

Discrete Laplace Operator: Theory and Applications

Dissertation

Presented in Partial Fulfillment of the Requirements for the Degree
Doctor of Philosophy in the Graduate School of The Ohio State
University

By

Pawas Ranjan, B.Tech.

Graduate Program in Computer Science and Engineering

The Ohio State University

2012

Dissertation Committee:

Tamal K. Dey, Advisor

Yusu Wang, Advisor

Rephael Wenger

Rick Parent

Daniel Burghelea

© Copyright by

Pawas Ranjan

2012

Abstract

The eigen-structures (eigenvalues and eigenfunctions) of the Laplace-Beltrami operator have been widely used in a broad range of application fields that include mesh smoothing, compression, editing, shape segmentation, matching, and parametrization, among others. While the Laplace operator is defined (mathematically) for a smooth domain, the underlying manifold is often approximated by a discrete mesh. Hence, the spectral structure of the manifold Laplacian is estimated from some discrete Laplace operator constructed from this mesh.

Recently, several different discretizations have been proposed, each with its own advantages and limitations. Although the eigen-structures have been found to be useful in graphics, not much is known about their behavior when a surface is deformed or modified. The objective of my thesis is two-fold. One is to study, and to develop theory for, changes in the eigen-structures of the discrete Laplace operator as the underlying mesh is changed. The other is to explore applications for the spectral theory of shape perturbations in areas like shape matching and deformation.

In particular, our work shows that the discrete Laplace is stable against noise and sampling. We also show that both the discrete and continuous Laplace change continuously as the underlying mesh or surface is deformed continuously, without introducing changes to the topology. Not only do these results help in providing a better theoretical understanding of the discrete Laplace operator, they also give us a

solid base for developing applications. Indeed, we present two such applications: one that deals with shape matching and another that performs fast mesh deformations.

Specifically, combining our theoretical results with concepts from persistent homology, we create a concise global shape signature that can be used for matching different shapes. Given our results regarding similarity of eigen-structures of similar shapes, our matching algorithm allows us to match even partially scanned or incomplete models, regardless of their pose or orientation. We also present a framework that uses eigenvectors to create an implicit skeleton of a shape and use it to deform the shape, producing smooth and natural looking deformations. By using the eigenvectors, we are able to reduce the problem size from the number of mesh vertices (hundreds to millions) to the number of eigenvectors used (tens to hundreds).

To my parents, for their untiring support and patience.

Acknowledgments

I am grateful to my advisors Prof. Tamal Dey and Prof. Yusu Wang for their support, their belief and their help which got me through some very tough times, both academic and personal. Their enthusiasm and creative thinking drove and shaped not only my work, but also my outlook. They introduced me to a world that artfully combines mathematics and computer science, and for that I am forever in their debt. I would also like to thank my lab mates, especially Issam, Josh, Kuiyu and Oleksiy for their help and support and for sharing their vast knowledge with the rest of us. This work would not have been possible without their help. They taught me a very valuable lesson indeed: when it comes to writing quality papers, conversations with friends and peers can be just as instrumental as advisors.

I would also like to thank Prof. Rephael Wenger, Prof. Mikhail Belkin, Prof. Rick Parent and Prof. Daniel Burghelea for agreeing to be on my candidacy and defense committees. Their questions, comments and suggestions helped vastly improve this dissertation.

Finally, I would like to thank my parents for bringing me into this world and indulging my selfish request of traveling halfway around the globe to pursue higher studies. Nothing would have been possible without their support and patience.

Vita

2002	AISSCE, Kendriya Vidyalaya, IIT Powai
2006	B.Tech Computer Engg., Dr. B.A.T.U., Lonere
2007-present	Direct PhD, Dept. Of Computer Science and Engg., The Ohio State University.

Publications

Research Publications

T. K. Dey, P. Ranjan, Y. Wang “Convergence, Stability, and Discrete Approximation of Laplace Spectra”. *Proceedings of the Twenty-first Annual ACM-SIAM Symposium on Discrete Algorithms (2010)*, 650–663.

T. K. Dey, K. Li, C. Luo, P. Ranjan, I. Safa, Y. Wang “Persistent Heat Signature for Pose-oblivious Matching of Incomplete Models”. *Proceedings of the Eighth Annual Eurographics Symposium on Geometry Processing (2010)*, 1545:1554.

T. K. Dey, P. Ranjan, Y. Wang “Eigen Deformations of 3D Models”. *The Visual Computer*, Volume 28, Issue 6-8, 585-595.

Fields of Study

Major Field: Computer Science and Engineering

Table of Contents

	Page
Abstract	ii
Dedication	iv
Acknowledgments	v
Vita	vi
List of Tables	x
List of Figures	xi
1. Introduction	1
1.1 Discrete Laplace Operator	2
1.1.1 Properties	4
1.1.2 Types	5
1.2 Applications	8
1.2.1 Use of eigenvalues	8
1.2.2 Use of eigenvectors	9
1.2.3 Use of eigenprojections	10
1.3 Contribution	12
1.3.1 Convergence and Stability of the mesh Laplacian spectra . .	13
1.3.2 Stability under topological noise	13
1.3.3 Applications in shape matching	14
1.3.4 Using eigenvectors for shape deformation	15
2. Discrete Laplace Operator	16
2.1 Mesh Laplace Operator	16
2.2 Gaussian-weight graph Laplacian	18

2.3	Heat operator	19
3.	Convergence, Stability, and Discrete Approximation of Laplace Spectra [26]	21
3.1	Approach Overview	23
3.1.1	Overview of Approaches and Results	23
3.2	Perturbation of Manifold and Stability	26
3.3	Spectra Convergence between Discrete and Continuous Laplacians .	34
3.4	Experiments	45
3.5	Conclusion and Discussion	48
4.	Weighted Graph Laplace Operator under Topological Noise [27]	50
4.1	Problem Formulation	52
4.2	Step 1: Correspondences	54
4.2.1	Bipartite graph construction	55
4.2.2	Bipartite Matching in G	58
4.3	Step 2: Bounding Spectra Distance	68
4.4	Experiments	72
5.	Persistent Heat Signature for Pose-oblivious Matching of Incomplete Models [24]	76
5.1	Heat Kernel Signature	79
5.1.1	Heat Kernel Signature	80
5.1.2	Discrete setting	81
5.2	Persistent Heat Maxima	81
5.2.1	Persistence	82
5.2.2	Region merging	85
5.3	Matching	90
5.4	Results	91
5.5	Conclusion and Discussion	96
6.	Eigen Deformation of 3D Models [25]	98
6.1	Eigen-framework	103
6.1.1	Eigen-skeleton	104
6.2	Algorithm	106
6.2.1	Step 1: Coarse Guess-Target Configuration	107
6.2.2	Step 2: Eigen-skeleton Deformation	108
6.2.3	Step 3: Shape Recovery	112
6.3	Implementation	114
6.3.1	Recovery details	114

6.3.2	Choice of number of eigenvectors	115
6.3.3	Additional modifications	117
6.3.4	Interactivity	119
6.4	Results	119
6.5	Conclusion and Discussion	123
7.	Conclusion	125
	Bibliography	129

List of Tables

Table	Page
5.1 Query models and top five matches returned for each query. Letters C, P, I indicate complete, partial, and incomplete models respectively.	92
5.2 Each entry shows Top-3 / Top-5 hit rates for our method, EVD, and LFD. “32 incompl.” includes both partial and incomplete queries and “18 compl.” includes only pose-altered queries. The database contains 300 models.	93
5.3 All experiments are carried out on a Dell computer with Intel 2.4GHz CPU and 6GB RAM.	95
5.4 Each entry shows Top-3 / Top-5 hit rates for our method using HKS, AGD and GC feature points.	96
6.1 Timing data (in seconds) for our algorithm	122
6.2 Comparison of relative RMS errors in deformations using as-rigid-as-possible (ARAP) and our method (ED)	123

List of Figures

Figure	Page
1.1 The horse model shown in (a) is reconstructed in (b)-(h) using the indicated number of eigenvectors of the mesh Laplacian. More eigenvectors are able to capture the the finer details.	8
1.2 Top 12 eigenvectors of the graph Laplacian. Nodal sets (vertices with zero eigenfunction value) are shown in gray.	10
3.1 Theorems relating different operators are shown on top of the arrows. Double arrows indicate the two main new results in this chapter, and lead to those results specified by dotted arrows.	25
3.2 Errors in the (a) eigenvalues and (b) eigenvectors of discrete Laplacian of meshes of unit sphere with increasing number of vertices.	45
3.3 Original, noisy, and non-uniform meshes for the same genus 3 surface. Bottom : comparison of their eigenvalues.	46
3.4 (a) Some near-isometric deformations of a human. (b) An example of non-isometric deformation. (c) Comparison of spectra computed from five isometric and two non-isometric deformations.	47
3.5 Snapshots of continuous deformation of an eight loop and plot of spectra of corresponding meshes.	48
4.1 (a) Dark region is the anchor-region R_M induced by the black anchor-nodes (other anchor-nodes are not shown): R_M consists of points from M within δ Euclidean distance to black points. Light regions are anomalous regions X_1, \dots, X_4 . (b) The intermediate region $R_N \subset M$ contains anomalous regions X_1 and X_4 fully, and X_2 partially. (c) The witness anchor-region $R_N^+ \subset N$ of R_M includes anomalous regions Y_1, Y_2 and Y_4 fully.	56

4.2	Left: Connecting two tori with increasingly larger bridges. Right: Comparison of eigenvalues	73
4.3	Left: Increasingly large perturbation of points on a surface. Right: Comparison of eigenvalues	73
4.4	Left: Increasingly large region of topological change on a torus. Right: Comparison of eigenvalues	74
4.5	Topological changes on the Armadillo. Top Row: Original Armadillo model; and a variation with two fingers joined. Bottom Row: Another variation with two fingers touching; and a model with two fingers touching and another finger touching the nose. Right: Comparison of eigenvalues for $t^2 = 0.0001$	75
5.1	Given a query Armadillo model that is pose-altered, incomplete, and partially scanned, our method first computes the heat kernel signature function at a certain scale, and then extract a set of HKS maxima (red dots) using persistent homology. Feature vectors computed at these maxima are then used to search for the most similar models, be it complete, partial, or incomplete, in a shape database. A few top matches are shown. The black curves are the boundary curves of either partial or incomplete models. Correspondence between segmentations of different models is shown with consistent coloring.	77
5.2	Consistent identification of the persistent HKS maxima for different human / animal models in different poses. The two human models on the right are incomplete and partially scanned models with black curves being boundary curves.	83
5.3	Persistence based merging on an Airplane model. (a) Initially, every triangle represents a region; (b) shows the segmentation after merging all zero-persistence regions. The central triangle of every remaining region corresponds to a maximum of input function h . (c) A and B are two maxima, for the gray and orange regions, respectively, with A having a larger h value; (d) after the merging, A stays the maximum for the new region (gray colored).	89
5.4	Top five matches for an incomplete Octopus query model by our algorithm, EVD, and LFD.	94

6.1	Creating correct cages for meshes	99
6.2	Comparing with green coordinates	100
6.3	Comparisons when we stretch the arm and bend the leg of the armadillo. Note that our method handles stretching better than as-rigid-as-possible (ARAP), and extreme bending better than harmonic coordinates (HC).	102
6.4	Stretching the arm of the armadillo. Note that spectral mesh deformation (SMD), causes the entire mesh to deform in order to preserve the mesh volume.	103
6.5	The dragon model with its eigen-skeleton created using 8 eigenvectors	108
6.6	Left picture: A coarse discontinuous initial guess. Rotating the entire region of interest (colored red) causes the discontinuity at its boundary. We use the mesh connectivity information from the original mesh to further emphasize this point. Right picture: After step 2, we obtain a smooth transition across the boundary.	111
6.7	Adding details back to the dragon	113
6.8	Far Left: head of camel. Right: Eigen-skeleton of the head of the camel constructed using 8, 50 and 300 eigenvectors, respectively.	116
6.9	Adding details back to the dragon; Left: Directly from eigen-skeleton, Right: After iterative improvement	118
6.10	Bending a bumpy plane (dense mesh)	120
6.11	Bending a bumpy plane (coarse mesh)	120
6.12	Moving the arm of Neptune using our method and spectral mesh deformation (SMD)	121
6.13	Twisting a bar using our method	121
6.14	Editing the dancing children	122

6.15 Deforming the elk model	123
--	-----

Chapter 1: Introduction

Spectral methods for mesh processing and analysis rely on the eigenvalues, eigenvectors, or eigenspace projections derived from appropriately defined mesh operators to carry out desired tasks. Early work in this area can be traced back to the seminal paper by Taubin [90], where spectral analysis of mesh geometry based on a combinatorial Laplacian aids our understanding of the low-pass filtering approach to mesh smoothing. Over the two decades, the list of applications in the area of geometry processing which utilize the eigen-structures of a variety of mesh operators in different manners have been growing steadily. Many works presented so far draw parallels from developments in fields such as graph theory, computer vision, machine learning, graph drawing, numerical linear algebra, and high-performance computing. Particularly, the Laplace operator has become increasingly important due to the special properties exhibited by its eigenvectors.

While the Laplace operator is defined (mathematically) for a smooth domain, it is not possible to store such surfaces on computers. Instead, the underlying manifold is often approximated by a discrete mesh, and the spectral structure of the manifold Laplacian is estimated from some discrete Laplace operator constructed from this mesh. The discrete Laplace operator and its eigen-structures are capable of capturing information about the shape at varying levels of detail.

Although the Laplace operator and its eigen-structures have many interesting properties useful for practical applications, not much is known about the behavior of the Laplace operator and its discrete counterpart when a surface is deformed or modified. The objective of my thesis is to study the behavior of the eigen-structures of the discrete Laplace operator as the underlying mesh is changed, possibly introducing changes to the topology (like adding or removing loops or boundary components). The work will also explore possible applications of these eigen-structures and their properties for performing shape matching, segmentation and deformation.

The rest of the chapter is organized as follows: in Section 1.1, we will look at some important properties and different versions of the discrete Laplace operator. in Section 1.2, we will briefly discuss some of the applications of the various eigen-structures of the discrete Laplacian. Finally, in Section 1.3, we will give an overview of the contributions of this thesis in both theoretical and application-based fields.

1.1 Discrete Laplace Operator

Laplace Operator Consider a smooth, compact manifold \mathbf{M} of dimension m isometrically embedded in some Euclidean space \mathbb{R}^d . Given a twice continuously differentiable function $f \in C^2(\mathbf{M})$, let $\nabla_{\mathbf{M}}f$ denote the gradient vector field of f on \mathbf{M} . The Laplace-Beltrami operator $\Delta_{\mathbf{M}}$ of f is defined as the divergence of the gradient; that is, $\Delta_{\mathbf{M}}f = \text{div}(\nabla_{\mathbf{M}}f)$. For example, if \mathbf{M} is \mathbb{R}^2 , then its Laplacian has the familiar form $\Delta_{\mathbb{R}^2}f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$.

The Laplace operator has many interesting properties. For example, it is isometry invariant. i.e. the Laplace operator remains the same for shapes undergoing isometric deformations. It is also known that the eigenfunctions of the Laplace operator are

real-valued, orthonormal and form a basis for all real-valued square integrable functions defined on the manifold. Further study of the Laplace operator, as well as its generalizations for higher order exterior derivatives can be found in the book [77] by Steve Rosenberg.

Discrete Setting In practice, the underlying manifold is often approximated by a discrete mesh. Discrete Laplacian operators are linear operators that act on functions defined on such meshes. These functions are specified by their values at the vertices. Thus, if a mesh K has n vertices, then functions on K will be represented by vectors with n components and a mesh Laplacian will be described by an $n \times n$ matrix.

Loosely speaking, a discrete Laplacian operator locally takes the difference between the value of a function at a vertex and a weighted average of its values at the first-order or immediate neighbor vertices. Although we will briefly discuss generalizations, for introductory purposes a Laplacian, L , will have a local form given by

$$(Lf)_i = b_i^{-1} \sum_{j \in N(i)} w_{ij}(f_i - f_j) \tag{1.1}$$

The edge weights, w_{ij} , are symmetric: $w_{ij} = w_{ji}$. The factor b_i^{-1} is a positive number. Its expression as an inverse allows for formulation of the Laplace operator as a product of two symmetric positive-semidefinite matrices $L = B^{-1}W$. This, in turn, enables us to use efficient numerical solvers for generalized eigenproblem $W\phi = \lambda B\phi$. A Laplacian satisfying Equation (1.1) is called a first order Laplacian because its definition at a given vertex involves only the one-ring neighbors. On a manifold

triangle mesh, the matrix of such an operator will be sparse, with an average of seven non-zero entries per row.

1.1.1 Properties

The discrete Laplace operator and its eigen-structures (i.e. eigenvalues and eigenvectors) have several important properties, some of which follow from the continuous counterpart, that make it an important operator commonly used in spectral methods. We will discuss some of those properties in this section.

Zero row sum.

An important property imposed by Equation (1.1) is the zero row sum. If f is a constant vector, i.e., one all of whose components are the same, then f lies in the kernel of L , since $Lf = 0$ for an operator L with zero row sum. This implies that the constant vectors are eigenvectors of L with eigenvalue zero. It is known [63, 62] that the multiplicity of the zero eigenvalue equals the number of connected components in the mesh.

Eigenvector orthogonality.

An operator that is locally expressed by 1.1 can be factored into the product of a diagonal and a symmetric matrix

$$L = B^{-1}S$$

where B^{-1} is a diagonal matrix whose diagonal entries are the b_i^{-1} , and S is a symmetric matrix whose diagonal entries are given by $S_{ii} = \sum_{j \in N(i)} w_{ij}$ and whose off diagonal entries are w_{ij} . Since L itself is not symmetric in general, its eigenvectors

are not necessarily orthogonal with respect to the standard dot product. However, if we define the inner (or dot) product as

$$\langle f, g \rangle_B = f^T B g \tag{1.2}$$

then the eigenvectors of L are orthogonal with respect to that product.

Positive semi-definiteness.

Equation (1.1) does not guarantee that L is positive semidefinite, but such a property is desirable in a Laplacian operator: the zero eigenvalue associated with the constant (zero frequency) eigenvectors should be the smallest one. However, it is easy to show that if the weights w_{ij} s are non-negative, then L is positive semi-definite with respect to the appropriate inner product 1.2.

1.1.2 Types

Over the last two decades, many different discretizations of the Laplace operator have been proposed. Some of them simply behave like the continuous counterpart, while others have been shown to actually converge to the Laplace operator of the original smooth surface being discretized as a mesh. We will briefly introduce some of the discretizations in this section. Further discussions and convergence properties of these operators can be found in Chapter 2.

Combinatorial Graph Laplace.

Given a mesh with edges E , the adjacency matrix W of the mesh is defined as:

$$W_{ij} = \begin{cases} 1 & \text{if } (i, j) \in E \\ 0 & \text{otherwise} \end{cases}$$

The degree matrix D is defined as

$$D_{ij} = \begin{cases} d_i = |N(i)| & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

d_i is said to be the degree of vertex i . W and D are $n \times n$ matrices, where $n = |V|$, number of vertices in the mesh. We define the graph Laplace matrix G as

$$G = D - W$$

Referring to Equation (1.1), G corresponds to setting $b_i = 1$ and $w_{ij} = W_{ij}$ for all i, j . The operator G is also known as the Kirchoff operator [66], as it has been encountered in the study of electrical networks by Kirchoff. In that context, the (weighted) adjacency matrix W is referred as the conductance matrix [38].

Weighted Graph Laplace.

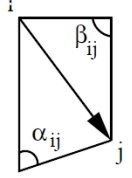
It is trivial to extend the above definition to weighted graphs, where the graph adjacency matrix W would be defined by $W_{ij} = w(e_{ij}) = w_{ij}$, for some edge weight $w : E \rightarrow \mathbb{R}^+$, whenever $(i, j) \in E$. Then, it is necessary to define the diagonal entries of the degree matrix D as $D_{ii} = \sum_{j \in N(i)} w_{ij}$. In particular, [5] uses Gaussian weights on a nearest neighbor graph constructed from point cloud sampled from a hidden manifold.

Cotangent Laplace.

The cotangent Laplace tries to discretize the smooth version of Laplace operator by introducing parameters based on local mesh geometry. The cotangent Laplace operator C is defined as:

$$(Cf)_i = \sum_{j \in N(i)} \frac{\cot \alpha_{ij} + \cot \beta_{ij}}{2} (f_i - f_j)$$

where angles α_{ij} and β_{ij} are subtended by the edge (i, j) as shown in figure on the right. In reference to 1.1, C is obtained by setting $b_i = 1$ for all i and $w_{ij} = \frac{\cot \alpha_{ij} + \cot \beta_{ij}}{2}$ for all (i, j) . If (i, j) is a boundary edge, the $\cot \beta_{ij}$ term vanishes. This corresponds to imposing von Neumann boundary conditions [93].



Mesh Laplace.

In [7], a discrete mesh-Laplacian L_t was proposed, where t is some parameter. It is defined by

$$(L_t f)_i = \frac{1}{t(4\pi t)^{m/2}} \sum_{j \in N(i)} A_j e^{-\frac{\|v_i - v_j\|^2}{4t}} (f_i - f_j)$$

where m is the intrinsic dimension of the mesh, and A_j is $\frac{1}{m+1}$ -th of the total volume of all m -simplices incident to the vertex v_j .

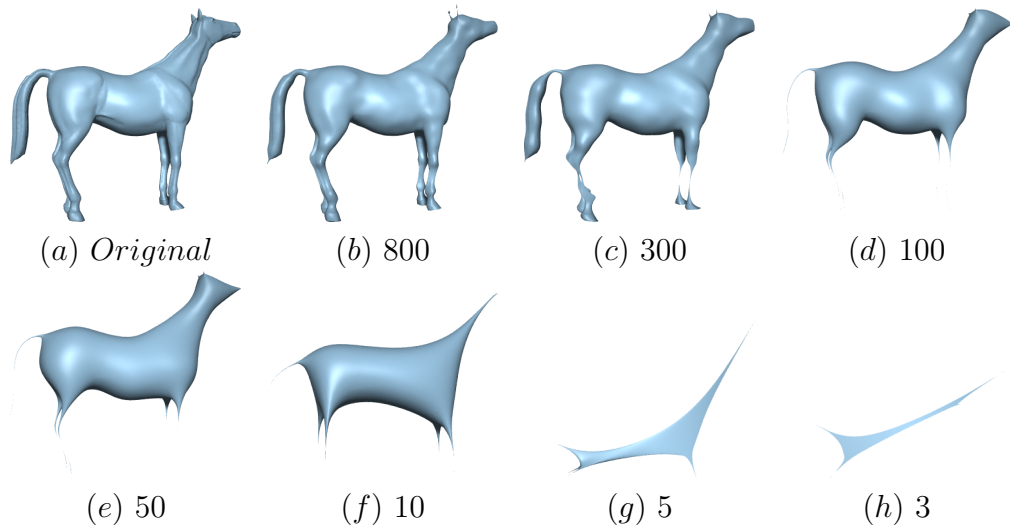


Figure 1.1: The horse model shown in (a) is reconstructed in (b)-(h) using the indicated number of eigenvectors of the mesh Laplacian. More eigenvectors are able to capture the the finer details.

1.2 Applications

The spectral methods have found a wide variety of usage in computer graphics. They can be broadly divided into methods that use the eigenvalues, and the ones that use the eigenvectors.

1.2.1 Use of eigenvalues

Drawing analogies from discrete Fourier analysis, one would treat the eigenvalues of a mesh Laplacian as measuring the frequencies of their corresponding eigenfunctions [90]. However, it is not easily seen what the term frequency means exactly in the context of eigenfunctions that oscillate irregularly over a manifold. Furthermore, since different meshes generally possess different operators and thus different eigenbases, using the magnitude of the eigenvalues to pair up corresponding eigenvectors

between the two meshes for shape analysis, e.g., correspondence, is unreliable [19]. Despite of these issues, much empirical success has been obtained using eigenvalues as global shape descriptors for graph [83] and shape matching [45].

Besides directly employing the eigenvalues as graph or shape descriptors, spectral clustering methods use the eigenvalues to scale the corresponding eigenvectors so as to obtain some form of normalization. [13] scale the eigenvectors by the squares of the corresponding eigenvalues, while [45] provide justification for using the square root of the eigenvalues as a scaling factor. The latter choice is consistent with the scaling used in spectral clustering [65], normalized cuts [80], and multidimensional scaling [22].

1.2.2 Use of eigenvectors

Eigenvectors are typically used to obtain an embedding of the input shape in the spectral domain. After obtaining the eigen-decomposition of a specific operator, the coordinates of vertex i in a k -dimensional embedding are given by the i -th row of matrix $\Phi_k = [\phi_1, \dots, \phi_k]$, where ϕ_1, \dots, ϕ_k are the first k eigenvectors from the spectrum (possibly after scaling). Whether the eigenvectors should be in ascending or descending order of eigenvalues depends on the operator that is being used. For the Laplacian operators, eigenvectors corresponding to the smallest eigenvalues are used to compute spectral embeddings.

For example, spectral clustering makes use of such embeddings. [65] present a method where the entries of the first k eigenvectors corresponding to the largest eigenvalues of a normalized affinity matrix are used to obtain the transformed coordinates of the input points. Additionally, the embedded points are projected onto the

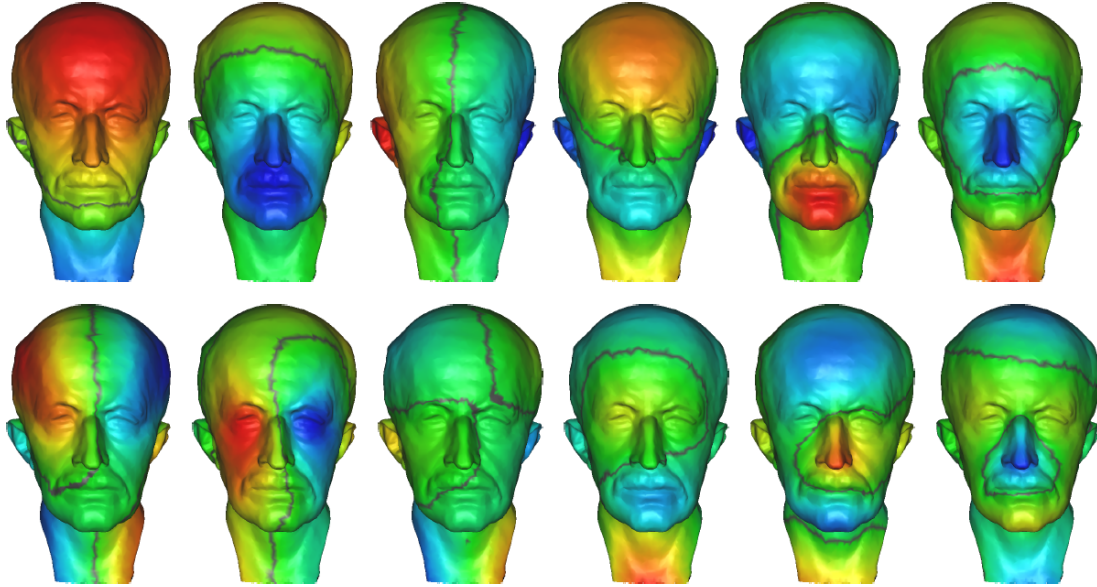


Figure 1.2: Top 12 eigenvectors of the graph Laplacian. Nodal sets (vertices with zero eigenfunction value) are shown in gray.

unit k -sphere. Points that possess high affinities tend to be grouped together in the spectral domain, where a simple clustering algorithm, such as k -means, can reveal the final clusters.

1.2.3 Use of eigenprojections

If a mesh operator possesses a set of orthogonal eigenvectors, with respect to inner product as defined in 1.2, and given by the columns of matrix Φ , then any discrete function defined on the mesh vertices, given by a vector x , can be transformed into the spectral domain by

$$\tilde{x} = V^T Bx$$

These spectral transforms are closely related to the Fourier transform that is the foundation of signal processing theory. In geometry processing, the signal considered is often the embedding function that specifies the 3D coordinates of each vertex. This signal is commonly referred to as the geometry of the mesh. Thus the geometry signal is an $n \times 3$ matrix P whose i -th row is the transpose of the position vector of the i -th vertex. The resulting coefficients $\tilde{P} = V^T B P$ are then a representation of the mesh geometry in the spectral domain.

As in the case of Fourier analysis, the intuition is that when the signal is transformed into the spectral domain, it might be easier to carry out certain tasks because of the relation of the coefficients to low and high frequency information. For example, the projections of P with respect to the eigenvectors of the graph Laplacian can be used for mesh compression [48]. That is, a set of the transformed coefficients from the high-frequency end of the spectrum can be removed without affecting too much the approximation quality of the mesh, when it is reconstructed by the inverse transform. Figure 1.1 shows the result of reconstructing a horse model by using different number of eigenvectors.

For spectral watermarking of meshes [66] however, it is the low-frequency end of the spectrum that is to be modulated. This way, the watermark is less perceptible and the watermarked mesh can become resilient against such attacks as smoothing.

An important reason for the sudden gain in popularity of the discrete Laplace operators in spectral methods is the fact that their eigenvectors possess similar properties as the classical Fourier basis functions. By representing mesh geometry using a discrete signal defined over the manifold mesh surface, it is commonly believed that a Fourier transform of such a signal can be obtained by an eigenspace projection of

the signal along the eigenvectors of a mesh Laplacian. Indeed, the classical Fourier transform of a periodic 1D signal can be seen as the decomposition of the signal into a linear combination of the eigenvectors of the Laplacian operator.

An important distinction between the mesh case and the classical Fourier transform however is that while the latter uses a fixed set of basis functions, the eigenvectors which serve as Fourier-like bases for mesh signal processing would change depending on mesh connectivity, geometry, and which type of Laplacian operator is adopted. Nevertheless, the eigenvectors of the mesh Laplacian all appear to exhibit harmonic behavior, loosely referring to their oscillatory nature. They are seen as the vibration modes or the harmonics of the mesh surface with their corresponding eigenvalues as the associated frequencies.

In Figure 1.2, we give color plots of the first 12 eigenvectors of the combinatorial graph Laplacian of the Max Planck mesh, where the entries of an eigenvector are color-mapped. As we can see, the harmonic behavior of the eigenvectors is evident.

1.3 Contribution

The purpose of this thesis is to study the theoretical properties of the discrete Laplace operator and to explore their possible applications in the field of computer graphics. In Chapters 3 and 4 ([26], [27]), we study the discrete Laplace spectra and its stability. Later on, in Chapters 5 and 6 ([24], [25]) we study some applications to shape matching and deformation, respectively.

1.3.1 Convergence and Stability of the mesh Laplacian spectra

We start with a study of the spectrum of mesh-Laplace operator and its relation to the spectrum of the (smooth) Laplacian. We pose the following two questions in an effort to learn about the stability of the Laplace operator, as well as its discrete counterpart.

- P1. Given a manifold and a simplicial mesh that approximates it, how does the spectrum of the (discrete) mesh-Laplacian relate to that of (smooth) Laplace operator? Does the former converge to the latter as the sampling becomes denser?

- P2. Given two 'similar' manifolds, how close are their (smooth) Laplacian spectra? What about the spectra of the discrete Laplacian computed from meshes approximating the two manifolds.

In Chapter 3, we will show that the mesh-Laplacian spectrum is, in fact, 'close' to the spectrum of the Laplace operator, and that the eigenvalues of the discrete operator actually converge to those of the smooth analog, as the sampling becomes denser. We will also show that the mesh-Laplacian spectrum of two meshes approximating 'similar' manifolds are also 'close'. The rigorous definitions of 'similarity' and 'closeness' can be found in Chapter 2. We close Chapter 3 with some experimental results to back up our theoretical findings.

1.3.2 Stability under topological noise

Next, we move on to study the behavior of discrete Laplace operators when modifications to a manifold change its topology. Our previous stability study required that

the input manifold be approximated by a mesh structure. The perturbation considered also needed to preserve the manifold topology and be smooth in the first order. We relax both the conditions. We concentrate on the Gaussian-weighted (discrete) graph Laplacian, which works directly on the point cloud data sampled from a manifold, without the need of a triangulation. The Gaussian-weighted (discrete) graph Laplacian is a popular and commonly used operator, known for its simplicity as well as the fact that it converges to the manifold Laplacian. We also modify the notion of 'similarity' used in Chapter 3 to allow topological changes. Since topological changes can alter geodesic distances dramatically, the stability of the smooth Laplacian cannot be guaranteed. However, we show that the weighted graph Laplacian under such perturbations is still stable. We use results from bipartite graph matching matrix perturbation theory to prove our claim. The details are presented in 4, along with experiments that corroborate our claim.

1.3.3 Applications in shape matching

In later chapters, we shift our focus from theory to practice. Our theoretical results tell us that not only is the spectrum of the mesh-Laplacian stable, but it is also similar for similar meshes. Hence, we should be able to use the spectrum to create a signature in order to match different shapes or meshes. In Chapter 5, we create such a signature. We use a function called the Heat Kernel Signature, or the HKS, which was first developed by Sun et al. in their paper [88]. The HKS has deep connections with the Laplace operator, which are explored in Chapter 2. In fact, the HKS can be computed by using the eigenvalues and the eigenvectors of the Laplace operator. This was one of the biggest reasons for choosing the HKS to create

our shape matching signature. Next, we used elements from persistent homology to create an algorithm to pick out the 'important' values of the HKS and discard the rest. The details of this procedure, along with various results and comparisons with other matching algorithms, are also presented in Chapter 5.

1.3.4 Using eigenvectors for shape deformation

Finally, we wrap up the thesis with an application that allows a user to freely deform shapes or models. A bulk of the current body of work on surface deformation and animation relies on extra structures like cages or skeletons which need to be provided by the user, in addition to the model they wish to deform. Our work aims to alleviate this problem by creating an implicit skeleton using the eigenvalues and eigenvectors of the mesh-Laplacian. We then use these eigenvectors, along with our skeleton, to guide the deformation. The result is a fast and easy to use software which is presented in Chapter 6, along with the details of the mechanism used to create the skeletons and guide the deformations.

Chapter 2: Discrete Laplace Operator

Before proceeding to the main contributions of this thesis, it is important to first introduce the discrete Laplace operators and related tools that will be used in the rest of the chapters. We will start with formal definitions and brief studies of the mesh-Laplacian and then move on to the Gaussian-weighted graph Laplacian. We will also take a look at the Heat operator, which is another operator defined on manifolds. It is of particular interest to us since it share its eigenvectors with the Laplace operator, a fact that we will exploit in Chapter 5

2.1 Mesh Laplace Operator

As seen in Chapter 1 several discretizations of the Laplace operator for meshes have been proposed. In [7], Belkin et al. proposed the so-called mesh-Laplace operator, which is the first discrete Laplacian that *pointwise converges* to the true Laplacian as the input mesh approximates a smooth manifold better. Specifically, for any C^2 -smooth scalar function f defined on a manifold M and its restriction \hat{f} on vertices of a mesh K , $|\Delta_M f(x) - \mathbf{D}_K \hat{f}(x)|_\infty$ converges to zero as K converges to M , where Δ_M and \mathbf{D}_K denote the Laplacian of M and its discrete approximation from K , respectively.

This result can be easily extended to higher dimensional manifolds¹. Experimental results also show that this operator indeed produces accurate approximation of the Laplace operator under various conditions, such as noisy data input, and different sampling conditions etc [92].

Given a simplicial mesh K with all vertices lying on M , we say that it ε -approximates a smooth manifold M if (i) for any point $p \in M$, there is a sample point (i.e, a vertex) from K that is at most $\varepsilon\rho(M)$ away; and (ii) the projection map ϕ from the underlying space $|K|$ of K onto M is a homeomorphism and its Jacobian is bounded by $1 + O(\varepsilon)$ at any point in the interior of the m -simplices. Intuitively, the first condition ensures that the mesh is sufficiently fine. However, a very fine mesh can still provide a poor approximation to the underlying surface. Hence we need the second condition to ensure that the distortion between $|K|$ and M is small. We remark that for an m -manifold embedded in \mathbb{R}^{m+1} (such as a surface embedded in \mathbb{R}^3), such an ε -approximation is equivalent to the (ε, η) -approximation used in [7] with $\eta = O(\varepsilon)$, which bounds both the sampling density and the normal deviation.

In the discrete setting, an input function f is only available at vertices of K , and thus can be represented as an n -dimensional vector $\hat{\mathbf{f}} = [f(v_1), \dots, f(v_n)]^T$ where $V = \{v_1, \dots, v_n\}$ is the set of vertices in K . In [7], Belkin et al. propose a discrete mesh-Laplacian \mathbf{L}_t^K , where t is some parameter. Being a linear operator, this discrete analog of the Laplace operator is an n by n matrix. It is defined by:

$$\mathbf{L}_t^K f(v_i) = \frac{1}{t(4\pi t)^{m/2}} \sum_{v_j \in V} A_j e^{-\frac{\|v_i - v_j\|^2}{4t}} (f(v_i) - f(v_j)),$$

¹The extension to d -manifolds embedded in \mathbb{R}^{d+1} is straightforward. When the co-dimension is greater than 1, one needs to define the sampling condition appropriately to guarantee the convergence of the normal space.

where A_j is $\frac{1}{m+1}$ -th of the total volume of all m -simplices incident to the vertex v_j . This discrete operator \mathbf{L}_t^K pointwise converges to the Laplace operator Δ_M of M . More precisely,

Theorem 2.1.1 ([7]). *Set $t(\varepsilon) = \varepsilon^{\frac{1}{2.5+\alpha}}$ for an arbitrary fixed positive number $\alpha > 0$. Then for any $f \in C^2(M)$ and any point $x \in M$,*

$$\limsup_{\varepsilon \rightarrow 0} \sup_{K(\varepsilon)} |\mathbf{L}_{t(\varepsilon)}^{K(\varepsilon)} \mathbf{f}(x) - \Delta_M f(x)| = 0,$$

where the supremum is taken over all ε -approximations $K(\varepsilon)$ of M .

2.2 Gaussian-weight graph Laplacian

Weighted graph Laplace operator. An underlying manifold is often approximated by simply a point cloud data (PCD) sampled from the manifold, instead of a triangulation. We thus need a discrete PCD version of the Laplace-Beltrami operator. A popular choice in this setting is the so-called Gaussian-weighted graph Laplacian, both for its simplicity and for its convergence to the manifold Laplacian with increasing number of uniformly randomly sampled points.

Let M be a smooth, compact m -dimensional manifold without boundary which is isometrically embedded in \mathbb{R}^d and thus equipped with a natural Riemannian metric induced from \mathbb{R}^d . We use $d_{\mathbb{R}^d}(x, y)$ to denote the Euclidean distance between two points $x, y \in \mathbb{R}^d$ and $d_M(x, y)$ to denote the geodesic distance between x and y on M when x, y are on $M \subseteq \mathbb{R}^d$. For simplicity of exposition, we replace $d_{\mathbb{R}^d}(x, y)$ with $\|x - y\|$ when it appears in the exponent.

Consider a set of discrete sample points $P = \{p_1, \dots, p_n\} \subset M$. Given a function $f : P \rightarrow \mathbb{R}$, the Gaussian-weighted graph Laplace is defined with respect to a

parameter $t > 0$ as

$$\mathbb{L}_P^t f(p_i) = \frac{1}{n} \cdot \frac{1}{(4\pi t)^{m/2} t} \sum_{j=1}^n e^{-\frac{\|p_i - p_j\|^2}{4t}} (f(p_i) - f(p_j)), \quad (2.1)$$

where $n = |P|$ is the number of sample points. Since a discrete function $f : P \rightarrow \mathbb{R}$ can be represented as an n -dimensional vector $[f(p_1), f(p_2), \dots, f(p_n)]^T$, \mathbb{L}_P^t is an $n \times n$ matrix where

$$\mathbb{L}_P^t[i][j] = \begin{cases} -\frac{1}{n} \cdot \frac{1}{(4\pi t)^{m/2} t} e^{-\frac{\|p_i - p_j\|^2}{4t}}, & \text{if } i \neq j \\ \frac{1}{n} \cdot \frac{1}{(4\pi t)^{m/2} t} \sum_{l \neq i, l \in [1, n]} e^{-\frac{\|p_i - p_l\|^2}{4t}}, & \text{if } i = j \end{cases} \quad (2.2)$$

It has been shown [4, 6] that if the set P samples \mathbf{M} uniformly randomly according to the volume measure on \mathbf{M} , then as n tends to infinity and t tends to 0 at appropriate rates, \mathbb{L}_P^t converges to $\Delta_{\mathbf{M}}$ both pointwise and in spectrum.

2.3 Heat operator

No discussion of the Laplace operator can be truly complete without the Heat operator. Given a Riemannian manifold M , the *heat operator* \mathcal{H}_t w.r.t. a parameter $t \in \mathbb{R}$ is an operator on $L^2(M)$, the space of square integrable functions on M . Specifically, imagine that there is an initial heat distribution on M at time 0. Now the heat starts to diffuse and this diffusion process is governed by the following *heat equation*, where $u(x, t)$ denotes the amount of heat at a point $x \in M$ at time t , and Δ is the Laplace-Beltrami operator of M : $\Delta u(x, t) = -\frac{\partial u(x, t)}{\partial t}$. Given a function $f : M \rightarrow \mathbb{R}$, the heat operator applied to f gives the heat distribution at time t with f being the initial heat distribution. That is, $\mathcal{H}_t f = u(\cdot, t)$ if $u(\cdot, 0) = f$. For a square integrable function f , a unique solution to the heat equation exists, and $\mathcal{H}_t f$ has the form: $\mathcal{H}_t f(x) = \int_M \mathbf{h}_t(x, y) f(y) d\mu_y$, where $d\mu_y$ is the volume form at y , and $\mathbf{h}_t : M \times M \rightarrow \mathbb{R}$ is the so-called *heat kernel* function.

Heat kernel. Intuitively, for two points $x, y \in M$, $\mathbf{h}_t(x, y)$ measures the amount of heat that passes from y to x within time t out of unit heat at y . If M is the Euclidean space \mathbb{R}^d , then the corresponding heat kernel is: $\mathbf{h}_t(x, y) = \frac{1}{(4\pi t)^{d/2}} e^{-\|x-y\|^2/4t}$. For a general manifold M , however, there is no known explicit expression for the heat kernel. There is fortunately an alternative way to represent the heat kernel. More specifically, the heat operator is compact, self-adjoint, and positive semi-definite. Thus it has discrete spectrum $1 = \rho_0 \geq \rho_1 \geq \dots \geq 0$ with $\mathcal{H}_t \phi_i = \rho_i \phi_i$. By the Spectral Theorem, the heat kernel can be written as:

$$\mathbf{h}_t(x, y) = \sum_{i \geq 0} \rho_i \phi_i(x) \phi_i(y). \quad (2.3)$$

i.e., if we know the spectrum and eigenfunctions of the heat operator, we can then compute the heat kernel function. We can compute the spectrum of the heat operator via the Laplace-Beltrami operator Δ of M , which is related to the heat operator as $\mathcal{H}_t = e^{-t\Delta}$. This means that \mathcal{H}_t and Δ share the same eigenfunctions ϕ_i , and their eigenvalues satisfy $\rho_i = e^{-t\lambda_i}$, where $\Delta \phi_i = \lambda_i \phi_i$. In other words, we can compute the heat kernel by $\mathbf{h}_t(x, y) = \sum_{i \geq 0} e^{-t\lambda_i} \phi_i(x) \phi_i(y)$.

Chapter 3: Convergence, Stability, and Discrete Approximation of Laplace Spectra [26]

One of the most popular discrete Laplacians is the *cotangent scheme* (which was briefly discussed in Section 1.1) for surfaces embedded in three-dimensional space, originally proposed in [28, 70], and its variants [23, 58, 59, 100]. The cotangent scheme has several nice properties, including the so-called *weak convergence* (which, roughly speaking, means convergence in the sense of inner product) [42, 97]. However, in general, it does not provide the standard pointwise convergence [100, 101], though there are some convergence results for certain special meshes and manifolds [100]. Nevertheless, in his Ph.D dissertation, Wardetzky showed a convergence result for spectra based on the cotangent scheme when the surface mesh satisfies some mild conditions on the aspect ratio of the triangles [96]. Reuter et al. computed a discrete Laplace operator using the finite element method, and obtained good practical performance [73].

In Chapter 2, we saw that the mesh Laplacian also has nice convergence properties and is also known to be robust against noise and sampling conditions. However, so far, no general convergence result is known for the eigen-structures of any discrete Laplacian for meshes in arbitrary dimensions, even though many practical applications rely on these structures. In general, pointwise convergence between two operators is not

strong enough to imply the convergence of their respective eigenvalues nor eigenfunctions. As mentioned above, partial spectrum convergence result was obtained for surface meshes based on the cotangent scheme [96]. For high dimensional manifolds, convergence result is known only under the statistical setting — if input points are *randomly* sampled from the underlying manifold, Belkin and Niyogi showed that the eigen-structure of the weighted graph Laplacian of these points converges to that of the manifold Laplacian [5].

In Section 3.3, we present the first result relating the eigen-structure of some discrete Laplacian from meshes with the manifold Laplacian for m -manifolds embedded in \mathbb{R}^d . We focus on the mesh-Laplacian proposed in [7] and show that its eigenvalues converge to those of the manifold Laplacian as the mesh approximates a smooth manifold better. The new result is achieved by showing that the mesh-Laplace operator converges to the manifold Laplacian not only pointwise, but in fact under a stronger operator norm when considered in a certain appropriate Sobolev space.

In Section 3.2, we investigate a related question of how stable the Laplacian spectrum and its discrete approximation are as the underlying manifold is perturbed. We give explicit bounds for the Laplacian spectra of two “close by” manifolds, and present a convergence result for their discrete approximations. This is the first stability result for discrete Laplace operators.

In Section 3.4, we provide experimental evidence showing that the mesh Laplacian indeed produces good estimates of spectra of the manifold Laplacian, and is robust to noise and deformations.

3.1 Approach Overview

Problem definition. In this chapter, we aim to understand the stability of the spectrum of the Laplace operator and its discrete analog. The first question we consider is:

- P1. How does the spectrum of the mesh-Laplacian \mathbf{L}_t^K relate to that of Δ_M . Does the former converge to the latter as the sampling becomes denser?

The second problem aims to understand the stability of the Laplacian spectrum (both the continuous and discrete versions) when the underlying manifold M is perturbed. Specifically, given two smooth and compact m -manifolds M and N embedded in \mathbb{R}^d , we say that M and N are δ_H -close if there is a homeomorphism $\Psi : M \rightarrow N$ such that (1) $\|x - \Psi(x)\| = O(\delta_H)$ for any $x \in M$, and (2) the Jacobian of the map Ψ is bounded by $|J\Psi - 1| = O(\delta_H)$ at any point of M .

- P2. How are the spectra of Δ_M and Δ_N , as well as the spectra of the discrete Laplacian computed from meshes approximating M and N , related.

3.1.1 Overview of Approaches and Results

To connect the Laplace operator and its approximation, we need an intermediate operator \mathcal{L}_t^M , called the *functional approximation of Δ_M* , first introduced in [3]. Given a point $p \in M$ and a function $f : M \rightarrow \mathbb{R}$, it is defined as:

$$\mathcal{L}_t^M f(x) = \frac{1}{t(4\pi t)^{m/2}} \int_{y \in M} e^{-\frac{\|x-y\|^2}{4t}} (f(x) - f(y)) dy. \quad (3.1)$$

The intuition behind using this operator is two-fold. First, the closed form of the Laplace operator is unavailable for general manifolds, making it hard to analyze directly. Secondly, while the Laplace operator is an unbounded operator, this functional

Laplacian is bounded with a simple spectral structure. This facilitates us to use the standard perturbation theory to analyze the stability of this operator. The connection between the functional Laplacian and Δ_M can be summarized in the following theorem [3, 5].

Theorem 3.1.1 ([3, 5]). *For a function $f \in C^2(M)$, we have that*

$$\lim_{t \rightarrow 0} \|\mathcal{L}_t^M f - \Delta_M f\|_\infty = 0.$$

Furthermore, let $\{\lambda_i\}$ and $\{\hat{\lambda}_i\}$ denote the discrete eigenvalues of Δ_M and \mathcal{L}_t^M enumerated in non-decreasing order. Then, for any fixed i and for t small enough (more precisely, $t < \frac{1}{2\lambda_i}$), we have $|\lambda_i - \hat{\lambda}_i| = O(t^{\frac{2}{m+6}})$.

In [7], it was shown that given a mesh K that ε -approximates M , \mathcal{L}_t^M can be approximated by the mesh Laplacian \mathbf{L}_t^K with pointwise convergence guarantee. When combined with the above theorem, this implies Theorem 2.1.1. However, to answer Question P1, we need a stronger (than pointwise) convergence result between \mathbf{L}_t^K and \mathcal{L}_t^M . Specifically, in Section 3.3, we show the following result, which is obtained by bounding the operator norm of the difference between \mathbf{L}_t^K and \mathcal{L}_t^M in an appropriate functional space.

Theorem 3.1.2. *Given a smooth m -manifold M , let $K(\varepsilon)$ denote a simplicial mesh K that ε -approximates M . Let $\{\hat{\lambda}_i\}$ and $\{\lambda_i^D(\varepsilon)\}$ denote the set of non-decreasing discrete eigenvalues of \mathcal{L}_t^M and of $\mathbf{L}_t^{K(\varepsilon)}$, respectively. Then, for any fixed i , we have that $\lim_{\varepsilon \rightarrow 0} |\hat{\lambda}_i - \lambda_i^D(\varepsilon)| = 0$.*

This result, combined with Theorem 3.1.1, gives an answer to Question P1 of this chapter, which is stated below. The relation between these results is illustrated in Figure 3.1.

Theorem 3.1.3. *Given a smooth m -manifold M and a simplicial mesh $K(\varepsilon)$ that ε -approximates M , let $\{\lambda_i\}$ and $\{\lambda_i^D(\varepsilon)\}$ denote the set of non-decreasing discrete eigenvalues of Δ_M and of $\mathbf{L}_t^{K(\varepsilon)}$, respectively. Then, for any fixed i , we have that $\lim_{t, \varepsilon, \frac{\varepsilon}{t^{\frac{m}{2}+3}} \rightarrow 0} |\lambda_i - \lambda_i^D(\varepsilon)| = 0$.*

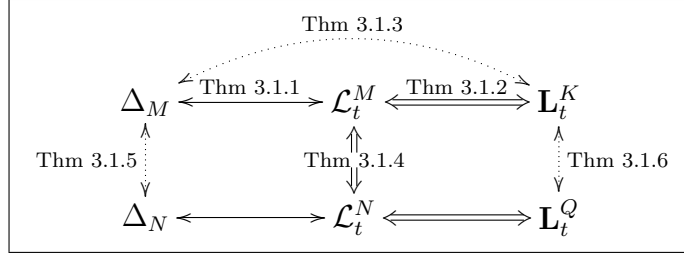


Figure 3.1: Theorems relating different operators are shown on top of the arrows. Double arrows indicate the two main new results in this chapter, and lead to those results specified by dotted arrows.

To answer Question P2, the main component is a perturbation result for the functional Laplace operator. Specifically, let $\text{Spec}(A)$ denote the spectrum of an operator A . We show that:

Theorem 3.1.4. *Given two δ_H -close m -manifolds M and N , the Hausdorff distance between $\text{Spec}(\mathcal{L}_t^M)$ and $\text{Spec}(\mathcal{L}_t^N)$ is $O(\frac{\delta_H}{t^{\frac{m}{4}+2}})$. That is, for any eigenvalue $\hat{\lambda} \in \text{Spec}(\mathcal{L}_t^M)$ and $\hat{\omega} \in \text{Spec}(\mathcal{L}_t^N)$, we have that $\text{dist}(\hat{\lambda}, \text{Spec}(\mathcal{L}_t^N)) = O(\frac{\delta_H}{t^{\frac{m}{4}+2}})$ and $\text{dist}(\hat{\omega}, \text{Spec}(\mathcal{L}_t^M)) = O(\frac{\delta_H}{t^{\frac{m}{4}+2}})$, where $\text{dist}(x, X) := \inf_{y \in X} |y - x|$.*

Combining this result with Theorem 3.1.1 bounds the spectra of Δ_M and of Δ_N (Theorem 3.1.5 below); and combining it with Theorem 3.1.2 leads to spectral convergence of discrete Laplacians for meshes approximating M and N , as N converges to M (Theorem 3.1.6 below). These relations are also illustrated in Figure 3.1.

Theorem 3.1.5. *Let $\{\lambda_i\}$ and $\{\omega_i\}$ be the non-decreasing eigenvalues of Δ_M and Δ_N with multiplicity. Then, for any λ_i , there exists $\delta_{H0} > 0$ such that if M and N are δ_H -close for any $\delta_H < \delta_{H0}$, then $|\lambda_i - \omega_i| = O(\delta_H^{\frac{8}{m^2+14m+56}})$.*

Theorem 3.1.6. *Let M and N be two m -manifolds that are δ_H -close, and $K(\varepsilon)$ and $Q(\varepsilon)$ be two simplicial meshes ε -approximating M and N , respectively. Let $\{\lambda_i^D\}$ and $\{\omega_i^D\}$ be the non-decreasing eigenvalues of \mathbf{L}_t^K and \mathbf{L}_t^Q with multiplicity. Then, for any fixed i , we have that as N converges to M and as the meshes approximate better, $\lim_{\delta_H, \varepsilon, \frac{\varepsilon}{\delta_H^{\frac{m}{2}+3}}, \frac{\delta_H}{t^{\frac{m}{4}+2}} \rightarrow 0} |\lambda_i^D - \omega_i^D| = 0$.*

Outline. In the rest of this chapter, instead of following the above order where we introduced the results, we first prove Theorem 3.1.4 and 3.1.5 in Section 3.2, as this will illustrate some of the main ideas of our approach. The proof for Theorem 3.1.2 is more technical, and we will present a sketch of it as well as proofs for the remaining results in Section 3.3.

3.2 Perturbation of Manifold and Stability

In this section, we study the behavior of the spectrum of Δ_M and its discrete approximation as the underlying manifold M is perturbed to another manifold N that is δ_H -close to M . The main component is to relate the spectrum of \mathcal{L}_t^M with that of \mathcal{L}_t^N (i.e, Theorem 3.1.4) which we focus on now. Here we consider the Hilbert spaces $L^2(M)$ and $L^2(N)$, which are the spaces of square integrable functions on M and on N , respectively. Notice that for any compact manifold X , the functional Laplacian \mathcal{L}_t^X is a self-adjoint and bounded operator in $L^2(X)$ (equipped with the standard L_2 norm).

Roughly speaking, if the norm of the difference between two operators is bounded in some space, then the distance between their spectra is also bounded. Hence, we wish to bound the operator norm of $\mathcal{L}_t^M - \mathcal{L}_t^N$. However, the two operators \mathcal{L}_t^M and \mathcal{L}_t^N are defined over two different spaces, $L^2(M)$ and $L^2(N)$, respectively. Thus, they are not directly comparable. Now assume $\Psi : M \rightarrow N$ is a homeomorphism between M and N that satisfies the δ_H -closeness conditions. We compare the operator \mathcal{L}_t^M with the pull-back operator of \mathcal{L}_t^N . Specifically, given an operator $\mathbf{A} : L^2(N) \rightarrow L^2(N)$, its *pullback via Ψ* , denoted by $\Psi^*(\mathbf{A}) : L^2(M) \rightarrow L^2(M)$, is defined by: given any function $f \in L^2(M)$, we obtain another function in $L^2(M)$ which is $\mathbf{A}(f \circ \Psi^{-1}) \circ \Psi$.

Lemma 3.2.1. *\mathbf{A} and $\Psi^*(\mathbf{A})$ share the same eigenvalues. The eigenfunctions of $\Psi^*(\mathbf{A})$ are $\{g_i \circ \Psi\}$ where g_i are the eigenfunctions of \mathbf{A} .*

Proof: Take an eigenfunction g_i of \mathbf{A} with eigenvalue ρ , that is, $\mathbf{A}g_i = \rho g_i$. Now consider $f = g_i \circ \Psi$ and consider $\Psi^*(\mathbf{A})f$. We have that

$$\Psi^*(\mathbf{A})f = \mathbf{A}(f \circ \Psi^{-1}) \circ \Psi = \mathbf{A}(g_i) \circ \Psi = \rho g_i \circ \Psi = \rho f.$$

The opposite direction is similar. ■

Since \mathcal{L}_t^N and its pullback share the same spectrum, it suffices to compare \mathcal{L}_t^M with $\Psi^*(\mathcal{L}_t^N)$. The following result will be needed later:

Claim 3.2.1. *Given an m -manifold M embedded in \mathbb{R}^d , for small enough $t > 0$,*

$$\int_M e^{-\frac{\|x-y\|^2}{4t}} dy = O(t^{\frac{m}{2}}).$$

Proof: Choose $r = (t)^{1/4} \leq \rho/2$ as a constant small enough, where ρ is the reach of the manifold M . Let B be the ball centered at point x with radius r , and M_B the

intersection between B and M . First, observe that $e^{-\frac{r^2}{4t}} \leq o(t^\alpha)$ for any $\alpha > 0$ when t is small enough, as

$$\lim_{t \rightarrow 0} e^{-\frac{r^2}{4t}} / t^\alpha = \lim_{t \rightarrow 0} e^{-\frac{1}{4\sqrt{t}}} / t^\alpha = 0.$$

It then follows that

$$\int_{M \setminus M_B} e^{-\frac{\|x-y\|^2}{4t}} dy \leq \text{Vol}(M) e^{-\frac{r^2}{4t}} = o(t^{m/2}). \quad (3.2)$$

On the other hand, consider the map from M_B to T_x , where T_x is the tangent space at x of M . Obviously, T_x is a m -dimensional subspace. Consider the projection map $\phi : M_B \rightarrow T_x$. For $r < \rho/2$, ϕ is injective. It is shown in [7] that the Jacobian of ϕ at any $y \in M_B$ is bounded by $1 + O(r^2/\rho^2)$. Same bound holds for the Jacobian of ϕ^{-1} for any $z \in \phi(M_B)$. This also implies that

$$\|\phi(y) - x\| \geq (1 - O(r^2/\rho^2))\|y - x\|.$$

Applying change of variables, we have:

$$\begin{aligned} \int_{M_B} e^{-\frac{\|x-y\|^2}{4t}} dy &= \int_{\phi(M_B)} e^{-\frac{\|x-\phi^{-1}(z)\|^2}{4t}} J_{\phi^{-1}}(z) dz \\ &\leq \int_{\phi(M_B)} e^{-\frac{(1-O(r^2/\rho^2))\|x-z\|^2}{4t}} (1 + O(\frac{r^2}{\rho^2})) dz \\ &\leq \int_{\phi(M_B)} e^{O(\frac{r^2\|x-z\|^2}{4t})} e^{-\frac{\|x-z\|^2}{4t}} (1 + O(\sqrt{t})) dz \\ &\leq \int_{\phi(M_B)} 2e^{O(\frac{r^4}{4t})} e^{-\frac{\|x-z\|^2}{4t}} dz \\ &\leq O(1) \cdot \int_{\phi(M_B)} e^{-\frac{\|x-z\|^2}{4t}} dz \\ &\leq O(1) \cdot \int_{\mathbb{R}^m} e^{-\frac{\|x-z\|^2}{4t}} dz \leq O(t^{m/2}). \end{aligned}$$

The last inequality follows from Claim 3.1 from [56]. The claim then follows from this and Equation (3.2). ■

Lemma 3.2.2. *The L_2 -norm of the difference of \mathcal{L}_t^M and $\Psi^*(\mathcal{L}_t^N)$ is bounded by $\|\mathcal{L}_t^M - \Psi^*(\mathcal{L}_t^N)\| = O(\frac{\delta_H}{t^{\frac{m}{4}+2}})$.*

Proof: Set $c = \frac{1}{t(4\pi t)^{m/2}}$ and $G_t(x, y) = e^{-\frac{\|x-y\|^2}{4t}}$. Given two points $x, y \in M$, note that $\|\Psi(x) - x\| = O(\delta_H)$ and $\|\Psi(y) - y\| = O(\delta_H)$ since M and N are δ_H -close. Thus

$$|G_t(\Psi(x), \Psi(y)) - G_t(x, y)| = O(\frac{\delta_H}{t})G_t(x, y).$$

Now, given a function $f : M \rightarrow \mathbb{R}$ and a point $x \in M$, note that $\Psi^*(\mathcal{L}_t^N)f(x) = \mathcal{L}_t^N(f \circ \Psi^{-1}) \circ \Psi(x)$. Setting $g = f \circ \Psi^{-1}$ and $p = \Psi(x)$, we have that:

$$\Psi^*(\mathcal{L}_t^N)f(x) = \mathcal{L}_t^N g(p) = c \int_N G_t(p, q)[g(p) - g(q)]dq.$$

By change of variables, we then obtain:

$$\begin{aligned} \Psi^*(\mathcal{L}_t^N)f(x) &= c \int_{\Psi^{-1}(N)} G_t(p, \Psi(y))[g(p) - g \circ \Psi(y)]J\Psi|_y dy \\ &= c \int_M G_t(\Psi(x), \Psi(y))[f(x) - f(y)]J\Psi|_y dy \end{aligned}$$

where $J\Psi|_y$ is the Jacobian of the map Ψ at $y \in M$, and is bounded by $|J\Psi|_y - 1| = O(\delta_H)$ due to the δ_H -closeness condition. Comparing this with $\mathcal{L}_t^M f(x)$ (recall

Equation (3.1)), we have that:

$$\begin{aligned}
\left| \Psi^*(\mathcal{L}_t^N)f(x) - \mathcal{L}_t^M f(x) \right| &\leq c \int_M |f(y)(G_t(\Psi(x), \Psi(y))[1 + O(\delta_H)] - G_t(x, y))| dy \\
&\quad + c \int_M |f(x)(G_t(\Psi(x), \Psi(y))[1 + O(\delta_H)] - G_t(x, y))| dy \\
&= c \left| \int_M G_t(\Psi(x), \Psi(y))f(y)dy - \int_M G_t(x, y)f(y)dy \right. \\
&\quad \left. + O(\delta_H) \int_M G_t(\Psi(x), \Psi(y))f(y)dy \right| \\
&\quad + c \int_M |f(x)(G_t(\Psi(x), \Psi(y))(1 + O(\delta_H)) - G_t(x, y))| dy \\
&\leq c \cdot O\left(\frac{\delta_H}{t}\right) \left[\int_M G_t(x, y)|f(y)|dy + |f(x)| \int_M G_t(x, y)dy \right] \\
&\leq c \cdot O\left(\frac{\delta_H}{t}\right) \left[\|f\| \sqrt{\int_M G_t^2(x, y)dy} + |f(x)| \int_M G_t(x, y)dy \right] \\
&\leq O\left(\frac{\delta_H}{t^{\frac{m}{4}+2}}\right) \|f\| + O\left(\frac{\delta_H}{t^2}\right) |f(x)|.
\end{aligned}$$

The last but one inequality follows from the fact that $\langle f, g \rangle \leq \|f\| \cdot \|g\|$ for any two functions. The last inequality follows from Claim 3.2.1. Hence the square of the L_2 -norm of $\Psi^*(\mathcal{L}_t^N)f - \mathcal{L}_t^M f$ is bounded by:

$$\begin{aligned}
\|\Psi^*(\mathcal{L}_t^N)f - \mathcal{L}_t^M f\|^2 &= \int_M [\Psi^*(\mathcal{L}_t^N)f(x) - \mathcal{L}_t^M f(x)]^2 dx \\
&\leq O\left(\frac{\delta_H^2}{t^{\frac{m}{2}+4}}\right) \int_M (\|f\|^2 + f^2(x) + 2\|f\| \cdot |f(x)|) dx \\
&\leq O\left(\frac{\delta_H^2}{t^{\frac{m}{2}+4}}\right) (\|f\|^2 \cdot \|\mathbf{1}\| + \|f\|^2 + 2\|f\|^2) \\
&\leq O\left(\frac{\delta_H^2}{t^{\frac{m}{2}+4}}\right) \|f\|^2,
\end{aligned}$$

where $\mathbf{1}$ is the constant function and $\|\mathbf{1}\| = \text{volume}(M)$. Hence $\|\Psi^*(\mathcal{L}_t^N)f - \mathcal{L}_t^M f\| = O(\delta_H/t^{\frac{m}{4}+2})\|f\|$ for any function f , where the big-O notation hides terms depending only on the underlying manifold M . The lemma then follows. \blacksquare

This result and Equation (2) from [94] imply that for any eigenvalue $\hat{\omega} \in \text{Spec}(\mathcal{L}_t^N)$, we have that $\text{dist}(\hat{\omega}, \text{Spec}(\mathcal{L}_t^M)) = O(\frac{\delta_H}{t^{\frac{m}{4}+2}})$. Now switching the role of M and N in Lemma 3.2.2, we obtain a symmetric result that for any eigenvalue $\hat{\lambda} \in \text{Spec}(\mathcal{L}_t^M)$, we have that $\text{dist}(\hat{\lambda}, \text{Spec}(\mathcal{L}_t^N)) = O(\frac{\delta_H}{t^{\frac{m}{4}+2}})$. Theorem 3.1.4 then follows from these two results. We remark that the distance between spectra of \mathcal{L}_t^M and \mathcal{L}_t^N depends not only on δ_H , the closeness between M and N , but also on t inversely. Intuitively, this is expected as the parameter t in the functional Laplacian \mathcal{L}_t specifies the width of the Gaussian kernel and thus the range of the region around $x \in M$ influencing $\mathcal{L}_t^M f(x)$. Hence, the larger t is, the stronger the smoothing effect it has, while the smaller t is, the more sensitive the functional Laplacian is to the perturbation of the underlying manifold, which leads to larger error between the corresponding spectra.

Proof of Theorem 3.1.5. It is well known that the Laplace operator only has real and isolated eigenvalues with finite multiplicity. We wish to build a one-to-one relationship between $\text{Spec}(\Delta_M)$ and $\text{Spec}(\Delta_N)$ and bound their distance. To achieve this using Theorems 3.1.1 and 3.1.4 (recall Diagram 3.1), there are two main technical issues to be addressed. First, the operator \mathcal{L}_t^X , although bounded and self-adjoint, is not compact. Hence, it may have non-isolated a continuous spectrum (e.g, all values within an interval are eigenvalues). Second, Theorem 3.1.4 only bounds the Hausdorff distance between spectra of \mathcal{L}_t^M and \mathcal{L}_t^N , while we wish to obtain a one-to-one relationship between (their lowest) eigenvalues.

For the first issue, given an operator T , recall that $\text{SpecDis}(T)$ denotes the set of isolated eigenvalues of T with finite multiplicity, and $\text{SpecEss}(T) = \text{Spec}(T) \setminus \text{SpecDis}(T)$ is the so-called essential spectrum of T .

Claim 3.2.2 ([5]). *The essential spectrum of \mathcal{L}_t^X is contained in $(\frac{1}{2}t^{-1}, \infty)$. The smallest eigenvalue of \mathcal{L}_t^X is 0, and the discrete spectrum of \mathcal{L}_t^X is contained in the interval $[0, \Theta(\frac{1}{t})]$.*

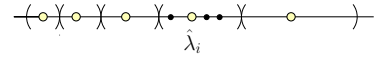
In other words, even though \mathcal{L}_t^M contains a continuous spectrum, those with low values (smaller than $\frac{1}{2}t^{-1}$) are isolated with finite multiplicity, and can be potentially related to those of \mathcal{L}_t^N in a one-to-one manner. These first few eigenvalues are also what are typically used in practice. As t goes to zero, the interval $[0, \frac{1}{2}t^{-1})$ will contain more and more isolated eigenvalues.

Claim 3.2.2 was shown in [5, 95]. We provide an intuition here: Set $c(t) = \frac{1}{t(4\pi t)^{m/2}}$ and $G_t(x, y) = e^{-\frac{\|x-y\|^2}{4t}}$. It turns out that the operator \mathcal{L}_t^X can be rewritten as $\mathcal{L}_t^X = \mathbf{M}_t^X - \mathbf{I}_t^X$, where $\mathbf{M}_t^X f(x) = g(x) \cdot f(x)$ with $g(x) = c(t) \int_X G_t(x, y) dy$, and $\mathbf{I}_t^X f(x) = c(t) \int_X G_t(x, y) f(y) dy$. In other words, \mathbf{M}_t^X is a multiplication operator and \mathbf{I}_t^X is an integral operator. It is easy to verify that both are self-adjoint in $L^2(X)$, and the former is bounded while the latter is compact. It then follows that the essential spectrum of \mathcal{L}_t^X coincide with the range of the function $g(\cdot)$ (i.e, $[\inf g(x), \sup g(x)]$). The range of this function $g(\cdot)$ was shown in [5] and Claim 3.2.2 thus follows.

For the second issue, consider the first k eigenvalues $\{\hat{\lambda}_i\}$ of \mathcal{L}_t^M and $\{\hat{\omega}_i\}$ of \mathcal{L}_t^N , in non-decreasing order, where k is an integer such that $\hat{\lambda}_k < \frac{1}{2}t^{-1}$ and $\hat{\omega}_k < \frac{1}{2}t^{-1}$ (i.e, the first k isolated eigenvalues). Theorem 3.1.1 from [5] states that for each $i < k$, $|\lambda_i - \hat{\lambda}_i| = O(t^{\frac{2}{m+6}})$. In other words, the first few eigenvalues from \mathcal{L}_t^M one-to-one correspond to the first few eigenvalues from Δ_M . The same statement holds for the lowest eigenvalues $\{\omega_i\}$ for Δ_N and $\{\hat{\omega}_i\}$ for \mathcal{L}_t^N .

Now we wish to also establish a one-to-one correspondence between (lowest) eigenvalues $\{\hat{\lambda}_i\}$ and $\{\hat{\omega}_i\}$. Imagine a sequence of manifolds $\{N(\delta_H)\}_{\delta_H \rightarrow 0}$ that converges to

M , where $N(\delta_H)$ is δ_H -close to M . This induces a sequence of functional Laplacians $\{\mathcal{L}_t^{N(\delta_H)}\}_{\delta_H \rightarrow 0}$, and Lemma 3.2.2 states that this sequence of functional Laplacians converges in operator norm to \mathcal{L}_t^M as δ_H goes to zero. It then follows from Proposition 6 in [95] that, for any isolated eigenvalue $\hat{\lambda} \in \text{SpecDis}(\mathcal{L}_t^M)$ with multiplicity m , and any open interval $I \in \mathbb{R}$ containing $\hat{\lambda}$ but no other eigenvalue from $\text{SpecDis}(\mathcal{L}_t^M)$, there exists some $\delta_{H0} > 0$ such that for any $\delta_H < \delta_{H0}$, exactly m (not necessarily distinct) eigenvalues of $\mathcal{L}_t^{N(\delta_H)}$ are contained in I . A similar result, in fact, holds for a finite set of consecutive isolated eigenvalues from $\text{SpecDis}(\mathcal{L}_t^M)$.

Now, imagine we plot the first k isolated eigenvalues  on a real line. See the right figure where each empty dot is a distinct eigenvalue of \mathcal{L}_t^M with multiplicity. For each one, we choose an open interval around it as shown in the figure (so their closures partition the line). Proposition 6 in [95] says that eventually (when δ_H is small enough), for the i th eigenvalue, only exactly m_i number of eigenvalues from $\text{Spec}(\mathcal{L}_t^{N(\delta_H)})$ will fall in the interval around it, where m_i is the multiplicity of $\hat{\lambda}_i$. (The right figure shows an example where $m_i = 3$, and the black dots represents eigenvalues of $\mathcal{L}_t^{N(\delta_H)}$.) This idea, combined with the one-to-one correspondence result between λ_i and $\hat{\lambda}_i$, eventually implies that when t is small enough and when $\frac{\delta_H}{t^{\frac{m}{4}+2}}$ is smaller than the separation gap between two consecutive λ_i s (which is a quantity depending only on the underlying manifold M when t is small enough), there is a one to one correspondence between $\hat{\lambda}_i$ and $\hat{\omega}_i$ and their distance is $O(\frac{\delta_H}{t^{\frac{m}{4}+2}})$. Specifically, each empty dot in the real line will be a set of m_i number of $\hat{\lambda}$ s clustered within a ball of radius $O(t^{\frac{2}{m+6}})$ where m_i is the multiplicity of the eigenvalue λ_i of Δ_M , and Proposition 6 in [95] states that there will be exactly m_i number of $\hat{\omega}$ s in the corresponding interval.

Finally, combining this with Theorem 3.1.1, we choose $t = \delta_H^{\frac{4(m+6)}{m^2+14m+56}}$ so that the two convergence rates, between Δ_M (resp. Δ_N) and \mathcal{L}_t^M (resp. \mathcal{L}_t^N), and between \mathcal{L}_t^M and \mathcal{L}_t^N , respectively, are balanced (i.e, $t^{\frac{2}{m+6}} = \frac{\delta_H}{t^{m/4+2}}$). Theorem 3.1.5 then follows. The various conditions in these theorems on the value of the eigenvalues are to ensure that the eigenvalues fall in the discrete spectrum for the functional Laplacian. Their existence does not matter for those lowest eigenvalues, which are the interesting ones in practice.

3.3 Spectra Convergence between Discrete and Continuous Laplacians

In this section, given a mesh K that ε -approximates a smooth compact m -manifold M embedded in \mathbb{R}^d , we relate the spectrum of Δ_M to that of its discrete approximation. By Theorem 3.1.1, we only need to show spectral convergence between the functional Laplace \mathcal{L}_t^M and the mesh-Laplacian \mathbf{L}_t^K (i.e, Theorem 3.1.2). Similar to previous section, we will achieve this by showing that the latter converges to the former in some operator norm. The main difference and challenge is that we now need to define the functional space we use to compare the relevant operators more carefully.

Specifically, the discrete Laplacian \mathbf{L}_t^K is a linear operator in \mathbb{R}^n where n is the number of vertices in K ; while \mathcal{L}_t^M is an operator in an infinite dimensional functional space. Hence, in Step 1, we first construct a continuous operator \mathcal{C}_t^K , which (almost) shares the same spectrum as the discrete operator \mathbf{L}_t^K , and which, at the same time, is well-defined in certain a common functional space along with \mathcal{L}_t^M . Next, in Step

2, we bound the operator norm of the difference between \mathcal{C}_t^K and \mathcal{L}_t^M in this space, which will in turn relate their spectra.

The Sobolev space H_s . The common functional space we use to compare \mathcal{C}_t^K and \mathcal{L}_t^M is the s -th Sobolev space H_s and we will choose $s = \frac{m}{2} + 1$. The norm in H_s is the Sobolev norm $\|g\|_{H_s} = [\sum_{i=0}^s \|g^{(i)}\|^2]^{1/2}$, where $g^{(i)}$ is the i -th weak derivative² of g and $\|\cdot\|$ denotes the standard L_2 norm. The key property of H_s for $s \geq \frac{m}{2} + 1$ that we will need is the following [91] and its corollary.

Lemma 3.3.1 ([91]). *Let $f \in H_s(M)$ with $s \geq m/2 + 1$ where m is the intrinsic dimension of the manifold M . Then f is Lipschitz with the Lipschitz constant bounded by $C\|f\|_{H_s}$ for some universal constant C .*

Corollary 3.3.1. *Given any $f \in H_s(M)$ with $s \geq m/2 + 1$, $\|f\|_\infty \leq C'\|f\|_{H_s}$ with some universal constant C' depending only on the underlying manifold M .*

Proof: The Lipschitz constant of f is bounded by $C\|f\|_{H_s}$ by Lemma 3.3.1. For any two points $x, y \in M$,

$$\begin{aligned} \left| |f(x)| - |f(y)| \right| &\leq |f(x) - f(y)| \leq C\|f\|_{H_s} \cdot |x - y| \\ &\leq C \cdot \text{Diameter}(M) \cdot \|f\|_{H_s}. \end{aligned}$$

Let $p \in M$ be a point so that $|f(p)| = \min_x |f(x)|$; note that $|f(p)| \leq \|f\| \leq \|f\|_{H_s}$. It then follows that $|f(x)| - |f(p)| \leq C \cdot \text{Diameter}(M) \cdot \|f\|_{H_s}$, implying

$$|f(x)| \leq |f(p)| + C \cdot \text{Diameter}(M) \cdot \|f\|_{H_s} \leq C'\|f\|_{H_s}.$$

The corollary then follows. ■

²The weak derivative is a generalization of the derivative of a function f , when f is not necessarily differentiable in the usual sense, and these two notions coincide when f is differentiable. For our purpose, the reader can think of it as the ordinary derivative.

From now on, we fix $s = m/2 + 1$. There are two main reasons behind relating the operators of interest in the space $H_s(M)$, instead of using some other spaces, say the space of square integrable functions $L^2(M)$.

(i) We can extend the discrete operator \mathbf{L}_t^K into a well-defined and well-behaved operator in $H_s(M)$. Intuitively, this is not possible in $L^2(M)$, as functions in $L^2(M)$ are not defined pointwise (two functions can be arbitrarily different at a finite set of points while the L_2 -norm of their difference is zero); while at the same time, \mathbf{L}_t^K requires point evaluations (as it is only defined at discrete sample points). Corollary 3.3.1 guarantees that the point evaluations in $H_{m/2+1}(M)$ are not only defined, but also bounded ($H_i(M)$ is, in fact, a reproducing kernel Hilbert space for $i \geq m/2 + 1$).

(ii) It turns out that we cannot bound the L_2 -norm distance of relevant operators (which is the operator norm in $L^2(M)$). As we will see later, this happens because the Lipschitz constant of the input function appears while bounding the L_2 -norm of the operator difference. Lemma 3.3.1 says that the Lipschitz constant can be bounded by the s -th Sobolev norm of f , which again suggests that we should use the space $H_s(M)$.

Step 1: Continuous extension for \mathbf{L}_t^K . We define operator $\mathcal{C}_t^K : H_s(M) \rightarrow H_s(M)$ as:

$$\mathcal{C}_t^K f(x) := \frac{1}{t(4\pi t)^{m/2}} \sum_{i=1}^n A_i G_t(x, v_i) (f(x) - f(v_i)).$$

Intuitively, we extend the kernel function from an n by n matrix (i.e, $G_t(v_j, v_i)$'s) to a continuous (Gaussian) kernel function defined on $M \times \mathbb{R}^n$. A similar extension was used in [95] to relate the graph Laplacian with the functional Laplacian.

Roughly speaking, there is a “one-to-one” correspondence between the eigenvalues (as well as eigenfunctions) of the operator \mathcal{C}_t^K and those of the discrete operator \mathbf{L}_t^K .

To make this correspondence more precise, set a function $d_K : M \rightarrow \mathbb{R}$ as

$$d_K(x) = \frac{1}{t(4\pi t)^{m/2}} \sum_{i=1}^n A_i G_t(x, v_i)$$

and define the multiplication operator $S_K : H_s(M) \rightarrow H_s(M)$ as $S_K f(x) = d_K(x)f(x)$.

Set $W_K : H_s(M) \rightarrow H_s(M)$ as

$$W_K f(x) = \frac{1}{t(4\pi t)^{m/2}} \sum_{i=1}^n [A_i G_t(x, v_i) f(v_i)].$$

It is easy to check that the operator $\mathcal{C}_t^K = S_K - W_K$. In space $H_s(M)$ where point evaluation is bounded (recall $H_s(M)$ is a reproducing kernel Hilbert space), S_K is a bounded multiplication operator and W_K is a compact operator [95]; implying that \mathcal{C}_t^K is bounded.

Unfortunately, the spectrum of \mathcal{C}_t^K may contain continuous spectrum. However, similar to the case of \mathcal{L}_t^X in Section 3.2, since W_K is compact, it turns out that $\text{SpecEss}(\mathcal{C}_t^K) = \text{SpecEss}(S_K) = \text{range}(d_K)$, where $\text{range}(d_K)$ is the range of the function d_K (i.e., $\text{range}(d_K) = [\inf_x d_K(x), \sup_x d_K(x)]$). Lemma 3.3.2 can then be derived by results from [95] and Lemma 3.3.3 follows from elementary calculations:

Lemma 3.3.2. *The essential spectrum of \mathcal{C}_t^K coincide with the range of the function d_K . For ε and t small enough, $\text{range}(d_K)$ (and thus $\text{SpecEss}(\mathcal{C}_t^K)$) is contained in $(\frac{1}{2}t^{-1}, \infty)$. The discrete spectrum of \mathcal{C}_t^K contains finite number of real eigenvalues, and is contained in the interval $[0, \Theta(\frac{1}{t})]$.*

Lemma 3.3.3. *1. If ρ is an eigenfunction of \mathcal{C}_t^K with arbitrary eigenvalue λ , then the n -vector $\hat{\rho} = [\rho(v_1), \dots, \rho(v_n)]^T \in \mathbb{R}^n$ is an eigenvector of \mathbf{L}_t^K with eigenvalue λ .*

2. If $\lambda \notin \text{range}(d_K) = \text{SpecEss}(\mathcal{C}_t^K)$ is an eigenvalue with multiplicity m , and ρ_1, \dots, ρ_m are the corresponding eigenfunctions, then \mathbf{L}_t^K has an eigenvalue λ also with multiplicity m , with the set of n -vectors $\hat{\rho}_1, \dots, \hat{\rho}_m$ being the corresponding m eigenvectors.
3. If $\lambda \notin \text{range}(d_K)$ is an eigenvalue for \mathbf{L}_t^K with multiplicity m , and $\hat{\rho}_1, \dots, \hat{\rho}_m$ being the corresponding m eigenvectors, then λ is an eigenvalue of \mathcal{C}_t^K with multiplicity m , corresponding to a set of eigenfunctions ρ_1, \dots, ρ_m such that

$$\rho_i(x) = \frac{1}{t(4\pi t)^{m/2}} \cdot \frac{\sum_{j=1}^n A_j G_t(x, v_j) \hat{\rho}_i[j]}{d_K(x) - \lambda}.$$

These results state that the interesting eigenvalues (i.e, with lowest values) are isolated with finite multiplicity, and that there is a one-to-one correspondence between such eigenvalues of \mathcal{C}_t^K and of \mathbf{L}_K . A similar extension was used in [95] to relate the graph Laplacian with the functional Laplacian.

Step 2: Relation between \mathcal{C}_t^K and \mathcal{L}_t^M . Let $\mathcal{D} = \mathcal{L}_t^M - \mathcal{C}_t^K$ denote the difference between operators \mathcal{L}_t^M and \mathcal{C}_t^K . We aim to show that $\|\mathcal{D}f\|_{H_s} \leq O(\varepsilon)\|f\|_{H_s}$ for any function $f \in H_s(M)$, which will then imply that $\|\mathcal{D}\|_{H_s} = O(\varepsilon)$. First, the following result bounds the derivatives of the Gaussian kernel function.

Lemma 3.3.4. $G_t^{(i)}(x, y) = \sum_{j=0}^{\lfloor \frac{i}{2} \rfloor} O(i^j) \frac{\|x-y\|^{i-2j}}{(2t)^{i-j}} G_t(x, y)$ The derivative is taken with respect to the variable x .

Proof: Recall that $G_t(x, y) = e^{-\frac{\|x-y\|^2}{4t}}$. We will prove the following statement by induction (which immediately implies Lemma 3.3.4).

$$G_t^{(i)}(x, y) = \sum_{j=0}^{\lfloor i/2 \rfloor} c_{j,i} \frac{\|x-y\|^{i-2j}}{(2t)^{i-j}} G_t(x, y) \text{ where}$$

$$c_{0,0} = 1, c_{0,i} = -c_{0,i-1},$$

$$c_{j,i} = (i - 2j + 1)c_{j-1,i-1} - c_{j,i-1}, 0 < j \leq \lfloor i/2 \rfloor,$$

$$c_{j,i} = 0, \text{ otherwise.}$$

$$\text{and } |c_{j,i}| = O((i+1)^i).$$

Now for the base case $i = 1$, we have

$$\begin{aligned} G_t^{(1)}(x, y) &= -\frac{\|x-y\|}{2t} G_t(x, y) = c_{0,1} \frac{\|x-y\|}{2t} G_t(x, y) \\ &= \sum_{j=0}^{\lfloor i/2 \rfloor} c_{j,i} \frac{\|x-y\|^{i-2j}}{(2t)^{i-j}} G_t(x, y). \end{aligned}$$

Thus the claim holds. For $G_t^{(i+1)}(x, y)$, we need to consider two cases - when i is odd and when i is even.

Case 1: i is odd: Inductive hypothesis states

$$G_t^{(i)}(x, y) = G_t(x, y) \left[c_{0,i} \frac{\|x-y\|^i}{(2t)^i} + c_{1,i} \frac{\|x-y\|^{i-2}}{(2t)^{i-1}} + \dots + c_{\frac{i-1}{2},i} \frac{\|x-y\|}{(2t)^{\frac{i+1}{2}}} \right]$$

We then have:

$$\begin{aligned} G_t^{(i+1)}(x, y) &= G_t(x, y) \left[i c_{0,i} \frac{\|x-y\|^{i-1}}{(2t)^i} - c_{0,i} \frac{\|x-y\|^{i+1}}{(2t)^{i+1}} \right. \\ &\quad + (i-2) c_{1,i} \frac{\|x-y\|^{i-3}}{(2t)^{i-1}} - c_{1,i} \frac{\|x-y\|^{i-1}}{(2t)^i} \\ &\quad \left. + \dots + c_{\frac{i-1}{2},i} \frac{1}{(2t)^{\frac{i+1}{2}}} - c_{\frac{i-1}{2},i} \frac{\|x-y\|^2}{(2t)^{\frac{i+3}{2}}} \right] \end{aligned}$$

Grouping terms together, we get

$$\begin{aligned}
G_t^{(i+1)}(x, y) &= G_t(x, y) \left[-c_{0,i} \frac{\|x-y\|^{i+1}}{(2t)^{i+1}} + [ic_{0,i} - c_{1,i}] \frac{\|x-y\|^{i-1}}{(2t)^i} \right. \\
&\quad + [(i-2)c_{1,i} - c_{2,i}] \frac{\|x-y\|^{i-3}}{(2t)^{i-1}} + \dots \\
&\quad \left. + [3 \cdot c_{\frac{i-3}{2},i} - c_{\frac{i-1}{2},i}] \frac{\|x-y\|^2}{(2t)^{\frac{i+3}{2}}} + c_{\frac{i-1}{2},i} \frac{1}{(2t)^{\frac{i+1}{2}}} \right] \\
&= \sum_{j=0}^{\frac{i+1}{2}} c_{j,i+1} \frac{\|x-y\|^{i+1-2j}}{(2t)^{i+1-j}} G_t(x, y)
\end{aligned}$$

where

$$c_{0,i+1} = -c_{0,i},$$

$$c_{j,i+1} = ((i+1) - 2j + 1)c_{j-1,i} - c_{j,i}, \quad 0 < j \leq \frac{i+1}{2},$$

$$c_{j,i+1} = 0, \text{ otherwise.}$$

Case 2: i is even: Can be shown by a similar argument to Case 1.

Finally, to bound the value of $c_{j,i}$, note that $c_{j,i} = (i - 2j + 1)c_{j-1,i-1} - c_{j,i-1} \leq i|c_{j-1,i-1}| - c_{j,i-1}$. One can then easily use the substitution method to show that $c_{j,i} = O((i+1)^i)$.

We remark that for simplicity, here we proceed as if x is a one-dimensional variable. In general, $x \in M$ is of m -dimension and one needs to compute the derivative w.r.t all mixed terms of coordinates. This will increase the bound by a factor that is exponential in m , but will not affect our final results. ■

Theorem 3.3.1. *Set $\mathcal{D}^{(j)}f = (\mathcal{D}f)^{(j)}$ to be the j -th (weak) derivative of the function $\mathcal{D}f$. We have that $\|\mathcal{D}^{(j)}f\| = O\left(\frac{\varepsilon}{i^{j+2}}(\|f\|_{H_j} + \|f\|_{H_{m/2+1}})\right)$ for $j \geq 0$, where the big- O notation hides constants exponential in j and dependent on the underlying manifold M .*

This implies that $\|\mathcal{L}_t^M - \mathcal{C}_t^K\|_{H^s} = O(\frac{\varepsilon}{t^{s+2}})$ for any $s \geq m/2 + 1$.

Proof: Recall that $\mathcal{D} = \mathcal{L}_t^M - \mathcal{C}_t^K$. Set $c(t)$ to be the constant $\frac{1}{t(4\pi t)^{m/2}}$, and let $|K|$ denote the underlying space of the mesh K . One way to interpret the mesh-Laplacian \mathbf{L}_t^K (as well as \mathcal{C}_t^K) is that, for any m -dimensional simplex $\sigma \in K$, subdivide it to $m + 1$ equal volume portions, with every portion σ' being represented by a different vertex, say v , of σ . We refer to the vertex v as the *pivot* p_z of every point z in this portion $\sigma' \subset \sigma$. The sampling condition of K implies that $\|z - p_z\| = O(\varepsilon)$. This way, we can rewrite

$$\mathcal{C}_t^K f(x) = c(t) \int_{|K|} G_t(x, p_z)(f(x) - f(p_z))dz,$$

and thus

$$\mathcal{D}f(x) = c(t) \int_M G_t(x, y)(f(x) - f(y))dy - c(t) \int_{|K|} G_t(x, p_z)(f(x) - f(p_z))dz.$$

Let $\phi : |K| \rightarrow M$ be the homeomorphism between $|K|$ and M so that K ε -approximates M . By change of variable $z = \phi^{-1}(y)$, we get the following where J_y is the Jacobian of the map $\phi^{-1} : M \rightarrow |K|$ at $y \in M$.

$$\begin{aligned} \mathcal{D}f(x) &= c(t) \int_M G_t(x, y)(f(x) - f(y))dy - c(t) \int_M G_t(x, p_y)(f(x) - f(p_y))J_y dy] \\ &= c(t) \left[\int_M G_t(x, p_y) f(p_y) J_y dy - \int_M G_t(x, y) f(y) dy \right] \\ &\quad - c(t) \left[\int_M G_t(x, p_y) f(x) J_y dy - \int_M G_t(x, y) f(x) dy \right]. \end{aligned}$$

It then follows that

$$\begin{aligned}
\mathcal{D}^{(j)} f(x) &= c(t) \left[\int_M G_t^{(j)}(x, p_y) f(p_y) J_y dy - \int_M G_t^{(j)}(x, y) f(y) dy \right] \\
&\quad + c(t) \left[\int_M \sum_{i=0}^j [G_t^{(i)}(x, y) f^{(j-i)}(x)] dy \right. \\
&\quad \left. - \int_M \sum_{i=0}^j [G_t^{(i)}(x, p_y) f^{(j-i)}(x)] J_y dy \right]
\end{aligned} \tag{3.3}$$

On the other hand, since $\|y - p_y\| = O(\varepsilon)$, we have

$$\|x - y\|^j - O(\varepsilon \|x - y\|^{j-1}) \leq \|x - p_y\|^j \leq \|x - y\|^j + O(\varepsilon \|x - y\|^{j-1}) \tag{3.4}$$

$$(1 - O(\varepsilon/t))G_t(x, y) \leq G_t(x, p_y) \leq (1 + O(\varepsilon/t))G_t(x, y) \tag{3.5}$$

Let G_t to denote $G_t(x, y)$ and $\alpha = O(\varepsilon^i)$ to simplify exposition. Using Lemma 3.3.4, we have:

$$\begin{aligned}
G_t^{(i)}(x, y) - G_t^{(i)}(x, p_y) &\leq \sum_{j=0}^{\lfloor i/2 \rfloor} |c_{j,i}| \left| \frac{\|x - p_y\|^{i-2j}}{t^{i-j}} G_t(x, p_y) - \frac{\|x - y\|^{i-2j}}{t^{i-j}} G_t \right| \\
&\leq \sum_{j=0}^{\lfloor i/2 \rfloor} \alpha \left[(1 + O(\frac{\varepsilon}{t})) \frac{\|x - p_y\|^{i-2j}}{t^{i-j}} G_t - \frac{\|x - y\|^{i-2j}}{t^{i-j}} G_t \right] \\
&\hspace{15em} \text{(Using 3.5)} \\
&\leq \sum_{j=0}^{\lfloor i/2 \rfloor} \alpha \left[(1 + O(\frac{\varepsilon}{t})) G_t \left(\frac{\|x - y\|^{i-2j}}{t^{i-j}} + \frac{\varepsilon \|x - y\|^{i-2j-1}}{t^{i-j}} \right) \right. \\
&\quad \left. - \frac{\|x - y\|^{i-2j}}{t^{i-j}} G_t \right] \\
&\hspace{15em} \text{(Using 3.4)} \\
&\leq \sum_{j=0}^{\lfloor i/2 \rfloor} \alpha G_t \left[O(\frac{\varepsilon}{t}) \frac{\|x - y\|^{i-2j}}{t^{i-j}} + O(\varepsilon) \frac{\|x - y\|^{i-2j-1}}{t^{i-j}} \right] \\
&\leq \sum_{j=0}^{\lfloor i/2 \rfloor} \alpha O\left(\frac{\varepsilon D^{i-2j}}{t^{i-j+1}}\right) G_t \leq O\left(\frac{i D^i \varepsilon}{t^{i+1}}\right) G_t(x, y),
\end{aligned}$$

where D is the diameter of the manifold M . Furthermore, as $\|x - y\| \leq D$ and $G_t(x, y) \leq 1$,

$$G_t^{(i)}(x, y) = O\left(\frac{i \cdot D^i}{t^i} G_t(x, y)\right) = O\left(\frac{1}{t^i}\right) G_t(x, y)$$

Combined with Equation (3.3) and that $|J_y - 1| = O(\varepsilon)$, we have that (again, G_t denotes $G_t(x, y)$):

$$\begin{aligned} \frac{1}{c(t)} \left| \mathcal{D}^{(j)} f(x) \right| &\leq \left| \int_M [G_t^{(j)}(x, p_y) f(p_y) J_y - G_t^{(j)} f(y)] dy \right| \\ &\quad + \sum_{i=0}^j \left| f^{(j-i)}(x) \int_M [G_t^{(i)} - G_t^{(i)}(x, p_y) J_y] dy \right| \\ &\leq \left| \int_M [(1 + O(\varepsilon)) [O\left(\frac{\varepsilon G_t}{t^{j+1}}\right) + G_t^{(j)}] f(p_y) - G_t^{(j)} f(y)] dy \right| \\ &\quad + \sum_{i=0}^j \left| f^{(j-i)}(x) \int_M [G_t^{(i)} - (1 + O(\varepsilon)) (O\left(\frac{\varepsilon G_t}{t^{i+1}}\right) + G_t^{(i)})] dy \right| \\ &\leq \left| \int_M O\left(\frac{\varepsilon G_t}{t^{j+1}}\right) f(p_y) dy + \int_M G_t^{(j)} [(1 + O(\varepsilon)) f(p_y) - f(y)] dy \right| \\ &\quad + \sum_{i=0}^j \left| f^{(j-i)}(x) \int_M [O\left(\frac{\varepsilon}{t^{i+1}}\right) G_t - O(\varepsilon) G_t^{(i)}] dy \right| \\ &\leq \left| \int_M O\left(\frac{\varepsilon}{t^{j+1}}\right) G_t [O(\varepsilon) \text{Lip}_f + f(y)] dy \right| \\ &\quad + \left| \int_M O\left(\frac{\varepsilon G_t}{t^j}\right) [\text{Lip}_f + f(y)] dy \right| + \sum_{i=0}^j f^{(j-i)}(x) \int_M O\left(\frac{\varepsilon G_t}{t^{i+1}}\right) dy \end{aligned}$$

Hence

$$\begin{aligned} \mathcal{D}^{(j)} f(x) &\leq c(t) O\left(\frac{\varepsilon \text{Lip}_f}{t^{j+1}}\right) \int_M G_t dy + c(t) O\left(\frac{\varepsilon}{t^{j+1}}\right) \left| \int_M G_t f(y) dy \right| \\ &\quad + c(t) \sum_{i=0}^j f^{(j-i)}(x) \cdot O\left(\frac{\varepsilon}{t^{i+1}}\right) \int_M G_t dy \end{aligned}$$

where Lip_f is the Lipschitz constant of the function f , which is bounded by $C\|f\|_{H_s}$ by Lemma 3.3.1. Furthermore, by Corollary 3.3.1, $f(y) \leq \|f\|_\infty \leq C'\|f\|_{H_s}$. Combining

these with Claim 3.2.1 we have that:

$$|\mathcal{D}^{(j)} f(x)| \leq O\left(\frac{\varepsilon}{t^{j+2}}\right) \|f\|_{H_s} + \sum_{i=0}^j f^{(j-i)}(x) O\left(\frac{\varepsilon}{t^{i+2}}\right).$$

This implies

$$\begin{aligned} \|\mathcal{D}^{(j)} f\| &\leq O\left(\frac{\varepsilon}{t^{j+2}}\right) \|f\|_{H_s} + O\left(\frac{\varepsilon}{t^{j+2}}\right) \sum_{i=0}^j \|f^{(j-i)}\| \\ &= O\left(\frac{\varepsilon}{t^{j+2}}\right) \|f\|_{H_s} + O\left(\frac{j\varepsilon}{t^{j+2}}\right) \|f\|_{H_j} \\ &= O\left(\frac{\varepsilon}{t^{j+2}}\right) (\|f\|_{H_s} + \|f\|_{H_j}). \end{aligned}$$

The bound on the s -th Sobolev norm for $\mathcal{L}_t^M - \mathcal{C}_t^K$ then follows easily. ■

Putting everything together. Now, for a fixed t , consider a sequence of meshes $\{K(\varepsilon)\}_{\varepsilon \rightarrow 0}$ that ε -approximates M and converges to M as ε goes to zero. This induces a sequence of discrete Laplace operators $\{\mathbf{L}_t^{K(\varepsilon)}\}_{\varepsilon \rightarrow 0}$ as well as a sequence of their continuous extensions $\{\mathcal{C}_t^{K(\varepsilon)}\}_{\varepsilon \rightarrow 0}$ in the Sobolev space $H_{m/2+1}(M)$. All these operators have only real eigenvalues. Theorem 3.3.1 implies that the sequence of operators $\{\mathcal{C}_t^{K(\varepsilon)}\}_{\varepsilon \rightarrow 0}$ converges in operator norm to the functional Laplacian \mathcal{L}_t^M as ε goes to zero. Using Proposition 6 in [95], by a similar argument as the one used in Section 3.2, when ε is small enough, there is a one to one correspondence between the lowest few eigenvalues of $\mathcal{C}_t^{K(\varepsilon)}$ and $\text{SpecDis}(\mathcal{L}_t^M)$ such that the i -th one from $\text{SpecDis}(\mathcal{C}_t^{K(\varepsilon)})$ converges to the i th one from $\text{SpecDis}(\mathcal{L}_t^M)$ as ε goes to zero. Since $\mathcal{C}_t^{K(\varepsilon)}$ shares discrete eigenvalues with $\mathbf{L}_t^{K(\varepsilon)}$ (precise statement in Lemma 3.3.3), this then implies Theorem 3.1.2. Finally, Theorem 3.1.3 follows from this and Theorem

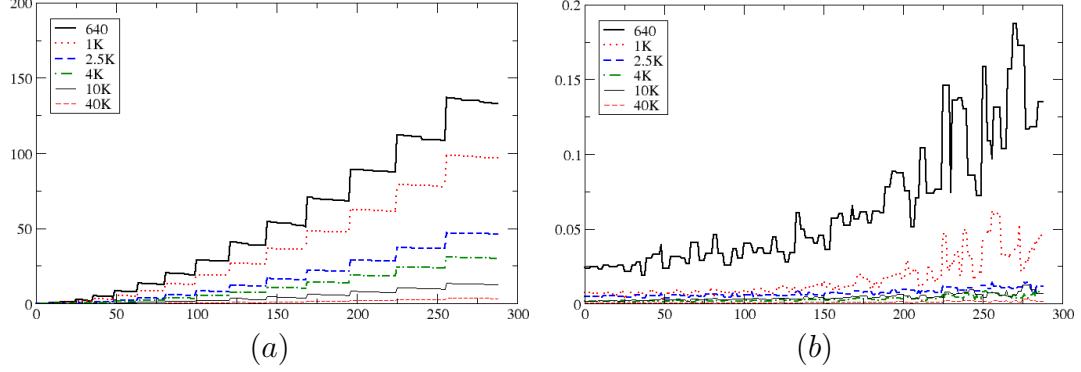


Figure 3.2: Errors in the (a) eigenvalues and (b) eigenvectors of discrete Laplacian of meshes of unit sphere with increasing number of vertices.

3.1.1. By 3.3.1, the condition $\frac{\varepsilon}{t^{\frac{m}{2}+3}} \rightarrow 0$ in the limit guarantees that the sequence of continuous extensions $\{\mathcal{L}_t^{K(\varepsilon)}\}_{\varepsilon \rightarrow 0}$ converges to \mathcal{L}_t^M in operator norm.

Proof of Theorem 3.1.6. Imagine that we have a sequence of manifolds $\{N_{\delta_H}\}_{\delta_H \rightarrow 0}$ that is δ_H -close to M and δ_H converges to zero. Now choose $t(\delta_H) = \Omega(\delta_H^{\frac{4}{m+8}-\nu})$ for some small constant $\nu > 0$ and denote $\mathcal{L}_{t(\delta_H)}^{N_{\delta_H}}$ by $\mathcal{L}^N(\delta_H)$. By Lemma 3.2.2, the sequence of manifolds $(N_{\delta_H})_{\delta_H \rightarrow 0}$ induces a sequence of operators $(\mathcal{L}^N(\delta_H))_{\delta_H \rightarrow 0}$ that converges to \mathcal{L}_t^M in operator norm. Combining Theorem 3.1.2, Theorem 3.1.6 then follows from a similar argument as above.

3.4 Experiments

In this section, we show through experiments that the spectrum of the mesh Laplacian [7] converges to that of the manifold Laplacian, is robust, and changes smoothly with smooth deformation of a surface. For all our experiments, we normalize the input surface to diameter 1. We use the code from Belkin et al. [7] to compute the mesh-Laplacian, and use MATLAB[®] to find its first 300 eigenvalues and eigenvectors.

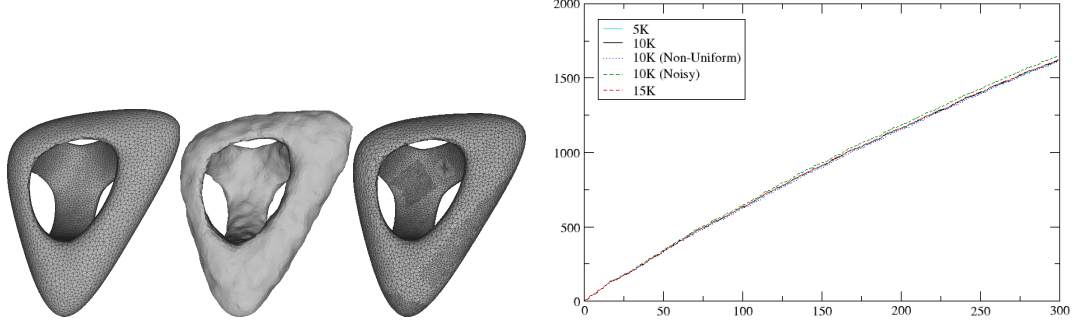


Figure 3.3: Original, noisy, and non-uniform meshes for the same genus 3 surface. Bottom : comparison of their eigenvalues.

To demonstrate the convergence behavior, we consider a sequence of increasingly denser meshes approximating a unit sphere, for which we can obtain the ground truth. We use an adaptive t , which is set to be 6 times the average edge length in the mesh. Hence, t becomes smaller as the meshes become denser. The results are shown in Figure 3.2, where we plot the error of each of the first 300 eigenvalues / eigenfunctions (x -axis is the index of the eigenvalue/eigenfunction). In (a) we plot for each i , the difference $|\lambda_i - \lambda_i^D|$, where λ_i and λ_i^D are the i th eigenvalue of the manifold and mesh Laplacians, respectively. In (b) we plot the error in eigenvectors. Specifically, note that the restriction of each ground truth eigenfunction ϕ_i to the vertices of the mesh gives us a vector $\widehat{\phi}_i$. We compute the error as the L_2 -norm distance between $\widehat{\phi}_i$ and the corresponding discrete eigenvector of the mesh Laplacian. If an eigenvalue has multiplicity more than 1, we project the discrete eigenvector into the eigenspace spanned by the restricted eigenfunctions corresponding to that eigenvalue and return the error as distance between this vector and its projection. As we can see, the eigenvalues and eigenvectors converge to ground truth as the sampling density increases.

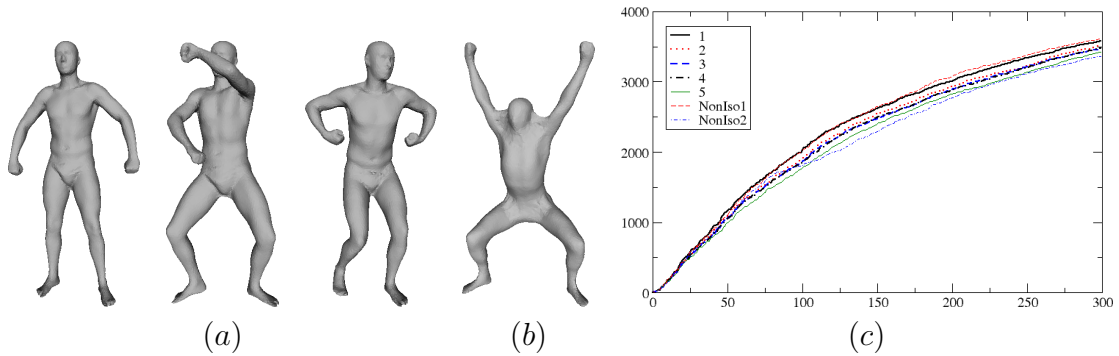


Figure 3.4: (a) Some near-isometric deformations of a human. (b) An example of non-isometric deformation. (c) Comparison of spectra computed from five isometric and two non-isometric deformations.

Next, we show that, with a fixed t , the mesh-Laplacian is robust against changes in the sampling density, noise, and quality of sampling. Here we use a more interesting genus 3 surface (see Figure 3.3), and plot the spectra of different meshes in the bottom picture, where x -axis is the index of each eigenvalue, and y -axis is the value. All these curves are close, indicating that the discrete Laplacian spectra are resilient to these changes.

For nearly isometric deformations, we use various poses of a human figure (Figure 3.4), and show that the discrete Laplacian spectrum is robust against such deformations. Finally, we investigate how the discrete Laplacian spectrum changes as the manifold undergoes larger deformations. Specifically, we continuously deform a figure-eight loop and plot the corresponding discrete Laplacian spectra. See Figure 3.5 and note the spectrum also changes continuously with the deformations.

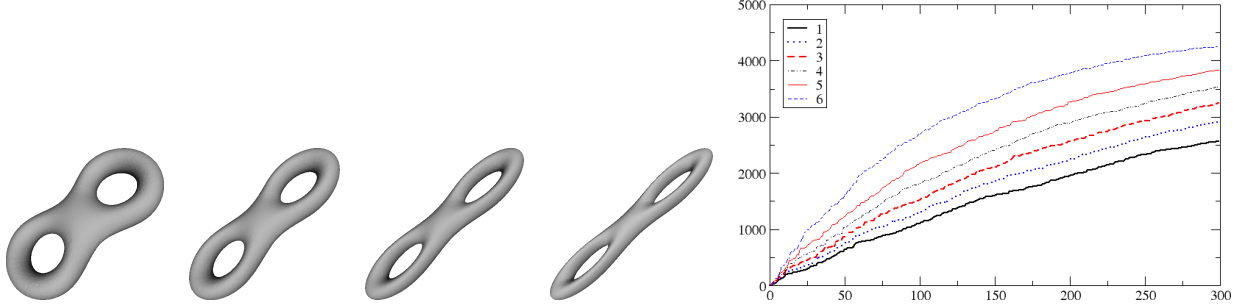


Figure 3.5: Snapshots of continuous deformation of an eight loop and plot of spectra of corresponding meshes.

3.5 Conclusion and Discussion

This chapter provides the first result showing that eigenvalues of a certain discrete Laplace operator [7] approximated from a general mesh in d -dimensional space converge to those of the manifold Laplacian as the mesh converges to a smooth manifold. It also shows that the spectrum of this discrete mesh-Laplacian is stable when the smooth manifold is perturbed, which is demonstrated by experimental studies. This helps to provide theoretical guarantees for applications using the mesh-Laplace operator.

In this chapter, we only focus on the eigenvalues of the Laplace operator. Another important family of eigen-structures is the set of Laplacian eigenfunctions. Indeed, these eigenfunctions have been widely used in spectral mesh processing applications. We believe that similar convergence results can be obtained for the eigenfunctions as

well ³ using the separation gap between consecutive distinct eigenvalues. Experimental results also show that eigen-spaces are stable. We leave the precise statement and formal proof of stability for eigenfunctions as an immediate future work.

Another future work is to investigate similar problems for discrete point-cloud Laplace operator, constructed from a set of unorganized points sampled from a hidden manifold. Such input is common as demonstrated by the plethora of high dimensional data in various scientific and engineering applications. As a result, many recent work focus on processing point data for spectral shape analysis. It appears that results from this chapter can be extended to the point-clouds Laplacian proposed in [55] when the input points is a so-called (ε, η) -sample of a manifold M ; namely, (i) for every point $p \in M$ there is a sample point at most $\varepsilon\rho(M)$ away, where $\rho(M)$ is the reach of M , and (ii) no two sample points are within distance $\eta\rho(M)$. It will be interesting to see whether similar results can be established for the more general ε -sampling without the η -sparsity condition. In Chapter 2 we will show that the weighted graph Laplacian proposed in [5] is stable against topological changes made in a small region with respect to a parameter t .

Finally, most of our results only show convergence instead of explicitly bounding the error between the discrete and true Laplacian spectra. An explicit error bound not only helps the theoretical understanding of discrete mesh Laplacian but also has practical implications. It will be interesting to explore this direction.

³To be more careful, for eigenvalues with multiplicity more than one, we should consider the eigenspace spanned by the corresponding eigenfunctions.

Chapter 4: Weighted Graph Laplace Operator under Topological Noise [27]

As discussed in Chapter 2, Gaussian-weighted graph Laplace is a popular operator due to the fact that it works directly on point cloud data (PCD), without relying on triangulations. Naturally, the question of robustness of this discrete Laplacian with respect to perturbations of the sampled manifold becomes a practical issue. In particular, estimating the change in spectrum of the discrete Laplacian for PCDs sampled from manifolds M and its perturbed version N becomes important. In the previous chapter, we showed that both manifold and discrete Laplacians are stable when the perturbation is “nice” in the sense that there is a homeomorphism between M and N that cause minor area distortion. This however forbids, for example, small topological changes between M and N .

In this chapter, we aim to study the stability of the weighted graph Laplacian under more general perturbations. In particular, we allow arbitrary changes to the hidden manifold as long as they are localized in the ambient space and the area distortion is small. Manifold Laplacians may change dramatically in this case. Nevertheless, we show that the weighted graph Laplacians computed from two sets of points uniformly randomly sampled from M and its perturbed version N maintain a small distance in their spectra (i.e, sequence of eigenvalues). This distance can

be bounded in terms of the perturbation and some intrinsic properties of M and N . Intuitively, the discrete graph Laplacians with parameters chosen appropriately are oblivious to perturbations that are localized in the ambient space.

Contributions. Our previous stability study requires that the input manifold is approximated by a mesh structure. The perturbation considered there needs to preserve the manifold topology and be smooth in the first order. In some sense, such a perturbation model is necessary as it also bounds changes in the manifold Laplacians. Now, we aim to



relax on both fronts. First, since we will focus on the Gaussian-weighted graph Laplacian, we do not need triangulations. Second, we introduce the δ -closeness between two manifolds which generalizes δ_H -closeness from Section 3.1 to allow for arbitrary though localized changes including topological ones. See the right figure for an example where the right hand of the human may either touch or not touch the human body within the circled region, causing a potential topological difference. Since this relaxed perturbation model can alter geodesic distances dramatically, the manifold Laplacian may change significantly under it. However, we show that the weighted graph Laplacian under such perturbations remains stable.

Specifically, we consider two sets of points P and Q uniformly randomly sampled from two δ -close manifolds M and N . We provide an asymptotic upper bound on the distance between the spectra of the weighted graph Laplacians L_P^t and L_Q^t constructed from P and Q respectively. This bound tells us how the spectra distance depends on the size of the perturbation δ , and on the parameter t used to construct the graph Laplacians. This bound is proven for two *uniformly randomly* sampled sets of points. This choice is reasonable since (i) the theoretical guarantee of the weighted graph

Laplacian is only established for such PCDs, and (ii) in practice, especially in high dimensional applications, input points are often sampled by certain random processes.

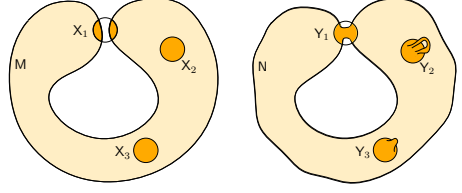
Our result is obtained by a novel way to establish a one-to-one correspondence between the two input sample point sets via the so-called “anchor-regions” that we will introduce later, and the Halls theorem, in the probabilistic setting. In Section 4.4, we show some experiments that confirm our theoretical results.

4.1 Problem Formulation

δ -closeness. In Section 3.1, we defined a notion of δ_H -closeness between homeomorphic manifolds. Now, we will relax the definition to allow topological changes. Recall that, given two topological spaces $A, B \subseteq \mathbb{R}^d$, a map $f : A \rightarrow B$ is a δ -*diffeomorphism* if f is a diffeomorphism, $d_{\mathbb{R}^d}(x, f(x)) \leq \delta$, and $1 - \delta \leq \|Jf_x\| \leq 1 + \delta$ for any $x \in A$, where Jf_x is the Jacobian of the map f at x . Now, our inputs are two smooth and compact m -manifolds \mathbf{M} and \mathbf{N} embedded in \mathbb{R}^d . Manifolds \mathbf{M} and \mathbf{N} are δ -close if there exists two sets of open regions $\mathcal{X} = \{X_1, \dots, X_m\}$, $X_i \subset \mathbf{M}$, and $\mathcal{Y} = \{Y_1, \dots, Y_m\}$, $Y_i \subset \mathbf{N}$ where the closures of X_i s (and of Y_i s) are pairwise disjoint and the following conditions hold.

- (C1) There is a δ -diffeomorphism $\Phi : \mathbf{M} \setminus \mathcal{X} \rightarrow \mathbf{N} \setminus \mathcal{Y}$ under which $\partial(\text{closure}(X_i))$ is mapped to $\partial(\text{closure}(Y_i))$ for any $i \in [1, m]$ where $\partial(\cdot)$ denotes the boundary.
- (C2) For any $i = 1, \dots, m$, X_i and Y_i are contained within a *Euclidean* d -ball of radius δ and $(1 - \delta)\text{vol}(X_i) \leq \text{vol}(Y_i) \leq (1 + \delta)\text{vol}(X_i)$.

Intuitively, \mathbf{N} is a perturbed version of \mathbf{M} . The regions in \mathcal{X} and \mathcal{Y} are the *anomalous regions* where arbitrary changes, including topological changes,



can happen. See the right figure for an illustration, where two types of topological changes happen in \mathcal{Y}_1 and \mathcal{Y}_2 , respectively. Such arbitrary changes can be tolerated as long as they are restricted to a small Euclidean ball, and the areas of corresponding anomalous regions are similar (i.e, condition C2).

High level framework. Given two δ -close manifolds \mathbf{M} and \mathbf{N} , let P and Q be two sets of n points, uniformly randomly sampled from \mathbf{M} and \mathbf{N} according to volume measures. We compute the weighted Graph Laplacians \mathbf{L}_P^t and \mathbf{L}_Q^t from P and Q respectively, for some parameter t . Our goal is to show that the spectrum of \mathbf{L}_P^t is close to that of \mathbf{L}_Q^t with high probability. We prove this via the following two steps.

Step 1. We show that, with high probability, there is a one-to-one correspondence

$$\psi : P \rightarrow Q \text{ such that } d_{\mathbb{R}^d}(p, \psi(p)) = O(\delta) \text{ for any } p \in P.$$

Step 2. Based on the one-to-one correspondence obtained from Step 1, we show that

the matrix norm $\|\mathbf{L}_P^t - \mathbf{L}_Q^t\|$ is bounded from above by a function $\mathcal{E}(\delta, t)$ which in turn gives an upper bound on the distance of the spectra for \mathbf{L}_P^t and \mathbf{L}_Q^t .

Our main result (stated below) follows from these two steps. Let $i(\mathbf{M})$ denote the injectivity radius of \mathbf{M} , and μ be the isoperimetric constant of \mathbf{M} , see Chavel [14] for definitions of these quantities.

Theorem 4.1.1. *Let \mathbf{M} be a smooth compact m -dimensional manifold embedded in \mathbb{R}^d with $\text{vol}(\mathbf{M}) = 1$, and \mathbf{N} be its δ -close perturbation with $\delta < \min\{\frac{1}{8}, i(\mathbf{M})\}$. Let P and Q be two sets of points uniformly randomly sampled from \mathbf{M} and \mathbf{N} , respectively,*

with $|P| = |Q| = n = \Omega(\frac{1}{\delta^{4m}})$. Let \mathbf{L}_P^t and \mathbf{L}_Q^t be the corresponding Gaussian-weighted graph Laplacians computed from them, with the parameter $t = \Omega(\delta^{2-\varepsilon})$ for any $\varepsilon > 0$. With high probability (at least $1 - O(\frac{1}{n^4})$), the eigenvalues of \mathbf{L}_P^t and \mathbf{L}_Q^t satisfy $|\lambda_i(\mathbf{L}_P^t) - \lambda_i(\mathbf{L}_Q^t)| = O(\frac{\delta^{4/5}}{t^{m/2+7/5}})$ for any $i \in [1, n]$. In particular, $|\lambda_i(\mathbf{L}_P^t) - \lambda_i(\mathbf{L}_Q^t)| = O(\delta^{\frac{1}{3}})$ if $t \geq \delta^{\frac{15}{14}m+3}$. The big- O and big- Ω notations hide constants that depend solely on M .

Isoperimetric constant. Our bounds depend on a concept called the isoperimetric constant. Given a m -dimensional compact Riemannian manifold A , the *isoperimetric constant* $\mu(A)$ of A is defined as $\mu(A) = \inf_{R \subset A} \frac{\text{vol}(\partial R)}{\text{vol}(R)}$, where R ranges over m -dimensional open subsets of A , with $\text{vol}(R) \leq \text{vol}(A)/2$, and $\text{vol}(\partial R)$ refers to the $(m-1)$ -dimensional volume of the boundary of R . This implies that given any m -dimensional open subset $R \subseteq A$ with volume at most $\text{vol}(A)/2$, we have $\text{vol}(\partial R) \geq \mu(A)\text{vol}(R)$. The isoperimetric constant, also called the *Cheeger isoperimetric constant*, is an intrinsic quantity of the manifold A and it is closely related to the first non-zero eigenvalue of the Laplace-Beltrami operator of A [14].

4.2 Step 1: Correspondences

We are given two random samples P and Q , of size n each, from two δ -close manifolds M and N . The goal is to show that, with high probability, there exists a one-to-one correspondence $\psi : P \rightarrow Q$ such that corresponding pairs of points are close. To achieve this goal, we construct a bipartite graph $G = (V, E)$ where $V = P \cup Q$ and $E \subseteq P \times Q$ so that $d_{\mathbb{R}^d}(p, q) = O(\delta)$ for each edge $(p, q) \in E$. Given a node $p \in P$, let $Ng(p) \subseteq Q$ denote the set of neighbors of p in Q , and define $Ng(S) = \bigcup_{p \in S} Ng(p)$ for any subset $S \subseteq P$. We then argue that with high

probability, we have $|S| \leq |Ng(S)|$ for all subsets S of P . It then follows from Hall's Theorem that with high probability, there is a perfect bipartite matching of G , inducing a one-to-one correspondence $\psi : P \rightarrow Q$ with $d_{\mathbb{R}^d}(p, \psi(p)) = O(\delta)$ for any $p \in P$.

4.2.1 Bipartite graph construction

Our construction of the bipartite graph for points $P \cup Q$ uses “anchor-nodes” and “anchor-regions” as detailed below. First, an $(\varepsilon_1, \varepsilon_2)$ -sample of a manifold M is a set of points $S \subset M$ such that (i) for any point $x \in M$, there is a sample point $s \in S$ within ε_1 geodesic distance away from x ; and (ii) any two sample points $s_1, s_2 \in S$ are at least ε_2 geodesic distance apart. A set of *anchor-nodes* $\mathcal{A} = \{\mathbf{a}_1, \dots, \mathbf{a}_r\}$ of M is simply a (δ, δ) -sample of M .

Constructing an (δ, δ) -sample. We can compute a set of anchor-nodes $\mathcal{A} = \{\mathbf{a}_1, \dots, \mathbf{a}_r\} \subset M$, which is simply a (δ, δ) -sample of M , using the following standard iterative procedure.

Initialize \mathcal{A} with an arbitrary point $\mathbf{a}_1 \in M$. In the i -th round after constructing $\mathcal{A}_{i-1} := \{\mathbf{a}_1, \dots, \mathbf{a}_{i-1}\}$, identify the point from M that is furthest away, in terms of geodesic distance, from points in \mathcal{A}_{i-1} and set it as \mathbf{a}_i if this distance is at least δ . We stop when this distance is smaller than δ . The procedure creates a (δ, δ) -sample. Indeed, no point in M can be further than δ away from its nearest neighbor in \mathcal{A} (otherwise, the procedure will continue), and no two \mathbf{a}_i and \mathbf{a}_j , $i < j$, are within δ distance since \mathbf{a}_j is at least δ away from \mathbf{a}_i .

The following observation is straightforward.

Observation 4.2.1. *Assume that $\delta < i(M)$. For a (δ, δ) -sampling \mathcal{A} of M , we have that $|\mathcal{A}| = O(1/\delta^m)$, where the big- O notation hides constants that depend on the intrinsic property of M .*

Proof: Compute the geodesic Voronoi diagram of \mathcal{A} on M . We claim that the Voronoi cell $\text{Vor}(\mathbf{a}_i)$ for each point $\mathbf{a}_i \in \mathcal{A}$ contains the geodesic ball of radius $\delta/2$ centered at \mathbf{a}_i . Indeed, if this is not the case, then there exists some point y on the boundary of $\text{Vor}(\mathbf{a}_i)$ such that $d_M(y, \mathbf{a}_i) < \delta/2$. Since y is on the boundary of $\text{Vor}(\mathbf{a}_i)$, there is another site, say $\mathbf{a}_j \in \mathcal{A}$ such that $d_M(y, \mathbf{a}_j) = d_M(y, \mathbf{a}_i)$ and thus $d_M(\mathbf{a}_i, \mathbf{a}_j) < \delta$ by triangle inequality. This contradicts the fact that \mathcal{A} is a (δ, δ) -sample. The volume of a geodesic ball with radius $\delta/2$ is $C_M \delta^m$ where C_M is a constant that depends on the intrinsic curvature of M and $\delta/2$ is smaller than the injectivity radius. It follows that $\text{vol}(\text{Vor}(\mathbf{a}_i)) = \Omega(\delta^m)$ implying $|\mathcal{A}| = O(1/\delta^m)$ by a packing argument. ■

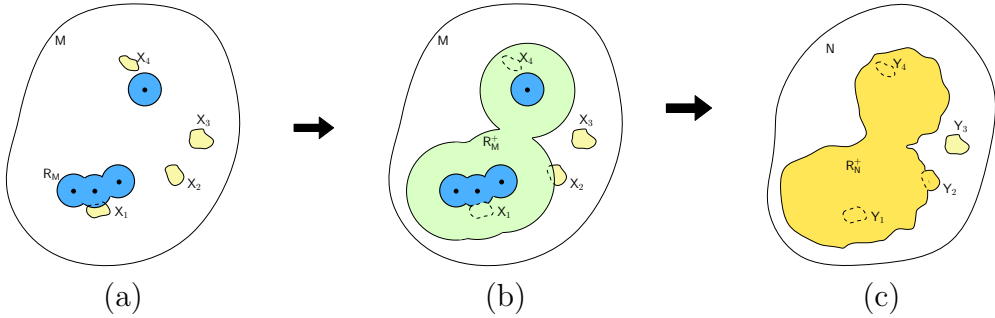


Figure 4.1: (a) Dark region is the anchor-region R_M induced by the black anchor-nodes (other anchor-nodes are not shown): R_M consists of points from M within δ Euclidean distance to black points. Light regions are anomalous regions X_1, \dots, X_4 . (b) The intermediate region $R_N \subset M$ contains anomalous regions X_1 and X_4 fully, and X_2 partially. (c) The witness anchor-region $R_N^+ \subset N$ of R_M includes anomalous regions Y_1, Y_2 and Y_4 fully.

Anchor-regions. Consider an arbitrary subset of anchor-nodes $A \subseteq \mathcal{A}$. Let $d_{\mathbb{R}^d}(x, A)$ denote the smallest Euclidean distance from x to any point in A . The *anchor-region* $R_M(A)$ on M induced by A is the set of points whose distance to A is at most δ ; that is, $R_M(A) = \{x \in M \mid d_{\mathbb{R}^d}(x, A) \leq \delta\}$. We call A the *defining subset* for $R_M(A)$. There are $2^{|\mathcal{A}|} = 2^{O(1/\delta^m)}$ number of anchor-regions on M , each defined by one subset of \mathcal{A} . Next, we define the set of anchor-regions on the manifold N . Each anchor-region $R_M = R_M(A)$ gives rise to one anchor-region $R_N^+ = R_N^+(A)$ on N which is constructed via an intermediate region R_N . We refer to R_N^+ as the *witness anchor-region* of R_M . See Figure 4.1 for an illustration.

First, we construct an intermediate region $R_N \subseteq M$, which contains all points from M within $\rho\delta$ Euclidean distance to R_M ; that is, $R_N = \{x \in M \mid d_{\mathbb{R}^d}(x, R_M) \leq \rho\delta\}$. The value of $\rho = O(1)$ will be specified shortly; it depends on the isoperimetric constant of the manifold M . Next, we “map” the region R_N to $R_N^+ \subseteq N$. Specifically, recall that there is a δ -diffeomorphism $\Phi : M \setminus \mathcal{X} \rightarrow N \setminus \mathcal{Y}$. We set $R_N^+ = \Phi(R_N \setminus \mathcal{X}) \cup \bigcup_{i \in I} Y_i$, where $I = \{i \in [1, m] \mid X_i \cap R_N \neq \emptyset\}$ is the set of indices of anomalous regions from M that intersect R_N . Intuitively, the witness anchor-region R_N^+ on N of an anchor-region $R_M \subseteq M$ is obtained by thickening R_M by $\rho\delta$ Euclidean distance on M , and then map R_N to R_N^+ on N . Since the diffeomorphism only exists between $M \setminus \mathcal{X}$ and $N \setminus \mathcal{Y}$, we need to process the intersection of R_N with anomalous regions separately when “mapping” R_N onto N .

Observation 4.2.2. (i) Given an anchor-region $R_M(A)$ induced by $A \subseteq \mathcal{A}$, its witness anchor-region in N satisfies that $R_N^+(A) = \bigcup_{a \in A} R_N^+(a)$. (ii) Given an anchor-node $a \in \mathcal{A}$, we have $d_{\mathbb{R}^d}(a, y) \leq (\rho + 2)\delta$ for any point $y \in R_N^+(a)$.

Graph construction. We now build a bipartite graph $G = (V, E)$ from input point sets $P \subset \mathbf{M}$ and $Q \subset \mathbf{N}$ as follows: The vertex set is $V = P \cup Q$. For each point $p \in P$, let $\mathbf{a}(p) \in \mathcal{A}$ denote the nearest neighbor of p in the set of anchor-nodes \mathcal{A} . We connect p to all points in Q falling inside the region $\mathbf{R}_{\mathbf{N}}^+(\mathbf{a}(p)) \subseteq \mathbf{N}$; that is, $Ng(p) = Q \cap \mathbf{R}_{\mathbf{N}}^+(\mathbf{a}(p))$ in G . For each point $q \in Ng(p)$, by Observation 4.2.2, $d_{\mathbb{R}^d}(\mathbf{a}(p), q) \leq (\rho + 2)\delta$. Furthermore, since \mathcal{A} is a (δ, δ) -sample of \mathbf{M} , we have that $d_{\mathbf{M}}(p, \mathbf{a}(p)) \leq \delta$. We thus have:

Claim 4.2.1. *For any edge (p, q) in G , $p \in P$, $q \in Q$, and $d_{\mathbb{R}^d}(p, q) \leq (\rho+3)\delta = O(\delta)$.*

Claim 4.2.2. *Given any subset $S \subseteq P$, let $\mathcal{A}_S = \{\mathbf{a}(s) \in \mathcal{A} \mid s \in S\}$ be the union of nearest anchor-nodes to each point in S . Then $S \subseteq P \cap \mathbf{R}_{\mathbf{M}}(\mathcal{A}_S)$ and $Ng(S) = Q \cap \mathbf{R}_{\mathbf{N}}^+(\mathcal{A}_S)$.*

4.2.2 Bipartite Matching in G

Consider the graph $G = (V, E)$ as constructed above. We wish to show that, for a uniform random sample P of \mathbf{M} and Q of \mathbf{N} , there exists a perfect bipartite matching in G with high probability.

In what follows, we assume that \mathbf{M} has volume 1. By the δ -closeness we have $(1 - \delta)\text{vol}(\mathbf{M}) \leq \text{vol}(\mathbf{N}) \leq (1 + \delta)\text{vol}(\mathbf{M})$; thus $1 - \delta \leq \text{vol}(\mathbf{N}) \leq 1 + \delta$. Let $\mu = \mu(\mathbf{M})$ be the isoperimetric constant for \mathbf{M} . We assume that $\delta \leq \min\{\frac{1}{8}, i(\mathbf{M})\}$, and $\rho = \max\{3, \frac{8}{\mu} + 2\}$. Note that by this choice of ρ , we have that $\rho\mu \geq 8$.

Given a region R , let $\#_X(R)$ denote the number of points from a point set X contained inside R . We prove the following key result later in this section.

Lemma 4.2.1. *Given two δ -close compact and smooth m -manifolds \mathbf{M} and \mathbf{N} with $\text{vol}(\mathbf{M}) = 1$ and $\delta < \min\{\frac{1}{8}, i(\mathbf{M})\}$, let P and Q be two sets of $n = \Omega(\frac{1}{\delta^{4m}})$ random samples from \mathbf{M} and \mathbf{N} respectively. Then, with probability at least $1 - O(\frac{1}{n^4})$, $\#_P(\mathbf{R}_M) \leq \#_Q(\mathbf{R}_N^+)$ for all anchor-regions $\mathbf{R}_M \subseteq \mathbf{M}$ and their witness anchor-regions $\mathbf{R}_N^+ \subseteq \mathbf{N}$.*

Our main result in this section follows from the above lemma.

Theorem 4.2.1. *Let \mathbf{M} and \mathbf{N} be two δ -close compact and smooth m -manifolds embedded in \mathbb{R}^d with $\text{vol}(\mathbf{M}) = 1$, and $\delta < \min\{\frac{1}{8}, i(\mathbf{M})\}$. Let P and Q be n uniform random samples of \mathbf{M} and \mathbf{N} , respectively. Then for large enough $n = \Omega(\frac{1}{\delta^{4m}})$, there is a one-to-one correspondence $\psi : P \rightarrow Q$ such that for any $p \in P$, $d_{\mathbb{R}^d}(p, \psi(p)) \leq (\rho + 3)\delta$ where $\rho = \max\{3, \frac{8}{\mu} + 2\}$.*

Proof: Consider the anchor-regions in \mathbf{M} and their witness anchor-regions in \mathbf{N} as described in Section 4.2.1. Then construct the bipartite graph $G = (V, E)$ as described earlier. Now consider the following two events:

(Event-1): $\#_P(\mathbf{R}_M) \leq \#_Q(\mathbf{R}_N^+)$ for all anchor-regions $\mathbf{R}_M \subseteq \mathbf{M}$ and their witness anchor-regions $\mathbf{R}_N^+ \subseteq \mathbf{N}$; and (Event-2): $|S| \leq |Ng(S)|$ for all subsets $S \subseteq P$.

By Claim 4.2.2, for each subset $S \subseteq P$, there is always an anchor-region \mathbf{R}_M such that $|S| \leq \#_P(\mathbf{R}_M)$ and $Ng(S) = \#_Q(\mathbf{R}_N^+)$. Hence Event-2 must happen if Event-1 happens. This means that $\text{Prob}[\text{Event-1}] \leq \text{Prob}[\text{Event-2}]$. By Lemma 4.2.1, we have that $\text{Prob}[\text{Event-1}] \geq 1 - O(\frac{1}{n})$, implying that $\text{Prob}[\text{Event-2}] \geq 1 - O(\frac{1}{n})$.

If Event-2 happens, then by Hall's Theorem there is a perfect bipartite matching in the graph G . This provides a one-to-one correspondence $\psi : P \rightarrow Q$, where there

is an edge in the bipartite graph G between each pair $(p, \psi(p))$. By Claim 4.2.1, $d_{\mathbb{R}^d}(p, \psi(p)) \leq (\rho + 3)\delta$, which completes the proof. \blacksquare

It remains to prove Lemma 4.2.1. To this end, we first take an arbitrary but fixed anchor-region $R_M \subseteq M$ and its witness anchor-region $R_N^+ \subseteq N$, and argue that $\#_M(R_M) \leq \#_N(R_N^+)$ with high probability. Let R_N be the thickening of R_M by $\rho\delta$ in M as used in the construction of R_N^+ in Section 4.2.1. We distinguish two cases depending on the volume of R_N . Lemma 4.2.1 then follows from these results and the union-bound.

Case 1: $\text{vol}(R_N) \leq \frac{\text{vol}(M)}{2}$.

First, we aim to obtain a relation between the volumes of R_M and R_N^+ . To do so, we relate the volumes of R_N and R_N^+ , and then those of R_M and R_N .

Claim 4.2.3. $\text{vol}(R_N^+) \geq (1 - \delta)\text{vol}(R_N)$.

Proof: By construction, $R_N^+ = \Phi(R_N \setminus \mathcal{X}) \cup \bigcup_{i \in I} Y_i$, where $I = \{i \in [1, m] \mid X_i \cap R_N \neq \emptyset\}$ is the set of indices of those anomalous regions X_i intersecting R_N . Since the map Φ is a δ -diffeomorphism, we have $\text{vol}(\Phi(R_N \setminus \mathcal{X})) \geq (1 - \delta)\text{vol}(R_N \setminus \mathcal{X})$. On the other hand, by the δ -closeness between M and N , for each $i \in I$,

$$\text{vol}(Y_i) \geq (1 - \delta)\text{vol}(X_i) \geq (1 - \delta)\text{vol}(R_N \cap X_i).$$

Putting these two together, we obtain the claim. \blacksquare

Claim 4.2.4. *If $\text{vol}(R_N) \leq \frac{\text{vol}(M)}{2}$, then $\text{vol}(R_N) \geq (1 + 8\delta)\text{vol}(R_M)$.*

Proof: Let $R_M(l)$ denote $\{x \in M \mid d_M(x, R_M) \leq l\}$, that is, the region expanded from R_M by *geodesic distance* l . For $l \leq \rho\delta$, we have $R_M \subseteq R_M(l) \subseteq R_M(\rho\delta) \subseteq R_N$ as R_N

contains all points within *Euclidean distance* $\rho\delta$ to R_M . Let $\text{bndVol}(l) = \text{vol}(\partial R_M(l))$. Since $\text{vol}(R_M(l)) \leq \text{vol}(R_N) \leq \text{vol}(M)/2$, we have that $\text{bndVol}(l) \geq \mu \cdot \text{vol}(R_M(l)) \geq \mu \cdot \text{vol}(R_M)$. It then follows that

$$\begin{aligned} \text{vol}(R_N) - \text{vol}(R_M) &\geq \text{vol}(R_M(\rho\delta)) - \text{vol}(R_M) \\ &= \int_0^{\rho\delta} \text{bndVol}(l) dl \geq \int_0^{\rho\delta} \mu \cdot \text{vol}(R_M) dl = \rho\mu\delta \cdot \text{vol}(R_M). \end{aligned}$$

Since $\rho\mu \geq 8$, we have that $\text{vol}(R_N) - \text{vol}(R_M) \geq 8\delta \cdot \text{vol}(R_M)$. The claim then follows.

■

Combining the above two claims we obtain the following corollary:

Corollary 4.2.1. *If $\text{vol}(R_N) \leq \frac{\text{vol}(M)}{2}$, then we have $\text{vol}(R_N^+) \geq (1 + 4\delta)\text{vol}(R_M)$.*

Lemma 4.2.2. *For an anchor-region R_M , with R_N and R_N^+ defined as above, if $\text{vol}(R_N) \leq \frac{\text{vol}(M)}{2}$, we have $\#_P(R_M) \leq \#_Q(R_N^+)$ with probability at least $1 - e^{-\Omega(\delta^{m+2}n)}$. The big- O and big- Ω notations contain constants depending on the intrinsic property of manifold M only.*

Proof: Set $r = \text{vol}(R_M)$. Since P is a uniform random sample of M according to volume measure, $\#_P(R_M)$ is a random variable and its expected value is rn . Furthermore, by construction, $\text{vol}(R_M)$ is at least the size of a geodesic ball with radius δ centered at some anchor-node $\mathbf{a} \in \mathcal{A}$. Hence for $\delta < i(M)$, $\text{vol}(R_M) \geq C_M\delta^m$, where m is the dimension of the manifold M , and C_M is a constant depending on the intrinsic curvature of M . It then follows from Chernoff bound (the upper tail) that

$$\text{Prob}[\#_P(R_M) \geq (1 + \delta)rn] \leq e^{-rn\delta^2/4} \leq e^{-C_M\delta^{m+2}n/4}.$$

On the other hand, the expected number of points from Q falling in R_N^+ , i.e, the expected size of $\#_Q(R_N^+)$, is $n \cdot \text{vol}(R_N^+)/\text{vol}(N)$ which is at least $\frac{1+4\delta}{1+\delta}rn$ by Corollary

4.2.1 and the bound on $\text{vol}(\mathbf{N})$. Note that since $\delta \leq \frac{1}{8}$, we have that $(1 + \delta) \leq \frac{(1-\delta)(1+4\delta)}{(1+\delta)}$. Hence using Chernoff bound (the lower tail), we have that

$$\begin{aligned} \text{Prob}[\#_Q(\mathbf{R}_N^+) \leq (1 + \delta)rn] &\leq \text{Prob}[\#_Q(\mathbf{R}_N^+) \leq (1 - \delta) \cdot \frac{(1 + 4\delta)}{(1 + \delta)} \cdot rn] \\ &\leq \text{Prob}[\#_Q(\mathbf{R}_N^+) \leq (1 - \delta)\text{Exp}[\#_Q(\mathbf{R}_N^+)]] \\ &\leq e^{-\text{Exp}[\#_Q(\mathbf{R}_N^+)]\delta^2/2} = e^{-\Omega(\delta^{m+2}n)}. \end{aligned}$$

The two inequalities imply that $\text{Prob}[\#_P(\mathbf{R}_M) \leq \#_Q(\mathbf{R}_N^+)] \geq 1 - e^{-\Omega(\delta^{m+2}n)}$. \blacksquare

Case 2: $\text{vol}(\mathbf{R}_N) \geq \text{vol}(\mathbf{M})/2$.

This case is more complicated to handle than the previous case. First, the relation between $\text{vol}(\mathbf{R}_M)$ and $\text{vol}(\mathbf{R}_N^+)$ as given in Corollary 4.2.1 is no longer true. Hence instead of relating the volumes of \mathbf{R}_M and of \mathbf{R}_N^+ , we now need to relate the volumes of their complements in \mathbf{M} and \mathbf{N} respectively: $\widehat{\mathbf{R}}_M = \mathbf{M} \setminus \mathbf{R}_M$ and $\widehat{\mathbf{R}}_N^+ = \mathbf{N} \setminus \mathbf{R}_N^+$. The technical details of the proofs in this section are also more involved. In what follows, we first show the following lemma. Let $\widehat{\mathbf{R}}_N = \mathbf{M} \setminus \mathbf{R}_N$ be the complement of the intermediate region \mathbf{R}_N .

Lemma 4.2.3. *If $\text{vol}(\mathbf{R}_N) \geq \frac{\text{vol}(\mathbf{M})}{2}$, then $\text{vol}(\widehat{\mathbf{R}}_N^+) \leq \frac{\text{vol}(\widehat{\mathbf{R}}_M)}{1+2\delta}$.*

Here, the aim is to obtain a relation between the volumes of $\widehat{\mathbf{R}}_M$ and $\widehat{\mathbf{R}}_N^+$. To do so, we first relate the volumes of $\widehat{\mathbf{R}}_N$ and $\widehat{\mathbf{R}}_N^+$, and then those of $\widehat{\mathbf{R}}_M$ and $\widehat{\mathbf{R}}_N$.

Claim 4.2.5. $\text{vol}(\widehat{\mathbf{R}}_N^+) \leq (1 + \delta)\text{vol}(\widehat{\mathbf{R}}_N)$.

Proof: Set $I = \{i \in [1, m] \mid \mathbf{X}_i \cap \mathbf{R}_N \neq \emptyset\}$ to be the set of indices of anomalous regions from \mathbf{M} intersecting the anchor-region \mathbf{R}_N . Given the δ -diffeomorphism Φ , it is easy to see that

$$\text{vol}(\widehat{\mathbf{R}}_N^+ \setminus \mathcal{Y}) = \text{vol}(\Phi(\widehat{\mathbf{R}}_N \setminus \mathcal{X})) \leq (1 + \delta)\text{vol}(\widehat{\mathbf{R}}_N \setminus \mathcal{X}).$$

Furthermore, since $\text{vol}(\mathbf{R}_N) \leq \text{vol}(\mathbf{R}_N \setminus \mathcal{X}) + \bigcup_{i \in I} \text{vol}(\mathbf{X}_i)$, we have that $\text{vol}(\widehat{\mathbf{R}}_N) \geq \text{vol}(\widehat{\mathbf{R}}_N \setminus \mathcal{X}) + \bigcup_{i \notin I} \text{vol}(\mathbf{X}_i)$. It then follows that

$$\begin{aligned} \text{vol}(\widehat{\mathbf{R}}_N^+) &= \text{vol}(\widehat{\mathbf{R}}_N^+ \setminus \mathcal{Y}) + \bigcup_{i \notin I} \text{vol}(\mathbf{Y}_i) \leq (1 + \delta) \text{vol}(\widehat{\mathbf{R}}_N \setminus \mathcal{X}) + (1 + \delta) \bigcup_{i \notin I} \text{vol}(\mathbf{X}_i) \\ &\leq (1 + \delta) \text{vol}(\widehat{\mathbf{R}}_N). \end{aligned}$$

■

Claim 4.2.6. *If $\text{vol}(\mathbf{R}_N) \geq \frac{\text{vol}(\mathbf{M})}{2}$, then $\text{vol}(\widehat{\mathbf{R}}_N) \leq \frac{\text{vol}(\widehat{\mathbf{R}}_M)}{(1+4\delta)}$.*

Proof: Since $\text{vol}(\mathbf{R}_N) \geq \text{vol}(\mathbf{M})/2$, we have that $\text{vol}(\widehat{\mathbf{R}}_N) \leq \text{vol}(\mathbf{M})/2$. First, suppose $\text{vol}(\mathbf{R}_M) \leq (1 - 4\delta)\text{vol}(\mathbf{M})/2$. The claim then follows in this case as we have:

$$\text{vol}(\widehat{\mathbf{R}}_M) \geq \text{vol}(\mathbf{M}) - \frac{(1 - 4\delta)\text{vol}(\mathbf{M})}{2} \geq \frac{(1 + 4\delta)\text{vol}(\mathbf{M})}{2} \geq (1 + 4\delta)\text{vol}(\widehat{\mathbf{R}}_N).$$

Hence we now consider the remaining case where $\text{vol}(\mathbf{R}_M) \geq (1 - 4\delta) \cdot \frac{\text{vol}(\mathbf{M})}{2}$. Similar to Section 4.2.2, let $\mathbf{R}_M(l) := \{x \in \mathbf{M} \mid d_M(x, \mathbf{R}_M) \leq l\}$. Set $\widehat{\mathbf{R}}_M(l) = \mathbf{M} \setminus \mathbf{R}_M(l)$, which is the region shrunk from $\widehat{\mathbf{R}}_M$ by geodesic distance l . Since \mathbf{R}_N includes all points within $\rho\delta$ Euclidean distance to \mathbf{R}_M , we have $\mathbf{R}_M \subseteq \mathbf{R}_M(l) \subseteq \mathbf{R}_M(\rho\delta) \subseteq \mathbf{R}_N$ for any $l \leq \rho\delta$, implying that

$$\text{vol}(\widehat{\mathbf{R}}_N) \leq \text{vol}(\widehat{\mathbf{R}}_M(\rho\delta)) \leq \text{vol}(\widehat{\mathbf{R}}_M(l)) \leq \text{vol}(\widehat{\mathbf{R}}_M).$$

Let $\text{bndVol}(l)$ denote the volume of the boundary of $\mathbf{R}_M(l)$ which is also the volume of the boundary of $\widehat{\mathbf{R}}_M(l)$; that is, $\text{bndVol}(l) = \text{vol}(\partial\mathbf{R}_M(l)) = \text{vol}(\partial\widehat{\mathbf{R}}_M(l))$. If $\text{vol}(\widehat{\mathbf{R}}_M(l)) \leq \frac{\text{vol}(\mathbf{M})}{2}$, then we have

$$\text{bndVol}(l) \geq \mu \cdot \text{vol}(\widehat{\mathbf{R}}_M(l)) \geq \mu \cdot \text{vol}(\widehat{\mathbf{R}}_N). \quad (4.1)$$

If $\text{vol}(\widehat{\mathbf{R}}_{\mathbf{M}}(l)) \geq \frac{\text{vol}(\mathbf{M})}{2}$, then we have that $\text{bndVol}(l) \geq \mu \cdot \text{vol}(\mathbf{R}_{\mathbf{M}}(l)) \geq \mu \cdot \text{vol}(\mathbf{R}_{\mathbf{M}})$. Since we have assumed that $\text{vol}(\mathbf{R}_{\mathbf{M}}) \geq (1 - 4\delta) \cdot \frac{\text{vol}(\mathbf{M})}{2}$, this implies that

$$\text{bndVol}(l) \geq \mu(1 - 4\delta) \cdot \text{vol}(\mathbf{M})/2 \geq \mu(1 - 4\delta) \cdot \text{vol}(\widehat{\mathbf{R}}_{\mathbf{N}}). \quad (4.2)$$

Putting Equations (4.1) and (4.2) together, and observing that $\rho\mu \geq 8$ and $\delta \leq \frac{1}{8}$, we have

$$\begin{aligned} \text{vol}(\widehat{\mathbf{R}}_{\mathbf{M}}) - \text{vol}(\widehat{\mathbf{R}}_{\mathbf{N}}) &\geq \text{vol}(\widehat{\mathbf{R}}_{\mathbf{M}}) - \text{vol}(\widehat{\mathbf{R}}_{\mathbf{M}}(\rho\delta)) \\ &= \int_0^{\rho\delta} \text{bndVol}(l) dl \geq \rho\delta\mu(1 - 4\delta) \cdot \text{vol}(\widehat{\mathbf{R}}_{\mathbf{N}}) \geq 4\delta \cdot \text{vol}(\widehat{\mathbf{R}}_{\mathbf{N}}). \end{aligned}$$

The claim then follows. ■

Combining Claims 4.2.5 and 4.2.6, Lemma 4.2.3 follows.

Next, we make the following observation:

Observation 4.2.3. *If $\widehat{\mathbf{R}}_{\mathbf{N}}^+ \neq \emptyset$, then $\widehat{\mathbf{R}}_{\mathbf{M}}$ contains at least a geodesic ball with radius $\rho\delta$.*

Proof: Let \tilde{u} be an arbitrary point in $\widehat{\mathbf{R}}_{\mathbf{N}}^+ \subseteq \mathbf{N}$. We now identify a “pre-image” u of \tilde{u} in \mathbf{M} . If $\tilde{u} \notin \mathcal{Y}$ then simply set $u = \Phi^{-1}(\tilde{u}) \in \widehat{\mathbf{R}}_{\mathbf{N}}$ to be the pre-image of \tilde{u} under the diffeomorphism $\Phi : \mathbf{M} \setminus \mathcal{X} \rightarrow \mathbf{N} \setminus \mathcal{Y}$. Otherwise, assume that $\tilde{u} \in \mathbf{Y}_j$. Then we pick an arbitrary point from \mathbf{X}_j as u ; note, $\mathbf{X}_j \cap \mathbf{R}_{\mathbf{N}} = \emptyset$ as $\tilde{u} \notin \mathbf{R}_{\mathbf{N}}^+$. Hence in either case, the “pre-image” u falls in the region $\widehat{\mathbf{R}}_{\mathbf{N}}$. Therefore its nearest Euclidean and geodesic distance to $\mathbf{R}_{\mathbf{M}}$ is at least $\rho\delta$, implying the claim in the observation. ■

Let us define $\mathbf{R}'_{\mathbf{M}}$ and $\mathbf{R}'_{\mathbf{N}}$ as follows. Imagine that we start with the region $\mathbf{R}_{\mathbf{M}}$, and obtain an intermediate region $\mathbf{R}'_{\mathbf{M}}$ by growing $\mathbf{R}_{\mathbf{M}}$ by $(\rho - 2)\delta$ Euclidean distance, instead of by $\rho\delta$ Euclidean distance as we construct $\mathbf{R}_{\mathbf{N}} = \mathbf{R}_{\mathbf{N}}(\rho)$. That is, $\mathbf{R}'_{\mathbf{M}} := \{x \in \mathbf{M} \mid d_{\mathbb{R}^d}(x, \mathbf{R}_{\mathbf{M}}) \leq (\rho - 2)\delta\}$. We then construct $\mathbf{R}'_{\mathbf{N}}$ from $\mathbf{R}'_{\mathbf{M}}$ in the same way

as constructing R_N^+ from R_N . Let $\widehat{R'_N}$ and $\widehat{R'_M}$ denote the complement of R'_N and R'_M . Obviously, $R'_M \subseteq R_N$ and $R'_N \subseteq R_N^+$. Reverse relations hold for their complements. We first prove that, under the condition that $\widehat{R'_N} \neq \emptyset$, we have that $\text{vol}(\widehat{R'_N}) \geq C_M \delta^k / 2$.

Claim 4.2.7. *If $\widehat{R'_N} \neq \emptyset$, then $\text{vol}(\widehat{R'_N}) \geq (1 - \delta) C_M \delta^k \geq C_M \delta^k / 2$.*

Proof: Since $\widehat{R'_N} \neq \emptyset$, consider any point $\tilde{u} \in \widehat{R'_N}$. We now compute a point $u \in \widehat{R'_N}$ as follows: if $\tilde{u} \notin X$, then simply set u to be the pre-image of \tilde{u} under the diffeomorphism $\Phi : M \setminus \mathcal{X} \rightarrow N \setminus \mathcal{Y}$. Otherwise, if $\tilde{u} \in Y_i$ for some i , then chose u as an arbitrary point from X_i . By construction of R_N^+ , $X_i \cap R_N = \emptyset$, hence u must lie in $\widehat{R'_N}$.

Now let $B_u(r)$ denote region $B_u(r) := \{x \in M \mid d_{\mathbb{R}^d}(x, u) \leq r\}$. (Note, we are using Euclidean distance $d_{\mathbb{R}^d}$ instead of geodesic distance d_M here.) Next, we claim that $B_u(2\delta) \subseteq \widehat{R'_M}$. Indeed, note that R_N is obtained by enlarging R'_M by 2δ Euclidean distance. If any point $x \in B_u(2\delta)$ is in R'_M , then u will be covered by R_N when we enlarge R'_M by 2δ Euclidean distance; contradiction. Hence $B_u(2\delta) \subseteq \widehat{R'_M}$.

Let $I = \{i \mid X_i \cap R'_M \neq \emptyset\}$ denote the set of indices of anomalous regions from M that intersects R'_M . Since each anomalous region is enclosed within a Euclidean ball of radius δ , and since $B_u(2\delta) \cap R'_M = \emptyset$, we have that the smaller region $B_u(\delta)$ satisfies that $B_u(\delta) \cap \bigcup_{i \in I} X_i = \emptyset$. Set B' to be $B_u(2\delta) \setminus \bigcup_{i \in I} X_i$. It then follows that $B_u(\delta) \subseteq B'$, and thus $\text{vol}(B') \geq \text{vol}(B_u(\delta)) \geq C_M \delta^k$.

On the other hand, compute the ‘‘image’’ of B' in N , denoted by $\tilde{B} \subseteq N$ as follows: Let $J = \{j \mid X_j \cap B' \neq \emptyset\}$, and set $\tilde{B} = \Phi(B' \setminus X) \cup \bigcup_{j \in J} Y_j$. Note $J \cap I = \emptyset$ since $B' = B_u(2\delta) \setminus \bigcup_{i \in I} X_i$. This means that $\tilde{B} \cap R'_N = \emptyset$, that is, $\tilde{B} \subseteq \widehat{R'_N}$ and thus $\text{vol}(\widehat{R'_N}) \geq \text{vol}(\tilde{B})$. By the δ -closeness between M and N , we have that $\text{vol}(\tilde{B}) \geq (1 - \delta) C_M \delta^k$. The claim then follows. \blacksquare

Lemma 4.2.4. *For an anchor-region R_M , with R_N and R_N^+ defined as above, if $\text{vol}(R_N) \geq \frac{\text{vol}(N)}{2}$, then $\#_P(R_M) \leq \#_Q(R_N^+)$ with probability at least $1 - e^{-\Omega(\delta^{m+2}n)}$. The big-O and big- Ω notations contain constants that depend on M only.*

Proof: First, if $\widehat{R_N^+} = \emptyset$, then the claim holds as $\#_Q(R_N^+) = n$. So from now on, we assume that $\widehat{R_N^+} \neq \emptyset$. In this case, it follows from Observation 4.2.3 that $\widehat{R_M}$ contains at least one geodesic ball with radius $\rho\delta$. Since $\rho \geq 1$, $\text{vol}(\widehat{R_M}) \geq C_M\delta^m$, for δ smaller than the injectivity radius, where C_M is a constant that depends on intrinsic curvature of the manifold M .

We now show that $\#_P(\widehat{R_M}) \geq \#_Q(\widehat{R_N^+})$ with high probability, which will imply the lemma. Since P is a uniform random sample of M with respect to the volume measure, we have that $\text{Exp}[\#_P(\widehat{R_M})] = \text{vol}(\widehat{R_M}) \cdot n$. By Chernoff bound (the lower tail), we have

$$\text{Prob}[\#_P(\widehat{R_M}) \leq (1 - \frac{\delta}{4})\text{vol}(\widehat{R_M})n] \leq e^{-\text{vol}(\widehat{R_M})n\delta^2/32} \leq e^{-C_M\delta^{m+2}n/32}. \quad (4.3)$$

Next, we aim to bound $m = \#_Q(\widehat{R_N^+})$. Since Q is a uniform random sample of N , $\text{Exp}[m] = \frac{\text{vol}(\widehat{R_N^+}) \cdot n}{\text{vol}(N)} \geq \frac{\text{vol}(\widehat{R_N^+}) \cdot n}{1+\delta}$. We first assume that $\text{vol}(\widehat{R_N^+}) \geq C_M\delta^m$. In this case, $\text{Exp}[m] \geq C_M\delta^k n / (1 + \delta) \geq C_M\delta^k n / 2$. Chernoff bound (the upper tail) provides:

$$\begin{aligned} \text{Prob}[m \geq (1 + \frac{\delta}{4})\text{vol}(\widehat{R_N^+})n] &\leq \text{Prob}[m \geq (1 + \frac{\delta}{4})(1 + 2\delta)\text{vol}(\widehat{R_N^+})n] \\ &\leq \text{Prob}[m \geq (1 + \frac{\delta}{4})(1 + 2\delta)(1 - \delta)\text{Exp}[m]] \\ &\leq \text{Prob}[m \geq (1 + \frac{\delta}{4})\text{Exp}[m]] \\ &\leq e^{-\delta^2 \text{Exp}[m]/64} \leq e^{-\Omega(\delta^{m+2}n)}. \end{aligned} \quad (4.4)$$

Combining Equation (4.4) with Equation (4.3), we have that $\#_Q(\widehat{R_N^+}) \leq \#_P(\widehat{R_M})$, thus $\#_P(R_M) \leq \#_Q(R_N^+)$, with probability at least $1 - e^{-\Omega(\delta^{k+2}n)}$, when $\text{vol}(\widehat{R_N^+}) \geq C_M\delta^m$.

What remains is to prove our lemma when $\text{vol}(\widehat{\mathbf{R}}_{\mathbf{N}}^+) \leq C_{\mathbf{M}}\delta^m$. In this case, we can no longer directly apply previous argument, because we cannot lower-bound $\text{Exp}[m]$, thus upper-bound the probability in Equation (4.4) any more. It turns out that Lemma 4.2.2 and 4.2.3 holds when replacing $\mathbf{R}_{\mathbf{N}}$ and $\mathbf{R}_{\mathbf{N}}^+$ with $\mathbf{R}'_{\mathbf{M}}$ and $\mathbf{R}'_{\mathbf{N}}$. (Indeed, in all the proofs, we only require that we grow $\mathbf{R}_{\mathbf{M}}$ by radius $(\rho - 2)\delta$.)

Now if $\text{vol}(\mathbf{R}'_{\mathbf{M}}) \leq \frac{\text{vol}(\mathbf{M})}{2}$, then by Lemma 4.2.2 we have that with probability at least $1 - e^{-\Omega(\delta^{m+2}n)}$ we have that $\#_P(\mathbf{R}_{\mathbf{M}}) \leq \#_Q(\mathbf{R}'_{\mathbf{N}})$. Since $\mathbf{R}'_{\mathbf{N}} \subseteq \mathbf{R}_{\mathbf{N}}^+$, we have $\#_P(\mathbf{R}_{\mathbf{M}}) \leq \#_Q(\mathbf{R}_{\mathbf{N}}^+)$ which proves our lemma.

Finally, consider the last case when $\text{vol}(\mathbf{R}'_{\mathbf{M}}) \geq \frac{\text{vol}(\mathbf{M})}{2}$. Set $s = \#_Q(\widehat{\mathbf{R}}_{\mathbf{N}})$ and we have, from Claim 4.2.7 and the bound on $\text{vol}(\mathbf{N})$ that

$$\frac{\text{vol}(\widehat{\mathbf{R}}'_{\mathbf{N}}) \cdot n}{1 - \delta} \geq \text{Exp}[s] = \frac{\text{vol}(\widehat{\mathbf{R}}'_{\mathbf{N}}) \cdot n}{\text{vol}(\mathbf{N})} \geq \frac{\text{vol}(\widehat{\mathbf{R}}'_{\mathbf{N}}) \cdot n}{1 + \delta} \geq \frac{1 - \delta}{1 + \delta} \cdot C_{\mathbf{M}}\delta^m \cdot n \geq C_{\mathbf{M}}\delta^m \cdot n/2.$$

On the other hand, by Lemma 4.2.3, $\text{vol}(\widehat{\mathbf{R}}_{\mathbf{M}}) \geq (1 + 2\delta)\text{vol}(\widehat{\mathbf{R}}'_{\mathbf{N}})$. Hence, similar to Equation (4.4), we have that:

$$\begin{aligned} \text{Prob}[s \geq (1 - \frac{\delta}{4})\text{vol}(\widehat{\mathbf{R}}_{\mathbf{M}})n] &\leq \text{Prob}[s \geq (1 - \frac{\delta}{4})(1 + 2\delta)\text{vol}(\widehat{\mathbf{R}}'_{\mathbf{N}})n] \\ &\leq \text{Prob}[s \geq (1 - \frac{\delta}{4})(1 + 2\delta)(1 - \delta)\text{Exp}[s]] \\ &\leq \text{Prob}[s \geq (1 + \frac{\delta}{4})\text{Exp}[s]] \\ &\leq e^{-\delta^2 \text{Exp}[s]/64} \leq e^{-\Omega(\delta^{m+2}n)}. \end{aligned} \tag{4.5}$$

Since $\mathbf{R}'_{\mathbf{N}} \subseteq \mathbf{R}_{\mathbf{N}}^+$, we have $m = \#_Q(\widehat{\mathbf{R}}_{\mathbf{N}}^+) \leq s$. It then follows that $\text{Prob}[m \geq (1 - \frac{\delta}{4})\text{vol}(\widehat{\mathbf{R}}_{\mathbf{M}})n] \leq \text{Prob}[s \geq (1 - \frac{\delta}{4})\text{vol}(\widehat{\mathbf{R}}_{\mathbf{M}})n] \leq e^{-\Omega(\delta^{m+2}n)}$. Combining this with Equation (4.3), we have that when $\widehat{\mathbf{R}}_{\mathbf{N}}^+ \neq \emptyset$, $\#_Q(\widehat{\mathbf{R}}_{\mathbf{N}}^+) \leq \#_P(\widehat{\mathbf{R}}_{\mathbf{M}})$, that is, $\#_P(\mathbf{R}_{\mathbf{M}}) \leq \#_Q(\mathbf{R}_{\mathbf{N}}^+)$, with probability at least $1 - e^{-\Omega(\delta^{m+2}n)}$. \blacksquare

Intuitively, by using a smaller expansion radius $(\rho - 2)\delta$, we show that the volume of \widehat{R}_N is at least (roughly) $C_M \delta^m$, so that we can continue to bound the size of s using Chernoff bound (the upper tail) as in Equation (4.4) ⁴.

Proof of Lemma 4.2.1. By Observation 4.2.1, there are $2^{\Theta(1/\delta^m)}$ number of anchor-regions in M and in N . By Lemma 4.2.2 and 4.2.4, for each anchor-region $R_M \subset M$ and its witness anchor-region $R_N^+ \subset N$, with probability at least $1 - e^{-\Omega(\delta^{m+2n})}$, $\#_P(R_M) \leq \#_Q(R_N^+)$. It then follows from union bound that $\#_P(R_M) \leq \#_Q(R_N^+)$ for all anchor-region R_M simultaneously with probability at least

$$1 - e^{-\Omega(\delta^{m+2n})} \cdot 2^{\Theta(\frac{1}{\delta^m})} \geq 1 - e^{-\Omega(\delta^{m+2n}) + \Theta(\frac{1}{\delta^m})} = 1 - O\left(\frac{1}{n^4}\right)$$

for sufficiently large n , say $n = \Omega(\frac{1}{\delta^{4m}})$, thus proves Lemma 4.2.1.

4.3 Step 2: Bounding Spectra Distance

We now assume that we are given two sets of n points $P = \{p_1, \dots, p_n\}$ and $Q = \{q_1, \dots, q_n\}$ sampled from hidden m -manifolds M and N , respectively, such that $d_{\mathbb{R}^d}(p_i, q_i) = \|p_i - q_i\| \leq (\rho + 3)\delta = O(\delta)$ for any $i \in [1, n]$, where $\rho = \max\{3, \frac{8}{\mu} + 2\}$ is a constant depending on the isoperimetric constant μ of the manifold M . Notice that Theorem 4.2.1 implies that we have such inputs with high probability when P and Q are uniformly randomly sampled from M and N according to their volume measures. Now consider the weighted graph Laplacians L_P^t and L_Q^t constructed from P and Q , respectively. Our goal is to show that the spectra of these two graph Laplacians are close. We achieve this by showing that the matrix 2-norm $\|L_P^t - L_Q^t\|$ of the matrix

⁴We remark that all our earlier results hold if we choose $(\rho - 2)\delta$ as the expansion radius, instead of using $\rho\delta$, when constructing the intermediate region R_N from R_M . The reason we use current choice of ρ is precisely to handle this case

$\mathbf{L}_P^t - \mathbf{L}_Q^t$ is bounded, which further bounds the spectra distance of \mathbf{L}_P^t and \mathbf{L}_Q^t by Weyl's Theorem for eigenvalue perturbations [64].

For simplicity, set $G_{ij} := \frac{1}{(4\pi)^{\frac{k}{2}t^{\frac{k}{2}+1}}} \cdot e^{-\frac{\|p_i - p_j\|^2}{4t}}$ and $\tilde{G}_{ij} := \frac{1}{(4\pi)^{\frac{k}{2}t^{\frac{k}{2}+1}}} \cdot e^{-\frac{\|q_i - q_j\|^2}{4t}}$ for any $i, j \in [1, n]$. Notice that $\mathbf{L}_P^t[i][j] = -\frac{1}{n}G_{ij}$ and $\mathbf{L}_Q^t[i][j] = -\frac{1}{n}\tilde{G}_{ij}$ for $i \neq j$.

First, we need the following key result. This lemma bounds $|G_{ij} - \tilde{G}_{ij}|$, which is then used to bound $\|\mathbf{L}_P^t - \mathbf{L}_Q^t\|$.

Lemma 4.3.1. $|G_{ij} - \tilde{G}_{ij}| = O(\frac{\delta^{4/5}}{t^{m/2+7/5}})$ if $t = \Omega(\delta^{2-\varepsilon})$ for any positive real $\varepsilon > 0$. In particular, $|G_{ij} - \tilde{G}_{ij}| = O(\delta^{1/3})$ for $t \geq \delta^{\frac{19}{14}m+3}$. The big- O and big- Ω notations hide constants depending on the isoperimetric constant μ of the manifold \mathbf{M} .

Proof: For simplicity, denote $C_\mu := \rho + 3$. By the one-to-one closeness between points in P and in Q , we have that $d_{\mathbb{R}^d}(p_i, q_i) = \|p_i - q_i\| \leq C_\mu \delta$ for constant $C_\mu \geq 6$. For every $i, j \in [1, n]$, the triangle inequality imply that $\|p_i - p_j\| - 2C_\mu \delta \leq \|q_i - q_j\| \leq \|p_i - p_j\| + 2C_\mu \delta$.

We distinguish two cases: $\|p_i - p_j\| \leq \tau$ and $\|p_i - p_j\| > \tau$ for some parameter $\tau \geq 4C_\mu \delta$ whose value is to be specified later.

Case 1: $\|p_i - p_j\| \leq \tau$.

Since $\tau \geq 4C_\mu \delta$, we have that $\delta\tau = \Omega(\delta^2)$. We can then bound \tilde{G}_{ij} from below as follows as long as $\frac{C_\mu^2 \delta^2}{t} < 1$ and $\frac{C_\mu \delta \tau}{t} < 1$:

$$\begin{aligned} \tilde{G}_{ij} &= \frac{1}{(4\pi)^{\frac{k}{2}t^{\frac{k}{2}+1}}} \cdot e^{-\frac{(\|q_i - q_j\|)^2}{4t}} \geq \frac{1}{(4\pi)^{\frac{k}{2}t^{\frac{k}{2}+1}}} \cdot e^{-\frac{(\|p_i - p_j\| + 2C_\mu \delta)^2}{4t}} \\ &\geq G_{ij} \cdot e^{-\frac{C_\mu^2 \delta^2}{t}} \cdot e^{-\frac{C_\mu \delta \|p_i - p_j\|}{t}} \geq G_{ij} \cdot e^{-\frac{C_\mu^2 \delta^2}{t}} \cdot e^{-\frac{C_\mu \delta \tau}{t}} \\ &\geq G_{ij} \left(1 - \frac{O(\delta^2)}{t}\right) \left(1 - \frac{O(\delta\tau)}{t}\right) \geq G_{ij} \cdot \left(1 - \frac{O(\delta\tau)}{t}\right). \end{aligned}$$

Now we bound \tilde{G}_{ij} from above. First, assume that $\|p_i - p_j\| \leq 2C_\mu\delta$. Since $e^{-\frac{\|q_i - q_j\|^2}{4t}} \leq 1$, we have $\tilde{G}_{ij} \leq \frac{1}{(4\pi)^{\frac{k}{2}} t^{\frac{k}{2}+1}}$. It then follows that, for $\frac{C_\mu^2\delta^2}{t} < 1$, we have

$$\begin{aligned}\tilde{G}_{ij} &\leq \frac{1}{(4\pi)^{\frac{k}{2}} t^{\frac{k}{2}+1}} = G_{ij} / e^{\frac{-(\|p_i - p_j\|)^2}{4t}} = G_{ij} \cdot e^{\frac{(\|p_i - p_j\|)^2}{4t}} \leq G_{ij} \cdot e^{\frac{C_\mu^2\delta^2}{t}} \\ &\leq G_{ij} \cdot (1 + O(\frac{\delta^2}{t})).\end{aligned}$$

Otherwise, $\tau \geq \|p_i - p_j\| \geq 2C_\mu\delta$. In this case we have that $\|q_i - q_j\|^2 \geq (\|p_i - p_j\| - 2C_\mu\delta)^2$. Hence for $\frac{C_\mu^2\delta\tau}{t} < 1$,

$$\begin{aligned}\tilde{G}_{ij} &\leq \frac{1}{(4\pi)^{\frac{k}{2}} t^{\frac{k}{2}+1}} \cdot e^{\frac{-(\|p_i - p_j\| - 2C_\mu\delta)^2}{4t}} \leq G_{ij} \cdot e^{\frac{-C_\mu^2\delta^2}{t}} \cdot e^{\frac{C_\mu\delta\|p_i - p_j\|}{t}} \leq G_{ij} \cdot e^{\frac{C_\mu\delta\|p_i - p_j\|}{t}} \\ &\leq G_{ij} \cdot (1 + O(\frac{\delta\tau}{t})).\end{aligned}$$

Putting the above inequalities together, and using the fact that $G_{ij} = O(\frac{1}{t^{m/2+1}})$, we have

$$E_{\leq} := |G_{ij} - \tilde{G}_{ij}| = G_{ij} \cdot O(\frac{\delta\tau}{t}) = O(\delta\tau/t^{\frac{m+4}{2}}) \text{ when } \|p_i - p_j\| \leq \tau.$$

Case 2: $\|p_i - p_j\| > \tau$.

Recall that $\|q_i - q_j\| \geq \|p_i - p_j\| - 2C_\mu\delta$. If $\|p_i - p_j\| > \tau$, and $\tau \geq 4C_\mu\delta$, then $\|q_i - q_j\| > \tau - \tau/2 > \tau/2$. It then follows

$$\tilde{G}_{ij} = \frac{1}{(4\pi)^{\frac{k}{2}} t^{\frac{k}{2}+1}} \cdot e^{\frac{-\|q_i - q_j\|^2}{4t}} \leq \frac{1}{(4\pi)^{\frac{k}{2}} t^{\frac{k}{2}+1}} \cdot e^{\frac{-\tau^2}{16t}}.$$

Since $e^{-\frac{1}{x}} \leq x^2$ for any $x > 0$, we have

$$E_{>} := |\tilde{G}_{ij} - G_{ij}| \leq \tilde{G}_{ij} \leq \frac{1}{(4\pi)^{\frac{k}{2}} t^{\frac{m}{2}+1}} \cdot O((\frac{t}{\tau^2})^2) = O(\frac{1}{\tau^4 t^{\frac{m}{2}-1}})$$

when $\|p_i - p_j\| > \tau$.

We balance the two error terms E_{\leq} and $E_{>}$ by choosing $\tau = \frac{t^{3/5}}{\delta^{1/5}}$ so that $E_{\leq} = E_{>} = \frac{\delta^{4/5}}{t^{m/2+7/5}}$. The condition $\frac{C_{\mu}^2 \delta^2}{t} < 1$, $\frac{C_{\mu}^2 \delta \tau}{t} < 1$, and $\tau \geq 4C_{\mu} \delta$ can be satisfied as long as $t = \Omega(\delta^{2-\varepsilon})$ for any $\varepsilon > 0$. Finally, if $t > \delta^{\frac{1}{14}k+3}$, we have that $E_{\leq} = E_{>} = O(\delta^{1/3})$.

The lemma follows. \blacksquare

Given a matrix (operator) D , let $\lambda_i(D)$ denote its i -th smallest eigenvalue. We have the following result.

Theorem 4.3.1. *Let $P = \{p_1, \dots, p_n\}$ and $Q = \{q_1, \dots, q_n\}$ be two sets of n points such that $\|p_i - q_i\| = O(\delta)$ for every $i \in [1, n]$. Let L_P^t and L_Q^t be the corresponding Gaussian-weighted graph Laplacians computed from P and Q , respectively. The eigenvalues of L_P^t and L_Q^t satisfy $|\lambda_i(\mathsf{L}_P^t) - \lambda_i(\mathsf{L}_Q^t)| = O(\mathcal{E}(\delta, t))$, where $\mathcal{E}(\delta, t) = \frac{\delta^{4/5}}{t^{m/2+7/5}}$. In particular, $\mathcal{E}(\delta, t) = O(\delta^{1/3})$ if $t \geq \delta^{\frac{1}{14}m+3}$.*

Proof: First, notice that the Gaussian-weighted graph Laplace matrix is symmetric. Set matrix D to be the difference between L_P^t and L_Q^t ; that is $D[i][j] = \mathsf{L}_P^t[i][j] - \mathsf{L}_Q^t[i][j]$ for all $i, j \in [1, n]$. Consider the definition of $\mathsf{L}_P^t[i][j]$ in Equation (2.2). By Lemma 4.3.1,

$$|D[i][j]| = \frac{1}{n} \cdot |G_{ij} - \tilde{G}_{ij}| = O(\mathcal{E}(\delta, t)/n), \quad \text{for each } i \neq j.$$

For diagonal entries, we have

$$|D[i][i]| \leq \sum_{j=1}^n \frac{1}{n} \cdot |G_{ij} - \tilde{G}_{ij}| = O(\mathcal{E}(\delta, t)), \quad i \in [1, n].$$

Hence the matrix 1-norm of D satisfies

$$\|D\|_1 = \max_{j=1}^n \sum_{i=1}^n |D[i][j]| = O(\mathcal{E}(\delta, t)),$$

and the matrix ∞ -norm of D satisfies

$$\|D\|_\infty = \max_{i=1}^n \sum_{j=1}^n |D[i][j]| = O(\mathcal{E}(\delta, t)).$$

Since the matrix 2-norm $\|D\|_2$ satisfies $\|D\|_2 \leq \sqrt{\|D\|_1 \|D\|_\infty}$, we have that $\|D\|_2 = O(\mathcal{E}(\delta, t))$. By Weyl's theorem for eigenvalue perturbation of Hermitian matrices (see e.g., [64]), the distance between corresponding eigenvalues of the matrix L_Q^t and $L_P^t = L_Q^t + D$ is bounded by the matrix 2-norm of the difference matrix D , which is $O(\mathcal{E}(\delta, t))$. ■

Our main result, Theorem 4.1.1 stated earlier, then follows from Theorems 4.2.1 and 4.3.1.

4.4 Experiments

In this section, we show through experiments that the discrete (weighted-graph) Laplace operator is indeed stable against small topological changes. We also show that the weighted-graph Laplace operator changes smoothly with the size of the region of topological change for a fixed t . In our experiments, we use the first 300 eigenvalues.

First, we show the effect of a small topological change on the eigenvalue, and how this effect changes with the size of the region of change. Figure 4.2 shows two tori connected using a very thin bridge, and then increasing the size of the connecting bridge. The graph depicts the top 300 eigenvalues at different stages of this process for $t^2 = 0.00001$. We can see that for smaller changes, the eigenvalues are similar, since the region of change is small when compared with our choice of t . However, for a larger region of change, the eigenvalues deviate more from the original surface.

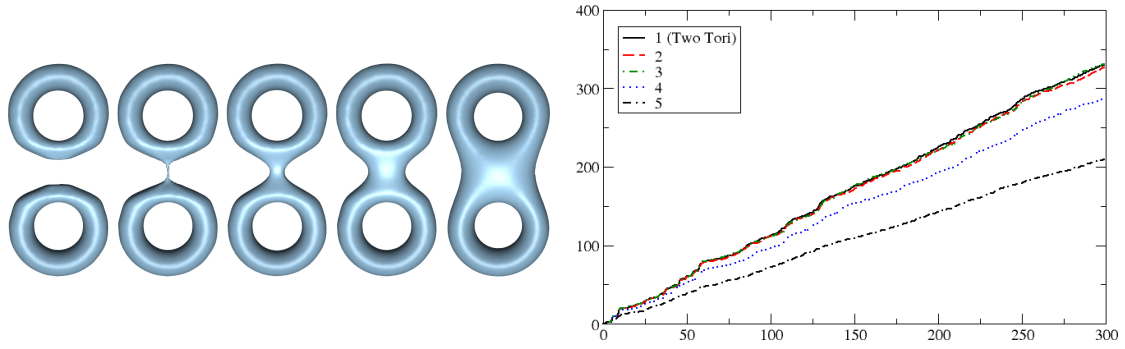


Figure 4.2: Left: Connecting two tori with increasingly larger bridges. Right: Comparison of eigenvalues

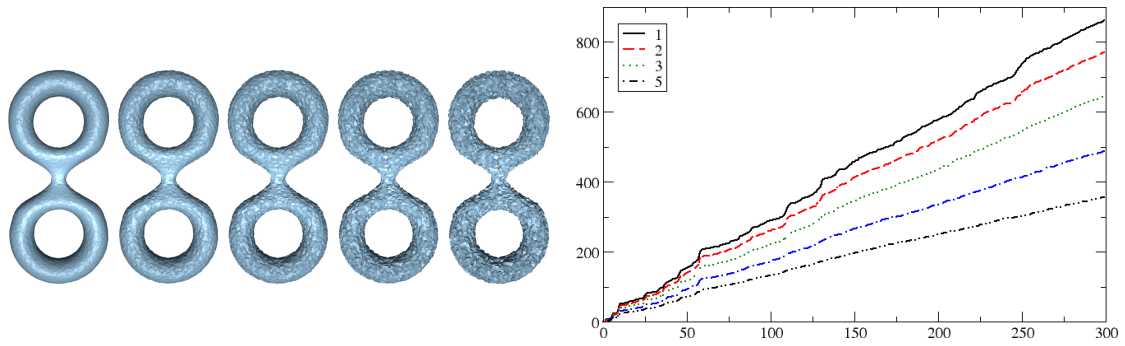


Figure 4.3: Left: Increasingly large perturbation of points on a surface. Right: Comparison of eigenvalues

A similar trend is also seen when non-topological noise is added to a surface. For example, Figure 4.3 shows a surface and with increasing perturbation of points. The eigenvalues change smoothly for small t as the perturbations become larger.

Figure 4.4 shows another example of a torus with another small torus attached to it, incrementing its genus, and hence changing its topology. Then, we gradually increase the size of this attached torus until it becomes as large as the original torus.

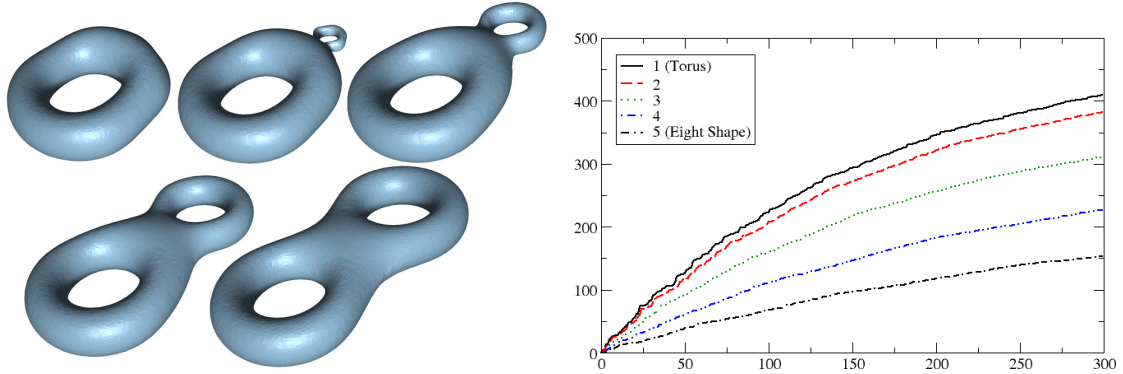


Figure 4.4: Left: Increasingly large region of topological change on a torus. Right: Comparison of eigenvalues

Here, we fix our t to be much larger ($t^2 = 0.001$). Note that the eigenvalues still differ a lot since the region of change is large in all the cases.

Finally, figure 4.5 shows the effect of multiple topological changes for the armadillo model. We fix our t to be $t^2 = 0.0001$. Here, the eigenvalues are similar despite multiple topological changes in different regions of the model. For a larger region of change (top-right model), however, the change in eigenvalues is noticeable.

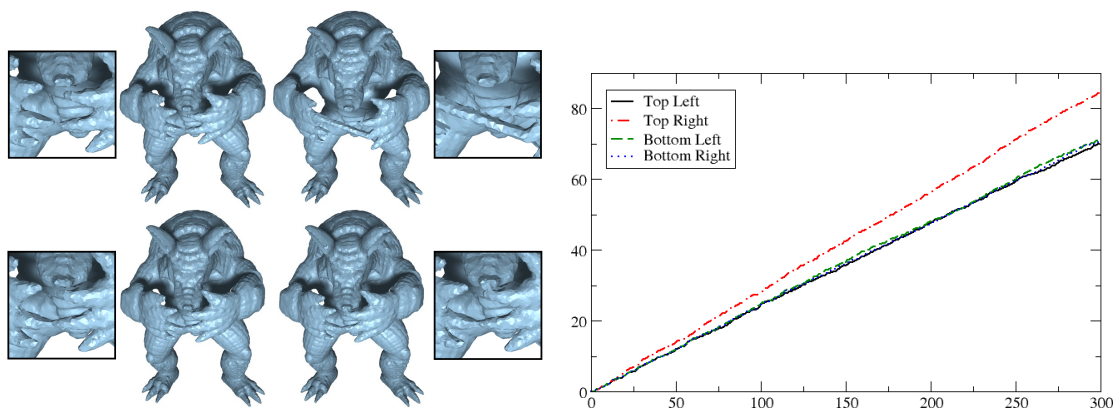


Figure 4.5: Topological changes on the Armadillo. Top Row: Original Armadillo model; and a variation with two fingers joined. Bottom Row: Another variation with two fingers touching; and a model with two fingers touching and another finger touching the nose. Right: Comparison of eigenvalues for $t^2 = 0.0001$

Chapter 5: Persistent Heat Signature for Pose-oblivious Matching of Incomplete Models [24]

With this chapter, we shift our focus from theoretical studies to building applications that are able to exploit our theoretical results. The first of two such applications that we will discuss lies in the field of shape matching.

The need for effective and efficient shape retrieval algorithms is ubiquitous in a broad range of applications in science and engineering. With the increasing understanding of shape geometry and topology in the context of shape similarity, workable solutions for shape retrieval are being produced. Numerous work drawing upon insightful ideas such as [16, 34, 35, 45, 67, 68] have made this possible. See also a comprehensive comparison study [82] and a survey [89]. The performances of some shape retrieval algorithms can also be seen at the *Shape retrieval contest* website [1].

The problem of shape matching and retrieval is not trivial even if only rigid body transformations are allowed to vary the shapes [49, 69]. Further difficulty ensues if shape variations include small deformations [8, 61]. Even more realistic assumption should allow the shapes to vary in pose meaning that the intrinsic metric is not distorted much though the three dimensional embedding varies widely. Spectral methods have shown remarkable resilience to shape variations caused by the extrinsic metric while remaining stable under small changes in the intrinsic metric [52, 78, 74].

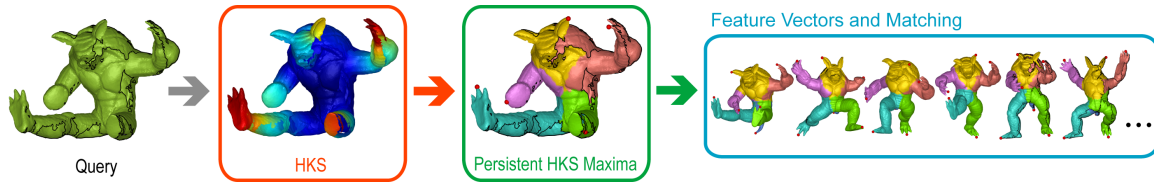


Figure 5.1: Given a query Armadillo model that is pose-altered, incomplete, and partially scanned, our method first computes the heat kernel signature function at a certain scale, and then extract a set of HKS maxima (red dots) using persistent homology. Feature vectors computed at these maxima are then used to search for the most similar models, be it complete, partial, or incomplete, in a shape database. A few top matches are shown. The black curves are the boundary curves of either partial or incomplete models. Correspondence between segmentations of different models is shown with consistent coloring.

As a result, researchers have started to study this approach more in depth [45, 68, 71, 88]. In this chapter we explore this approach to address one of the main difficulties still existing for shape retrieval systems— *pose-oblivious* matching of *partial* and *incomplete* 3D models.

The matching of a partial or an incomplete model against a complete one cannot rely on features that are too global. At the same time, any matching relying on only local measures becomes susceptible to noise caused by small perturbations. Therefore, we need something in between which can describe shape features at different scales. The Heat Kernel Signature (*HKS*) recently introduced in [88] bears this multi-scale property. However, it has not yet been demonstrated how this signature can be used effectively for partial or incomplete shape retrievals from a database that may itself contain partial or incomplete models. We provide such a scheme which is both robust and scalable for large data sets.

Our method is based on a novel synergy between *HKS* and persistent homology. Given a surface M with an initial distribution of unit heat concentrated at any point $x \in M$, the *HKS* at x at time (i.e, duration of the flow) t provides the amount of heat retained at x after heat dissipates for time t according to the well known heat equation. This heat dissipation is determined by the intrinsic geometry of M and the influence of shape features on the heat flow can be controlled by regulating the time. Small time scales dissipate heat over a small region and thereby allow local features to regulate the heat whereas large time scales allow global features to exert more influence. This suggests that one can use *HKS* at different time values to describe features at multiple scales. Indeed, Ovsjanikov et al. use this as a shape descriptor for every vertex of an input surface, and quantize the space of shape descriptors by k -means clustering to obtain a succinct set of representative features [68]. They represent an input shape based on the distribution of its shape descriptors with respect to these representative features, and develop an efficient shape retrieval system.

The distribution-based features tend to be less discriminating for partial matching. The *HKS* function is more likely to remain similar on a surface and its partial counterpart when t is relatively small. However, a small diffusion interval tends to increase the local variation in the *HKS* function values, and makes it more sensitive to noise. To counter this, we bring in the tool of persistent homology [31] to identify important features.

We argue that the critical points, in particular, the maxima of *HKS*, serve as good candidates for feature points. We consider only *persistent features*, which are *HKS* maxima that persist beyond a given threshold. To compute such maxima, instead of using the standard persistent algorithm, we employ a region merging algorithm

that bypasses detecting persistence values of all critical points and focuses only on eliminating those maxima that are not persistent. As a byproduct we also obtain a segmentation of the surface that appears to be robust, and consistent between a shape and its partial versions even if they are in different poses (see the last box in Figure 5.1). This may be of independent interest.

The persistent feature points, together with a multi-scale *HKS* description at each points, provide a concise yet discriminating shape descriptor for the input surface, which is also robust under near-isometric deformations and partial occlusions. Indeed, experimental results show that our algorithm is able to match partial, incomplete, and pose-altered query shapes in a database of moderate size efficiently and with a high success rate. In our paper [24], for large input data, we also develop an efficient algorithm to approximate *HKS* values to make our algorithm scalable.

We remark that using point signatures for matching is not new, and there has been much previous work proposing and investigating various point signatures; see e.g [18, 37, 81]. However, our method relies on a novel synergy between the multi-scale *HKS* and the topological persistence algorithm. Our algorithm is the first software designed specifically for matching partial, incomplete, as well as pose-modified shapes in a database. Experimental results show that our method is robust, very efficient, and quite effective in partial shape retrieval.

5.1 Heat Kernel Signature

Our matching algorithm relies on the *Heat Kernel Signature* (*HKS*) proposed and analyzed in [88] (a scale-invariant version was used in [36]). This shape descriptor is derived from the heat operator which we briefly describe first.

5.1.1 Heat Kernel Signature

The heat kernel function has many nice properties; see [88] for a detailed discussion. The family of heat kernel functions uniquely defines the underlying manifold up to an isometry; thus it is very informative and a potentially good tool to construct a shape descriptor.

Specifically, Sun et al. propose the following Heat Kernel Signature (*HKS*) as a shape descriptor for a manifold M [88].

$$HKS_t(x) = \mathbf{h}_t(x, x) = \sum_{i \geq 0} e^{-t\lambda_i} \phi_i(x)^2. \quad (5.1)$$

Where \mathbf{h}_t is the kernel of Heat operator, which was briefly discussed in Section 2.3. Intuitively, it measures how much heat is left at time t for the point $x \in M$ if unit amount of heat is placed at point x when $t = 0$. *HKS* inherits many nice properties of the heat kernel. It is invariant to isometric deformations, and not sensitive to noise or even slight topological changes. It is multi-scale: as t gets larger, the eigenfunctions corresponding to large eigenvalues play a smaller role, hence only the main features of the shape detected by small eigenvalues matter. Most importantly, *HKS* is almost as informative as the family of functions $\mathbf{h}_t(\cdot, \cdot)_{t > 0}$ (see Theorem 1 in [88]). At the same time, it reduces the family of two-variables kernel functions to a family of single-variable functions, and is hence more succinct and much easier computationally.

***HKS* Maxima.** It is well-known [40] that as $t \rightarrow 0$, there is an asymptotic expansion of the *HKS* function at every point $x \in M$ of the form:

$$HKS_t(x) = \mathbf{h}_t(x, x) = (4\pi t)^{-d/2} \sum_{i \geq 0} a_i t^i,$$

where $a_0 = 1$ and $a_1 = \frac{1}{6}s(x)$ with $s(x)$ being the scalar curvature at point x . For a 2-manifold M , $s(x)$ is simply the Gaussian curvature at x . Thus intuitively, the heat

diffusion for small t is governed by intrinsic curvature. Heat tends to diffuse slower at points with positive curvature and faster with negative curvature. This suggests that critical points of HKS correspond to features of the shape, where maxima of HKS capture the tips of protrusions or the bottoms of concave areas. Hence, we propose to use HKS maxima as feature points for shape matching. It turns out that we can select only a handful of *persistent* feature points for matching.

5.1.2 Discrete setting

In the discrete setting, the input is a triangular mesh K whose underlying space is homeomorphic to a smooth surface M . To compute the HKS , one needs to compute the eigenvectors and eigenvalues of the Laplace-Beltrami (thus heat) operator. As observed in Chapter 1, several discrete Laplace operators for meshes have been developed in the literature. The most popular ones are the cotangent-scheme [28, 23, 70, 96] and the finite element method based approach [72, 74]. We use the mesh-Laplacian proposed in [7] for two reasons. First, the construction of this mesh-Laplacian is based on the heat diffusion idea, making it consistent and perhaps natural for computing Heat Kernel Signatures. Second, the mesh-Laplacian is the only current discrete Laplace operator so that both the operator itself and its spectrum converge to those of the manifold Laplacian with increasing mesh density while requiring no constraints on triangle qualities, as seen in Chapter 5.

5.2 Persistent Heat Maxima

Our goal is to compute a signature of a shape from its HKS functions. To make the size of the signature concise, we want to select a subset of vertices of the mesh that may find correspondences in HKS values in a shape which we want it to

match. We may take the critical points, in particular, the maxima of the HKS as this subset. This subset itself could possibly be large. Furthermore, discretization errors and small variations in the shapes may cause many maxima to lose their correspondences in a similar shape. Therefore, we need a mechanism to select a small subset of these maxima that hopefully remain stable under small perturbations but allow pose variations.

One might think that the HKS at a large time scale t provides a few such stable maxima since such a HKS function describes the input shape at a large scale. However, the criticality of a point p is decided not only by the function value at p , but also by its relative value compared to its neighbors. Some unimportant local maxima at small t may survive for large t thereby invalidating t alone as a discriminating factor in filtering maxima. Furthermore, computing HKS at a large t value makes it more sensitive to partial and incomplete shapes. Hence we take the strategy that, first, we compute the HKS function for a moderately small t so that partial similarity tends to be well preserved. We next use the concept of persistent homology [31] to select a subset of the maxima that have large persistence. Experimental results suggest that this strategy produces robust feature points across partial and incomplete shapes.

5.2.1 Persistence

We now briefly review the concept of persistent homology. We refer the readers to the original persistence paper [31] and the survey [30] for detailed discussions. Let K be the triangulation whose underlying space, denoted $|K|$, is a surface possibly with boundaries. Suppose we have a HKS function $h: V \rightarrow \mathbb{R}$ defined on vertices V of

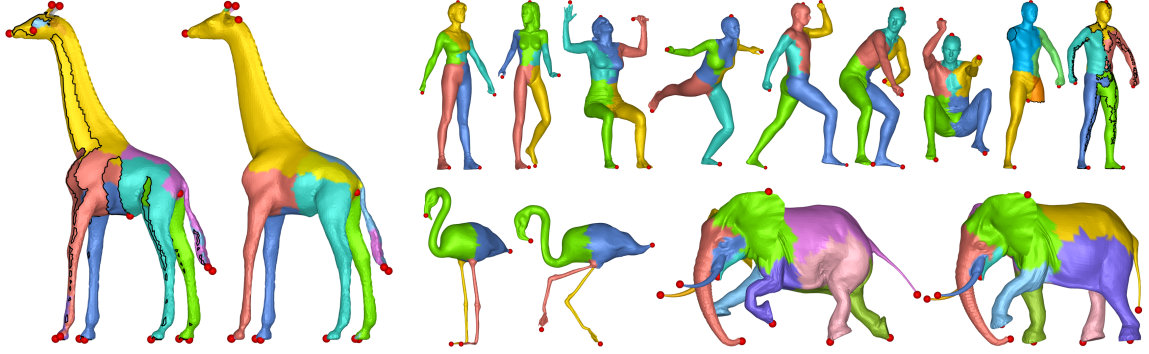


Figure 5.2: Consistent identification of the persistent HKS maxima for different human / animal models in different poses. The two human models on the right are incomplete and partially scanned models with black curves being boundary curves.

K . One can linearly interpolate h over all edges and triangles to obtain a piecewise linear function, still denoted by $h: |K| \rightarrow \mathbb{R}$.

The critical points of h occur at vertices of K where the homology of the *sub-level sets* $\{x \in |K| \text{ s.t. } h(x) < \alpha\}$ changes. Using the concept of persistent homology [31], one can define the persistence of these critical vertices. Informally speaking, while sweeping $|K|$ from $\min h$ to $\max h$, we inspect the topology change of the sub-level set at critical values which are values taken by critical vertices: either new topology is generated or some topology is destroyed, where topology is quantified by a class of ‘cycles’. A critical vertex is a *creator* if new topology appears and a *destroyer* otherwise. It turns out that every destroyer v_2 is uniquely *paired* with a creator v_1 in the sense that v_2 destroys a topology created at v_1 . The *persistence of v_1 and v_2* is defined as $\text{Pers}_h(v_1) = \text{Pers}_h(v_2) = h(v_2) - h(v_1)$, which indicates how long a class of cycles created at v_1 lasts before it gets destroyed at v_2 . A creator is either paired with a unique destroyer, or unpaired, in which case it has infinite persistence.

For a surface, there are three types of critical points, *maxima*, *minima*, and *saddles*. We are interested in computing the maxima with large persistence, which roughly correspond to feature points that are hard to remove by perturbing the *HKS* function values. The persistence pairings and values can be computed by the standard persistent algorithm on a filtration of K dictated by h . Specifically, modify h as:

$$\bar{h}: K \rightarrow \mathbb{R}, \sigma \mapsto w \text{ where } w \text{ is maximum } h \text{ over vertices of } \sigma.$$

Sort the simplices in K with increasing order of \bar{h} values. If two simplices have the same value but different dimensions, break the tie by putting the lower dimensional one first. Otherwise, break the tie arbitrarily. Note, this enforces any face of a simplex to appear before this simplex in the sorted order. Hence this sorted order $\sigma_1, \sigma_1, \dots, \sigma_n$ defines a filtration of K : $\emptyset = K_0 \subset K_1 \subset \dots \subset K_n = K$, where each K_i is a sub-complex of K and $K_{i+1} \setminus K_i = \sigma_i$.

The persistence algorithm when run on this filtration pairs up triangles with edges and edges with vertices. For paired simplices σ_1, σ_2 with $\bar{h}(\sigma_1) \leq \bar{h}(\sigma_2)$, we define their persistence as $\text{Pers}_{\bar{h}}(\sigma_1) = \text{Pers}_{\bar{h}}(\sigma_2) = \bar{h}(\sigma_2) - \bar{h}(\sigma_1)$. It is known that Pers_h and $\text{Pers}_{\bar{h}}$ have the following relation for a piecewise-linear function h . This justifies the discretized framework.

Proposition 5.2.1 ([20]). *Assume that σ_1 is paired with σ_2 , and let v_1 and v_2 be the vertices of σ_1 and σ_2 respectively with the highest h value. Either $v_1 = v_2$ in which case it is a regular (non-critical) vertex. Otherwise, v_1 is paired with v_2 for h and $\text{Pers}_h(v_1) = \text{Pers}_h(v_2) = \text{Pers}_{\bar{h}}(\sigma_1) = \text{Pers}_{\bar{h}}(\sigma_2)$.*

By the above result, if a maximum v of h is paired with a saddle point u , then a triangle incident to v gets paired with an edge incident to u in the filtration induced

by \bar{h} . The persistence value for v can also be computed by persistence value of that triangle. Hence it is safe to work with the filtration induced by \bar{h} to compute persistence of critical points of h .

5.2.2 Region merging

It turns out that for a piecewise-linear function defined on a triangulation of a surface, one can compute the persistence pairing between maxima and saddles (as well as minima and saddles) by sweeping the function value from large to small and maintaining the connected components throughout the course via a union-find data structure [31]. The process takes $O(n \log n)$ time (or $O(n\alpha(n))$ time if vertices are already sorted), where n is the number of vertices and $\alpha(n)$ is the inverse Ackermann's function which grows extremely slowly. We are interested only in computing important *HKS* maxima, that is, those corresponding triangles whose persistence values are more than a prescribed threshold. Instead of computing the complete list of persistence pairings, we employ a *region merging* algorithm that computes these pairings up to a level dictated by an input parameter λ .

Specifically, this algorithm cancels maxima-saddle or equivalently triangle-edge pairs till only those maxima whose persistence values are more than a threshold are left. As a by product, this merging algorithm also produces an explicit segmentation of the surface which seems to be stable w.r.t. partial and incomplete input and which could be of independent interest (see Figure 5.2). This induced segmentation is also later used to help us to approximate the *HKS* maxima for large data sets [24].

The merging algorithm partitions the entire surface $M = |K|$ into regions which are grown iteratively. Each region maintains a *central triangle* and its *pair* edge which

together provide a persistence value. At any stage, the central triangle t of R is the triangle with the highest function value in R , and its pair edge $\text{pr}(t)$ is the edge with the highest function value among all edges in the boundary ∂R of R . Equivalently, the pair edge $\text{pr}(t)$ is also the edge among all edges in ∂R such that the difference between $\bar{h}(t)$ and $\bar{h}(\text{pr}(t))$ is minimized. The current *persistence* of region R is defined as this difference: $\text{p}(R) = \bar{h}(t) - \bar{h}(\text{pr}(t))$.

Initially, each region R consists of a single triangle $t \in K$, The central triangle of R is t itself whose pair-edge $\text{pr}(t)$ is: $\text{pr}(t) = \text{argmax}_{e \in \partial t}(\bar{h}(e))$. At any generic step, let R be the region whose persistence value $\text{p}(R)$ is minimal among all regions — for regions with the same persistence values, we first process the one whose central triangle has a smaller \bar{h} value. Let t be the central triangle in R paired with the edge $e \in \partial R$. Let R' be the *unique* region adjacent to e other than R . Let t' be the central triangle of R' and e' its pair edge. We merge R and R' to region $R \cup R'$. The central triangle t' of R' becomes the central triangle of this merged region whose pair edge is updated with the edge e'' in $\partial(R \cup R')$ that has the largest \bar{h} value. Note that it is necessary that $\bar{h}(t') \geq \bar{h}(t)$ and $\bar{h}(e'') \leq \max\{\bar{h}(e), \bar{h}(e')\}$ as R is the region with the smallest persistence chosen for merging. Hence the persistence $\text{p}(R \cup R')$ for the merged region becomes larger than or equal to both $\text{p}(R)$ and $\text{p}(R')$. If $\partial(R \cup R')$ is empty, then the central triangle is unpaired. We continue this process till the minimal persistence value of any current region exceeds an input threshold λ . The following result explains the utility of the merging algorithm.

Proposition 5.2.2. *At any stage of the merging algorithm, if R is the region with minimal persistence value $\text{p}(R)$, then $\text{Pers}_{\bar{h}}(t) = \text{p}(R)$ where t is the central triangle in R , and the edge $\text{pr}(t)$ pairs with t w.r.t the function \bar{h} .*

In particular, the set of central triangles t of regions that survive after merging with threshold $\lambda > 0$ are exactly the set of triangles with persistence $\text{Pers}_{\bar{h}}(t) \geq \lambda$.

Proof: Recall that at any stage of the merging process, we maintain a central triangle t and its pair edge e for every region R . We always choose the region with the current smallest persistence for merging. It is easy to verify that this greedy strategy maintains the following invariant: t has the highest \bar{h} value among all triangles in R , and e has the highest \bar{h} value among all edges in the boundary ∂R .

We prove the proposition by taking the matrix view of the persistence algorithm which reduces the initial incidence matrix iteratively. In the original persistence algorithm [31], when a new triangle t is considered, columns for certain triangles which are on the left (older triangles) are added to the column of t till the lowest 1 in the column of t does not have another such lowest 1 in its row. This lowest 1 corresponds to the edge to which t pairs. If no such 1 exists, t is declared unpaired. It turns out that [21], it is not necessary to execute this algorithm sequentially as proposed in the original persistence algorithm. Let D be the original edge-triangle incidence matrix that respects the filtration order. One can add columns of D arbitrarily with the constraint that a column is added only to a column on its right, i.e., a triangle's column is added only to another triangle's column which is younger. If these additions provide a unique lowest 1 for each triangle column, then Cohen-Steiner et al. showed that the triangle-edge pairing is same as the original persistence pairing [21].

In our merging algorithm we are simulating this column additions with the guarantee that we reach a unique pairing. Each column of a triangle t represents the boundary of the region R it is currently central to. The edge e currently paired with t corresponds to the lowest 1 in this column (as it has the highest function value

among all edges in ∂R). We merge the region R with minimal persistence value to another region R' adjacent to e . That is, we add the column of t to the column of another triangle t' which is central to R' , and t' has 1 at the entry corresponding to e .

Since e is also in $\partial R'$, it is necessary that the pairing edge e' for t' at this point has a function value that is at least as large as that of e ; i.e, $\bar{h}(e') \geq \bar{h}(e)$. On the other hand, as R is the region currently with the smallest persistence, we have $p(R) \leq p(R')$. It follows that $\bar{h}(t) < \bar{h}(t')$. Hence the central triangle t' of R' must be younger than the central triangle of R . This means our algorithm simulates the column addition to the right. Furthermore, when we merge R with R' , the paired edge of its central triangle is absorbed in the merged region $R \cup R'$ as we consider modulo-2 addition in persistence homology. This edge will not be paired in the future. Using induction, one can show that the central triangle of R pairs uniquely with its pair edge at the time of merging. The first statement of proposition 2 follows.

Since we merge in non-decreasing order of persistence values of the regions, combining the first half of the proposition with Proposition 5.2.1, the second half of the claim follows. ■

In other words, the above result implies that our algorithm merges regions with respect to the persistence order induced by the function \bar{h} , which by Proposition 5.2.1, is then connected to the persistence order of the function h . Note that at the beginning, all the regions we merge have zero-persistence value. These do not correspond to pairings between critical points of h (the case where $v_1 = v_2$ in Proposition 5.2.1). An example of the merging process is shown in Figure 5.3.

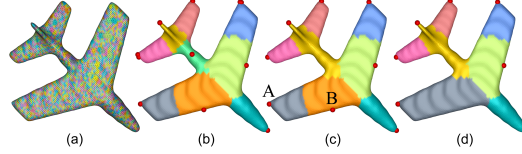


Figure 5.3: Persistence based merging on an Airplane model. (a) Initially, every triangle represents a region; (b) shows the segmentation after merging all zero-persistence regions. The central triangle of every remaining region corresponds to a maximum of input function h . (c) A and B are two maxima, for the gray and orange regions, respectively, with A having a larger h value; (d) after the merging, A stays the maximum for the new region (gray colored).

Using the union-find data structure to maintain intermediate regions, and a priority-queue to maintain the order of regions to be processed, this merging process takes $O(n \log n)$ time in the worst case. However, it terminates faster for lower threshold λ , and it also produces a segmentation of the input domain, which is in some sense a combinatorial version of the stable manifold decomposition induced by the input function. We remark that the merging idea has been used before to produce segmentations [15, 39]. However, we show that our merging process and the resulting segmentation respect the topological persistence pairing.

Finally, the above procedure is for handling 2-manifold without boundary. Given a surface M with boundary, assume E' is the set of boundary edges, the algorithm simply ignores edges in E' when computing the boundary of each region. This is equivalent to first seal all the holes in M and convert M to a manifold M' without boundary; and then perform the persistence algorithm on M' . To seal a hole in M (a loop C in ∂M), we add a dummy vertex v with function value lower than all vertices in M , and then connect all edges in C to v .

5.3 Matching

Feature vector. We now identify a small set of persistent heat maxima as *feature points* to concisely represent an input shape. Specifically, we use a fixed number κ , and perform the region-merging process till the remaining number of regions left equals κ . Experimentally we observe that $\kappa = 15$ maxima are usually sufficient to capture most prominent features, which we fix from now on. See the two Giraffe models in Figure 5.2 for an example, where similar feature points are captured for both the partial scan (left picture) and the complete version (right picture).

Next, we construct a feature vector for each one of the $\kappa = 15$ selected feature point. Since *HKS* for different time values describe local geometry at different scales, we compute *HKS* values for different t at each of the feature points — such a multi-scale description of local shape helps to enhance the discriminability of our feature vectors. We choose 15 different time scales to compute a 15D feature vector for each feature point. The time scales are chosen relative to a constant τ which is the parameter used for computing the discrete Laplace operator. The algorithm in [7] approximates the discrete Laplacian based on a heat-dissipation process whose duration is determined by τ . Hence we take τ as the *time unit*, as it is not meaningful to use the eigenvectors of a *HKS* constructed at a resolution smaller than τ . In our experiments we fix τ to be 0.0002 and we consider $t = \alpha * \tau$ where α varies over 5, 20, 40, 60, 100, 150, 200, 300, 400, 500, 600, 700, 800, 900, 1000.

Scoring. Let F_1 and F_2 be the set of feature vectors computed for two surfaces M_1 and M_2 , respectively. Each F_i is a collection of 15 feature vectors, where each vector is of dimension 15. For two vectors $f_1 \in F_1$ and $f_2 \in F_2$, we use the L_1 -norm

$\|f_1 - f_2\|_1$ to measure their distance. The matching score between M_1 and M_2 is computed as

$$\sum_{f_1 \in F_1} \min_{f_2 \in F_2} \|f_1 - f_2\|_1 + \sum_{f_2 \in F_2} \min_{f_1 \in F_1} \|f_1 - f_2\|_1.$$

Note that this scoring function does not satisfy the triangle inequality. One could use more sophisticated distance functions for scoring. We resort to this simple scoring function since it already provides good results. Once F_1 and F_2 are given with a preprocessing, computation of the scores becomes very efficient for such a small set of vectors.

5.4 Results

Database. To test our matching method, we build a database of 300 shapes divided into 21 classes (dogs, horses, airplanes, chairs, glasses, humans, etc.), where each class has more than one shape including partial and incomplete versions. We create our own database mainly because we want to emphasize the robustness of our method for pose-oblivious partial and incomplete matching and existing shape databases do not necessarily cater to this need. Models in our database are collected from available sources [17, 84], to which we add partial and incomplete versions of them. The partial scan data is generated by taking a random viewing direction and keeping all vertices of a complete model that are visible from this direction. The incomplete data is generated by removing one or more portions off a complete model using cutting planes. For every pose of a model, only one version of it (i.e, either the complete model, its partial scan, or the incomplete version of it) is in the database or in the set of query models. There are 50 query models. Among them, 18 are complete











































Query	Top five matches				
 C	 P	 C	 P	 C	 P
 P	 I	 C	 C	 I	 C
 C	 C	 C	 C	 C	 C
 I	 P	 P	 I	 C	 C
 C	 C	 C	 C	 C	 C
 I	 C	 C	 C	 C	 C
 P	 P	 C	 P	 C	 P

Table 5.1: Query models and top five matches returned for each query. Letters C, P, I indicate complete, partial, and incomplete models respectively.

models, and 32 are partial or incomplete models. Finally, we scale each shape to a unit bounding box to factor out the global scaling.

Parameters. For all models, we compute mesh-Laplacian and its eigenvectors using a universal time-unit $\tau = 0.0002$. For each model, we then compute 15 persistent maxima for the *HKS* function at time 5τ , which we observe provides robust feature points across partial and incomplete models. We then construct, for every persistent maximum, a feature vector of 15D as described in Section 5.3. A shape is now represented by 15 feature vectors, each of dimension 15.

Hit rate. Given a query shape, we compute the matching score between it and every shape in the database. The top 5 matches for some queries are shown in Table 5.1. We also compute a *hit rate* as follows. For a query shape in a class, there is a *Top-k*

hit if a model is retrieved from the same class within the top k matches. For N query shapes, the Top- k hit rate is the percentage of the Top- k hits with respect to N .

#queries	ours	EVD	LFD
32 incompl.	88% / 91%	62% / 62%	56% / 59%
18 compl.	78% / 83%	100% / 100%	39% / 39%
50 total	84% / 88%	76% / 76%	50% / 52%

Table 5.2: Each entry shows Top-3 / Top-5 hit rates for our method, EVD, and LFD. “32 incompl.” includes **both** partial and incomplete queries and “18 compl.” includes only pose-altered queries. The database contains 300 models.

We compare our method with two competitive shape retrieval methods: Eigenvalue descriptor (EVD) method of [45] and Light Field Distribution (LFD) method of [16]. We chose these two methods because EVD represents a state-of-the-art spectral technique for articulated shape retrieval, and LFD represents a technique that has been reported to be one of the best in literature for rigid models [82]. Table 5.2 shows the Top-3 and Top-5 hit rates of all three methods for 50 query models.

We find that, if pose variations are disallowed and models are complete or even near-complete, all three methods work very well achieving a hit rate close to 100% with LFD performing the best (details of this result omitted here). If only pose variations are allowed, EVD performs quite well for retrieving different pose variations of input queries as the second row in Table 5.2 shows. However, for pose-altered models with significant incompleteness our method beats LFD and EVD by large margin as the first row in Table 5.2 exhibits. An example of the top 5 matches returned for an incomplete Octopus query by these methods is shown in Figure 5.4.

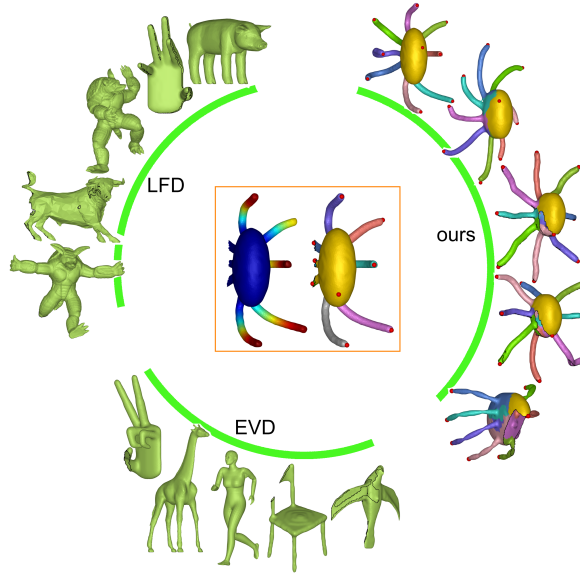


Figure 5.4: Top five matches for an incomplete Octopus query model by our algorithm, EVD, and LFD.

An additional advantage of our method over EVD and LFD is that it can also provide correspondences between matching features in models.

Timing data. We present the timing data of all three algorithms in Table 5.3. To make our algorithm scalable, we use the exact algorithm for models with 25K vertices or less, and run the approximation algorithm *HKS-simp* to compute *HKS* for larger data. For EVD algorithm, the timing reported is obtained by an optimized version of the original implementation obtained from the authors. To handle large meshes using EVD, we also follow their original strategy to first decimate the models with QSlim to 3K vertices, and then process them for matching. For LFD algorithm, we use the original implementation from the authors. LFD is slowest in terms of the retrieval time though its preprocessing time for models is the best. EVD has the

fastest retrieval time and its preprocessing time is comparable to ours, although its accuracy for partial and incomplete query shapes is worse than our algorithm.

		Pre-processing (sec)		
Model	#v	ours	EVD	LFD
Plier	4.8k	9.4	7.9	0.7
Hand	8.7k	19.8	8.1	1.0
Octopus	11.0k	25.2	8.5	1.3
Teddy	12.6k	30.7	8.6	1.4
Human	15.2k	42.8	8.9	1.5
Dragon	1000k	712.0	28.0	40.0
Elephant	1500k	1032.0	37.0	60.0
Retrieval time (sec)		0.02	0.006	36.0

Table 5.3: All experiments are carried out on a Dell computer with Intel 2.4GHz CPU and 6GB RAM.

Validity of Persistent Maxima. We choose persistent maxima as feature points for matching. We provide some experimental results to validate this choice. First, Figure 5.2 shows that the persistent maxima and their induced segmentations are rather robust in practice for different models in various poses, and in their partial and incomplete versions.

Next, we conduct the following experiment: we replace the *HKS* maxima by extrema of the average geodesic distance (AGD) as in [103] and by extrema of discrete Gaussian curvature (GC) computed using the method from [60]. We compute the top 15 persistent AGD maxima and GC maxima by the region merging algorithm and then construct feature vectors for these feature points to match shapes. The Top-3 and Top-5 hit rates of the three algorithms using different feature points (i.e, persistent

HKS , AGD, or GD maxima) are reported in Table 5.4. AGD takes global information into account, thus does not match well for partial / incomplete models. GC takes only a very local neighborhood into account, thus the function value at each point does not reflect global features very well, and performs the worst. Interestingly, the hit-rate for the GC method will improve if one first *sparsify* the set of potential feature points by keeping only one extremum within every neighborhood of an appropriate size. However, the question is then how one should choose the size of this “neighborhood”, and it is not clear whether there is any natural choice for it.

#queries	ours (HKS)	AGD	GC
32 incompl.	88% / 91%	78% / 81%	38% / 41%
18 compl.	78% / 83%	72% / 94%	83% / 89%
50 total	84% / 88%	76% / 86%	55% / 58%

Table 5.4: Each entry shows Top-3 / Top-5 hit rates for our method using HKS , AGD and GC feature points.

5.5 Conclusion and Discussion

In this chapter we combined techniques from spectral theory and computational topology to design a method for matching partial and incomplete shapes. Heat Kernel Signature functions from spectral theory provide a way of capturing features of a shape at various scales. However, discretization and approximations at various stages inject noise into this function which requires a filtering. We achieve this filtering by identifying maxima of this function that are persistent. Heat values of these maxima

at different time scales become the signature of the shape with which a simple scoring scheme can be adopted for matching.

Our results show that the method is quite effective in shape matching. It outperforms existing techniques for pose-oblivious matching of partial and incomplete models. Our current method requires manifold meshes, although it can tolerate mild discrepancies in this regard. It would be interesting to investigate whether this method can be adapted to handle triangle soups in general. We also remark that our algorithm consistently fails to match shapes in the category “Snake” in our experiments. This is perhaps because that snakes do not possess many “features” that the *HKS* describes. It will be interesting to investigate how one can enhance our algorithm so that it can handle a broader range of shapes.

Finally, we remark that in this chapter, we compute eigenvectors of the mesh-Laplacian without enforcing any boundary condition. It is believed that the mesh-Laplacian assumes certain implicit boundary condition, although it has not been shown what this boundary condition is. It will be interesting to further investigate in this direction to see for example whether changing the boundary condition (by using, say, the accurate high-order finite element method based approach in [72]) will affect our matching results significantly.

Chapter 6: Eigen Deformation of 3D Models [25]

The final contribution of this thesis is another application. In this chapter, we use our theoretical understanding of the Laplace operator to create a framework that will allow us to freely deform shapes without asking the user for extra structures, like cages, skeletons etc.

The creation of deformed models from an existing one is a quintessential task in animations and geometric modeling. A user availing such a system would like to have the flexibility in controlling the deformation in real-time while preserving the isometry. In recent years, considerable progress has been made to meet these goals and a number of approaches have been suggested.

Some of the earliest techniques for mesh deformation involved using skeletons [102]. A user typically creates a skeletal shape which is bound to the mesh. The mesh is then deformed by deforming the skeleton and transforming the changes back to the mesh. This approach puts a burden on the user to create an appropriate skeleton and bind the mesh to it. Later work [29, 2, 99] sought to reduce this burden by automatically creating a skeleton. Automatic generation of good skeletons and accurate transformation of deformations from skeleton to mesh remain challenging till today.

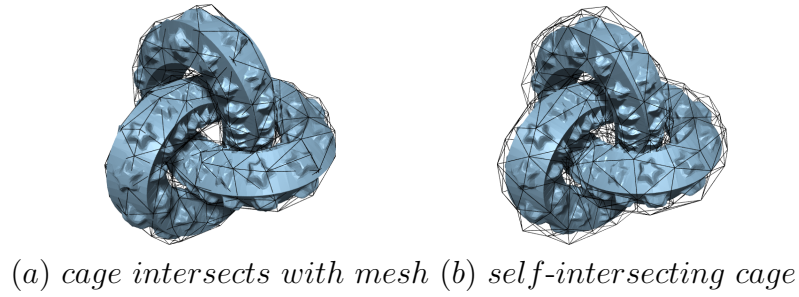
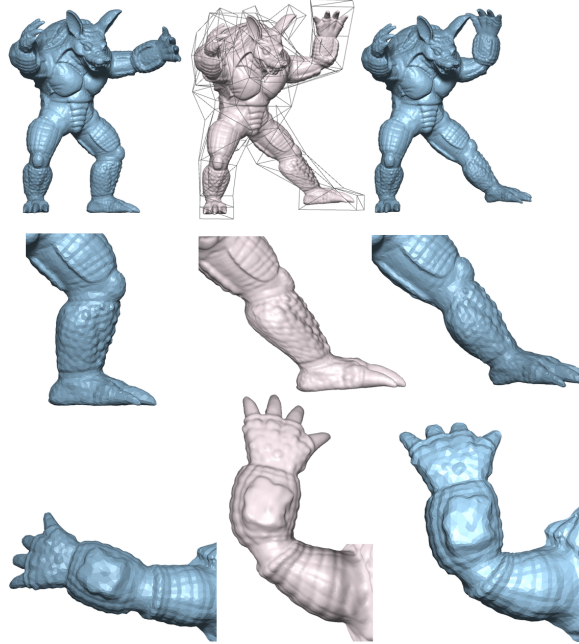


Figure 6.1: Creating correct cages for meshes

Later approaches replaced the skeletons with a sparse cage surrounding the mesh and then controlled the deformation through the movement of the cage. The use of a cage is akin to the concept of control polyhedron that is used for free-form deformations. The authors in [79] introduced the concept of control polyhedron and others refined it later [50, 57]. It is well recognized that a control polyhedron does not provide sufficient flexibility to deform meshes with complicated topology and geometry [46, 54]. More recent techniques increase this flexibility by introducing sophisticated coordinate functions that bind the cage to the mesh. In general, each vertex of the mesh is associated with weights called *coordinates* for each vertex of the cage. This allows the mesh vertices to be represented as a linear combination of the vertices of the cage. Cage based techniques vary in how the weights of the vertices are computed. Early attempts include extending the notion of barycentric coordinates to polyhedra [47, 70, 98]. More recently, Mean Value Coordinates [32, 33, 51], Harmonic [46] and Green [54] Coordinates have been proposed for the purpose. The authors in [54] pointed out that the Mean Value and Harmonic Coordinates do not necessarily preserve shapes though they provide affine invariant deformations. They overcome this difficulty by providing a real-time deformation tool that preserves



(a) Original Mesh (b) Green Coordinates (c) Our Method

Figure 6.2: Comparing with green coordinates

shapes. Recently, in [44], Sorkine et al. use biharmonic coordinates to integrate cages and skeletons under a single framework. This allows the user to use different types of techniques simultaneously based on the result desired. Also, in [9], Ben-Chen et al. use a small set of control points on the original mesh to guide the deformation of the cage. Nevertheless, the limitation of creating pseudo structures like cages and skeletons by users still persists.

Creating pseudo structures, especially cages, can be time-consuming and tricky, as Figure 6.1 illustrates. The cage on the left, for example, fails to envelop the mesh correctly. The cage on the right envelops the entire mesh, but has self-intersections leading to incorrect calculation of coordinates. It falls upon the user to manually move the cage vertices to rectify the cage, which can become time-consuming. A

user typically spends more time creating a good cage, than deforming the mesh. The state-of-the art would be enhanced if one can have a tool that has the capabilities of Green Coordinates but without the need for the cage. Our approach is geared towards that. Figure 6.2 illustrates the point by showing how our method produces similar quality deformations as Green Coordinates but without any cage.

There are other approaches that impart the deformation directly to the surface mesh and thus eliminate the need for intermediate structures like cages or skeletons; see e.g. [10, 11, 43, 87, 86, 105]. These techniques usually optimize an energy function tied to the deformation and user control to achieve high quality deformations. However, they either require non-linear solvers or multiple iterations of linear solvers to compute new vertex positions, making them slower for large meshes. Also see [12] for a survey on various deformation techniques that use the Laplacian operator to formulate the energy.

Our work We introduce a novel approach that allows the user to apply the deformation directly to the mesh but without solving any non-linear system and thus improving both time and numerical accuracy. The method uses a skeleton but without explicitly constructing one. It computes the eigenvectors of the Laplace-Beltrami operator to provide a low frequency harmonic functional basis which helps creating an *implicit* skeleton. The skeleton is a high-level abstraction of the shape of the mesh, lacking small features and details. It deforms this skeleton by computing a new set of eigen coefficients. These coefficients are solutions of a *linear* system which can be computed in real-time. Finally, it adds the details back to the skeleton to get the

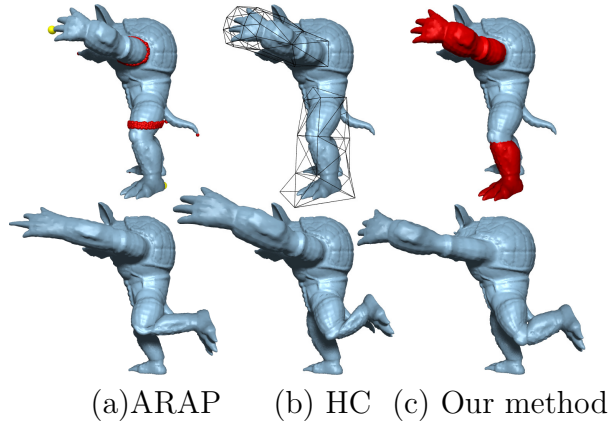


Figure 6.3: Comparisons when we stretch the arm and bend the leg of the armadillo. Note that our method handles stretching better than as-rigid-as-possible (ARAP), and extreme bending better than harmonic coordinates (HC).

deformed mesh. We point out that, unlike other skeleton-based approaches, our implicit skeleton is simply the original mesh whose vertex coordinates are derived from a truncated set of eigenvectors.

Our eigen-framework retains the advantages of both the cage-based and cage-less approaches. Figures 6.2,6.10,6.13 illustrate this point. Our deformation software is easy to use and efficient. We are also able to handle both isometric as well as non-isometric deformations like stretching gracefully, as shown in Figures 6.3 and ??.

Comparison with previous work on spectral deformation Recently, Rong et al. [75, 76] proposed a deformation framework using the eigenvectors of the Laplace operator. Although this seems similar to our approach, the reasons why we use eigenvectors are different. In particular, Rong et al. perform as-rigid-as-possible [86] deformations by trying to preserve the Laplace operator. However, they use the eigenvectors to change the problem domain from spatial to spectral, thereby reducing

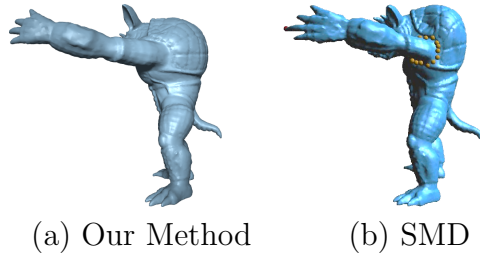


Figure 6.4: Stretching the arm of the armadillo. Note that spectral mesh deformation (SMD), causes the entire mesh to deform in order to preserve the mesh volume.

the size of the optimization problem to the number of eigenvectors used. Usually, they need at least 100 eigenvectors, and more eigenvectors make their final deformation look better.

Our method, on the other hand, just needs a functional basis of low frequency harmonic functions in which the meshes are represented. We use low frequency eigenvectors in order to get a smooth fit for a target deformation since our goal is to guarantee a *smoothly varying deformation* rather than *isometry*. Hence, we can guide deformations by using as few as 8 eigenvectors. Furthermore, since we do not try to preserve the Laplace operator, and hence isometry, we can handle stretching better than [75, 76], as Figure 6.4 illustrates.

Also, [75, 76] use deformation transfer based techniques to recover details, and sometimes produce artifacts in the deformed mesh, as shown in Figure 6.12. We develop a more sophisticated iterative technique to recover the details with greater accuracy. Finally, in [75, 76], the positions of the constrained vertices need to be changed each time the user wish to change the scope of the deformation, increasing users' burden to specify the intended deformation.

6.1 Eigen-framework

As mentioned in Chapter 1, the Laplace-Beltrami operator has many useful properties and has been widely used in many geometric processing applications. For example, it is well known that the Laplace operator uniquely decides the intrinsic geometry of the input manifold M . Hence, isometric manifolds share the same Laplacian, which makes the Laplace operator a natural tool to capture or describe isometric deformation. Indeed, this idea has been used to build local coordinates for mesh editing and deformation to help produce as-rigid-as possible type of deformation [85, 105]. The eigenfunctions of the Laplace operator form a natural basis for square integrable functions defined on M . Analogous to Fourier harmonics for functions on a circle, Laplacian eigenfunctions with lower eigenvalues correspond to low frequency modes, while those with higher eigenvalues correspond to high frequency modes that describe the details of the input manifold M .

In our problem, the input is a triangular mesh approximating a hidden surface M . In such case, we need a discrete version of the Laplace operator computed from this mesh. Several choices are available in the literature [7, 23, 41, 70, 73]. Again, we use the mesh-Laplacian developed in [7], although other discretizations of the Laplace operator should also be fine. See [12] for a discussion of the effects that the various discretizations have on Laplacian based surface deformation techniques.

6.1.1 Eigen-skeleton

Given the Laplace operator Δ_M of an input manifold, let ϕ_1, ϕ_2, \dots denote the eigenfunctions of Δ_M . These eigenfunctions form a basis for $\mathcal{L}^2(M)$, the family of square integrable functions on M . Hence we can re-write any function $f \in \mathcal{L}^2(M)$ as

$f = \sum_{i=1}^{\infty} \alpha^i \phi_i$, where $\alpha^i = \langle f, \phi_i \rangle$ and $\langle \cdot, \cdot \rangle$ is the inner product in the functional space $\mathcal{L}^2(\mathbf{M})$. Under this view, the function f can be considered as a vector $\alpha = [\alpha^1, \alpha^2, \dots]$ in the infinite-dimensional *eigenspace* spanned by the Laplacian eigenfunctions.

Now, consider the *coordinate functions* (f_x, f_y, f_z) defined on \mathbf{M} whose values at each point are simply the x , y and z -coordinate values of the point, respectively. By re-writing these coordinate functions, we can represent a surface by three vectors $(\alpha_x, \alpha_y, \alpha_z)$ in its eigenspace. We call these the *coordinate weights* of \mathbf{M} . The embedding of a manifold is fully decided by its coordinate weights once the eigenfunctions are given.

Finally, since higher eigenfunctions have higher frequencies and hence capture smaller details, we can truncate the number of eigenfunctions (i.e, use only the top few coordinate weights) for reconstructing the surface to get varying levels of detail.

Eigen-skeleton Given a surface mesh, also denoted by \mathbf{M} , with n vertices, we compute the eigenvectors of the mesh-Laplacian computed from \mathbf{M} , denoted still by ϕ_1, \dots, ϕ_n . We now restrict ourselves to the first few, say $m < n$, eigenvectors of the shape. This gives us a higher level abstraction of the surface that captures its coarser features. Specifically, let $P = \{p_1, p_2, \dots, p_n\}$ be the set of points reconstructed from the vertex set $V = \{v_1, v_2, \dots, v_n\}$ of \mathbf{M} using only the first m eigenvectors $\phi_1, \phi_2, \dots, \phi_m$ of the mesh-Laplacian. That is, if

$$\hat{f}_x = \sum_{i=1}^m \alpha_x^i \phi_i, \quad \hat{f}_y = \sum_{i=1}^m \alpha_y^i \phi_i, \quad \hat{f}_z = \sum_{i=1}^m \alpha_z^i \phi_i,$$

then $p_i = \{\hat{f}_x(v_i), \hat{f}_y(v_i), \hat{f}_z(v_i)\}$ for $i = 1, \dots, n$. Consider the mesh $\mathbf{K} = \mathbf{K}_m$ with vertices p_i and the connectivity same as that of \mathbf{M} . We call this mesh the *eigen-skeleton*⁵ of \mathbf{M} . For different values of m , the eigen-skeleton \mathbf{K}_m abstracts the input surface \mathbf{M} at different levels of detail.

6.2 Algorithm

Our algorithm will compute the target configuration by deforming the eigen-skeleton. In particular, the eigen-skeleton \mathbf{K}_m is decided by the $3m$ coordinate weights $\alpha_x^1, \dots, \alpha_x^m$; $\alpha_y^1, \dots, \alpha_y^m$ and $\alpha_z^1, \dots, \alpha_z^m$. We will deform the eigen-skeleton by computing a new set of coordinate weights by solving only a linear system. Since the number of eigenvectors used is typically much smaller than the number of vertices involved in deformation, a solution can be obtained in real-time. We will see later that, other than being efficient, the use of coordinate weights also has the advantage that the deformation tends to be smooth across the entire shape. Since the new eigen-skeleton lacks smaller features, we design a novel and effective algorithm to add back details using the one-to-one correspondence between the vertices of the eigen-skeleton and the original mesh. The high level framework for our algorithm is described in Algorithm 1. Next, we describe each step in detail.

6.2.1 Step 1: Coarse Guess-Target Configuration

Our software uses a standard and simple interface for the user to specify the intended target configuration. First, the user selects a mesh region that he wishes to deform. We call it the *region of interest*, R , and let $V_R \subseteq V$ denote the set of vertices

⁵The concept of eigen-skeletons is not new and has been used for mesh compression in [48]. For more applications, please refer to the survey papers [53, 104].

Algorithm 1: Deformation Framework

Input : Input mesh M
Output: Deformed mesh M^*

```
1 begin
2   Compute eigen-skeleton  $K$  for  $M$ 
3   While (user initiates deformation) {
4     Step 1: Interpret user-specified deformation and
5           compute a coarse target configuration  $\tilde{K}$ 
6     Step 2: Obtain  $K^*$ , a smooth approximation to  $\tilde{K}$ 
7     Step 3: Add shape details to obtain  $M^*$ 
8   }
```

in this region. Next, the user specifies the type of transformation desired for the region of interest, which can be either a **translation-type** or a **rotation-type**. The user then indicates the target configuration by simply dragging some point, say $\mathbf{v} \in V_R$, to its target position $\tilde{\mathbf{v}}$.

From the type of transformation combined with the position of \mathbf{v} and $\tilde{\mathbf{v}}$, our algorithm computes either a translational vector $\mathbf{t} = \tilde{\mathbf{v}} - \mathbf{v}$ if the desired transformation is a translation-type, or a rotational pivot \mathbf{p} and a rotation matrix \mathbf{r} if the desired transformation is a rotation-type. We then compute a coarse target configuration \tilde{K} for the eigen-skeleton K using the following simple procedure: For all points $v_i \notin V_R$, the target position for the corresponding point p_i in the eigen-skeleton is simply $\tilde{p}_i = p_i$. For each point $v_i \in V_R$, if the type of transformation is translation, then the target position is $\tilde{p}_i = p_i + \mathbf{t}$. If the type of transformation is rotation, then the target position is $\tilde{p}_i = \mathbf{r}(p_i - \mathbf{p}) + \mathbf{p}$.

In other words, we simply cut the region of interest and apply to it the target transformation indicated by the user, while the rest of the shape remains intact. Such an initial guess of target configuration is of course rather unsatisfactory. In fact,

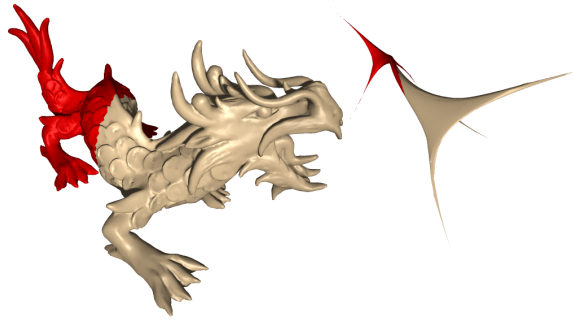


Figure 6.5: The dragon model with its eigen-skeleton created using 8 eigenvectors

the deformation is not even continuous (along the boundary of the region of influence R , there is a dramatic, non-continuous change in the deformation). However, we will see later that in Step 2, our algorithm takes this initial target configuration and produces a much better, smoothly bent eigen-skeleton. See Figures 6.5 and 6.6 for an example: in order to bend the body of the dragon, we specify a rotation on the back half of the dragon. We then apply this rotation on the entire region of interest in the eigen-skeleton to obtain a target configuration \tilde{K} , as shown in the left image in Figure 6.6. Note that Step 2 will produce a nice, global deformation for the eigen-skeleton, as shown on the right in Figure 6.6.

Observe that the translation-type and rotation-type motions are only high-level guidance for producing the final deformation in Step 2. The final deformation is of course not necessarily rigid. A stretching effect, for example, can be achieved by a simple translation-type motion. From the user's point of view, the amount of work to specify the deformation is very little and rather intuitive, while the algorithm reconstructs a more complex deformation from the user's coarse input.

6.2.2 Step 2: Eigen-skeleton Deformation

After Step 1, we have a guess-target configuration $\tilde{\mathbf{K}}$ for the eigen-skeleton \mathbf{K} . In this step, we wish to compute an improved target deformed eigen-skeleton \mathbf{K}^* from the guess-target configuration $\tilde{\mathbf{K}}$. In Step 3 described in next section, we will add details back to \mathbf{K}^* to obtain a deformed surface \mathbf{M}^* for the input surface \mathbf{M} . The following diagram illustrates successive structures.

$$\begin{array}{ccccccc} \mathbf{M} & \xrightarrow[\text{skeleton}]{\text{Get}} & \mathbf{K} & \xrightarrow[\text{guess target}]{\text{Step 1}} & \tilde{\mathbf{K}} & \xrightarrow[\text{improve target}]{\text{Step 2}} & \mathbf{K}^* & \xrightarrow[\text{add details}]{\text{Step 3}} & \mathbf{M}^* \\ \\ v_i & \longrightarrow & p_i & \longrightarrow & \tilde{p}_i & \longrightarrow & p_i^* & \longrightarrow & v_i^* \end{array}$$

Recall that \tilde{p}_i is the position of the i th vertex v_i in the guess-target skeleton $\tilde{\mathbf{K}}$. Now consider the coordinate functions $(\tilde{f}_x, \tilde{f}_y, \tilde{f}_z)$ of the guess-target skeleton $\tilde{\mathbf{K}}$. Note, each function \tilde{f}_a , where $a \in \{x, y, z\}$, is a function $\mathbf{M} \rightarrow \mathbb{R}$ on the input surface \mathbf{M} . The guess configuration $\tilde{\mathbf{K}}$ is often far from being satisfactory. In particular, by cutting the region of influence and simply translating and rotating this part, a discontinuity exists at the boundary of region of influence. In other words, there is no smooth transition across the cut. See the enlarged picture in Figure 6.6 left image. This means that the coordinate functions \tilde{f}_a are not smooth across the cut. To get a smooth deformed skeleton, we wish to find a smooth approximation f_a^* for each \tilde{f}_a . This will give rise to an improved deformed skeleton \mathbf{K}^* with the i th vertex $p_i^* = (f_x^*[i], f_y^*[i], f_z^*[i])$.

To this end, note that since the eigenfunctions ϕ_j s of \mathbf{M} form a basis for the family of square-integrable functions on \mathbf{M} , each \tilde{f}_a can be written as a linear combination of all eigenfunctions ϕ_i s for $i = 1, \dots, n$. Furthermore, eigenfunctions with low eigenvalues are analogous to modes with low-frequency while those with high eigenvalues

correspond to high-frequency modes. Since we aim to obtain a smooth reconstruction of \tilde{f}_a , we want to ignore high frequency modes. Hence we find a smooth reconstruction of \tilde{f}_a using only the top m low-frequency eigenfunctions ϕ_1, \dots, ϕ_m of \mathbf{M} . This is achieved as follows: Suppose $f_a^* = \sum_{j=1}^m \tilde{\alpha}_a^j \phi_j$, and $A_j = (\tilde{\alpha}_x^j, \tilde{\alpha}_y^j, \tilde{\alpha}_z^j)$ for $j = 1, \dots, m$. We want to find weights $(\tilde{\alpha}_x, \tilde{\alpha}_y, \tilde{\alpha}_z)$ that minimize the following energy function where $\phi_j[i]$ is the value of the j -th eigenfunction ϕ_j on the vertex v_i :

$$E = \sum_{i=1}^n \left\| \sum_{j=1}^m A_j \phi_j[i] - \tilde{p}_i \right\|^2. \quad (6.1)$$

Intuitively, the discontinuity in the coordinates of guess-target configuration $\tilde{\mathbf{K}}$ requires high frequency eigenfunctions to reconstruct it, and using only low-frequency modes produces smoother f_a^* s, which induces better deformed skeleton \mathbf{K}^* . See the right picture of Figure 6.6 — the skeleton reconstructed from new coordinate weights $(\tilde{\alpha}_x, \tilde{\alpha}_y, \tilde{\alpha}_z)$ after Step 2 shows a smooth transition from the region of interest to the rest.

An alternative interpretation Before we describe how we minimize the above energy function, we provide an alternative interpretation for the formulation of our energy function. Recall after Step 1, we have a guess-target configuration $\tilde{\mathbf{K}}$ for the eigen-skeleton \mathbf{K} , and \tilde{p}_i is the position of vertex v_i in this skeleton $\tilde{\mathbf{K}}$. Intuitively, if $\tilde{\mathbf{K}}$ turns out to be the eigen-skeleton of an isometric deformation $\tilde{\mathbf{M}}$ of \mathbf{M} , then there exist new coordinate weights $(\tilde{\alpha}_x, \tilde{\alpha}_y, \tilde{\alpha}_z)$, such that

$$\left\| \sum_{j=1}^m A_j \phi_j[i] - \tilde{p}_i \right\|^2 = 0 \quad \forall v_i \in V$$

where $A_j = (\tilde{\alpha}_x^j, \tilde{\alpha}_y^j, \tilde{\alpha}_z^j)$ represents the j -th entry of each $\tilde{\alpha}_a$. This is true because the manifold $\tilde{\mathbf{M}}$ has the same eigenfunctions as \mathbf{M} , and its corresponding coordinate

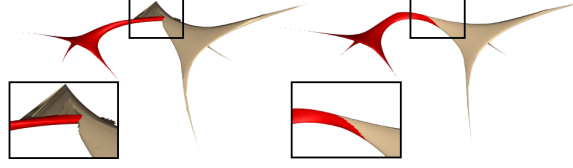


Figure 6.6: Left picture: A coarse discontinuous initial guess. Rotating the entire region of interest (colored red) causes the discontinuity at its boundary. We use the mesh connectivity information from the original mesh to further emphasize this point. Right picture: After step 2, we obtain a smooth transition across the boundary.

functions can be written as a linear combination of the eigenfunctions of \tilde{M} (i.e. ϕ_1, \dots, ϕ_n). The new coordinate weights $\tilde{\alpha}_a$ are simply the first m coefficients for ϕ_1, \dots, ϕ_m in this linear combination.

If the deformation is not isometric, then we can try to find the best fit $(\tilde{\alpha}_x^j, \tilde{\alpha}_y^j, \tilde{\alpha}_z^j)$ for $j \in [1, m]$ by minimizing the above quantity over all vertices in V , that is, minimizing the energy function as defined in Equation (6.1). Experimental results show that our method tends to preserve isometry in practice when such a deformation is possible; see for example Figure 6.10 and Table 6.2. At the same time, since we do not try to preserve the Laplace operator, we can handle non-isometric deformations like stretching in a more natural manner, compared to [86, 76, 75]; see for example Figures 6.4 and ??.

Minimizing the Energy function E There are $3m$ variables in the energy function in Equation (6.1). To minimize E , we compute its gradient with respect to A_k

$$\begin{aligned} \frac{\partial E}{\partial A_k} &= 2 \sum_{i=1}^n \phi_k[i] \left(\sum_{j=1}^m A_j \phi_j[i] - \tilde{p}_i \right) \\ &= 2 \left(\sum_{j=1}^m A_j \langle \phi_k \cdot \phi_j \rangle - (\langle \phi_k \cdot \tilde{f}_x \rangle, \langle \phi_k \cdot \tilde{f}_y \rangle, \langle \phi_k \cdot \tilde{f}_z \rangle) \right) \end{aligned}$$

where $(\tilde{f}_x, \tilde{f}_y, \tilde{f}_z)$ are the coordinate functions of the guess-target skeleton. Now, setting the partial derivatives to zero for all A_k , we get

$$\sum_{j=1}^m \langle \phi_k \cdot \phi_j \rangle A_j = (\langle \phi_k \cdot \tilde{f}_x \rangle, \langle \phi_k \cdot \tilde{f}_y \rangle, \langle \phi_k \cdot \tilde{f}_z \rangle)$$

which leads to a linear system of equations in the following form: $\Phi A^* = b$, where Φ is an m by m matrix with $\Phi_{i,j} = \langle \phi_i \cdot \phi_j \rangle$ ⁶, A^* is an m by 3 matrix with $A_{i,\cdot}^* = A_i$ and b is also an m by 3 matrix with the i th row as $(\langle \phi_i \cdot \tilde{f}_x \rangle, \langle \phi_i \cdot \tilde{f}_y \rangle, \langle \phi_i \cdot \tilde{f}_z \rangle)$. Using A^* as coordinate weights, we reconstruct the new deformed eigen-skeleton K^* .

6.2.3 Step 3: Shape Recovery

We now have the deformed eigen-skeleton K^* . Since we use only the top few eigenvectors for deformation, this skeleton lacks small features and fine details of the original mesh. To obtain the deformed mesh M^* , we need to add appropriate details back to K^* .

In order to keep track of all the shape details, when creating the original eigen-skeleton K , we also keep track of the difference between v_i and its reconstruction p_i . We call it the *detail vector* which is given by $d_{v_i} = v_i - p_i$. However, since the mesh is deforming, we cannot simply add d_{v_i} back to \tilde{p}_i .

To address this issue, we keep track of d_{v_i} in a *local coordinate frame* around p_i . In particular, for each p_i , we compute three axes that are given by: (i) the normal at p_i , (ii) projection of an edge incident at p_i onto a tangent plane at p_i , and (iii) a third vector orthogonal to the previous two. This frame remains consistent with the local orientation of the vertex. For each detail vector d_{v_i} , we record its coordinates in this

⁶In the ideal case, the Laplace-Beltrami operator is symmetric, which makes its eigenvectors orthonormal, and Φ the identity matrix. However, we use area weights when building the Laplace-Beltrami operator, which makes it asymmetric, and Φ a matrix with non-zero off diagonal entries.

local frame, which is the projection of d_{v_i} onto the three axes. After the eigen-skeleton is deformed to a new configuration \mathbf{K}^* , we compute the new frames, and reconstruct \tilde{d}_{v_i} using the same coordinates but in the new frame. We then obtain the deformed location \tilde{v}_i for vertex v_i by adding the new detail vector back to the skeleton point \tilde{p}_i ; that is, $\tilde{v}_i = \tilde{p}_i + \tilde{d}_{v_i}$.

A drawback of this local-frame based scheme is that the resulting eigen-skeleton becomes very thin near the extremities, with a lot of small features collapsing together, when very few eigenvectors are used. This leads to poor normal estimation around sharp features. See the dragon foot in Figure 6.7(a). To overcome this difficulty, we compute two sets of new detail vector \tilde{d}_{v_i} : one is obtained by using the local-frames as described above, denoted by $\tilde{d}_{v_i}^{(1)}$; and the other, denoted by $\tilde{d}_{v_i}^{(2)}$, is obtained by simply applying the target deformation transformation computed in Step 1 to the original d_{v_i} . The advantage of $\tilde{d}_{v_i}^{(2)}$ is that it tends to preserve local details. However, just using $\tilde{d}_{v_i}^{(2)}$ alone has the problem that the changes around boundary of R are often dramatic. See the body of the dragon in Figure 6.7(b).

To get the best out of both strategies, we obtain a final detail vector \tilde{d}_{v_i} by interpolating between the two detail vectors $\tilde{d}_{v_i}^{(1)}$ and $\tilde{d}_{v_i}^{(2)}$. In particular, we assign a large weight to the local-frame based detail vector (i.e, $\tilde{d}_{v_i}^{(1)}$) near the boundary of R and diminish away from the boundary both inside and outside R . We do so because we observed that the normal estimation (and hence $\tilde{d}_{v_i}^{(1)}$) is reliable away from the extremities and near the boundary of the region of interest and that $\tilde{d}_{v_i}^{(2)}$ provides accurate recovery of the sharp features. This works for most models commonly used in the real world, although it is theoretically possible for a mesh and its corresponding skeleton to be very thin even in regions away from the extremities. The details of

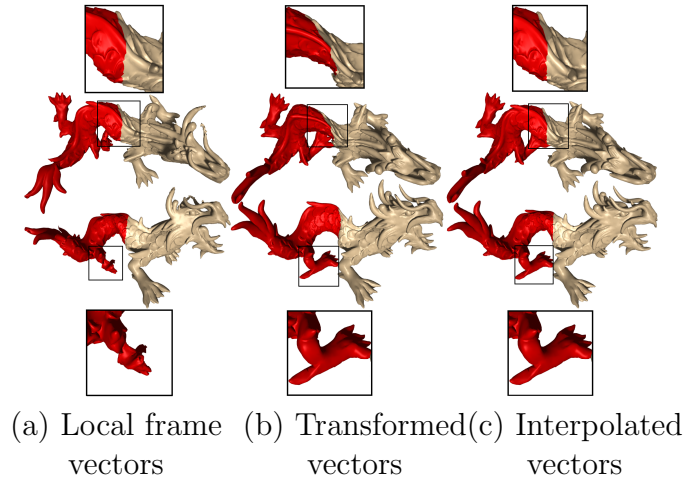


Figure 6.7: Adding details back to the dragon

this interpolation scheme are described in Section 6.3.1. Figure 6.7 shows results for recovering the details of the deformed dragon using each individual strategies (a, b) and using the integrated strategy (c).

6.3 Implementation

6.3.1 Recovery details

For interpolating the detail vectors, we need to assign a weight to each vertex which should depend on how far it is from the boundary of the region of interest. To do this, we need to (1) identify the boundary ∂R of the region of interest R ; (2) compute a per-vertex function denoting the distance from the boundary ∂R ; and (3) use this function to assign the interpolation weights.

The boundary vertices are identified by considering all vertices in R and simply choosing the ones whose one-ring neighborhood contains vertices that are not in R .

We precompute the one-ring neighborhoods on the original mesh just once to reduce computation time during actual deformation of the mesh.

Next, we first compute the following function for each vertex v_i : $g(v_i) = \min_{v \in \partial R} d_g(v_i, v)$, where ∂R is the set of boundary vertices of the region of interest R , and $d_g(v_i, v)$ denotes the geodesic distance between two vertices. Again, we precompute the all-pair geodesic distance matrix once for the original mesh and use it subsequently for all deformations.

Once we have g , we find the approximate diameter of R as $\text{diam}R = \max_{v \in R} g(v)$. We use the diameter to compute two cutoff values $\delta_1 = \frac{\text{diam}R}{8}$, and $\delta_2 = \frac{\text{diam}R}{4}$. The interpolation weights are then computed as:

$$w(v_i) = \begin{cases} 1 & \text{if } g(v_i) < \delta_1 \\ 0 & \text{if } g(v_i) > \delta_2 \\ \frac{\delta_2 - g(v_i)}{\delta_2 - \delta_1} & \text{otherwise} \end{cases}$$

The final detail vector at each vertex v_i is then

$$\tilde{d}_{v_i} = \left(w(v_i) \cdot \frac{\tilde{d}_{v_i}^{(1)}}{\|\tilde{d}_{v_i}^{(1)}\|} + (1 - w(v_i)) \cdot \frac{\tilde{d}_{v_i}^{(2)}}{\|\tilde{d}_{v_i}^{(2)}\|} \right) \cdot \|\tilde{d}_{v_i}^{(1)}\|$$

Note that the two detail vectors $\tilde{d}_{v_i}^{(1)}$ and $\tilde{d}_{v_i}^{(2)}$ have the same length. The above formula simply interpolates their directions to obtain \tilde{d}_{v_i} . Intuitively, the closer a point is to the boundary ∂R of the region of interest, the larger role the local-frame detail vector $\tilde{d}_{v_i}^{(1)}$ plays to guarantee smooth transition. When a point is far from ∂R , the skeleton tends to be much thinner, and in this case we rely more on the transformation-based detail vector $\tilde{d}_{v_i}^{(2)}$ to reconstruct \tilde{d}_{v_i} .

6.3.2 Choice of number of eigenvectors

The eigenvectors capture details at different scales. Consequently, the use of different number of eigenvectors for deformation causes changes at different scales.

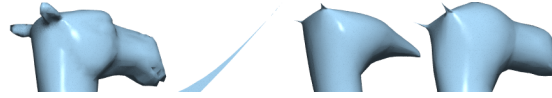


Figure 6.8: Far Left: head of camel. Right: Eigen-skeleton of the head of the camel constructed using 8, 50 and 300 eigenvectors, respectively.

In general, the eigen-skeleton created with only the top few eigenvectors causes shape changes at a global level. To capture local changes, we need a larger number of eigenvectors. Specifically, if the region of interest R is small, then we need more eigenvectors to build the skeleton so that R is reconstructed reasonably well in this skeleton and the change of the corresponding coordinate weights are sufficient to deform R . See Figure 6.8 where if we choose too few eigenvectors, the eigen-skeleton of the ear collapses into roughly a point, and cannot represent the ear at all. Since deformation is computed for the eigen-skeleton, the deformation of the ear cannot be described by such a skeleton. Using more eigenvectors we can capture the ear in the skeleton and further deform it.

On the other hand, if the region of interest is large, the change usually needs to be spread over a large area. If we now choose too many eigenvectors, minimizing the energy function in Step-2 tries to preserve local details of the eigen-skeleton (as there are more terms, i.e, A_j s with large j , describing them). Roughly speaking, the optimization of the weights of the lower eigenvectors is overwhelmed by the large

number of higher eigenvectors. Hence, the deformation of the eigen-skeleton returned in Step 2 tends to have some dramatic changes for a few points while trying to preserve local details elsewhere. Therefore in the case of a large region of interest, we need to choose a small number of eigenvectors to build the eigen-skeleton so that the weight for global deformation is emphasized.

In summary, the number of eigenvectors n_{ev} to be used to reconstruct the eigen-skeleton should be chosen based on the size of R , the region of interest. At the same time, it turns out that the deformation returned by our algorithm is rather robust with respect to n_{ev} , as long as n_{ev} is within a reasonable range. We thus use the following simple strategy to decide n_{ev} . First, compute $\delta_3 = \frac{\text{diam}R}{\text{diam}(V \setminus R)}$, where $\text{diam}(V \setminus R) = \max_{v \notin R} g(v)$ is the approximate diameter of the complement of the region of interest. Now choose n_{ev} as:

$$n_{ev} = \begin{cases} 8 & \text{if } \delta_3 \geq 0.75 \\ 50 & \text{if } 0.75 > \delta_3 \geq 0.1 \\ 300 & \text{otherwise} \end{cases}$$

This simple strategy works well for all the models we experimented with. However, the user can easily override these defaults to choose their own value for n_{ev} .

6.3.3 Additional modifications

Finally, we observe that since the eigen-skeleton can be rather coarse when n_{ev} is small (8 or 50), the local frame estimation sometimes simply becomes too error-prone on $\mathbf{K}_{n_{ev}}^*$ to recover a smooth shape through the interpolation strategy.

For this reason, we iteratively improve the quality of the eigen-skeleton based on the algorithm introduced in Section 6.2 as follows: Recall that \mathbf{K}_m^* denotes the

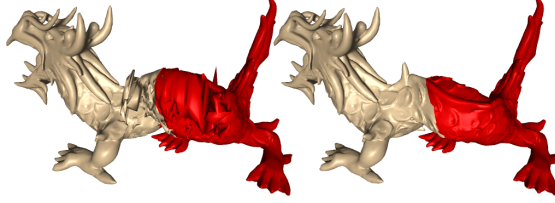


Figure 6.9: Adding details back to the dragon; Left: Directly from eigen-skeleton, Right: After iterative improvement

deformed eigen-skeleton reconstructed using m eigenvectors. Instead of directly recovering the deformed mesh M^* from K_m^* , we first recover another intermediate eigen-skeleton $\tilde{K}_{n'}$ from $K_{n_{ev}}^*$, with $n' > n_{ev}$ using Algorithm 1. This is achieved by using the detail vectors to record the change from $\tilde{K}_{n_{ev}}$ to $\tilde{K}_{n'}$, instead of $\tilde{K}_{n_{ev}}$ to the original mesh M . In particular, in our software, $n_{ev} = 8$ or 50 , and $n' = 300$ (this iterative approach is not needed if $n_{ev} = 300$). The result is an eigen-skeleton that already captures the main deformation, and that also contains sufficient details.

Next, we feed $\tilde{K}_{n'}$ as the coarse-guess configuration to the linear solver in Step 2 to obtain a new deformed eigen-skeleton $K_{n'}^*$. This is done to smooth out any errors that may have been introduced due to poor local frame estimation on $K_{n_{ev}}^*$. We then use this new eigen-skeleton $K_{n'}^*$ and local-frame based detail estimation (instead of the interpolation method) to recover the shape-detail of the deformed mesh M^* . See Figure 6.9 for an example. The final deformation algorithm for the case that $n_{ev} = 8$ or 50 is summarized in the following diagram.

Iteration 1:

$$K_{n_{ev}} \xrightarrow{\text{Step 1}} \tilde{K}_{n_{ev}} \xrightarrow{\text{Step 2}} K_{n_{ev}}^* \xrightarrow[\text{Interpolation based}]{\text{Step 3}} \tilde{K}_{n'}$$

Iteration 2:

$$\tilde{\mathbf{K}}_{n'} \xrightarrow{\text{Step 2}} \mathbf{K}_{n'}^* \xrightarrow[\text{local-frame based}]{\text{Step 3}} \mathbf{M}^*$$

For the case where $n_{ev} = 300$, the original algorithm 1 is applied as before. We remark that potentially one can perform more iterations to improve the deformation quality. However, we observe in practice that two iterations provide a good trade-off between quality and simplicity / efficiency.

6.3.4 Interactivity

To make the software interactive, we precompute the eigenvectors for the mesh along with the matrix Φ since it depends on the original mesh only. Notice that Φ is symmetric and hence can be factored using Cholesky decomposition. We also precompute the all-pairs geodesic distance matrix used for interpolating detail vectors. To maintain interactive rates, we only deform the eigen-skeleton. Once the user is satisfied with the shape of the eigen-skeleton, the details are added. When deforming the eigen-skeleton, the right-hand side (b) for our linear-solver can be quickly computed by multiplying the matrix of eigenvectors with a matrix containing the coarse guess. We can then compute the new coordinate weights by performing simple backward and forward substitutions. The entire process is simple and can be computed in real-time. Adding details can be a little slow (see Table 6.1) since we need to compute the normal for each vertex and hence is separated from the interactive part.

6.4 Results

We implemented our deformation algorithm using C, OpenGL and MATLAB. For comparisons, we wrote our own code for as-rigid-as-possible deformations [86]

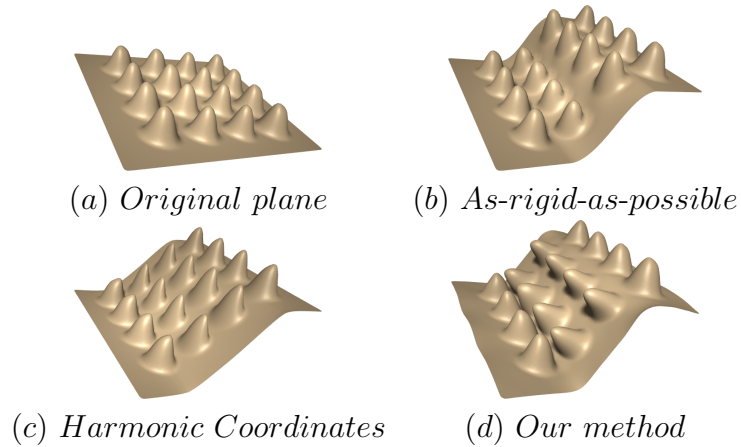


Figure 6.10: Bending a bumpy plane (dense mesh)

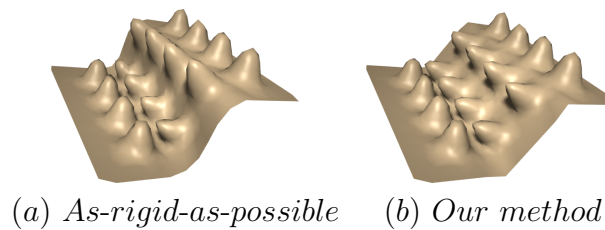


Figure 6.11: Bending a bumpy plane (coarse mesh)

and used the implementation of cage-based deformation using harmonic coordinates provided in open-source software called BLENDER. For spectral surface deformation, we used the code provided by the authors.

Figure 6.3 compares our method with harmonic coordinates and as-rigid-as-possible deformations. For as-rigid-as-possible deformation, red dots denote the fixed vertices, while yellow dots represent the vertices that are moved. The partial cages used for deforming using harmonic coordinates are depicted using black edges. For our method, the red portions are the regions of interest. Figures 6.10 and 6.11 show the results

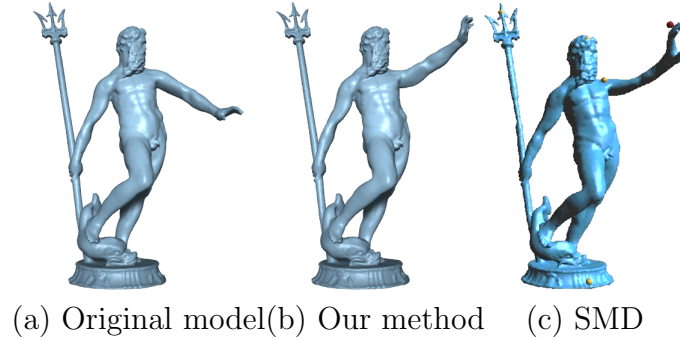


Figure 6.12: Moving the arm of Neptune using our method and spectral mesh deformation (SMD)

of bending a plane with smooth bumps using different techniques. Harmonic coordinates are not able to orient the details correctly while for as-rigid-as-possible, the quality of the deformation seems to depend on mesh density.

The timing data for different stages of our algorithm are presented in Table 6.1. Timings of Step 1 and 2 are coupled together since they are used in each step of interactive deformation. Step 3 is used after the user is satisfied with the shape of the skeleton. Table 6.2 compares the root mean square error in edge lengths. Our method introduces very little error in edge lengths, similar to as-rigid-as-possible approach which aims to optimize such error. Figure 6.13 shows the result of twisting a bar using our method, while Figures 6.15 and ?? shows that we can handle meshes of arbitrary genus.

We also present comparisons with spectral mesh deformation in Figures 6.4, 6.12. The results of spectral mesh deformation are global and cannot be constrained to small regions. For example, in Figure 6.4, even though only the arm was stretched, the entire mesh got deformed in an attempt to preserve the volume and the Laplace operator of the mesh. Our method is able to handle such deformations more naturally.

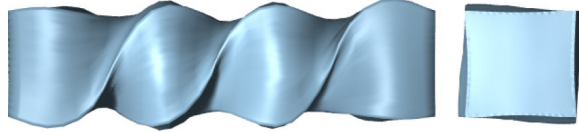


Figure 6.13: Twisting a bar using our method

Model (# vertices)	n_{ev}	Step1 & 2	Step 3
Armadillo (25k)	50	0.018	0.125
dragon (22.5k)	8	0.013	0.161
camel (7k)	8	0.002	0.049
	300	0.012	0.013
plane (10k)	50	0.016	0.061
bar (13.5k)	50	0.017	0.074
children (20k)	50	0.017	0.095

Table 6.1: Timing data (in seconds) for our algorithm

Also, the detail recovery method used in spectral mesh deformation can introduce artifacts into the deformed mesh. For example, in Figure 6.12, although only the arm was moved, the staff of Neptune got slightly deformed as well. Even the hand looks unnatural after the deformation. This happens because 100 eigenvectors are not enough to capture the finer details of the model.

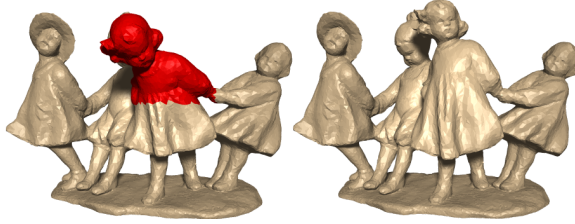


Figure 6.14: Editing the dancing children

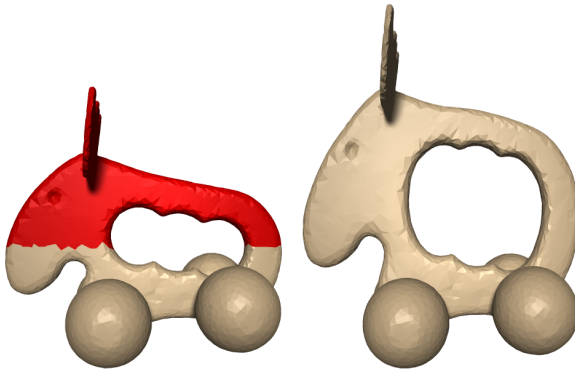


Figure 6.15: Deforming the elk model

6.5 Conclusion and Discussion

In this chapter, we presented an eigen-based framework for mesh deformation that works in real-time without any help from an intermediate structure. It allows isometric deformations as well as non-isometric deformations such as stretching. Several experimental results confirm the effectiveness of the method.

As any other existing technique, one drawback of our method is that it does not guarantee the deformed mesh to be free of self-intersections. Although it is not frequent, self-intersections may happen near the boundary of the region of interest

Model	ARAP	ED
Armadillo (Stretch Arm)	0.0023	0.0022
Armadillo (Bend Knee)	0.001	0.0007
Armadillo (Combined)	0.0052	0.0021
plane	0.0028	0.0022

Table 6.2: Comparison of relative RMS errors in deformations using as-rigid-as-possible (ARAP) and our method (ED)

in case of relatively large deformations. Designing a software that deforms meshes in real-time without self-intersections and intermediate structures remains to be a challenging open question.

Chapter 7: Conclusion

The purpose of this thesis is two-fold: to study the theoretical properties of the mesh Laplace and to provide applications that exploit these properties in the field of computer graphics. In Chapter 3, we provided theoretical proofs about the stability of the mesh Laplace operator under various types of perturbations such as noise and non-isometric topology preserving deformations. We also showed that the eigenstructures of the manifold Laplace are stable as well and change smoothly as the underlying manifold is deformed.

In Chapter 4, we considered modifications that altered the topology of the mesh in a small region and provided theoretical proofs and error bounds on the Gaussian-weighted graph Laplace when a mesh is subjected to such alterations.

In Chapter 5, we used the Heat operator, which is closely related to the Laplace operator, to construct a global shape descriptor that allowed us to match partial or incomplete models in a pose-oblivious manner. Here, we exploited the fact that the Laplace operator does not change when a mesh is deformed isometrically. We also used the fact that the Heat operator can capture details at multiple scales and hence is resilient to the the partial nature of the meshes at small scales.

In Chapter 6, we used the eigenvectors of the Laplace operator to deform meshes. The eigenvectors are capable of capturing the features of a mesh at multiple scales.

The main focus was to project the mesh into a spectral domain using the top few eigenvectors and performing deformations in spectral coordinates. Since low frequency eigenvectors capture coarse details and higher frequencies capture finer details, we formulated our approach so that the globalness of the deformation could be controlled by simply changing the number of eigenvectors used.

Our current results for stability of discrete Laplace operator under topological noise use the Gaussian-weighted graph Laplace operator built from some nearest neighbor graph of a point cloud data that has been uniformly randomly sampled from a manifold. Such samples are common in high dimensions since most data is created by sampling manifold based on some probability distribution. This, however, is not true for 3D data. Point clouds in 3D are often obtained by scanning actual objects. The resulting point clouds are rarely a uniformly random sampling of the original surface.

Our current stability results for the mesh Laplace are only valid so long as there are no topological changes. However, scanned data often has a lot of errors including topological noise such as small holes and handles not present on the original object. Hence, there is a need to extend our results to other variations of the discrete Laplace operator, such as the mesh Laplace, so that the stability results can be more general and practical for three dimensional data.

An important point to note is that our error bounds under topological noise are valid only for the discrete Laplace operator and cannot be extended to the continuous Laplace-Beltrami operators of the manifolds from which the point clouds were sampled. An immediate question is whether it is even possible to give an error bound

on the Laplace-Beltrami operator when topological changes are allowed. Answering this question can be another line of future work.

On the applications side, although there has been a lot of work on automatic mesh segmentation, performing labeled segmentation still remains an open and challenging problem. Labeled segmentation involves segmenting an object and assigning labels to the segments, usually based on a user provided labeled segmentation of a generic object. For example, the user may provide a labeled segmentation of a horse model and ask for a similar segmentation of other animal models. A common way to achieve this is to first segment the target shape and then transfer the labels by matching segments. This, however, requires the initial segmentation to be good. An alternate approach is to use a data-driven approach that learns the labeled segmentation from a training data set, and then performs simultaneous labeling and segmentation.

Since the eigen-structures of discrete Laplace of similar shapes are also similar, it should be possible transfer labels from one shape to another. The similarity of eigen-structures can also make the task of learning labels easier for data-driven approaches.

Deformation transfer is also important area of research in computer graphics which involves transferring deformations from a source mesh onto a target mesh. This is of particular importance in the animation community, since it allows an animator to animate one model and then simply transfer the deformations to other models.

In Chapter 6, we saw how eigenvectors can be used to obtain a coarse representation of a shape by using the top few eigenvectors. We also saw how this skeletal representation can be made more and more detailed by using the higher eigenvectors. A possible application of this can be multi-scale deformation transfer. Observing and transferring the changes in source eigen-skeleton onto the target eigen-skeleton alone

can reduce the problem size from number of vertices in the mesh to the number of eigenvectors used. Furthermore, by increasing or decreasing the number of eigenvectors used, we can control the scale unto which the deformations are observed and transferred.

Bibliography

- [1] Shape retrieval contest. <http://www.aimatshape.net/event/SHREC>.
- [2] I. Baran and J. Popović. Automatic rigging and animation of 3D characters. In *Proc. SIGGRAPH '07*, pages 72:1–72:8.
- [3] M. Belkin and P. Niyogi. Laplacian Eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.
- [4] M. Belkin and P. Niyogi. Convergence of laplacian eigenmaps. In *NIPS*, pages 129–136, 2006.
- [5] M. Belkin and P. Niyogi. Convergence of Laplacian Eigenmaps. Preprint, 2008.
- [6] M. Belkin and P. Niyogi. Towards a theoretical foundation for laplacian-based manifold methods. *Journal of Computer and System Sciences*, 74(8):1289–1308, 2008.
- [7] M. Belkin, J. Sun, and Y. Wang. Discrete laplace operator on meshed surfaces. In *Proceedings of the twenty-fourth annual symposium on Computational geometry*, SCG '08, pages 278–287, New York, NY, USA, 2008. ACM.
- [8] M. Ben-Chen and C. Gotsman. Characterizing shape using conformal factors. In *Proc. Workshop on Shape Retrieval '08*.
- [9] M. Ben-Chen, O. Weber, and C. Gotsman. Variational harmonic maps for space deformation. In *Proc. SIGGRAPH '09*, pages 34:1–34:11.
- [10] M. Botsch and L. Kobbelt. Real-time shape editing using radial basis functions. *Comput. Graph. Forum*, 24(3):611–621, 2005.
- [11] M. Botsch, M. Pauly, M. Gross, and L. Kobbelt. Primo: coupled prisms for intuitive surface modeling. In *Proc. SGP '06*, pages 11–20.
- [12] M. Botsch and O. Sorkine. On linear variational surface deformation methods. *IEEE Trans. on Visualization and Comput. Graph.*, 14(1):213–230, Jan. 2008.

- [13] T. Caelli and S. Kosinov. An eigenspace projection clustering method for inexact graph matching. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26:515–519, April 2004.
- [14] I. Chavel. *Riemannian Geometry: A Modern Introduction*. Cambridge University Press, second edition, 2006.
- [15] F. Chazal, L. J. Guibas, S. Oudot, and P. Skraba. Analysis of scalar fields over point cloud data. In *Proc. 20th ACM-SIAM Sympos. Discrete Algs.*, pages 1021–1030, 2009.
- [16] D.-Y. Chen, X. P. Tian, Y.-T. Shen, and M. Ouhyoung. On visual similarity based 3D model retrieval. *Comput. Graph. Forum*, 22(3):223–232, 2003.
- [17] X. Chen, A. Golovinskiy, , and T. Funkhouser. A benchmark for 3D mesh segmentation. *Proc. SIGGRAPH '09*, pages 73:1–73:12.
- [18] C. Chua and R. Jarvis. Point signatures: A new representation for 3D object recognition. *International Journal of Computer Vision*, 25(1):63–85, 1996.
- [19] A. Clements and H. Zhang. Robust 3d shape correspondence in the spectral domain. In *Proc. of Shape Modeling International*, pages 118–129, 2006.
- [20] D. Cohen-Steiner, H. Edelsbrunner, and J. Harer. Extending persistence using poincaré and lefschetz duality. *Foundations of Computational Mathematics*, 9(1):79–103, 2009.
- [21] D. Cohen-Steiner, H. Edelsbrunner, and D. Morozov. Vines and vineyards by updating persistence in linear time. In *Proc. 22nd Annu. ACM Sympos. Comput. Geom.*, pages 119–126, 2006.
- [22] T. Cox and M. Cox. *Multidimensional Scaling*. Chapman & Hall, London, 1994.
- [23] M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr. Implicit fairing of irregular meshes using diffusion and curvature flow. *Computer Graphics*, 33(Annual Conference Series):317–324, 1999.
- [24] T. K. Dey, K. Li, C. Luo, P. Ranjan, I. Safa, and Y. Wang. Persistent heat signature for pose-oblivious matching of incomplete models. *Comput. Graph. Forum*, 29(5):1545–1554, 2010.
- [25] T. K. Dey, P. Ranjan, and Y. Wang. Eigen deformation of 3d models. *The Visual Computer*, 28:585–595.

- [26] T. K. Dey, P. Ranjan, and Y. Wang. Convergence, stability, and discrete approximation of laplace spectra. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '10, pages 650–663, Philadelphia, PA, USA, 2010. Society for Industrial and Applied Mathematics.
- [27] T. K. Dey, P. Ranjan, and Y. Wang. Stability of weighted graph laplace spectra under topological noise. Under revision, 2012.
- [28] J. Dodziuk. Finite-difference approach to the hodge theory of harmonic forms. *American Journal of Mathematics*, 98(1):79–104, 1978.
- [29] H. Du and H. Qin. Medial axis extraction and shape manipulation of solid objects using parabolic PDEs. In *Proc. ACM Sympos. Solid Modeling Appl. '04*, pages 25–35.
- [30] H. Edelsbrunner and J. Harer. Persistent homology — a survey. In *Twenty Years After*, eds. J. E. Goodman, J. Pach and R. Pollack, AMS., 2007.
- [31] H. Edelsbrunner, D. Letscher, and A. Zomorodian. Topological persistence and simplification. *Discrete Comput. Geom.*, 28:511–533, 2002.
- [32] M. S. Floater. Mean value coordinates. *Comput. Aided Design*, 20(1):19–27, 2003.
- [33] M. S. Floater, G. Kos, and M. Reimers. Mean value coordinates in 3D. *Comput. Aided Design*, 22(7):623–631, 2005.
- [34] T. Funkhouser and P. Shilane. Partial matching of 3D shapes with priority-driven search. In *Proc. SGP '06*, pages 131–142.
- [35] R. Gal and D. Cohen-Or. Salient geometric features for partial shape matching and similarity. *ACM Trans. Graph.*, 25(1):130–150, 2006.
- [36] K. Gebal, J. A. Bærentzen, H. Aanaes, and R. Larsen. Shape analysis using the auto diffusion function. *Comput. Graph. Forum*, 28(5):1405–1413, 2009.
- [37] N. Gelfand, N. J. Mitra, L. J. Guibas, and H. Pottmann. Robust global registration. In *Proc. Symp. Geom. Processing (SGP)*, pages 197–206, 2005.
- [38] S. Guattery and G. L. Miller. Graph embeddings and laplacian eigenvalues. *SIAM J. Matrix Anal. Appl.*, 21:703–723, February 2000.
- [39] A. Gyulassy, V. Natarajan, V. Pascucci, P.-T. Bremer, and B. Hamann. A topological approach to simplification of three-dimensional scalar functions. *IEEE Trans. Vis. Comput. Graph.*, 12(4):474–484, 2006.

- [40] J. H. P. McKean and I. M. Singer. Curvature and the eigenvalues of the Laplacian. *J. Differential Geom.*, 1(1–2):43–69, 1967.
- [41] K. Hildebrandt and K. Polthier. On approximation of the laplacebeltrami operator and the willmore energy of surfaces. In *Proc. SGP '11*, pages 1513–1520.
- [42] K. Hildebrandt, K. Polthier, and M. Wardetzky. On the convergence of metric and geometric properties of polyhedral surfaces. *Geometriae Dedicata*, 123(1):89–112, December 2006.
- [43] K. Hildebrandt, C. Schulz, C. V. Tycowicz, and K. Polthier. Interactive surface modeling using modal analysis. *ACM Trans. Graph.*, 30(5):119:1–119:11, Oct 2011.
- [44] A. Jacobson, I. Baran, J. Popović, and O. Sorkine. Bounded biharmonic weights for real-time deformation. *Proc. SIGGRAPH '11*, pages 78:1–78:8.
- [45] V. Jain and H. Zhang. A spectral approach to shape-based retrieval of articulated 3d models. *Comput. Aided Des.*, 39:398–407, May 2007.
- [46] P. Joshi, M. Meyer, T. DeRose, B. Green, and T. Sanocki. Harmonic coordinates for character articulation. In *Proc. SIGGRAPH '07*, pages 71:1–71:10.
- [47] T. Ju, S. Schaefer, J. Warren, and M. Desbrun. A geometric construction of coordinates for convex polyhedra using polar duals. In *Proc. SGP '05*, pages 181–186.
- [48] Z. Karni and C. Gotsman. Spectral compression of mesh geometry. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, SIGGRAPH 2000, pages 279–286, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.
- [49] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz. Symmetry descriptors and 3D shape matching. In *Proc. SGP '04*, pages 115–123.
- [50] K. G. Kobayashi and K. Ootsubo. T-ffd:free-form deformation by using triangular mesh. In *Proc. Sympos. Solid Modeling Appl. '03*, pages 226–234.
- [51] T. Langer, A. Belyaev, and H.-P. Seidel. Spherical barycentric coordinates. In *Proc. SGP '06*, pages 81–88.
- [52] B. Lévy. Laplace-Beltrami eigenfunctions towards an algorithm that “understands” geometry. In *Proc. Internat. Conf. Shape Model. Applications '06, Invited Talk*.

- [53] B. Levy. Laplace-beltrami eigenfunctions: Towards an algorithm that understands geometry. In *IEEE International Conference on Shape Modeling and Applications, invited talk*, 2006.
- [54] Y. Lipman, D. Levin, and D. Cohen-Or. Green coordinates. In *Proc. SIGGRAPH '08*, pages 78:1–78:10.
- [55] C. Luo, I. Safa, and Y. Wang. Approximating gradients for meshes and point clouds in R^d via diffusion metric. *Computer Graphics Forum*, 28(5):1497–1508, 2009.
- [56] C. Luo, J. Sun, and Y. Wang. Integral estimation from point cloud in d-dimensional space: A geometric view. In *Proc. 25th Annu. ACM Sympos. Comput. Geom.*, 2009.
- [57] R. MacCracken and K. I. Joy. Free-form deformations with lattices of arbitrary topology. In *Proc. SIGGRAPH '96*, pages 181–188.
- [58] U. F. Mayer. Numerical solutions for the surface diffusion flow in three space dimensions. *comput. Appl. Math*, 20(3):361–379, 2001.
- [59] M. Meyer, M. Desbrun, P. Schröder, and A. H. Barr. Discrete differential geometry operators for triangulated 2-manifolds. In *Proc. VisMath'02*, 2002.
- [60] M. Meyer, M. Desbrun, P. Schröder, and A. H. Barr. Discrete differential-geometry operators for triangulated 2-manifolds. In *VisMath'02*, 2002.
- [61] N. J. Mitra, L. J. Guibas, J. Giesen, and M. Pauly. Probabilistic fingerprints for shapes. In *Proc. SGP '05*.
- [62] B. Mohar. Some applications of laplace eigenvalues of graphs. In *Graph Symmetry: Algebraic Methods and Applications, volume 497 of NATO ASI Series C*, pages 227–275. Kluwer, 1997.
- [63] B. Mohar and S. Poljak. Eigenvalues in combinatorial optimization. In *IMA Volumes in Mathematics and Its Applications.*, volume 50, pages 107–151. Springer-Verlag, 1993.
- [64] Y. Nakatsukasa. Absolute and relative Weyl theorems for generalized eigenvalue problems. *Linear Algebra and its Applications*, 432(1):242–248, 2010.
- [65] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS*, pages 849–856. MIT Press, 2001.

- [66] R. Ohbuchi, S. Takahashi, T. Miyazawa, and A. Mukaiyama. Watermarking 3d polygonal meshes in the mesh spectral domain. In *Graphics interface 2001*, GRIN'01, pages 9–17. Canadian Information Processing Society, 2001.
- [67] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin. Matching 3D models with shape distributions. In *Proc. SMI '01*, pages 154–166.
- [68] M. Ovsjanikov, A. M. Bronstein, M. M. Bronstein, and L. J. Guibas. Shape google: a computer vision approach to invariant shape retrieval. In *Proc. NOR-DIA (ICCV Workshops) '09*, pages 320–327.
- [69] M. Pauly, R. Keiser, and M. Gross. Multi-scale feature extraction on point-sampled surfaces. *Comput.s & Graph.*, 22:281–289, 2003.
- [70] U. Pinkall and K. Polthier. Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics*, 2(1):15–36, 1993.
- [71] M. Reuter. Hierarchical shape segmentation and registration via topological features of laplace-beltrami eigenfunctions. *Internat. J. Computer Vision*, 89(2):287–308, 2010.
- [72] M. Reuter, S. Biasotti, D. Giorgi, G. Patane, and M. Spagnuolo. Discrete Laplace-Beltrami operators for shape analysis and segmentation. *Comput.s & Graph.*, 33(3):381–390, 2009.
- [73] M. Reuter, F.-E. Wolter, and N. Peinecke. Laplace-beltrami spectra as "shape-dna" of surfaces and solids. *Computer-Aided Design*, 38(4):342–366, 2006.
- [74] M. Reuter, F.-E. Wolter, and N. Peinecke. Laplace-beltrami spectra as "shape-dna" of surfaces and solids. *Computer-Aided Design*, 38(4):342–366, 2006.
- [75] G. Rong, Y. Cao, and X. Guo. Spectral surface deformation with dual mesh. In *Proc. Internat. Conf. on Comput. Animation and Social Agents '08*, pages 17–24.
- [76] G. Rong, Y. Cao, and X. Guo. Spectral mesh deformation. *The Visual Comput.*, 24(7-9):787–796, 2008.
- [77] S. Rosenberg. *The Laplacian on a Riemannian Manifold*. Cambridge University Press, 1997.
- [78] R. M. Rustamov. Laplace-Beltrami eigenfunctions for deformation invariant shape representation. In *Proc. SGP '07*, pages 225–233.
- [79] T. W. Sederberg and S. R. Parry. Free-form deformation of solid geometric models. In *SIGGRAPH '86*, pages 151–160.

- [80] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2000.
- [81] P. Shilane and T. Funkhouser. Selecting distinctive 3d shape descriptors for similarity retrieval. In *Shape Modeling International*, 2006.
- [82] P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser. The Princeton shape benchmark. In *SMI '04*.
- [83] A. Shokoufandeh, D. Macrini, S. Dickinson, K. Siddiqi, and S. W. Zucker. Indexing hierarchical structures using graph spectra. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27:1125–1140, July 2005.
- [84] K. Siddiqi, J. Zhang, D. Macrini, A. Shokoufandeh, S. Bouix, and S. Dickinson. Retrieving articulated 3D models using medial surfaces. *Machine Vision and Applications*, 19(4):261–274, 2008.
- [85] O. Sorkine. Differential representations for mesh processing. *Computer Graphics Forum*, 25(4):789–807, 2006.
- [86] O. Sorkine and M. Alexa. As-rigid-as-possible surface modeling. In *Proc. SGP '07*, pages 109–116.
- [87] O. Sorkine, Y. Lipman, D. Cohen-Or, M. Alexa, C. Rössl, and H.-P. Seidel. Laplacian surface editing. In *Proc. SGP '04*, pages 179–188.
- [88] J. Sun, M. Ovsjanikov, and L. J. Guibas. A concise and provably informative multi-scale signature based on heat diffusion. In *Proc. SGP '09*, pages 1383–1392.
- [89] J. W. Tangelder and R. C. Veltkamp. A survey of content based 3D shape retrieval methods. *Multimedia Tools Appl.*, 39(3):441–471, 2008.
- [90] G. Taubin. A signal processing approach to fair surface design. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, SIGGRAPH '95*, pages 351–358, New York, NY, USA, 1995. ACM.
- [91] M. E. Taylor. *Partial Differential Equations: Basic Theory*. Springer-Verlag New York Inc., 1996.
- [92] K. Thangudu. Practicality of laplace operator, 2009. Master Thesis, The Ohio State University, Computer Science and Engineering Department.
- [93] B. Vallet and B. Lvy. Spectral geometry processing with manifold harmonics. *Computer Graphics Forum (Proceedings Eurographics)*, 2008.

- [94] K. Veselić. Spectral perturbation bounds for selfadjoint operators i. *Operators and Matrices*, 2(3):307–339, 2008.
- [95] U. von Luxburg, M. Belkin, and O. Bousquet. Consistency of spectral clustering. *Ann. Statist.*, 36(2):555–586, 2008.
- [96] M. Wardetzky. *Discrete Differential Operators on Polyhedral Surfaces – Convergence and Approximation*. PhD thesis, Freie Universität Berlin, 2006.
- [97] M. Wardetzky. Convergence of the cotangent formula: An overview. In A. I. Bobenko, J. M. Sullivan, P. Schröder, and G. Ziegler, editors, *Discrete Differential Geometry*, pages 89–112. Birkhuser, to appear.
- [98] J. Warren. Barycentric coordinates for convex polytopes. *Advances in Computational Math.*, 6(2):97–108, 1996.
- [99] O. Weber, O. Sorkine, Y. Lipman, and C. Gotsman. Context-aware skeletal shape deformation. pages 265–274.
- [100] G. Xu. Discrete laplace-beltrami operators and their convergence. *Comput. Aided Geom. Des.*, 21(8):767–784, 2004.
- [101] Z. Xu, G. Xu, and J.-G. Sun. Convergence analysis of discrete differential geometry operators over surfaces. In *IMA Conference on the Mathematics of Surfaces*, pages 448–457, 2005.
- [102] S. Yoshizawa, A. G. Belyaev, and H.-P. Seidel. Free-form skeleton-driven mesh deformations. In *Proc. ACM Sympos. Solid Modeling Appl. '03*, pages 247–253.
- [103] H. Zhang, A. Sheffer, D. Cohen-Or, Q. Zhou, O. van Kaick, and A. Tagliasacchi. Deformation-drive shape correspondence. *Computer Graphics Forum*, 27(5):1431–1439, 2008.
- [104] H. Zhang, O. van Kaick, and R. Dyer. Spectral mesh processing. *Computer Graphics Forum*, *accepted*, 2009.
- [105] K. Zhou, J. Huang, J. Snyder, X. Liu, H. Bao, B. Guo, and H.-Y. Shum. Large mesh deformation using the volumetric graph laplacian. *ACM Trans. Graph.*, 24(3):496–503, 2005.