified intrinsic framework of both topology and geometry for deformable 3D shape retrieval.

5.2.3 From Image Processing to Geometry Processing

Generally speaking, this thesis provides two bridges to borrow ideas from image processing for geometry processing. More precisely, it generalizes methods in the Euclidean space to the weighted graph space, resulting in a fruitful way to understand 3D shapes by extending sophisticated methods in image domain via these tools. Our not-so-distant future plan is to explore other tools to link these two fields, such as finding a proper generalization of sparse coding and low rank matrix recovery based methods in the image domain for 3D surfaces.

References

- [1] G. Taubin, “A signal processing approach to fair surface design,” in *Proc. ACM SIGGRAPH*, 1995, pp. 351–358.
- [2] Y. Ohtake, A. Belyaev, and I. Bogaevski, “Polyhedral surface smoothing with simultaneous mesh regularization,” in *Proc. Geometric Modeling and Processing*, 2000, pp. 229–237.
- [3] G. Taubin, “Linear anisotropic mesh filtering,” IBM Research Report, Tech. Rep. RC2213, 2001.
- [4] S. Petitjean, “A survey of methods for recovering quadrics in triangle meshes,” *ACM Computing Surveys*, vol. 34, no. 2, pp. 211–262, 2002.
- [5] Y. Shen and K. Barner, “Fuzzy vector median-based surface smoothing,” *IEEE Trans. Visualization and Computer Graphics*, vol. 10, no. 3, pp. 252–265, 2004.
- [6] H. Yagou, Y. Ohtake, and A. Belyaev, “Mesh smoothing via mean and median filtering applied to face normals,” in *Proc. Geometric Modeling and Processing*, 2002, pp. 124–131.
- [7] S. Fleishman, I. Drori, and D. Cohen-Or, “Bilateral mesh denoising,” in *Proc. ACM SIGGRAPH*, 2003, pp. 950–953.
- [8] T. Jones, F. Durand, and M. Desbrun, “Non-iterative, feature preserving mesh smoothing,” in *Proc. ACM SIGGRAPH*, 2003, pp. 943–949.
- [9] M. Desbrun, M. Meyer, P. Schröder, and A. Barr, “Implicit fairing of irregular meshes using diffusion and curvature flow,” in *Proc. ACM SIGGRAPH*, 1999, p. 317324.

- [10] Y. Zhang and A. Ben Hamza, "Vertex-based diffusion for 3d mesh denoising," *IEEE Trans. Image Processing*, vol. 16, no. 4, pp. 1036–1045, 2007.
- [11] W. Rey, *Introduction to Robust and Quasi-Robust Statistical Methods*. Springer, 1983.
- [12] Y. Yang, H. Lin, and Y. Zhang, "Content-based 3-D model retrieval: A survey," *IEEE Trans. Systems, Man, and Cybernetics, Part C*, vol. 37, no. 6, pp. 1081–1098, 2007.
- [13] A. DelBimbo and P. Pala, "Content-based retrieval of 3D models," *ACM Trans. Multimedia Computing, Communications, and Applications*, vol. 2, no. 1, pp. 20–43, 2006.
- [14] J. Tangelder and R. Veltkamp, "A survey of content based 3D shape retrieval methods," *Multimedia Tools and Applications*, vol. 39, no. 3, pp. 441–471, 2008.
- [15] B. Bustos, D. Keim, D. Saupe, T. Schreck, and D. Vranic, "Feature-based similarity search in 3D object databases," *ACM Computing Surveys*, vol. 37, no. 4, pp. 345–387, 2005.
- [16] V. Jain and H. Zhang, "A spectral approach to shape-based retrieval of articulated 3D models," *Computer-Aided Design*, vol. 39, no. 5, pp. 398–407, 2007.
- [17] Y. Shinagawa, T. Kunii, and Y. Kergosien, "Surface coding based on morse theory," *IEEE Computer Graphics and Applications*, vol. 11, no. 5, pp. 66–78, 1991.
- [18] K. Siddiqi, A. Shokoufandeh, S. Dickinson, and S. Zucker, "Shock graphs and shape matching," *Int. Journal of Computer Vision*, vol. 35, no. 1, pp. 13–32, 1999.
- [19] M. Hilaga, Y. Shinagawa, T. Kohmura, and T. Kunii, "Topology matching for fully automatic similarity estimation of 3D shapes," in *Proc. SIGGRAPH*, 2001, pp. 203–212.
- [20] K. Siddiqi, J. Zhang, D. Macrini, A. Shokoufandeh, S. Bouix, and S. Dickinson, "Retrieving articulated 3-D models using medial surfaces," *Machine Vision and Applications*, vol. 19, no. 4, pp. 261–275, 2008.
- [21] N. Cornea, M. Demirci, D. Silver, A. Shokoufandeh, S. Dickinson, and P. Kantor, "3D object retrieval using many-to-many matching of curve skeletons," in *Proc. Int. Conf. Shape Modeling and Applications*, 2005, pp. 368–373.

- [22] M. Hassouna and A. Farag, “Variational curve skeletons using gradient vector flow,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 31, no. 12, pp. 2257–2274, 2009.
- [23] A. Tagliasacchi, H. Zhang, , and D. Cohen-Or, “Curve skeleton extraction from incomplete point cloud,” *ACM Trans. Graphics*, vol. 28, no. 3, 2009.
- [24] M. Ankerst, G. Kastenmüller, H. Kriegel, and T. Seidl, “3D shape histograms for similarity search and classification in spatial databases,” in *Proc. Int. Sympo. Advances in Spatial Databases*, 1999, pp. 207–226.
- [25] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin, “Shape distributions,” *ACM Trans. Graphics*, vol. 21, no. 4, pp. 807–832, 2002.
- [26] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz, “Rotation invariant spherical harmonic representation of 3D shape descriptors,” in *Proc. Eurographics/ACM SIGGRAPH Symposium on Geometry Processing (SGP’03)*, 2003, pp. 156–164.
- [27] A. Ion, N. Artner, G. Peyré, W. Kropatsch, and L. Cohen, “Matching 2D and 3D articulated shapes using the eccentricity transform,” *Computer Vision and Image Understanding*, vol. 115, pp. 817–834, 2011.
- [28] D.-Y. Chen, X.-P. Tian, Y.-T. Shen, and M. Ouhyoun, “On visual similarity based 3D model retrieval,” *Computer Graphics Forum*, vol. 22, no. 3, pp. 223–232, 2003.
- [29] P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser, “The Princeton shape benchmark,” in *Proc. Shape Modeling International (SMI’04)*, 2004, pp. 167–178.
- [30] R. Coifman and S. Lafon, “Diffusion maps,” *Applied and Computational Harmonic Analysis*, vol. 21, no. 1, pp. 5–30, 2006.
- [31] P. Perona and J. Malik, “Scale space and edge detection using anisotropic diffusion,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 12, no. 7, pp. 629–639, 1990.
- [32] L. Rudin, S. Osher, and E. Fatemi, “Nonlinear total variation based noise removal algorithms,” *Physica D*, vol. 60, pp. 259–268, 1992.

- [33] Y. You, W. Xu, A. Tannenbaum, and M. Kaveh, “Behavioral analysis of anisotropic diffusion in image processing,” *IEEE Trans. Image Processing*, vol. 5, no. 11, pp. 1539–1553, 1996.
- [34] P. Charbonnier, L. Blanc-Féraud, G. Aubert, , and M. Barlaud, “Deterministic edge-preserving regularization in computed imaging,” *IEEE Trans. Image Processing*, vol. 6, no. 2, pp. 298–311, 1997.
- [35] J. Weickert, *Anisotropic Diffusion in Image Processing*. Teubner-Verlab, 1998.
- [36] Y. Bao and H. Krim, “Smart nonlinear diffusion: a probabilistic approach,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 26, no. 1, pp. 63–72, 2004.
- [37] M. Giaquinta and S. Hildebrandt, *Calculus of variations I*. Springer-Verlag, 2004.
- [38] M. Desbrun, M. Meyer, P. Schröder, and A. Bar, “Anisotropic feature-preserving denoising of height fields and bivariate data,” in *Proc. Graphics Interface*, 2000, pp. 145–152.
- [39] U. Clarenz, U. Diewald, and M. Rumpf, “Anisotropic geometric diffusion in surface processing,” in *Proc. IEEE Visualization*, 2000, pp. 397–405.
- [40] U. Clarenz, U. Diewald, , and M. Rumpf, “Processing textured surfaces via anisotropic geometric diffusion,” *IEEE Trans. Image Processing*, vol. 13, no. 2, pp. 248–261, 2004.
- [41] C. Bajaj and G. Xu, “Anisotropic diffusion of subdivision surfaces and functions on surfaces,” *ACM Trans. Graphics*, vol. 22, no. 1, pp. 4–32, 2003.
- [42] T. Tasdizen, R. Whitaker, P. Burchard, and S. Osher, “Geometric surface smoothing via anisotropic diffusion of normals,” in *Proc. IEEE Visualization*, 2002, pp. 125–132.
- [43] ———, “Geometric surface processing via normal maps,” *ACM Trans. Graphics*, vol. 22, no. 4, pp. 1012–1033, 2003.
- [44] K. Hildebrandt and K. Polthier, “Anisotropic filtering of non-linear surface features,” *Computer Graphics Forum*, vol. 23, no. 3, pp. 391–400, 2004.
- [45] B. Silverman, *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, 1986.

- [46] A. Elmoatz, O. Lezoray, and S. Bogleux, “Nonlocal discrete regularization on weighted graphs: A framework for image and manifold processing,” *IEEE Trans. Image Processing*, vol. 17, no. 7, pp. 1047–1060, 2008.
- [47] C. Tomasi and R. Manduchi, “Bilateral filtering for gray and color images,” in *Proc. IEEE ICCV*, 1998, pp. 839–846.
- [48] D. Barash, “A fundamental relationship between bilateral filtering, adaptive smoothing, and the nonlinear diffusion equation,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 6, pp. 844–847, 2002.
- [49] A. Buades, B. Coll, and J. Morel, “A review of image denoising algorithms, with a new one,” *Multiscale Modeling and Simulation*, vol. 4, no. 2, pp. 490–530, 2005.
- [50] K. Tarmissi and A. Ben Hamza, “Multivariate kernel diffusion for surface denoising,” *Signal, Image and Video Processing*, vol. 5, no. 2, pp. 191–201, 2011.
- [51] Z. Karni and C. Gotsman, “Spectral compression of mesh geometry,” in *Proc. ACM SIGGRAPH*, 2000, pp. 279–286.
- [52] R. LeVeque, *Finite Volume Methods for Hyperbolic Problems*. Cambridge University Press, 2002.
- [53] C. Hirsch, *Numerical Computation of Internal and External Flows*. Butterworth-Heinemann, 2007.
- [54] P. Rousseeuw and A. Leroy, *Robust Regression and Outlier Detection*. Wiley, 1987.
- [55] P. Basser and C. Pierpaoli, “Microstructural and physiological features of tissues elucidated by quantitative-diffusion-tensor mri,” *Journal of magnetic resonance Series B*, vol. 111, no. 3, pp. 209–219, 1996.
- [56] H. Huang and U. Ascher, “Fast denoising of surface meshes with intrinsic texture,” *Inverse Problems*, vol. 24, no. 3, 2008.
- [57] A. Fomenko and T. Kunii, *Topological Modeling for Visualization*. Springer-Verlag, 1997.

- [58] T. Grigorishin, G. Abdel-Hamid, and Y. Yang, “Skeletonization: an electrostatic field-based approach,” *Pattern Analysis and Applications*, vol. 1, no. 3, pp. 163–177, 1998.
- [59] P. Min, M. Kazhdan, and T. Funkhouserz, “A comparison of text and shape matching for retrieval of online 3D models,” in *Proc. ECDDL*, 2004, pp. 209–220.
- [60] J. Milnor, *Morse Theory*. Princeton University Press, 1963.
- [61] X. Ni, M. Garland, and J. Hart, “Fair morse functions for extracting the topological structure of a surface mesh,” in *Proc. Int. Conf. Computer Graphics and Interactive Techniques*, 2004, pp. 613–622.
- [62] J. Tierny, J.-P. Vandeborre, and M. Daoudi, “Partial 3D shape retrieval by Reeb pattern unfolding,” *Computer Graphics Forum*, vol. 28, no. 1, pp. 41–55, 2008.
- [63] D. Aouada and H. Krim, “Squigraphs for fine and compact modeling of 3D shapes,” *IEEE Trans. Image Processing*, vol. 19, no. 2, pp. 306–321, 2010.
- [64] S. Biasotti, D. Giorgi, M. Spagnuolo, and B. Falcidieno, “Reeb graphs for shape analysis and applications,” *Theoretical Computer Science*, vol. 392, no. 1-3, pp. 5–22, 2007.
- [65] V. Pascucci, G. Scorzelli, P. Bremer, and A. Mascarenhas, “Robust on-line computation of Reeb graphs: simplicity and speed,” *ACM Trans. Graphics*, vol. 26, no. 3, 2007.
- [66] G. Patane, M. Spagnuolo, and B. Falcidieno, “A minimal contouring approach to the computation of the Reeb graph,” *IEEE Trans. Visualization and Computer Graphics*, vol. 15, no. 4, pp. 583–595, 2009.
- [67] M. Belkin, P. Niyogi, and V. Sindhvani, “Manifold regularization: a geometric framework for learning from labeled and unlabeled examples,” *Journal of Machine Learning Research*, vol. 7.
- [68] B. Lévy, “Laplace-Beltrami eigenfunctions: Towards an algorithm that “understands” geometry,” in *Proc. IEEE Int. Conf. Shape Modeling and Applications*, 2006, p. 13.

- [69] M. Reuter, “Hierarchical shape segmentation and registration via topological features of Laplace-Beltrami eigenfunctions,” *Int. Journal of Computer Vision*, vol. 89, no. 2, pp. 287–308, 2010.
- [70] R. Rustamov, “Laplace-Beltrami eigenfunctions for deformation invariant shape representation,” in *Proc. Symposium on Geometry Processing*, 2007, pp. 225–233.
- [71] A. Bronstein, M. Bronstein, L. Guibas, and M. Ovsjanikov, “Shape Google: Geometric words and expressions for invariant shape retrieval,” *ACM Trans. Graphics*, vol. 30, no. 1, 2011.
- [72] J. Sun, M. Ovsjanikov, and L. Guibas, “A concise and provably informative multi-scale signature based on heat diffusion,” *Computer Graphics Forum*, vol. 28, no. 5, pp. 1383–1392, 2009.
- [73] K. Gøbal, J. A. Bærentzen, H. Aanæs, and R. Larsen, “Shape analysis using the auto diffusion function,” *Computer Graphics Forum*, vol. 28, no. 5, pp. 1405–1513, 2009.
- [74] Y. Shi, R. Lai, S. Krishna, N. Sicotte, I. Dinov, and A. Toga, “Anisotropic Laplace-Beltrami eigenmaps: Bridging Reeb graphs and skeletons,” in *Proc. IEEE Computer Vision and Pattern Recognition Workshops (CVPRR’08)*, 2008, pp. 1–7.
- [75] E. Kreyszig, *Introduction to Differentiable Geometry and Riemannian Geometry*. University of Toronto Press, 1969.
- [76] A. Bronstein, M. Bronstein, and R. Kimmel, *Numerical Geometry of Non-rigid Shapes*. Springer, 2008.
- [77] S. Rosenberg, *The Laplacian on a Riemannian Manifold*. Cambridge University Press, 1997.
- [78] M. Kac, “Can one hear the shape of a drum?” *The American Mathematical Monthly*, vol. 73, no. 4, pp. 1–23, 1966.
- [79] C. Gordon and D. Webb, “You can’t hear the shape of a drum,” *American Scientist*, vol. 84, no. 1, pp. 46–55, 1996.

- [80] S. Gallot, D. Hulin, and J. Lafontaine, *Riemannian Geometry*. Springer-Verlag, 1990.
- [81] M. Wardetzky, S. Mathur, F. Kälberer, and E. Grinspun, “Discrete Laplace operators: no free lunch,” in *Proc. Eurographics Symposium on Geometry Processing (SGP’07)*, 2007, pp. 33–37.
- [82] K. Uhlenbeck, “Generic properties of eigenfunctions,” *American Journal of Mathematics*, vol. 98, no. 4, pp. 1059–1078, 1976.
- [83] W. Mohamed and A. Ben Hamza, “Reeb graph path dissimilarity for 3D object matching and retrieval,” *The Visual Computer*, vol. 28, no. 3, pp. 305–318, 2012.
- [84] X. Bai and L. Latecki, “Path similarity skeleton graph matching,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 30, no. 7, pp. 1282–1292, 2008.
- [85] M. Demirci, A. Shokoufandeh, Y. Keselman, L. Bretzner, and S. Dickinson, “Object recognition as many-to-many feature matching,” *Int. Journal of Computer Vision*, vol. 69, no. 2, pp. 203–222, 2006.
- [86] H. Ling and D. Jacobs, “Shape classification using inner-distance,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 2, pp. 286–299, 2007.
- [87] X. Bai, L. Latecki, and W.-Y. Liu, “Skeleton pruning by contour partitioning with discrete curve evolution,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 3, pp. 449–462, 2007.
- [88] A. Kacem and A. Ben Hamza, “Regularized kernel weighted diffusion for 3D shape smoothing,” *International Journal of Computer Graphics*, vol. 3, no. 2, pp. 27–50, 2012.
- [89] —, “Fast surface denoising using finite volumes of the dual mesh,” in *Proc. Canadian Conf. Computer and Robot Vision*, 2012, pp. 70–77.
- [90] A. Kacem, W. Mohamed, and A. Ben Hamza, “Spectral geometric descriptor for deformable 3D shape matching and retrieval,” in *Proc. Int. Conf. Image Analysis and Recognition, LNCS*, 2013.

[91] —, “Nonrigid 3D shape retrieval using a spectral geometric signature,” *Under review*, 2013.