January 2019

3d Surface Registration Using Geometric Spectrum Of Shapes

Hajar Hamidian
Wayne State University, hamidian.nasim@gmail.com

3D SURFACE REGISTRATION USING GEOMETRIC SPECTRUM OF SHAPES

by

HAJAR HAMIDIAN

DISSERTATION

Submitted to the Graduate School

of Wayne State University,

Detroit, Michigan

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

2019

MAJOR: COMPUTER SCIENCE

Approved By:

_____

Advisor Date

_____

_____

_____

_____

DEDICATION

To my FATHER and MOTHER and my partner, PAUL

ACKNOWLEDGMENTS

First of All, I thank my advisors Dr. Jing Hua and Dr. Farshad Fotouhi for their continued support throughout my PhD. I have learned a lot from Dr. Hua: conducting research, writing good papers, and presenting my ideas. I am grateful for all his knowledge and his enthusiasm to teach me. I thank Dr. Fotouhi, who is like a father to me. He always supported me in all the stages of my PhD and always reminded me that pursuing PhD is not all about studying, but learning how to treat other people.

I thank my Father and Mother who love me, believe in me, and support me in every moment of my life. They taught me how to give love without asking for it and how to support loved ones unconditionally.

Special thanks to my partner, Paul Janiczek, who is my Refigh (close friend) in all moments of happinesses and sadness, health and sickness. Without him, I don't know if I could finish my PhD. I gratefully appreciate his unconditional love and support.

My love goes out to my dear friends Abdelrahman Hassane, Khayyam Hashmi, Erfan Najmi, and Elahe Rashedi who brought joy to many moments of my life and supported me during my PhD.

Finally, I must express my profound gratitude to Dr. Penelope Hale for her tremendous support and care. She encouraged me to follow my heart and helped me to know myself and be myself.

Last but not the least, I thank Dr. Hamid Soltanian-Zadeh, Dr. Shiyong Lu, and Dr. Zichun Zhong for serving as my committee and providing invaluable feedback.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1: INTRODUCTION

Shape registration is one of the important research topics for biomedical research and clinical diagnosis. Its application ranges from analyses of cardiac deformations [2] to brain structures deformations cause by diseases such as Epilepsy [33] or Alzheimer [15]. Considering that the deformations of most organs such as heart or brain structures are non-isometric, it is very difficult to find the correspondence between the shapes before and after deformation, and therefore, very challenging for diagnosis purposes.

For solving these challenges, there are two categories of methods for detection and categorization of 3D shape deformation: spatial registration methods and spectral methods. Spatial registration methods usually require well-defined features or landmarks to map two shapes [66, 56]. It becomes even more challenging when the landmarks are difficult to define in certain 3D shapes, such as hippocampus, heart, etc. Iterative Closest Point (ICP) method introduced by Besl and Mckay in [4] is one of the popular approaches in spatial registration-based methods. In this approach, the initial transformation for global matching is first estimated and then the closest points are found by minimizing the distance between two shapes. In spite of the simplicity of the algorithm, it is computationally costly to identify the closest points and it does not converge very fast. The ICP method has been used in many research areas such as multi-modality image registration [54]. They tried to find the most-likely correspondence between two shapes using principal direction tree search. Han et al. [10] enhanced the ICP method for registering the large-scale 3D environment models. They tried to avoid the local minima to reach an optimal registration. According to their experiments, the computational time is around 20 times more than standard ICP. There are many other spatial registration methods that can be applied on 3D shapes for medical applications, such as Flirt [19] and surface parameterization-based approaches [65]. These methods use atlases and landmarks as well.

Spectral methods, on the other hand, do not need any landmarks. Shape spectrum, inspired by Fourier transform in signal processing [43], is another method to represent and differentiate shapes. This method was applied on graphs earlier. Considering a discrete meshes as a graph, shape spectrum is defined by a Laplacian matrix of the vertices and their connections. Using the concept of Fourier transform, the eigenvalues of the Laplacian matrix define the spectrum of the graph, and the eigenfunctions are the orthogonal bases. Therefore, the functions defined on graphs can be projected to the orthogonal bases and analysed in the spectrum domain. Karni and Gotsman [20] used this projection for smoothing and mesh comparison purposes. Jain and Zhang [18] employed the extended version of this method for shape registration in the spectrum domain. However, the Laplace spectrum approach focuses more on the connectivity of the graph which may lead to distorted mappings [64]. Along this direction, Reuter [43] and Lévy [28] defined a shape spectrum approach with the Laplace-Beltrami operator on a manifold and employed the eigenvalues and eigenvectors as a global shape descriptor [41, 39]. Shape spectrum is invariant to isometric deformations and different triangulations. Also, the computing time is affordable and it can reveal the fine characteristics of the shape. The eigenvectors are orthogonal basis functions; therefore, the shape can be projected to the orthogonal bases and then analyzed and reconstructed using these bases [51]. As the geometry changes, the spectrum of the shape changes as well. Because of these important advantages of Laplace-Beltrami spectrum, this method is used by lots of studies and many shape analysis approaches such as shape recognition and shape matching [37]. Some studies employed the spectrum of this operator to classify and differentiate shapes [26, 23]. However, the spectrum through these methods can only show the global difference between shapes and cannot map and quantify the non-isometric shape differences due to the lack of non-isometric registration with spectrum.

There are some other methods, besides the spectral and ICP-like approaches. One of these approaches is diffeomorphism, which usually yields global optima. For example, Windheuser et al. [58] proposed a framework for computing an elastic orientation-preserving matching of non-rigid 3D shapes. Some other approaches are based on minimizing metric distortion, such as [6, 46]. There also exists some works [7, 11, 12] for visualization of shape deformations.

In order to address the above challenging problems, we present two new spectral based methods for quantifying and visualizing the deformations and point to point correspondence between surface shapes through the variation of shape spectrum. We also integrate all these methods in a workflow system in order to utilize the execution of our methods using web browsers. There are three major contributions presented in this work.

- We present a novel approach based on spectral geometry to quantify and visualize non-isometric deformations of 3D surfaces by mapping two manifolds. The proposed method can determine multi-scale, non-isometric deformations through the variation of Laplace-Beltrami spectrum of two shapes. Given two triangle meshes, the spectra can be varied from one to another with a scale function defined on each vertex. The variation is expressed as a linear interpolation of eigenvalues of the two shapes. In each iteration step, a quadratic programming problem is constructed, based on our derived spectrum variation theorem and smoothness energy constraint, to compute the spectrum variation. The derivative of the scale function is the solution of such a problem. Therefore, the final scale function can be solved by integral of the result from each step, which, in turn, quantitatively describes non-isometric deformations between two shapes. This method can detect the shape deformation but cannot detect the point to point correspondence between two shapes.

- We extend our method to use eigenvectors in some features points in addition to eigenvalues to find the point to point correspondence between two shapes. In order to register two surfaces, we map both eigenvalues and eigenvectors of the Laplace-Beltrami of the shapes by optimizing an energy function. The function is defined by the integration of a smooth term to align the eigenvalues and a distance term between the eigenvectors at feature points to align the eigenvectors. The feature points are generated using the static points of certain eigenvectors of the surfaces. By using both the eigenvalues and the eigenvectors on these feature points, the computational efficiency is improved considerably without losing the accuracy in comparison to the approaches that use the eigenvectors for all vertices. The variation of the shape is expressed using a scale function defined at each vertex. Consequently, the total energy function to align the two given surfaces can be defined using the linear interpolation of the scale function derivatives. Through the optimization the energy function, the scale function can be solved and the alignment is achieved. After the alignment, the eigenvectors can be employed to calculate the point to point correspondence of the surfaces. Therefore, the proposed method can accurately define the displacement of the vertices.

- We present a workflow system named DATAVIEW that implement all above methods and give the user the accessibility to integrate, customize and execute all these methods and can execute up to 64 task runs in parallel. For employing this workflow system, users do not require to install any software tools and can create, save, share, reuse, and run their workflows only using a web browser without knowing the implementation details of the service. For employing the DATAVIEW workflow system, we extend this workflow system by integrating the software tools we used in this study, to this platform which is one of the challenges of this research. Also, in order

to provide processing the data in parallel, we integrate high-end computing resource like grids to the system to speed up the processing, which is another challenge for this work.

This dissertation proposal is organized as follows:

Chapter 2 presents definitions and detailed description of shape spectrum. This provides a mathematical foundation on which the remainder of the chapters will be based.

Chapter 3 presents our first innovative method to define the deformation of the shape using the variation of the eigenvalues of the Laplace-Beltrami operator of the shape.

Chapter 4 presents our second innovative approach that can calculate the point to point correspondence between two shapes using the eigenvalues and eigenvectors variation of the shape.

In chapter 5, we demonstrate the workflow system that integrate our methods and give the users the capability of running our codes using their web-browsers without installing any software tools.

Chapter 6 presents the conclusion of our work.

Chapter 2: BACKGROUND

2.1 Related work

Shape spectrum is a method to represent the shape. There is a powerful tool called Laplace-Beltrami (LB) operator that can analyze the intrinsic property of the shape. Employing this operator, Reuter [43] and Lévy [28] defined a shape spectrum approach with the Laplace-Beltrami operator on a manifold and employed the eigenvalues and eigenvectors as a global shape descriptor [41, 39]. The eigenvectors are orthogonal basis functions; therefore, the shape can be projected to the orthogonal bases and then analyzed and reconstructed using these bases [51]. As the geometry changes, the spectrum of the shape changes as well. Some studies employed the spectrum of this operator to classify, register, and differentiate shapes [26, 23, 27, 14]. However, the spectrum through these methods can only show the global difference between shapes and cannot map and quantify the non-isometric shape differences due to the lack of non-isometric registration with spectrum.

There exists other work that employs shape spectrum to match shapes. Rodolà et al. [45] proposed a method based on the Laplace–Beltrami eigenvectors for computing partial functional correspondence between non-rigid shapes that have isometric deformation. Litany et al. [31] extended this study to match partial shapes that undergo topological noise and non-isometric deformation within the same framework. There are some limitations for this method. The main limitation lies in its reliance on good local features to drive the matching process. Shi et al. [52] used the difference between the eigenvectors of two surfaces to generate a conformal mapping, but the method is computationally expensive. While many promising techniques were developed, most of the existing methods can find the corresponding points for the shapes that were under isometric deformation and there is still a lack of a method that can generate the correspondence between points for non-isometric shape structure change in a timely efficient fashion.

In this chapter, we are going to explain the definition of shape spectrum and how to describe a shape using the spectrum of the Laplace-Beltrami operator of the shape.

2.2 Eigenvalues and Eigenfunctions of the Shape

In this work, we use Laplace-Beltrami operator to compute the geometric spectrum of a manifold. Let $f_1 \in C^2$ be a real function defined on a Riemannian manifold M. The Laplace-Beltrami operator $\triangle$ is defined as:

$$\triangle f_1 = \nabla \cdot (\nabla f_1), \tag{2.1}$$

where ∇f_1 is the gradient of f_1 and $\nabla \cdot$ is the divergence on the Manifold M. The Laplace-Beltrami operator is linear differential and can be calculated in local coordinates. Let ψ be a local parametrization of a submanifold of M such that, $\psi : R^n \rightarrow R^{n+k}$, $g_{ij} = < \partial_i\psi, \partial_j\psi >$, $G = (g_{ij})$, $W = \sqrt{\det G}$, and $(g^{ij}) = G^{-1}$, where $i, j = 1, 2, ..., n$, $<,>$ is the dot product and $\det$ is the determinant. The Laplace-Beltrami operator then is defined on the submanifold as $\Delta f_1 = \frac{1}{W} \sum_{i,j} \partial_i (g^{ij} W \partial_j f_1)$. If $M \subset R^2$, the Laplace-Beltrami operator reduces to the Laplacian:

$$\Delta f_1 = \frac{\partial^2 f_1}{(\partial x)^2} + \frac{\partial^2 f_1}{(\partial y)^2}. \tag{2.2}$$

We compute the eigenvalue of the Laplacian equation defined as follows:

$$\triangle f = -\lambda f, \tag{2.3}$$

where the family solution $\{\lambda_i\}$ is a real nonnegative scalar and will result in the corresponding real family functions of $\{f_i\}$ for $i = 0, 1, 2, ...$ The spectrum is defined to be the

eigenvalues arranged increasingly as $0 \leq \lambda_0 \leq \lambda_1 \leq \lambda_2 \leq ... \leq +\infty$. In the case of a close manifold or an open manifold with Neumann boundary condition, the first eigenvalue, λ_0, will always be zero, and the corresponding eigenvector, f_0, is a constant vector. The spectrum is an isometric invariant because it only depends on the gradient and divergence which are dependent only on the Riemannian structure of the manifold.

To solve the differential equations, different methods can be employed such as finite element method (FEM) and discrete differential operator. In [43], Reuter et al. discretized the manifold and calculate the Laplace-Beltrami operator by using FEM. In this work, we use discrete differential operator to solve this problem. In this framework, a 2D manifold is discretized to triangle meshes of $M = (V, E, F)$, where V is the set of vertices, E is the set of edges and F is the set of faces. All triangular faces assume counterclockwise orientation. Each triangular face represents the local manifold. We can define property on each element, e.g., vertex, edge, and face, which is a spatial average around such element. The properties on the vertices are considered as discrete samplings on the manifold. The discrete operators are also defined on each vertex.

Assuming the neighborhood of a vertex is approximated with the area of its Voronoi region, a discrete Laplace-Beltrami operator can be defined with the average value over the area of the Voronoi region. Using this concept, the Laplacian-Beltrami matrix for the vertices of a triangle mesh can be constructed as

$$L_{ij} = \begin{cases} -\dfrac{\cot \alpha_{ij} + \cot \beta_{ij}}{2A_i} & \text{if } i, j \text{ are adjacent,} \\ \displaystyle\sum_k \dfrac{\cot \alpha_{ik} + \cot \beta_{ik}}{2A_i} & \text{if } i = j, \\ 0 & \text{otherwise,} \end{cases} \tag{2.4}$$

where α_{ij} and β_{ij} are the two angles opposite to the edge in the two triangles sharing the edges i, j and A_i is the area of Voronoi region at vertex i. (Figure 2.1). k is the index of triangles within 1-ring neighborhood of the vertex i.

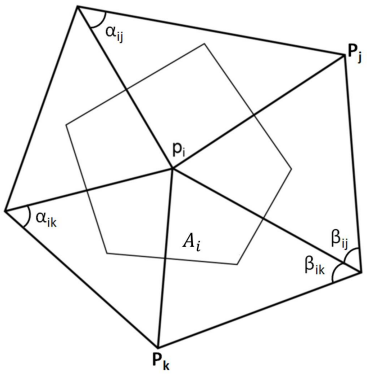


Figure 2.1: The Voronoi region for the vertex P_i within its one-ring neighborhood. Also, a set of angles for calculating Laplace-Beltrami matrix are shown.

Therefore, Equation 2.3 turns to:

$$\mathbf{L}f = \lambda f, \tag{2.5}$$

where f is n dimensional vector for each λ and represents the function value at each vertex on the mesh. This equation is a generalized eigenvalue problem and is solved numerically by constructing a sparse matrix $\mathbf{W}$ and a diagonal matrix $\mathbf{S}$ such that:

$$W_{ij} = \begin{cases} -\dfrac{\cot\alpha_{ij} + \cot\beta_{ij}}{2} & \text{if } i,j \text{ are adjacent,} \\ \displaystyle\sum_k \dfrac{\cot\alpha_{ik} + \cot\beta_{ik}}{2} & \text{if } i = j, \\ 0 & \text{otherwise,} \end{cases}$$

and $S_{ii} = A_i$. Thus, the Laplace Matrix $\mathbf{L}$ is decomposed as $\mathbf{L} = \mathbf{S}^{-1}\mathbf{W}$ and the generalized eigenvalue problem can be presented as:

$$\mathbf{W}f = \lambda \mathbf{S}f. \tag{2.6}$$

$\mathbf{W}$ and $\mathbf{S}$ are symmetric matrices and $\mathbf{S}$ is diagonal and has positive values. λ is the eigenvalue and f presents the eigenvectors. The eigenvalues and eigenvectors are real and eigenvalues are positive values. The eigenvectors corresponding to different eigenvalues are orthogonal in terms of $\mathbf{S}$ dot product. This can be described as follows:

$$< f_i, f_j >_{\mathbf{S}} = f_i^T \mathbf{S} f_j, \tag{2.7}$$

where f_i and f_j are eigenvectors of Equation 2.6 and can be reduced, respectively, to:

$$< f_i, f_j >_{\mathbf{S}} = 0, i \neq j, \tag{2.8}$$

$$F_1 = \sum_{i=1}^{n} f_i c_i, \tag{2.9}$$

and

$$c_i = < F_1, f_i >_{\mathbf{S}}, \tag{2.10}$$

where F_1 is a real function defined on a Riemannian manifold M. Under this setting, the spectrum, $\{0, \lambda_1, \lambda_2, ..., \lambda_{n-1}\}$, the family of eigenvalues, and $\{f_0, f_1, f_2, ..., f_{n-1}\}$, the family of corresponding eigenvectors, can be calculated.

Chapter 3: SURFACE REGISTRATION USING EIGENVALUES VARIATION

In this chapter, we focus on spectrum alignment of general shapes using the eigenvalue variation in order to quantify the non-isometric deformations between surface shapes. In our approach, shapes are automatically aligned by calculating the metric scaling on both shapes. Our method defines the surface shape deformation by the variation of Laplace-Beltrami spectrum of the shapes and quantifies the multi-scale deformations through the use of different sets of eigenvalues. Compared to the traditional approaches, it can detect and localize small non-isometric deformations in addition to global difference of the shapes, without using any defined landmarks on the manifolds. This is because the spectrum only depends on the intrinsic geometry of the shape and is invariant to spatial translation, rotation, scaling and isometric deformation. This method is computationally affordable and suitable to map surface shapes for non-isometric deformation analysis.

3.1 Related Work

Reuter et al. in [41] discussed that spectrum is invariant to small amount of noise but high level noise or small non-isometic deformation can change the spectrum dramatically. The change of shape spectrum is less studied in the literature. In reality, many deformations, such as heart motion, brain surface development, and so on, break isometry. Hence, applying geometric spectrum methods for analyzing non-isometric deformations is very challenging. Recent approaches [50, 52] showed the shape spectrum can be controlled with a scale function on the Riemann metric. Shi et al. discussed in [50] that the eigenvalues and eigenfunctions change according to the Riemann metric of the manifold. Since their work focused on generating a dense registration, the eigenvalue variation was not studied. In [52], Shi et al. used difference between the eigenfunctions of two sur-

faces to generate a conformal map directly between them. To this end, they minimize the difference between surfaces in the Laplace-Beltrami embedding space using optimization approaches. This study focused on eigenfunction variation and the method was time costly. For instance, for mapping two hippocampal surfaces with 1000 vertices, the procedure took around 20 minutes on a 2.6-GHz Intel Xeon CPU and the maximal memory consumption is around 60 MB.

This chapter presents a new method for quantifying and visualizing the deformations between surface shapes through the variation of geometric spectrum. Given two triangle meshes, the spectra can be varied from one to another with a scale function defined on each vertex. The variation is expressed as a linear interpolation of eigenvalues of the two shapes. In each iteration step, a quadratic programming problem is constructed based on our derived spectrum variation theorem and smoothness energy constraint to compute the spectrum variation. The derivative of the scale function is the solution of such a problem. Therefore, the final scale function can be solved by integral of the result from each step, which, in turn, quantitatively describes non-isometric deformations between the two shapes. Our major contributions in this work can be summarized as follows:

- **We prove a theorem that the shape spectrum is a piecewise analytic to a scale function defined on the Riemann metric of the manifold.** Our theorem holds in both continuous domains and discrete triangle meshes and supports the variation of eigenvalues for non-isometrically deformed shapes. Therefore, it enables a quantitative method for deformation analysis.

- **We present a spectrum alignment algorithm for triangle meshes, supporting non-isometric deformation analysis.** In the discrete domain, the variation of eigenvalues in terms of the scale vector can be turned into a matrix form, which introduces a linear system. Together with the smoothness and local bound constraints, the linear

system can be solved efficiently. After the eigenvalues are matched, the eigenfunction distributions are aligned as well. This means the shape spectrum can be controlled analytically through a scale vector, hence, non-isometric analysis is available within shape spectra.

- **Our developed system verifies the spectrum variation theorem and demonstrates the accuracy and efficiency of the spectral variation algorithm on visualization of non-isometrically deformed shapes.** The applications to biomedical imaging problems show that it is a viable solution for morphometric analysis and visualization in biomedical applications and clinical diagnoses.

3.2 Variation of the Eigenvalues and Eigenfunctions

As mentioned before in this work, we use Laplace-Beltrami operator to compute the geometric spectrum of a manifold. As a result of non-isometric deformation, the eigenvalues and eigenfunctions of the shape dramatically change. On a compact closed manifold M with Riemann metric g, we define shape deformation as a time variant positive scale function $\omega(t) : M \rightarrow R^{+}$ such that $g_{ij}^{\omega} = \omega g_{ij}$ and $d\sigma^{\omega} = \omega d\sigma$, where $\omega(t)$ is nonnegative and continuously differentiable. By definition, the weighted Laplace-Beltrami operator becomes $\Delta^{g^{\omega}} = \frac{1}{\omega}\Delta^{g}$. Consider the ith solution of the weighted eigen problem, this equation can be written as:

$$\Delta^{g^{\omega}} f_i = -\lambda_i f_i, \tag{3.1}$$

or rewritten as

$$\Delta^{g} f_i = -\lambda_i \omega f_i, \tag{3.2}$$

where the eigenfunction f_i is normalized as

$$\int_M f_i^2 d\sigma^\omega = 1 \textbf{ for } i = 0, 1, 2, \ldots, \tag{3.3}$$

and orthogonal to other eigenfunctions, such that:

$$\int_M f_i f_j d\sigma^\omega = 0, j \neq i. \tag{3.4}$$

Next, we will explain and prove two theorems that guarantee the existence of a scale function which aligns shapes with non-isometric deformations.

Theorem 1. *$\lambda_i(t)$ is piecewise analytic and, at any regular point, the t-derivative of $\lambda_i(t)$ is given by:*

$$\dot{\lambda_i} = -\lambda_i \int_M \dot{\omega} {f_i}^2 d\sigma. \tag{3.5}$$

Proof. ω is a nonnegative and continuously differentiable function, and Δ^g is analytic. We can compute the derivative of the eigenvalue in Equation 3.2, and get

$$\Delta^g \dot{f_i} = -\dot{\lambda_i}\omega f_i - \lambda_i \dot{\omega} f_i - \lambda_i \omega \dot{f_i}.$$

Then, we multiply both sides with f_i and take the integral on M to get

$$\int_M f_i \Delta^g \dot{f_i} d\sigma = -\dot{\lambda_i} \int_M \omega {f_i}^2 d\sigma - \lambda_i \int_M \dot{\omega} {f_i}^2 d\sigma - \int_M \dot{f_i} \lambda_i \omega f_i d\sigma,$$

which can be simplified, based on Equations 3.2 and 3.3, as

$$\int_M f_i \Delta^g \dot{f_i} d\sigma = -\dot{\lambda_i} - \lambda_i \int_M \dot{\omega} {f_i}^2 d\sigma + \int_M \dot{f_i} \Delta^g f_i d\sigma.$$

Note that, M is a closed manifold. According to divergence theorem, we can have

$$\int_M f_i \Delta^g \dot{f}_i d\sigma = -\int_M \nabla \dot{f}_i \cdot \nabla f_i d\sigma = \int_M \dot{f}_i \Delta^g f_i d\sigma,$$

hence we get Equation 3.5. □

This theorem can be applied to discrete matrix as well. Assume that $\mathbf{\Omega}$ is a nonnegative, continuously differentiable and diagonal matrix, based on Equation 4.1, a weighted generalized eigenvalue problem can be presented as follows:

$$\mathbf{W} f_i = \lambda_i \mathbf{\Omega S} f_i, \tag{3.6}$$

where λ_i and f_i are ith corresponding solution. The eigenvectors can be normalized as

$$< f_i, f_i >_{\mathbf{\Omega S}} = 1 \textbf{ for } i = 0, 1, 2, \ldots, \tag{3.7}$$

and orthogonal to each other, i.e.,

$$< f_i, f_j >_{\mathbf{\Omega S}} = 0, \textbf{ when } i \neq j. \tag{3.8}$$

Theorem 2. *λ_i is piecewise analytic and, at any regular point, the t-derivative of λ_i is given by:*

$$\dot{\lambda_i} = -\lambda_i {f_i}^T \dot{\mathbf{\Omega}} \mathbf{S} f_i, \tag{3.9}$$

Proof. We can compute the derivative of the eigenvalue equation, Equation 3.6, and get

$$\mathbf{W} \dot{f}_i = \dot{\lambda}_i \mathbf{\Omega S} f_i + \lambda_i \dot{\mathbf{\Omega}} \mathbf{S} f_i + \lambda_i \mathbf{\Omega S} \dot{f}_i,$$

multiply f_i^T from the left to obtain

$$f_i^T \mathbf{W} \dot{f}_i = \dot{\lambda}_i f_i^T \mathbf{\Omega S} f_i + \lambda_i f_i^T \dot{\mathbf{\Omega}} \mathbf{S} f_i + f_i^T \lambda_i \mathbf{\Omega S} \dot{f}_i,$$

and simplify it, based on Equations 3.6 and 3.7, as

$$f_i^T \mathbf{W} \dot{f}_i = \dot{\lambda}_i + \lambda_i {f_i}^T \dot{\mathbf{\Omega}} \mathbf{S} f_i + {f_i}^T \mathbf{W}^T \dot{f}_i.$$

Finally, we obtain Equation 3.9 as $\mathbf{W}$ is symmetric. □

Our theorems show that the spectrum is smooth and analytical to non-isometric local scale deformation. They support the variation of eigenvalues for the alignment of non-isometrically deformed shapes, hence an automatic registration-free method for deformation analysis.

3.3 Algorithm for Computing Spectral Variations

Based on the theorems proved in Section 3.2, this section will detail a discrete algorithm for the alignment of non-isometrically deformed shapes through the variation of eigenvalues. Consider two closed manifolds, M and N, represented with discrete triangle meshes, their first k_1 nonzero eigenvalues and eigenvectors are

$$\lambda_{M_i}, f_{M_i}, \lambda_{N_i}, \text{ and } f_{N_i}, \text{ for } i = 1, 2, \ldots, k_1.$$

To align two shapes we use first k_1 smallest eigenvalues. By increasing k_1, some high frequency deformations may be detected. As we mentioned before, the deformation is not isometric, thus the first k_1 eigenvalues of two manifolds are not the same. In order to align

the first k_1 eigenvalues of N to those of M, a continuous scale diagonal matrix $\mathbf{\Omega(t)}$ is applied on N. $\mathbf{\Omega}$ is an n by n matrix, where n is the number of vertices on N. The element Ω_{ii} at the diagonal is a scale factor defined on each vertex on N. According to Theorem 2, the derivative of each eigenvalue is expressed by those of Ω_{ii} analytically. Thus, the scale matrix $\mathbf{\Omega}$ will introduce a variation and alignment from N to M on eigenvalues. The following will explain the details how to calculate the diagonal matrix $\mathbf{\Omega}$ numerically.

3.3.1 Matrix Eigenvalue Variation

We assume that the eigenvalues of N vary linearly towards those of M. This linear interpolation is represented as:

$$\lambda_i(t) = (1-t)\lambda_{N_i} + t\lambda_{M_i}, t \in [0,1]. \tag{3.10}$$

At the beginning, $t = 0$, and $\lambda_i(0)$ starts as λ_{N_i}, while t reaches 1, $\lambda_i(1)$ aligned to λ_{M_i}. At any regular time $t \in [0,1]$, the derivative is constant value and can be calculated as:

$$\dot{\lambda}_i(t) = \lambda_{M_i} - \lambda_{N_i}, t \in [0,1]. \tag{3.11}$$

Combining Equations 3.11 and 3.9, the derivative of each $\lambda_i(t)$ leads to an equation of $\mathbf{\Omega}$ as follows:

$$-\lambda_i(t) f_i(t)^T \dot{\mathbf{\Omega}} \mathbf{S} f_i(t) = \lambda_{M_i} - \lambda_{N_i}, t \in [0,1]. \tag{3.12}$$

Each diagonal element Ω_{ii} represents a scale factor at vertex i on manifold N. $\mathbf{\Omega(0)}$ is an identity matrix on N, and $\mathbf{\Omega(1)}$ aligns the first k_1 nonzero eigenvalues of N to those of M. Although the time derivative of $\mathbf{\Omega}$ can be calculated in Equation 3.11 but solving this equation is not straightforward. We need to transform the individual integration equation into a linear system. We achieve this by extracting the diagonals as vectors $\mathbf{v_\Omega}$ and $\mathbf{v_S}$ and

then employing Hadamard product, which is an element wise matrix product as follows:

$$\mathbf{A} \circ \mathbf{B} = \mathbf{C} \text{ such that } A_{ij} \cdot B_{ij} = C_{ij}. \tag{3.13}$$

Then, Equation 3.9 can be rewritten in a linear form as follows:

$$(\mathbf{v_S} \circ f_i \circ f_i)^T \cdot \mathbf{v_{\dot{\Omega}}} = \frac{\lambda_{N_i} - \lambda_{M_i}}{\lambda_i(t)}, t \in [0, 1]. \tag{3.14}$$

Note that, as the first k_1 eigenvalues are going to be aligned, we can get k_1 independent equations, which lead to a linear system as follows:

$$\mathbf{A} \cdot \mathbf{v_{\dot{\Omega}}} = \mathbf{b}, \tag{3.15}$$

where $\mathbf{A}$ is a row stack of $(\mathbf{v_S} \circ f_i \circ f_i)^T$ with k_1 rows and $\mathbf{b}$ is the right side of Equation 3.14.

Considering that practically k_1 is much less than the number of nodes in the mesh, the system is underdetermined and has no unique solution. Thus, suitable constrains are necessary to provide an optimized solution for the linear system.

3.3.2 Smoothness Constraints

In this work, we focus on the global smoothness of the scale factors distributed on N. Consider a scalar function $f \in C^2$ is define on the continuous manifold $< N_c, g >$. The gradient of f describes the local change of f. A smoothness energy of f is defined with the total squared magnitude of the gradient ∇f on N_c as:

$$E = \int_{N_c} \|\nabla f\|^2 d\sigma. \tag{3.16}$$

Note that ∇f is a vector, and the squared magnitude is calculated as a dot product:

$$\|\nabla f\|^2 = \nabla f \cdot \nabla f. \tag{3.17}$$

Then, the integral on N_c becomes

$$E = -\int_{N_c} f \Delta^g f d\sigma. \tag{3.18}$$

At time t, we investigate the scale function $\omega(t)$ and $d\omega|_t$. Then, we can obtain the following smoothness energy:

$$\begin{aligned} E &= -\int_{N_c} (\omega + d\omega)\Delta^g(\omega + d\omega)d\sigma \\ &= -\int_{N_c} d\omega\Delta^g d\omega d\sigma - 2\int_{N_c} \omega\Delta^g d\omega d\sigma - \int_{N_c} \omega\Delta^g\omega d\sigma. \end{aligned} \tag{3.19}$$

On the discrete triangle mesh N, the scale function is a vector $\mathbf{v_\Omega}$, which is the diagonal of matrix $\mathbf{\Omega}$. The integral is a matrix product as follows:

$$\begin{aligned} \mathbf{E} &= < \mathbf{v_\Omega} + \mathbf{v_{\dot\Omega}}, \mathbf{L} \cdot (\mathbf{v_\Omega} + \mathbf{v_{\dot\Omega}}) >_{\mathbf{S}} \\ &= (\mathbf{v_\Omega} + \mathbf{v_{\dot\Omega}})^T \cdot \mathbf{S} \cdot \mathbf{L} \cdot (\mathbf{v_\Omega} + \mathbf{v_{\dot\Omega}}) \\ &= (\mathbf{v_\Omega} + \mathbf{v_{\dot\Omega}})^T \cdot \mathbf{W} \cdot (\mathbf{v_\Omega} + \mathbf{v_{\dot\Omega}}) \\ &= \mathbf{v}_{\dot\Omega}^T \cdot \mathbf{W} \cdot \mathbf{v_{\dot\Omega}} + 2\mathbf{v}_{\Omega}^T \cdot \mathbf{W} \cdot \mathbf{v_{\dot\Omega}} + \mathbf{v}_{\Omega}^T \cdot \mathbf{W} \cdot \mathbf{v_\Omega}. \end{aligned} \tag{3.20}$$

Assume that $\mathbf{v_\Omega}$ is known at each time t and $\mathbf{v_{\dot\Omega}}$ is to be solved in Equation 3.15. $\mathbf{v_{\dot\Omega}}$ is going to minimize the quadratic smooth energy E_q at any time,

$$\mathbf{E}_q = \mathbf{v}_{\dot\Omega}^T \cdot \mathbf{W} \cdot \mathbf{v_{\dot\Omega}} + 2\mathbf{c}^T \cdot \mathbf{v_{\dot\Omega}}, \tag{3.21}$$

where $\mathbf{c} = \mathbf{W} \cdot \mathbf{v}_{\mathbf{\Omega}}$. In order to preserve the physical availability, $\mathbf{v}_{\mathbf{\Omega}}$ must be bounded, i.e., the scale factor cannot be zero or negative; and it cannot be infinite either. We denote a lower bound and an upper bound with $\mathbf{h}_l, \mathbf{h}_u > \mathbf{0}$, where $\mathbf{h}_l$ and $\mathbf{h}_u$ are n dimensional constant vector. $\mathbf{v}_{\dot{\mathbf{\Omega}}}$ must satisfy

$$\mathbf{h}_l \leq \mathbf{v}_{\mathbf{\Omega}} + \mathbf{v}_{\dot{\mathbf{\Omega}}} \leq \mathbf{h}_u. \tag{3.22}$$

This inequality bound can be written into a matrix form:

$$\mathbf{G} \cdot \mathbf{v}_{\dot{\mathbf{\Omega}}} \leq \mathbf{h}, \tag{3.23}$$

where $\mathbf{G}$ is stack of identity matrices as

$$\mathbf{G}_{2n \times n} = \begin{pmatrix} -\mathbf{I}_{n \times n} \\ \mathbf{I}_{n \times n} \end{pmatrix}, \tag{3.24}$$

and $\mathbf{h}$ is a $2n$ dimensional vector as

$$\mathbf{h}_{2n \times 1} = \begin{pmatrix} \mathbf{v}_{\mathbf{\Omega}} - \mathbf{h}_l \\ \mathbf{h}_u - \mathbf{v}_{\mathbf{\Omega}} \end{pmatrix}. \tag{3.25}$$

The linear system (Equation 3.15), smoothness constraint (Equation 3.21), and constant bound (Equation 3.23) introduce a quadratic programming problem at each time t. Assume the eigenvalues and eigenvectors are known at each time t, the derivative of the scale matrix $\dot{\mathbf{\Omega}}$ is the solution of such quadratic programming.

3.3.3 Linear Integration

In Section 3.3.2, we have shown that at each time t, the derivative of the scale function can be calculated as a solution of a quadratic programming. In the initial state, the scale function is identity which starts from manifold N itself. The final scale matrix can be achieved by the following integral:

$$\mathbf{\Omega}(\mathbf{1}) = \mathbf{I} + \int_0^1 \dot{\mathbf{\Omega}} dt. \tag{3.26}$$

This equation aligns the first k_1 nonzero eigenvalues from N to M. This integration is discretely approximated with an iteration. We divide the time interval $[0, 1]$ into K steps which we index them as q. In the initial state, $q = 0$, $\mathbf{\Omega}(0) = \mathbf{I}$, $\lambda_i(0) = \lambda_{N_i}$, and $f_i(0) = f_{N_i}$. In order to reduce the numerical error, we reinitialize the problem at the beginning of each step $q = 0, 1, \cdots, K$. Instead of aligning λ_{N_i} to λ_{M_i}, we align $\lambda_i(q)$ to λ_{M_i}. $\lambda_i(q)$ and $f_i(q)$ are re-calculated with Equation 4.1 and current $\mathbf{\Omega}(q)$. The result $\dot{\mathbf{\Omega}}(q)$ can be used to calculate $\mathbf{\Omega}(q+1)$ as follows:

$$\mathbf{\Omega}(q+1) = \mathbf{\Omega}(q) + \frac{1}{K-q}\dot{\mathbf{\Omega}}(q). \tag{3.27}$$

After K steps, the desired $\mathbf{\Omega}(K)$ will be achieved and manifold M will be aligned to manifold N. The summary of the algorithm can be found in Algorithm 1.

3.4 Experiments and Applications

The proposed algorithm and system are implemented using Python and C++ on a 64-bit Linux platform. The experiments are conducted on a computer with an Intel Core i7-3770 3.4 GHz CPU and 8 GB RAM. We apply our algorithm on 2D manifolds, represented

Algorithm 1 Eigenvalue Alignment

Input: Closed 2D manifolds N and M, represented by triangle meshes, and constant k_1;
Output: Diagonal weight matrix $\mathbf{\Omega}(q)$ on N, aligning first k_1 nonzero eigenvalues from N to M;
Initialize $\mathbf{\Omega}(0) \leftarrow \mathbf{I}$, calculate matrices $\mathbf{W}$ and $\mathbf{S}$ on N, and $\lambda_{M_i}, f_{M_i}, \lambda_{N_i},$ and $f_{N_i},$ for $i = 1, 2, \ldots, k_1$;
while $q < K$ **do**
  Calculate $\lambda_i(q), f_i(q),$ for $i = 1, 2, \ldots, k_1$ using Equation 4.1 with $\mathbf{\Omega}(q)$;
  Construct the quadratic programming problem using Equations 3.15, 3.21, and 3.23;
  Solve the quadratic programming problem to get $\dot{\mathbf{\Omega}}(q)$ and calculate $\mathbf{\Omega}(q+1)$;
  $q \leftarrow q + 1$;
end while

with triangle meshes. In the experiments, we use hippocampi extracted from brain MR images and their surface meshes have around 5000 vertices. Besides the vertex number, there are two constants, K iteration and the first k_1 nonzero eigenvalues to be aligned. According to the algorithm described in Section 3, each iteration is an independent quadratic programming problem. Thus, the complexity is linear to the step number K. k_1 determines how many eigenvalues to be re-initialized at the beginning of each step. The algorithm calculates the eigenvalues by iterations with the complexity of $O(n^2)$ to the number of vertices and linear to k_1. The average computing time for around 5000 nodes, with $k_1 = 100$ and $K = 10$, is around 17 seconds. Note that, the larger the K is, the more accurate the approximation is, in terms of the linear interpolation. In practice, we found $K = 10$ is sufficient to get the accurate result with a reasonable computational time. Ideally, including more eigenvalues for alignment can be more accurate. However, the numeric eigenvalue calculation is not reliable on higher indexed eigenvalues, which will bring more unsuitability. It is noted that the unstable eigenfunctions are introduced by the symmetry. This is avoided by applying some preprocessing with existing symmetry analysis algorithms. Our experiments show that the first 100 eigenfunctions carry sufficient geometry information and are also quite reliable. The middle range frequencies provide sufficient geometry information

for the fine deformation. So we usually choose $k_1 = 100$ in our experiments. Details about how we choose K and k_1 in our experiments are provided in Section 3.4.1.

3.4.1 Experiments on Synthetic Data

Sensitivity Analysis

We analyze the sensitivity of our model with respect to the number of iteration (K) and eigenvalues (k_1). To this end, we employ the Bunny model with 3000 vertices and make a bump on the back of the bunny. Figure 3.1 shows the original and target bunnies. The dashed circle shows the bump made on the back of the bunny. We conduct an empirical study to map the original bunny to target one using 30, 50, 80 and 100 eigenvalues and 10 iterations. The results are presented in Figure 3.2 and shown that by increasing the number of eigenvalues, the outcome is better to match the synthetic target results. By comparing the result for 80 and 100 eigenvalues, one can conclude that the results don't change after 80 eigenvalues. Therefore 80 is sufficient for the number of eigenvalues. We conduct another experiment and map the original bunny to the target one using 80 eigenvalues and 1, 5, 10 and 15 iterations, respectively. Figure 3.3 shows the results of this mapping. As one can conclude, the outcome is improved from 1 iteration to 10 iterations and is not improved between 10 and 15. Therefore, 10 is the sufficient number of iteration.

Hippocampus Data

In order to evaluate the efficiency of our method, we synthetically generate some non-isometric deformations based on an initial shape. In this experiment, we use a hippocampus segmented from 3D brain MR images. The surface is then deformed manually to make a non-isometric deformation. Our spectrum alignment is applied on the first 100 nonzero eigenvalues. Note that, no correspondence information is used in the experiments.

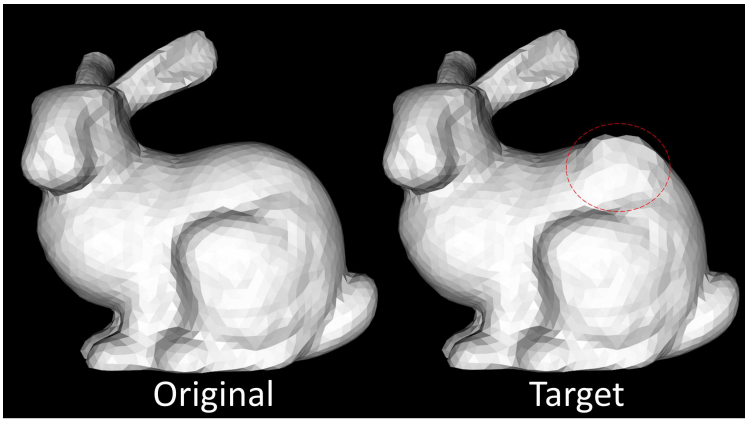


Figure 3.1: We made a bump on the back of the bunny. The original and target bunnies are shown in this picture.

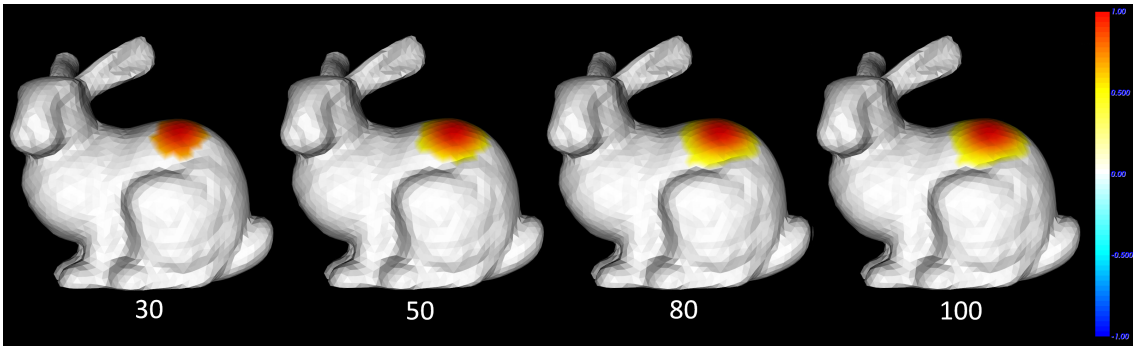


Figure 3.2: The result of mapping the original bunny to the target one in Figure 3.1 using 30, 50, 80 and 100 eigenvalues and 10 iterations.

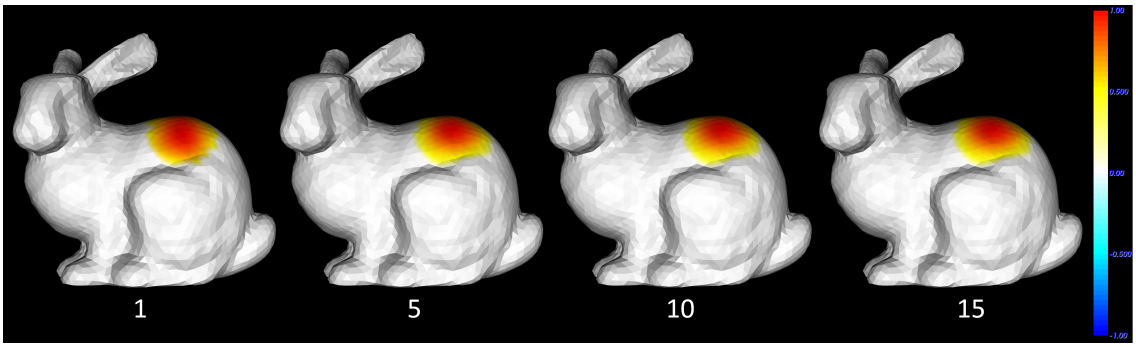
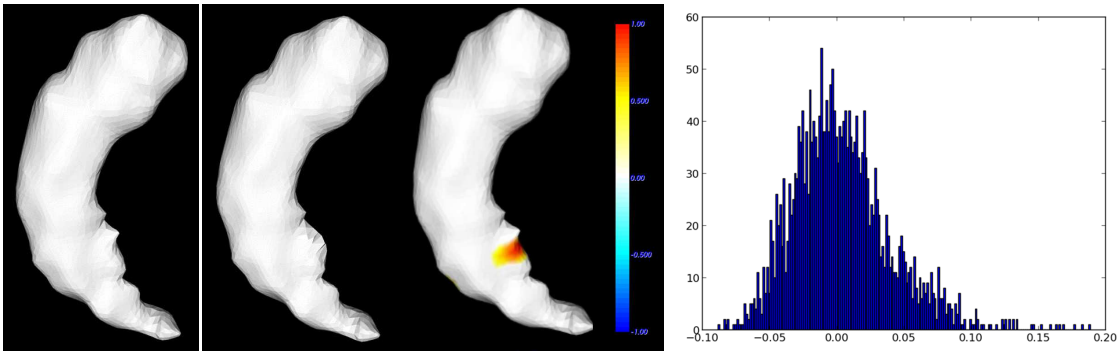


Figure 3.3: The result of mapping the original bunny to the target one in Figure 3.1 using 80 eigenvalues and 1, 5, 10 and 15 iterations.

In the first experiment, we synthetically generate a bump on the original surface. Then we align the original object to the bumped one to obtain the scale function. Figure 3.4a and 3.4b show the original and deformed objects, respectively. Figure 3.4c shows the result of mapping the original shape to the deformed one. The spectrum variation can detect and localize the non-isometric deformation clearly. The red color indicates the located dilating area. In order to see the values of scale function, we also provide the histogram in Figure 3.4d. The histogram has more positive areas than negative, which concludes to the shape expansion when mapping from the shape in Figure 3.4a to the one in Figure 3.4b. It is worthy to mention that we use a threshold to show the deformed area in Figure 3.4c. The threshold value ε is defined as follows:

$$\varepsilon = min(|min(\mathbf{\Omega})|, |max(\mathbf{\Omega})|). \tag{3.28}$$

area A_i which has scale values over the threshold ε indicates the expansion as shown in Figure 3.4c. When area A_i having scale values less than $-\varepsilon$, it indicates the shrinkage.

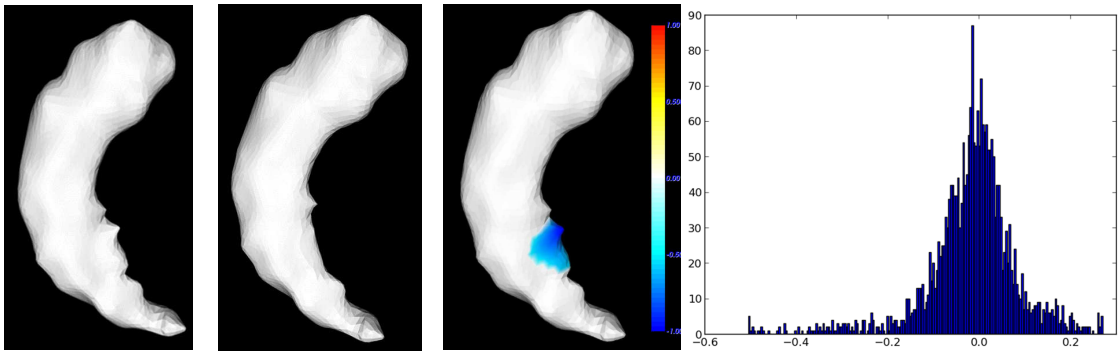


(a) (b) (c) (d)

Figure 3.4: The result of mapping the original 3D object to the synthetic one. (a) shows the original object. (b) is obtained by generating a bump on the original shape. (c) shows the result of mapping shape (a) to the one in (b) where the expansion is determined and located. The histogram of the scale matrix is presented in (d). The histogram has more positive area, which mean the shape has an expansion.

In the second experiment, we shrink one part of the original manifold. Then, we align the original shape to the shrunk one. Figure 3.5a and 3.5b shows the original and deformed shapes. Figure 3.5c shows the results of mapping. As can be seen, the local shrunk region is detected and pinpointed by our method. The blue color shows the contraction of the area. In order to see the distribution of the computed scale function, we present its histogram in 3.5d. The histogram has more negative areas, which concludes to the shape contraction when mapping from the shape in Figure 3.5a to the one in Figure 3.5b.

In the third experiment, we scale the manifold by a factor of 0.5 and align the original shape to the scaled one. Figure 3.6a shows the comparison of the original and scaled objects. Figure 3.6b shows the result of mapping the original shape to the scaled one. As can be seen the whole surface is blue which means it is globally shrunk. The histogram result from this mapping is shown in Figure 3.6c. The center of the histogram is moved to -0.661 (equal to the changes of the Voronoi areas of the two shapes) which shows



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

Figure 3.5: The result of mapping the original 3D object to a modified object generated by shrinking a local area of the original data. (a) shows the original object. (b) shows the modified object generated by shrinking a local part of the original shape. (c) shows the result of mapping the original shape in (a) to the shrunk shape in (b), where shrinkage is localized. The histogram of the computed scale function is presented in (d). The histogram has more negative area, which means the shrinkage of the shape.

the shrinkage of the whole object while there exist almost no local spectral variations across different vertices.

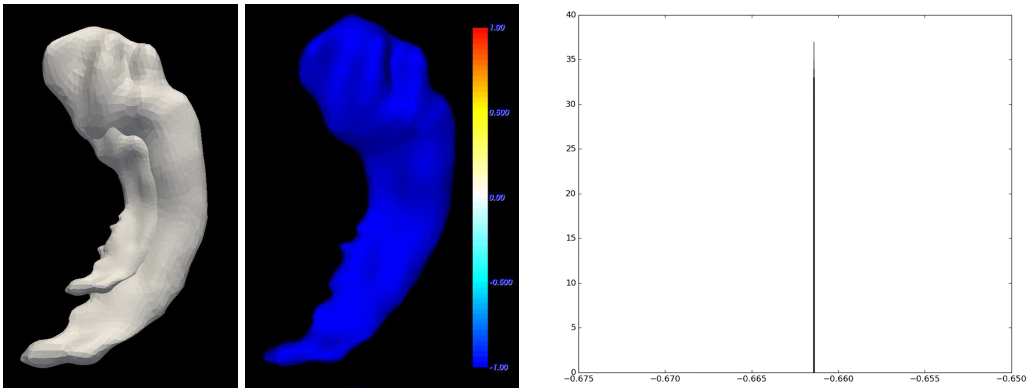
These experiments clearly demonstrate that our method is able to detect and localize non-isometric deformations as well as global deformations.

3.4.2 Applications on Real Patient Imaging Data

To evaluate our method on real imaging data, we have conducted two studies: Epilepsy study and Alzheimer study. In both studies, hippocampus, which is located in temporal lobe of brain, is affected by these diseases.

Epilepsy Study and Diagnosis

Mesial temporal lobe epilepsy (mTLE) is one of the most common types of focal epilepsy. Among mTLE structural abnormalities, hippocampus is one of the most frequent structures that can be affected. As indicated in [17], epilepsy may cause shrinkage of the



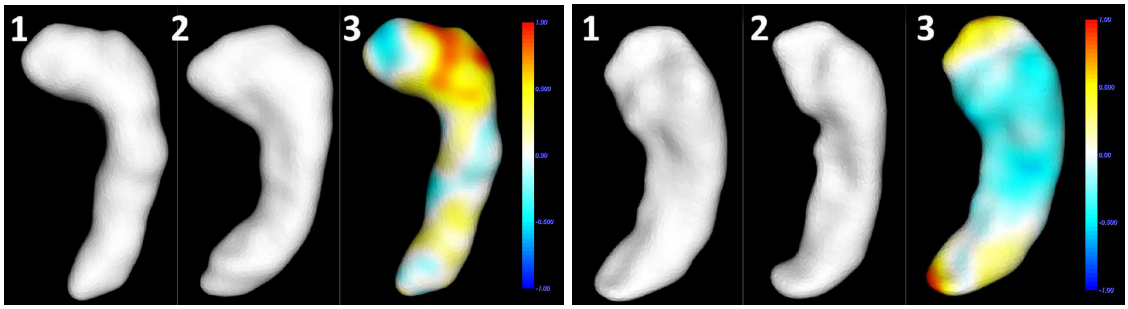
(a) (b) (c)

Figure 3.6: The result of mapping the original 3D object to the object which generated by scaling the original data by a factor of 0.5. (a) shows the original object in comparison to the synthetically scaled one. (b) shows the result from mapping the original shape to the scaled one. The whole object surface is blue which means the object is globally shrunk. The histogram of the scale matrix is presented in (c). The histogram center has moved to -0.661 with very little eigenvalue variations across vertices.

affected hippocampus in comparison to the non-affected one. Note that, traditional voxel-based approaches can determine expansion or shrinkage but cannot localize deformations. We apply our method on twenty TLE patients for localizing and quantifying the shape variation between left and right hippocampi. In our data, half number of the patients are reported to have left defected hippocampus and the other half have abnormality in the right hippocampus. To generate the 3D hippocampus surface, right and left hippocampi are segmented manually using MRIcro Tool from 3D T1 images. Right hippocampi are then mirrored in order to have the same orientation as the left ones. For epilepsy study, we have also applied our spectrum alignment to the first 100 nonzero eigenvalues in order to obtain the scale function.

The abnormal deformations are accurately localized and quantified, which are consistent with clinical findings in the patients' medical records. In Figure 3.7, column 1 and 2 show samples of left and right hippocampi. Column 3 shows the computed scale function distributions on the left hippocampus surface when mapping from the left one to the right. The colors denote the values of scale function in each vertices. Red means dilating, blue means contraction, and white means no distortion. According to the clinical record, Figure 3.7a is for a patient case that has left abnormal hippocampus, therefore, mapping from the left hippocampus to the right displays more expansion (indicated in red), i.e., the left hippocampus is shrunk (diseased) compared to the right, normal one. Figure 3.7b depicts another patient case that has the right defected hippocampus. When mapping from the left hippocampus to the right, the scale distribution displayed on the left hippocampus surface mainly shows the shrinkage (indicated in blue) which indicates the right hippocampus is shrunk (diseased) in comparison to the left hippocampus.

To check how the eigenfunctions vary after changing the eigenvalues, we select the 12th eigenvalue and show the eigenfunctions corresponding to this eigenvalue on the source manifold before and after mapping to the target manifold. Figure 3.8 shows the 12th eigen-



(a) (b)

Figure 3.7: The results of mapping the left hippocampus to right one for two cases. (a) shows a case in which the left hippocampus is affected by epilepsy. (b) shows a case that has right abnormal hippocampus. The first, second, and third columns represent left hippocampus, right hippocampus, and the result of mapping left hippocampus to right one, respectively.

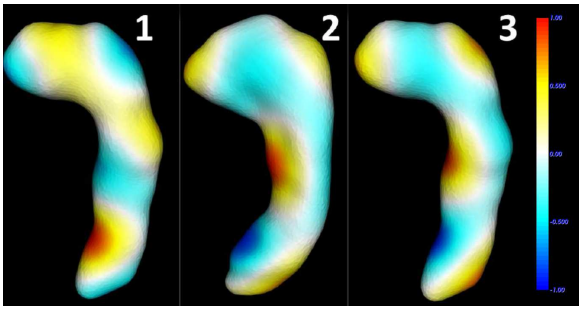


Figure 3.8: The 12th eigenfunction of the original left hippocampus (first column), the original right hippocampus (second column), and the spectrum-aligned left hippocampus (third column). The pattern of the eigenfunction for the left hippocampus shape has been changed in order to map the right one.

function of the original left hippocampus (first column), the original right hippocampus (second column), and the spectrum-aligned left hippocampus (third column). The eigenfunctions are normalized between -1 and 1. The values of eigenfunction at each vertex are expressed with color map, where red means larger value, blue means smaller ones, and white means zero. Comparing the eigenfunction patterns before and after the alignment, a great improvement is obtained and the pattern of the eigenfunction in the source manifold has changed in order to well align to the target manifold.

In order to show the variation of eigenvalues of the manifolds before and after alignment, we list the 2nd to 8th eigenvalues of left hippocampus (before and after mapping) and right hippocampus in Table 3.1. The eigenvalues are normalized by the first nonzero one to remove the scale factor. It can be seen that after applying the spectrum alignment algorithm, the eigenvalues of the source manifold have changed to well align with the target ones.

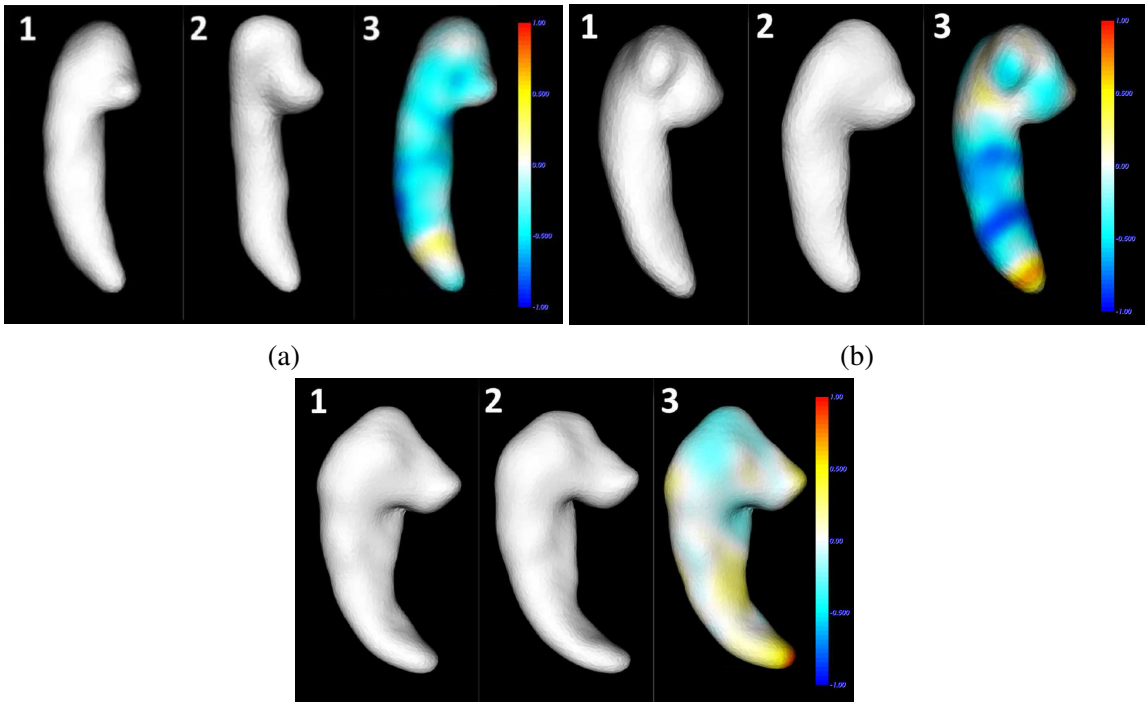
Table 3.1: The result of aligning eigenvalues from the left hippocampus to right one using the same case as in Figure 3.7a.

Manifold	$\lambda_i/\lambda_1, i \in [2, 8]$
Left Hippocampus	3.87, 7.76, 11.93, 14.22, 15.88, 18.49, 20.62
Right Hippocampus	4.36, 7.75, 11.20, 12.62, 16.60, 18.35, 21.73
Aligned Left One	4.36, 7.75, 11.19, 12.62 , 16.59, 18.34, 21.73

Alzheimer Study and Diagnosis

Alzheimer disease (AD) is a brain mis-functionality that is caused by the loss of neurons and neural volume. Hippocampus, a part of the mesial temporal lobe memory system, is vulnerable to damage in the early stage of Alzheimer. Volumetric longitudinal studies [60, 57] using MR images show hippocampal atrophy during time in comparison to healthy cases. In many clinical studies, it was observed that the variation is more in the left hippocampus than the right one and there exists a deformation of CA1 region in hippocampus which can be extended to the subiculum region [29]. In this work, we conduct experiments on the dataset provided by the Alzheimer's Disease Neuroimaging Initiative (ADNI) to show that our shape variation analysis method confirms the aforementioned observation.

We employ ten healthy and ten AD cases aging between 74-80 which have longitudinal study for one year to track and compare the deformation of left and right hippocampi.



(a) (b)

(c)

Figure 3.9: Longitudinal study for Alzheimer disease. Column 1 shows the baseline hippocampus and column 2 shows the hippocampus after 1 year. Column 3 shows the deformation (scale function) results from aligning the baseline hippocampus to the one after one year. (a) and (b) show the left and right hippocampi results for an AD patient, respectively. (c) shows the result for a hippocampus in a normal case.

Our motivating data example consists of ten healthy and ten AD patients which were downloaded from the hippocampal study in ADNI and were segmented using Freesurfer software [**?**]. The 3D object and surface meshes are generated using the method mentioned before and the scale function is obtained by applying our proposed method. According to our results, the AD patients' left and right hippocampi are shrunk after one year. The average of shrinkage for AD cases is more than health cases in both right and left hippocampi. We also confirm that in AD patients the average of shrinkage is more in the left hippocampus than right one. In addition to the global deformation, the local analysis using our method shows that the middle region of hippocampus, called CA1, is affected more severely in both left and right hippocampi. Figure 3.9 shows the results of applying the

proposed method on an AD and healthy cases for left and right hippocampi. The baseline hippocampus is aligned to the hippocampus after one year and the results are shown on the source manifold. As one can see, the left and right hippocampi in AD cases have shrinkage in the middle of the hippocampus. This can be extend to the tail of hippocampus in some cases. Also Figure 3.9 shows that in the normal case the hippocampus has little change after one year. Therefore, our results confirm the clinical observation.

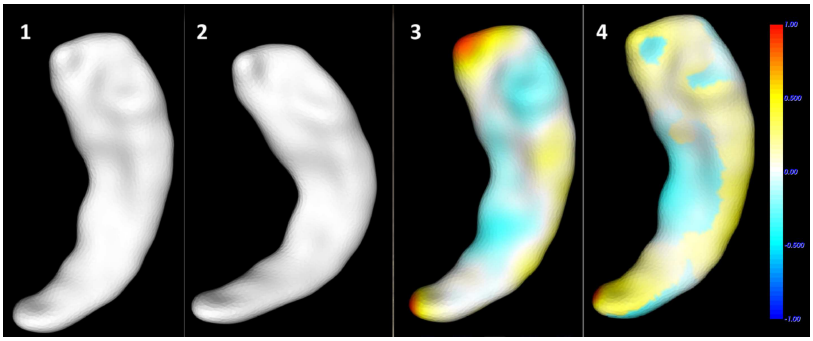
3.4.3 Comparisons to Spatial Registration Methods

In order to further demonstrate the capabilities of our method, we compare the results of our algorithm with the ones employing non-rigid Iterative Closest Point (ICP) algorithm [4]. In the non-rigid ICP method, we fist find the rigid transformation of the source manifold to the target. Then, using the non-rigid registration method, we further register the manifold to the target to build the point-to-point correspondence. The displacement of each corresponding point pairs between the rigid registered source manifold and the target manifold presents the deformation distribution. In order to compare two methods, we synthetically deform a shape by stretching and bending the geometry from both top and bottom sides of the shape as shown in Figure 3.10. Column 1 and 2 show the original and the synthetically deformed shapes, respectively. Figure 3.10a and 3.10b show different views of the shapes. Column 3 and 4 show the results of spectrum alignment and non-rigid ICP methods, respectively. From the results, we can see that our method detects the expansions on both ends while the non-rigid ICP method detects the shape deformation at the bottom end instead of both ends.

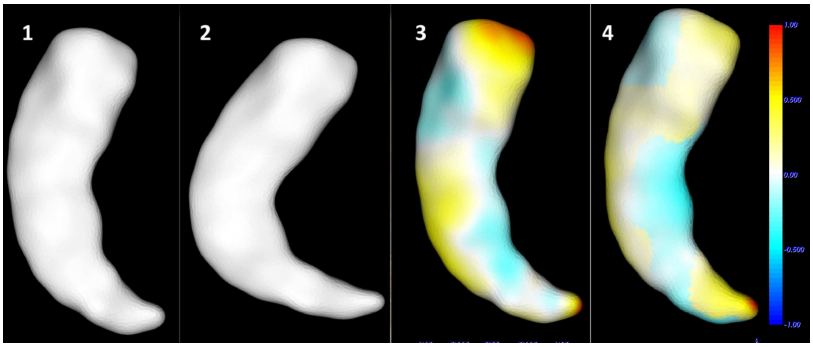
In order to further compare two methods, we also make an expansion in one part of the synthetically deformed shape as in Figure 3.10 and compare the results. Figure 3.11 shows the original and synthetically deformed shapes in column 1 and 2, respectively. The

locally deformed area is marked with a red circle. The results of the spectrum alignment and non-rigid ICP registration method are shown in column 3 and 4. As we can see, our spectrum alignment method detects the deformation more accurately than the non-rigid ICP method. The local expansions, including previously deformed ends in Figure 3.10, are exactly localized.

Our method can also detect the deformation of the shapes using different number of eigenvalues. This allows us to analyze shape deformation using lower or higher frequency bands. Figure 3.12 shows the results of spectrum alignment of a shape in 3.12a to one

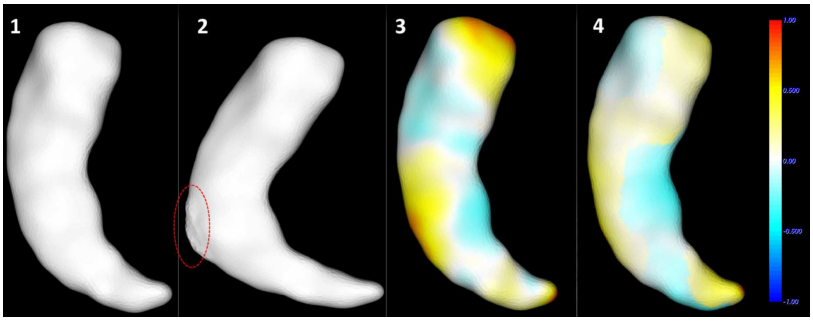


(a)

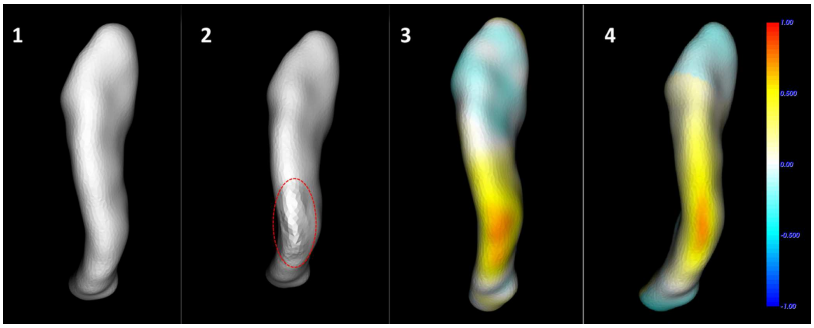


(b)

Figure 3.10: The results of mapping the original shape to the synthetically deformed shape using our method and non-rigid ICP. Column 1 shows the original shape. Column 2 shows the synthetic data which is bent and stretched at both ends. Column 3 and 4 show the results of mapping the shape in column 1 to column 2 using our method and non-rigid ICP technique, respectively. (a) and (b) show the different views of the shape.

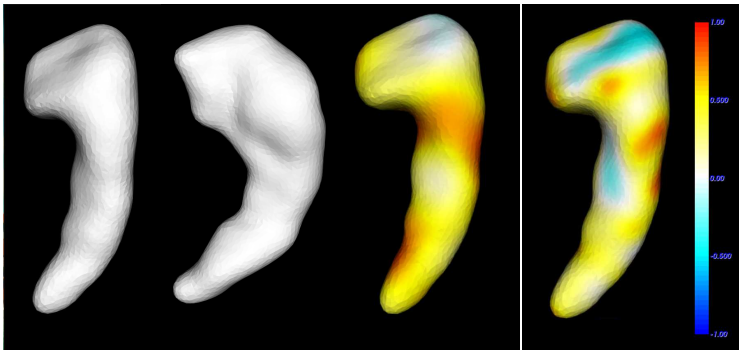


(a)



(b)

Figure 3.11: We manually add an expansion to the deformed shape in Figure 3.10 to compare our spectrum alignment method and the non-rigid ICP results. The locally deformed area is marked with a red circle. Column 1 shows the original shape. Column 2 shows the locally expanded shape from Figure 3.10 (Column 2). Column 3 and 4 shows the results of aligning column 1 to column 2 using our method and the non-rigid ICP technique, respectively. (a) and (b) show different views of the shapes.



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

Figure 3.12: The results of spectrum alignment of a shape in (a) to one in (b) using different sets of eigenvalues. (c) and (d) show the results of using first 20 and 80 eigenvalues.

in 3.12b using different sets of eigenvalues. As we can see, subtle deformations will be added to the results when using higher indexed number of eigenvalues.

The qualitative and quantitative comparisons of our method, non-rigid ICP method and voxel-based method are documented in Table 3.2. The voxel-based method is based on the number of the voxels in the original 3D MR images [17]. Using this method only the global volume of the hippocampus can be detected and the running time for this method is more than 45 seconds. Based on our experiments, the computing time for our method is less than 20 seconds while ICP method costs over 60 seconds. In order to quantitatively evaluate the capabilities of these methods in localizing deformations, we use the following metric to compute their agreement, A, as follows:

$$A = \frac{D \cap G}{D \cup G}, \tag{3.29}$$

where D is the localized deformation region and G is the known ground truth of the deformed region. The experiments demonstrate that the average outcome for our method is about 92%; for ICP method is about 81%; and for voxel-based method is about 85%. Therefore, our method outperforms the non-rigid ICP method in these shapes with few landmarks and features. Our method performs much better when there is mixture of different types of deformations.

Table 3.2: Comparison among our method, non-rigid ICP and voxel-based method.

Capabilities	Our Method	Non-rigid ICP	Voxel-based
Registration-Free	✓		
Global Deformation	✓	✓	✓
Local Deformation	✓	✓	
Multi-scale Deformation Quantification	✓		
Average accuracy: A	92%	81%	85%
Computation	$\approx$ 20s	$>$ 60s	$>$ 45s

Chapter 4: SURFACE REGISTRATION WITH EIGENVALUES AND EIGENVECTORS

In this Chapter, we focus on spectrum alignment of the non-isometrically deformed surfaces using both eigenvalue and eigenvector variations in order to find the correspondence and map the non-isometric deformations. To search for the alignment, we utilize a scale function on the surface that deforms one surface to a targeted one. Compared to the traditional approaches through the experiments, our method can accurately and automatically map and localize the point-to-point non-isometric deformations in addition to global difference of the shapes. Because the spectrum of shape is only depended on the intrinsic geometry, our method is invariant to spatial translation, rotation, scaling, and isometric deformation. Our method is computationally efficient and takes considerably less time to execute compared with existing methods [52].

4.1 Related Work

By definition, shape spectrum represents the information of intrinsic local geometry. It is invariant to isometric deformations and different triangulations. Reuter et al. [43] defined the spectrum of the Laplace-Beltrami operator of a shape as the signature or fingerprint of the shape. Rustamov in [47] employed the spectrum of this operator for shape clustering and classification purposes. Lévy in [28] employed the theory of *stationary waves* to study the behavior of eigenvectors and the static points of the eigenvectors. These points correspond to the locations that do not move in the theory of stationary waves. This study shows that the static points are strongly linked to the geometry of a shape and these points' locations change when the geometry varies. As these points are extracted from eigenvectors, they are invariant to isometric deformations of the shape. Thus, these points can be employed as the feature points to describe the geometry of the shape. Reuter et

al. [44, 40] employed these points, together with the domains generated by these static points, as topological features to segment and register different parts of the shapes. But the deformation of the shapes is restricted to be isometric in these studies. In reality, many deformations, such as heart motion, brain development, and so on, are not isometric. Hence, applying geometric spectrum methods for analyzing non-isometric deformation and registration is very challenging. Some recent work [50, 52] showed that the shape spectrum can be controlled with a scale function on the Riemannian metric. Shi et al. [50] discussed that the eigenvalues and eigenvectors change according to the Riemannian metric of a manifold. Later in [52], Shi et al. employed this metric to measure the difference between the eigenvectors of two surfaces in order to generate a conformal mapping between them. To this end, they minimized the difference between surfaces in the Laplace-Beltrami embedding space using an optimization approach. This work focused on eigenvector variation, but the eigenvalue variation was not investigated, and the method is very computationally expensive. For instance, to map two hippocampal surfaces with 1000 faces, the procedure took around 20 minutes on a computer with a 2.6-GHz Intel Xeon CPU and approximately 60 MB memory consumption. Instead, Hamidian et al. presented a method to align two surfaces by mapping their eigenvalues [9, 13]. This method provided a deformation matrix showing the deformation of the initial surface to the target one but did not generate a point-to-point correspondence mapping of the vertices. Cosmo et al. [8] employed a similar method to align eigenvalues. After the eigenvalue alignment, they used the similarity between the eigenvectors to find the corresponding points. This method is applicable to cases in which the eigenvectors turn out to be similar after aligning the eigenvalues. But in the cases of dramatic deformations like heart beating, just aligning the eigenvalues cannot make eigenvectors very similar. Therefore, the correspondence among deformations cannot be identified correctly. In this kind of dramatic deformation, eigenvectors need to be involved in the alignment process to find the correct corresponding points.

There are recent advances in the field of spectral shape analysis closely related to the proposed approach. For instance, Kovnatsky et al. [25] showed how to modify (align) the eigenvectors of the Laplace-Beltrami operator in order to match non-isometric shapes. Ovsjanikov et al. [36] proposed a spectral method for shape matching which is to find an alignment between eigenvectors based on a set of linear constraints. Later, they [35] presented a method for finding functional correspondence between manifolds based on the geometric matrix completion framework [24]. In [35, 49, 61], visualizing shape deformations based on a spectral representation of the correspondence was shown. However, the key difference between the methods mentioned above and our proposed approach lies in the fact that our method is using both eigenvalues and eigenvectors to align two manifolds versus these methods employed only the eigenvectors. Also, our method extracts feature points from eigenvectors, instead of using all the points, and employs them to align two surfaces without loosing accuracy.

In this paper, we present a novel method that can align two surfaces and visualize the corresponding points through the variation of geometric spectrum. This is achieved by mapping eigenvalues and certain feature points extracted from the eigenvectors of two surfaces. Given two triangle meshes, the spectra can be varied from one to another with a scale function defined on each vertex. In order to compute the alignment, we aim to minimize an energy function which is the integration of a smoothness term for aligning the eigenvalues and a distance term describing the distance between the corresponding feature points. Optimizing this energy function is a quadratic programming problem which can be solved using an iterative method. Furthermore, we assume that the variation of eigenvalue is expressed as a linear interpolation of eigenvalues of the two surfaces. The derivative of the scale function is the solution of such a problem. Therefore, the final scale function can be computed by an integral of the derivatives from each step. Subsequently, the scale function can describe the mapped surface eigenvectors that can be employed to find the

point-to-point correspondence. Our major contributions in this work can be summarized as follows:

- **We present a spectrum alignment algorithm using eigenvalue and eigenvector variations for 3D surfaces, supporting non-isometric global and local deformation analysis.** In the discrete domain, the variation of eigenvalues and eigenvectors in terms of the scale function can be presented as matrices. Employing these matrices, together with the smoothness function to align the eigenvalues and a distance function to align the feature points extracted from eigenvectors, a linear system can be defined. By solving this system, the eigenvalues and eigenvectors are aligned and the corresponding points of the surfaces can be determined.

- **Feature points automatically extracted from eigenvectors of the surfaces, along with the defined distance between the corresponding feature points, can lead to an improved correspondence with considerably reduced computational cost.** Because our method aligns both eigenvalues and eigenvectors at the same time, a limited number of feature points for the eigenvectors are sufficient to warrant the alignment. This helps to improve the accuracy and reduce the computational time considerably. These feature points are proven to be highly related to the geometry of the shape and they change when deforming the shape.

- **Our developed system demonstrates the accuracy and efficiency of the spectral variation and registration algorithm on visualization of non-isometrically deformed shapes.** The applications to biomedical imaging problems show that it is a viable solution for morphometric analysis and visualization in biomedical applications and clinical diagnoses.

4.2 Surface Registration Using Spectral Optimization

In this work, we employ Laplace-Beltrami operator to compute the geometric spectrum of a manifold. As we mentioned in chapter 2, Laplace-Beltrami operator can be presented as:

$$\mathbf{W}f_n = \lambda_n \mathbf{S}f_n, \tag{4.1}$$

where f_n and λ_n are the nth eigenvector and eigenvalue, respectively. The eigenvectors for different eigenvalues are orthogonal in term of $\mathbf{S}$ dot product. Using this concept, an embedding $\mathbf{I}_M : M \rightarrow R^\infty$ [48] is proposed as follows:

$$\mathbf{I}_M^\Phi = (\frac{f_1(x)}{\sqrt{\lambda_1}}, \frac{f_2(x)}{\sqrt{\lambda_2}}..., \frac{f_n(x)}{\sqrt{\lambda_n}}) \ \forall x \in M, \tag{4.2}$$

where $\Phi = \{f_0, f_1, f_2, ...\}$. By finding the proper map between the eigenvector embeddings after solving the sign ambiguity, two shapes can be aligned. Considering that for each eigenvalue, there is a vector of size n eigenvector, mapping eigenvectors of two surfaces for all the vertices is time-consuming. Therefore, we propose to use the eigenvector values for certain feature points to map the shapes.

4.2.1 Calculating the Feature Points

Using the spectrum of Laplace-Beltrami operator, Lévy [28] employed the theory of the stationary waves to model the shape. The spectrum contains a lot of information about the shape which can therefore be used for matching and mapping among different shapes. Looking closely to the eigenvectors, it shows that the n-th eigenvector can have at most n nodal domains. The nodal domains are the partitions of the surface that have the same sign. In this work, we are interested in the points, called *nodal sets*, which are the

static points between two nodal domains. In other words, the nodal sets separate the nodal domains. These nodes are the still zones in the theory of stationary waves. Lévy in [28] showed that these points are strongly linked to the geometry of the shapes. In our method, we use the nodal sets of certain eigenvectors as the feature points to map the eigenvectors of two shapes. As mentioned before, the n-th eigenvector has at most $n-1$ nodal sets which partition the n nodal domains. We use this concept and employ the nodal sets of the eigenvectors corresponding to the first non-zero eigenvalue as the first set of feature points. Figure 4.1a shows the first non-zero eigenvector for a sample left ventricle of heart and the red set of points in Figure 4.1b show this first feature set. The second feature sets are the nodal sets for the eigenvector corresponding to the second or third non-zero eigenvalue that are parallel to the first set of feature points. We use the parallel points in order to get exclusive sets of points for mapping the eigenvector of two shapes. Figure 4.1c shows the eigenvector corresponding to the second non-zero eigenvector and the blue sets of points on Figure 4.1b present the second sets of feature points. This approach can provide us three sets of points that are used for matching the eigenvectors of different shapes. When needed, more nodal sets can be used. Using these feature points for mapping the eigenvectors, instead of using all of the points, reduces the computational time considerably.

To illustrate the changes of nodal sets in the non-isometric deformation, we generate a bump on the surface of a bunny and compute these points for the original and deformed surfaces. Figure 4.2 shows these static points and that the nodal sets for two shapes vary when applying a non-isometric deformation. Therefore, these points are valuable in identifying landmark features of surfaces.

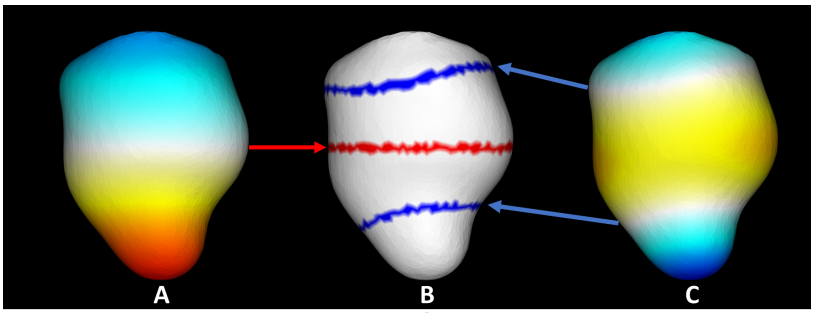
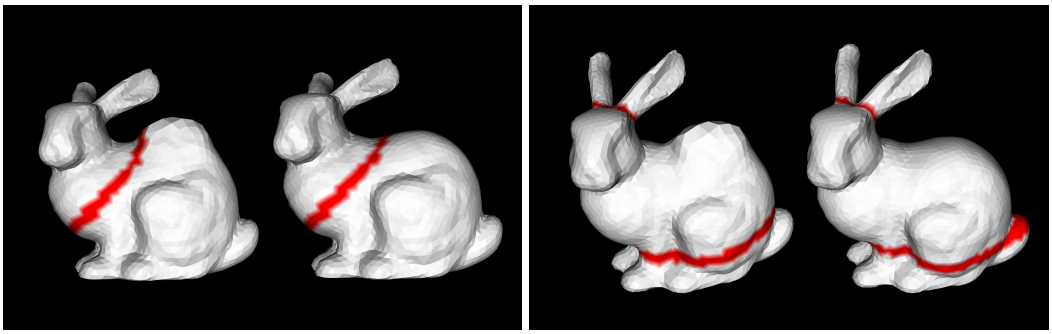


Figure 4.1: (A) The eigenvector corresponds to the first non-zero eigenvalue. (B) The three nodal sets. The red set shows the static points for eigenvector corresponding to the first non-zero eigenvalue. The blue sets show the static points for eigenvectors corresponding the second or third non-zero eigenvalue. (C) The eigenvector corresponds to the third non-zero eigenvalue.



(a) (b)

Figure 4.2: Static points for an original and deformed bunny. (a) shows the static points for eigenvectors corresponding to the first eigenvalue; (b) shows the static points for eigenvectors corresponding to the second eigenvalue.

4.2.2 Spectral Registration Using Eigenvector and Eigenvalue

To register two surfaces, the challenge is to minimize the difference between the two shapes in the spectral space. In order to align two surfaces using the LB spectral space, we aim to maximize the similarity between both eigenvalues and eigenvectors of the LB operator of the surfaces.

As a result of non-isometric deformation, the eigenvalues and eigenvectors of the shape dramatically change. On a compact closed manifold M with Riemann metric g, we

define shape deformation as a time variant positive scale function $\omega(t) : M \rightarrow R^{+}$ such that $g_{ij}^{\omega} = \omega g_{ij}$ and $d\sigma^{\omega} = \omega d\sigma$, where $\omega(t)$ is non-negative and continuously differentiable.

To increase the similarity between eigenvectors, we try to minimize the distance of the eigenvectors on the feature points. Therefore, we need a distance function in the embedding space. We employ the distance measure that was proposed in [52]. In order to find the optimal scale function ω for two surfaces $(N, \omega g_1)$ and (M, g_2), the energy function is defined as follows:

$$\begin{aligned} E(\omega, \Phi_1, \Phi_2) = & \int_N \left[d_{\Phi_1}^{\Phi_2}(x, M)\right]^2 d_N(x) \\ & + \int_M \left[d_{\Phi_1}^{\Phi_2}(N, y)\right]^2 d_M(y), \end{aligned} \tag{4.3}$$

where $d_{\Phi_1}^{\Phi_2}(x, M)$ and $d_{\Phi_1}^{\Phi_2}(N, y)$ are defined as follows:

$$\begin{aligned} d_{\Phi_1}^{\Phi_2}(x, M) &= inf_{y \in M} \parallel I_N^{\Phi_1}(x) - I_M^{\Phi_2}(y) \parallel_2, \forall x \in N, \\ d_{\Phi_1}^{\Phi_2}(N, y) &= inf_{x \in N} \parallel I_N^{\Phi_1}(x) - I_M^{\Phi_2}(y) \parallel_2, \forall y \in M, \end{aligned} \tag{4.4}$$

and ω is the scale function that applies on N and Φ_1 and Φ_2 are the eigenvector basis for LB embedding of $(N, \omega g_1)$ and (M, g_2). In this work, we focus on mapping one manifold to another and aim to find such a metric optimization. Therefore, we assume that the manifold M is fixed and N changes using the scale function ω to minimize the distance between M and ωN on those feature points.

To increase the similarity of eigenvalues between two manifolds N and M, we employ the theorem proved in previous chapter ([13]) using Equation 3.5. This theorem shows that the spectrum is smooth and analytical to non-isometric local scale deformation. Based on this theorem, non-isometrically deformed shapes can be registered when minimizing the energy function between eigenvectors on the feature points.

4.3 Numerical Optimization Using Spectral Variation

In this section, we will detail a discrete algorithm for the alignment of non-isometrically deformed shapes through the variation of eigenvalues and eigenvectors. Consider two closed manifolds, N and M with the eigenvalues of λ_1 and λ_2, and eigenvector basis of Φ_1 and Φ_2. These two manifolds are represented with discrete triangle meshes. We use their first k_1 non-zero eigenvalues and eigenvectors to align two surfaces. As we mentioned before, the deformation is not isometric; thus the first k_1 eigenvalues of these two surfaces are not the same. In order to align the first k_1 eigenvalues of N to those of M, a continuous scale diagonal matrix $\mathbf{\Omega(t)}$ is applied on N. $\mathbf{\Omega}$ is an m by m matrix, where m is number of vertices on N. The element Ω_{ii} at the diagonal is a scale factor defined on each vertex on N and will introduce a variation and alignment from N to M. It is a non-negative, continuously differentiable matrix.

To solve the numerical problem, we use the time interval of t and we divide the time interval of $t \in [0, 1]$ into K steps which we will index them as q. For each step of q, we solve an optimization equation to increase the similarity of eigenvalues and eigenvectors of $\mathbf{\Omega}N$ toward those of manifold M. At the beginning, $t = 0$, the eigenvectors and eigenvalues are Φ_1 and λ_1 and $\mathbf{\Omega}(0) = I$. When t reaches 1, the eigenvalues and eigenvectors will be λ_2 and Φ_2. In order to do that, we employ the Equation 3.11 mentioned in Chapter 3 which shows that the eigenvalues of N vary linearly toward those of M.

For mapping of eigenvectors, in each step we minimize the distance function described in the Equation 4.3 between $\mathbf{\Omega}N$ and M. The following will explain the details how to calculate the optimization function to minimize the distance between eigenvalues and eigenvectors in each step.

4.3.1 Eigenvector Optimization Equation

To minimize the energy function in Equation 4.3, we need to calculate the distance between two manifolds using Equation 4.4. In order to do that, we compute the k_1 eigenvalues and eigenvectors for both manifolds using Equation 4.1. One of the concerns about the calculating the eigenvectors is the sign ambiguity. This means that either f_n or $-f_n$ can be the eigenvector of a specific eigenvalue. For a target surface M, we fix the eigenvectors by picking random signs for Φ_2. Then, we calculate the feature points as described before. For the surface N, we start with $\mathbf{\Omega} = 1$ and in each step we update the surface using the optimized $\mathbf{\Omega}$ to minimize the energy function E.

At each step, we first calculate the k_1 eigenvalues and eigenvectors of the updated surface N using Equation 4.1. Then we calculate the 3 sets of feature points using the eigenvectors as explained before. We need to find the corresponding feature sets on two surfaces for solving the sign ambiguity of eigenvectors. As shown in Figure 4.1A, there is one nodal set for the eigenvector corresponding to the first non-zero eigenvalue. Therefore, these sets are matched on two surfaces. For the other two nodal sets, we calculate the corresponding sets using their signs on the eigenvector corresponding to the first non-zero eigenvalue. We first calculate and determine the sign of this eigenvector using the histogram of the positive and negative eigenvector values for both surfaces. As we mentioned before, the second sets of the nodal nodes are parallel to the first set. Also, the sets of nodes are at the opposite sides of the first nodal set. Therefore, each set of nodes has a different sign value on the first eigenvector. By knowing the sign of the first eigenvector, we can categorize and determine the corresponding sets of nodes for the second feature sets.

By detecting the corresponding feature sets in each step, we find the nearest feature points of surface N to the feature points of M that minimize the distance equation, Equation 4.4. To achieve this, we consider all combinations of signs for k_1 eigenvectors

for surface N to minimize the distance equation. After finding the corresponding points and the signs, the points are employed to generate matrix $\mathbf{C}$ as the nearest feature points map from $I_N^{\Phi_1}$ to $I_M^{\Phi_2}$. This map can be presented as $Id(\mathbf{B}V_1) = \mathbf{C}V_2$ where V_1 and V_2 are the vertices of surfaces N and M, respectively. Matrix $\mathbf{B}$ is a diagonal matrix of size V_1 in which the diagonal elements for the feature points are 1 and 0 otherwise. Matrix $\mathbf{C}$ has value of 1 only for the feature points; therefore, the projection relation Id can present a linear interpolation map from feature points of surface N to M. Using this map, we write the energy function, Equation 4.3, in a discrete numerical form as:

$$E = \sum_{n=1}^{k_1} \left(\frac{1}{\mathbf{S}(N)} \left(\frac{\mathbf{B}f_{1,n}}{\sqrt{\lambda_{1,n}}} - \frac{\mathbf{C}f_{2,n}}{\sqrt{\lambda_{2,n}}} \right)^T \mathbf{\Omega} \left(\frac{\mathbf{B}f_{1,n}}{\sqrt{\lambda_{1,n}}} - \frac{\mathbf{C}f_{2,n}}{\sqrt{\lambda_{2,n}}} \right) \right), \tag{4.5}$$

where $\mathbf{S}(N)$ is the surface area of N and $\mathbf{\Omega}$ is the scale function. In this work, because we change the surface N toward surface M and surface M does not change in each step, the second part of the Equation 4.3 is zero and only the first part is used to calculate the numerical equation. Considering that in each iteration the corresponding feature points are calculated and the eigenvalues do not change, the derivative of E with respect to time can be defined as follows:

$$E_f = \frac{\partial E}{\partial t} = \sum_{n=1}^{k_1} \left(\frac{1}{\mathbf{S}(N)} Ds_n^T \, \dot{\mathbf{\Omega}} \, Ds_n \right), \tag{4.6}$$

where $Ds_n = \left(\frac{\mathbf{B}f_{1,n}}{\sqrt{\lambda_{1,n}}} - \frac{\mathbf{C}f_{2,n}}{\sqrt{\lambda_{2,n}}} \right)$ and $\dot{\mathbf{\Omega}} = \frac{\partial \mathbf{\Omega}}{\partial t}$. Because $\mathbf{\Omega}$ is a diagonal matrix, we extract the diagonal elements as a vector $\mathbf{v_\Omega}$ and Equation 4.6 can be rewritten as:

$$E_f = \sum_{n=1}^{k_1} \left(\frac{1}{\mathbf{S}(N)} ((Ds_n)^2)^T \mathbf{v}_{\dot{\mathbf{\Omega}}} \right). \tag{4.7}$$

Using this equation, we update the eigenvectors through the numerical optimization of the gradient of the energy function in each step.

4.3.2 Eigenvalue Optimization Equation

In order to increase the similarity of two eigenvalues of λ_1 and λ_2, we employed the method mentioned in Chapter 2. As we mentioned in Chapter 2, the eigenvalue derivative can be defined as 3.9:

$$\dot{\lambda_n} = -\lambda_n f_n^T \dot{\mathbf{\Omega}} \mathbf{S} f_n. \tag{4.8}$$

Using this Equation, Equation 3.12 can be rewritten in a linear form as:

$$(\mathbf{v}_{\mathbf{S}_N} \circ f_{1,n} \circ f_{1,n})^T \cdot \mathbf{v}_{\dot{\mathbf{\Omega}}} = \frac{\lambda_{1,n} - \lambda_{2,n}}{\lambda_{1,n}(t)}, t \in [0,1]. \tag{4.9}$$

Note that, as the first k_1 eigenvalues are going to be aligned, we can get k_1 independent equations, which lead to a linear system as follows:

$$\mathbf{a} \cdot \mathbf{v}_{\dot{\mathbf{\Omega}}} = \mathbf{b}, \tag{4.10}$$

where $\mathbf{a}$ is a row stack of $(\mathbf{v}_{\mathbf{S}_N} \circ f_{1,n} \circ f_{1,n})^T$ with k_1 rows and $\mathbf{b}$ is the right side of Equation 4.9.

Considering that we use the first k_1 eigenvalues for this work and that practically k_1 is much less than the number of nodes in the mesh, the system is underdetermined and has no unique solution.

We solve this by assuming that the scale factors distributed on N are smooth as mentioned in Chapter 3. On the discrete triangle mesh N, with the scale function vector

$\mathbf{v_\Omega}$, the smoothness energy of E is define as:

$$E_\lambda = \mathbf{v}_{\dot{\mathbf{\Omega}}}^T \cdot \mathbf{W}_N \cdot \mathbf{v}_{\dot{\mathbf{\Omega}}} + 2\mathbf{z}^T \cdot \mathbf{v}_{\dot{\mathbf{\Omega}}}, \tag{4.11}$$

where $\mathbf{z} = \mathbf{W}_N \mathbf{v}_{\cdot\mathbf{\Omega}}$. Through the combination of this energy function and the energy function calculated for eigenvectors, the distance between the eigenvalues and eigenvectors of two surfaces can be minimized.

4.3.3 Energy Equation Integration

In order to increase the similarity of eigenvalues and eigenvectors of two surfaces, we integrate the energy function calculated for both eigenvalues and eigenvectors in order to find a scale matrix that minimizes the total energy function as follows:

$$\begin{gathered} E_T = E_\lambda + E_f = \\ \mathbf{v}_{\dot{\mathbf{\Omega}}}^T \cdot \mathbf{W}_N \cdot \mathbf{v}_{\dot{\mathbf{\Omega}}} + 2\mathbf{z}^T \cdot \mathbf{v}_{\dot{\mathbf{\Omega}}} + \sum_{n=1}^{k_1} \left(\frac{1}{\mathbf{S}(N)} ((Ds_n)^2)^T \mathbf{v}_{\dot{\mathbf{\Omega}}} \right). \end{gathered} \tag{4.12}$$

In order to preserve the physical availability, $\mathbf{v_\Omega}$ must be bounded, i.e., the scale factor cannot be zero or negative; and it cannot be infinite either. We employ the same lower and upper bound as mentioned in Chapter 3:

$$\mathbf{h}_l \leq \mathbf{v_\Omega} + \mathbf{v}_{\dot{\mathbf{\Omega}}} \leq \mathbf{h}_u. \tag{4.13}$$

The linear system (Equation 4.10), energy function (Equation 4.12), and constant bound (Equation 4.13) form a quadratic programming problem at each time t. Assume the eigenvalues and eigenvectors are known at each time t, the derivative of the scale matrix $\dot{\mathbf{\Omega}}$ is the solution of such quadratic programming.

The summary of the algorithm can be found in Algorithm 2. As shown, after K steps surface N will be aligned to surface M and the correspondence can be computed using the aligned eigenvectors.

Algorithm 2 Spectrum Alignment

Input: Closed 2D manifolds N and M, represented by triangular meshes, and constant k_1;
Output: Diagonal weight matrix $\mathbf{\Omega}(q)$ on N, aligning first k_1 non-zero eigenvalues and corresponding eigenvectors of feature points from N to M;

1: Initialize $\mathbf{\Omega}(0) \leftarrow \mathbf{I}$, calculate matrices $\mathbf{W}_N$ and $\mathbf{S}_N$, and $\lambda_{2,n}, f_{2,n}, \lambda_{1,n},$ and $f_{1,n},$ for $n = 1, 2, \ldots, k_1$;
2: Compute the feature points for surfaces M.
3: **while** $q < K$ **do**
   a: Calculate $\lambda_{1,n}(q), f_{1,n}(q),$ for $n = 1, 2, \ldots, k_1$ using Equation 4.1 with $\mathbf{\Omega}(q)$;
   b: Calculate the feature points for surface $\mathbf{\Omega}(q)N$ using $f_{1,n}(q)$; solve the eigenvector sign ambiguity; and find the corresponding features points between surface $\mathbf{\Omega}(q)N$ and M;
   c: Construct the quadratic programming problem using Equations 3.15, 4.12, and 3.23;
   d: Solve the quadratic programming problem to get $\dot{\mathbf{\Omega}}(q)$ and calculate $\mathbf{\Omega}(q+1)$;
   e: $q \leftarrow q + 1$;
4: **end while**
5: The correspondence of surface N and M can be computed using the aligned eigenvectors.

4.4 Experiments and Applications

The proposed algorithm and system are implemented using Python and C++ on a 64-bit Linux platform. For visualization purposes, we employ MATLAB and VTK library in Python. The experiments are conducted on a computer with an Intel Core i7-3770 3.4 GHz CPU and 8 GB RAM. We apply our algorithm to 2D manifolds, represented with triangle meshes. We employ the approach in [63, 59] to generate the uniform meshes. The number of vertices in those meshes is about 3000 for most of the experiment data. Besides the vertex number, there are two constants, i.e., K iteration and the first k_1 nonzero eigenvalues and eigenvectors to be aligned. For our experiments we choose $K = 10$.

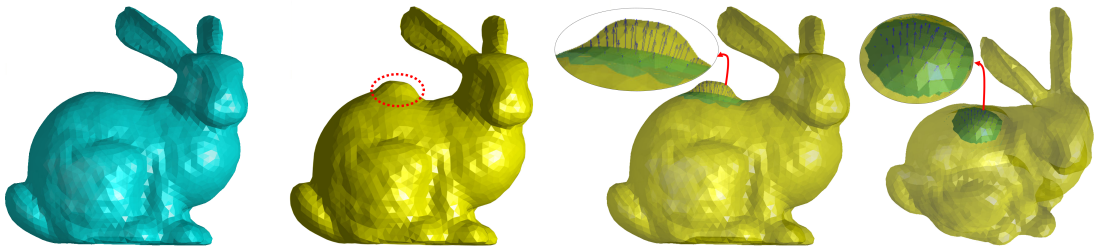
This number is sufficient to generate accurate results. We use different k_1 for different experiments. Depending on the resolution that we need in our experiments, the number k_1 may vary. The average computational time for 3000 nodes, with $k_1 = 8$ and $K = 10$, is around 12 seconds.

4.4.1 Experiments on Synthetic Data

Our Results

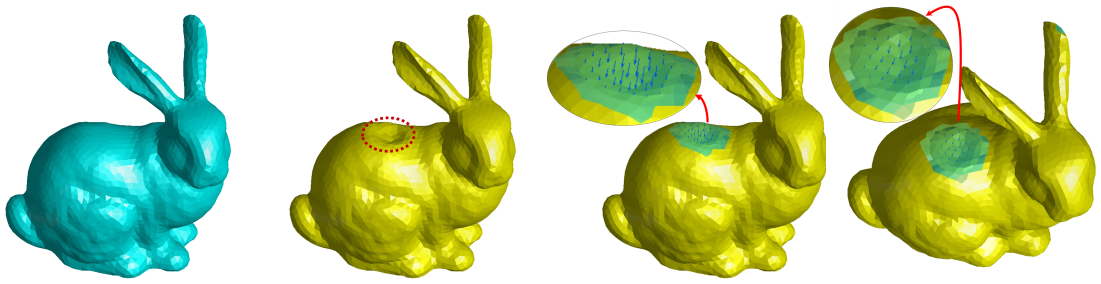
In order to evaluate our method, we manually make some non-isometric deformations on the surface of the shape and then we register the initial shape to the deformed one. In these experiments, we use a Stanford bunny model and make a non-isometric deformation on the surface and then generate uniform triangle meshes on both surfaces. We employ the first 10 non-zero eigenvalues and the corresponding eigenvectors to do the alignments. The processing time for $k_1 = 10$, $K = 10$, and 3000 mesh vertices is about 43 seconds. Note that, no correspondence information is used in the experiments.

In the first experiment, we manually generate a bump on the back of a bunny and align the original surface to the deformed one. Figure 4.3 shows the original surface in



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

Figure 4.3: The result of mapping the original 3D object to the synthetic one. (a) shows the original object. (b) is obtained by generating a bump on the original surface. (c) and (d) show the results of point-to-point mapping the original surface (cyan) to the target one (yellow) from different angles.



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

Figure 4.4: The result of mapping the original 3D object to the synthetic one. (a) shows the original object. (b) is obtained by generating an indentation on the original surface. (c) and (d) show the results of point-to-point mapping the original surface (cyan) to the target one (yellow) from different angles.

cyan and target surface in yellow. The location of the bump is marked by a red circle. Figure 4.3c and 4.3d present the results of point-to-point mapping of the surfaces from different angles. The original and targeted shapes are overlaid and the arrows in the bump area show the deformation of each vertex from the original to the targeted surface.

In the second experiment, we manually create an indentation on the surface of a bunny and align the original surface to the dent one. Figure 4.4a and 4.4b show the original and the surface results of creating the non-isometric dent on the surface, respectively. Figure 4.4c and 4.4d present the result of point-to-point mapping of surfaces using our method. The original and targeted shapes are overlaid and the arrows in the bump area show the deformation of each vertex from the original to the targeted surface. These results confirm that our method can accurately detect and localize the non-isometric deformation and find the corresponding points.

For more complex and challenging deformations, we use a hammer model and create 2000 uniform meshes on the surface. Then we create different non-isometric deformations on the surface and align the original surface to the deformed one. Figure 4.5 show the original hammer in yellow color and target deformed hammers in the cyan color.

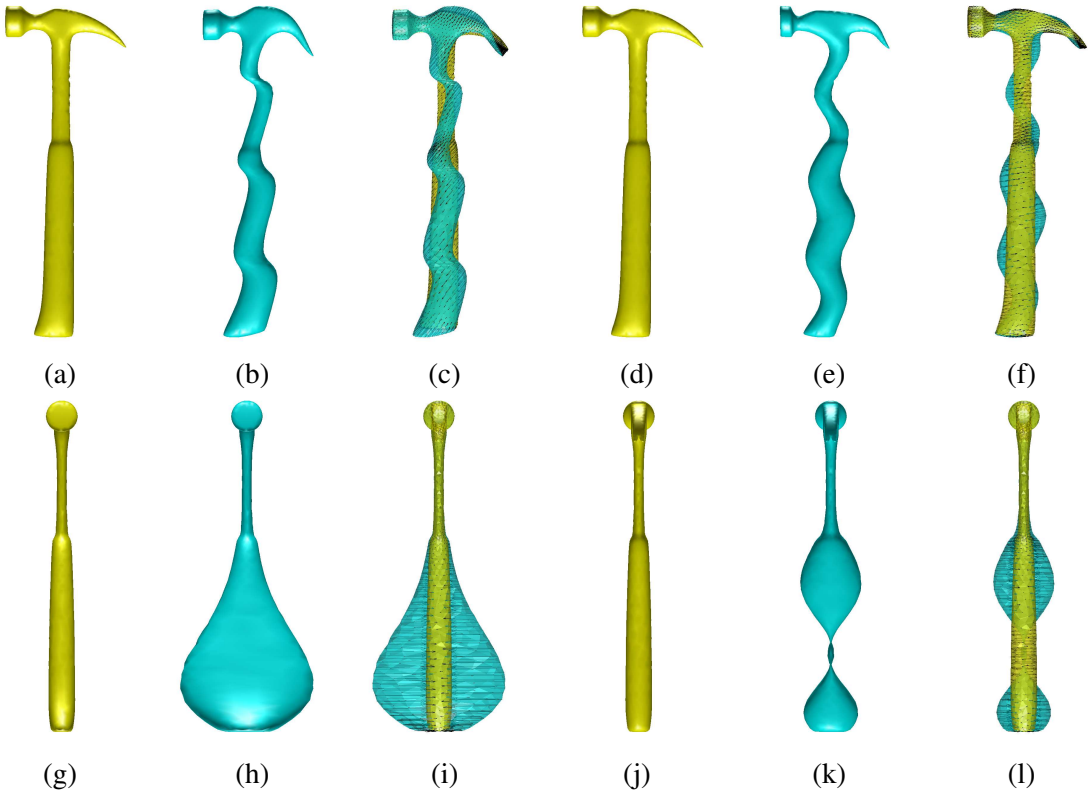


Figure 4.5: The result of mapping the original 3D hammers to the synthetic deformed ones. (a), (d), (g), and (j) show the original hammers. (b), (e), (h), and (k) are obtained by generating non-isometric deformation on the original surface. (c), (f), (i), and (l) show the results of point-to-point mapping the original surface (yellow) to the target one (cyan).

Figures 4.5c, 4.5f, 4.5i, and 4.5l show the results of aligning the original surface to the deformed ones using our method. The point-to-point alignments are demonstrated using the arrows that connect the corresponding points on the original and deformed surfaces. The original and target surfaces are overlaid in these figures. These results conclude that our method can detect the simple and complex non-isometric deformations on the surface accurately.

Comparisons to Spatial Registration Methods

In order to further demonstrate the capabilities of our method, we compare the results of our algorithm with the ones from non-rigid Iterative Closest Point (ICP) algorithm. ICP is introduced by Besl and Mckay in [4] and is one of the popular approaches in spatial registration-based methods. In this approach, the initial transformation for global matching is first estimated and then the closest points are found by minimizing the distance between two shapes. Therefore, using this method we first register the original surface to the target one rigidly and then the corresponding points between the rigidly registered original shape and the target shape are calculated. For our method, we employ the first 12 non-zero eigenvalues and the corresponding eigenvectors for alignment purposes. The processing time for 3000 vertices, $k_1 = 12$ and 10 iterations is 118 seconds.

In order to compare two methods, we use a template hippocampus and synthetically deformed the shape by bending and stretching the shape from upper and lower sides. The original and deformed surfaces are shown in Figures 4.6a and 4.6b, respectively, which exhibit global variation. Figure 4.6c presents the overlay of the original and deformed shapes. As can be seen, the top and the bottom parts of the surface are stretched and the shape is bent in the middle part. Figure 4.6d shows the result of performing ICP rigid registration on the original shape to map it to the deformed shape. Comparing Figures 4.6c and 4.6d, one can notice that the rigid ICP does not match the shapes correctly, especially in the top and bottom regions of the shapes. Figures 4.6e and 4.6f present the results of our method and non-rigid ICP, and the arrows show the displacement of each vertex on the surface. Because the ICP method fails in the rigid registration stage, the corresponding points calculated using non-rigid ICP do not reflect the accurate deformation, especially in the top and bottom region. On the other hand, because our method does not require pre-rigid registration to find the corresponding points, this variation can be captured by

our registration and mapping method accurately. These results justify the advantage of our method over the rigid and non-rigid ICP method. In order to demonstrate that our method can handle both global and local deformation simultaneously, we create a bump on the shape of Figure 4.6b as shown in Figure 4.6g. Then we align the surface in Figure 4.6a to Figure 4.6g using our method. Both global and local deformations (bump) can be captured by our method as shown by arrows in Figure 4.6h. Therefore, these results confirm that our method can detect and localize the non-isometric deformation and find the correspondence and their displacements resulted from both global and local deformations.

Comparison to a Spectral-based Method

In order to compare our method with a similar spectral-based technique, we employ an approach suggested by Shi et al. in [52]. They proposed a method based on aligning the eigenvectors of two surfaces via optimization of a conformal metric on the surfaces. They employed the eigenvectors for all the points of the surfaces therefore the computation is very expensive.

In this experiment, we employ a template hippocampus and create 1500 uniform vertices on the surface. Then, similar to the previous section, we synthetically deform the shape by bending and stretching the shape from upper and lower sides. The bending and stretches are larger in this experiment than in the previous one. Figures 4.7a and 4.7b show the original and synthetically deformed surfaces, respectively. Figures 4.7c and 4.7d show the results of the alignment using our and Shi et al.'s methods. As can be seen, the bottom tip of the surface cannot be aligned correctly using Shi et al.'s method but our method can align all the points accurately.

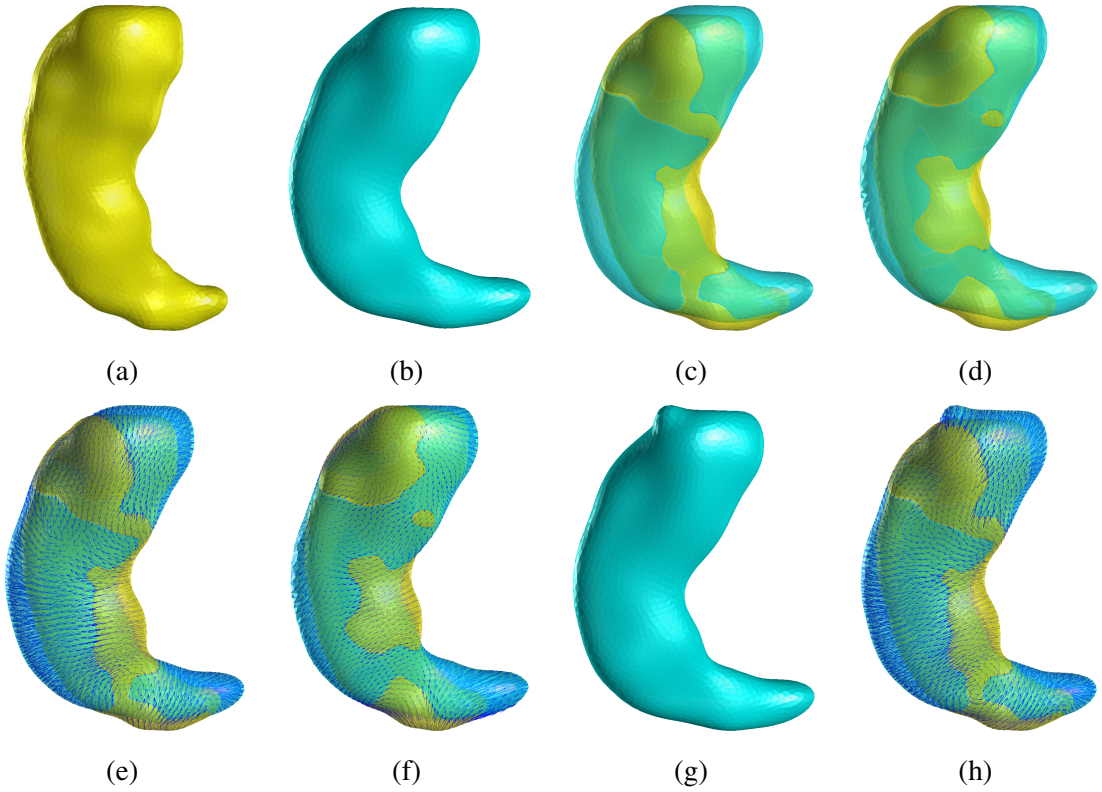
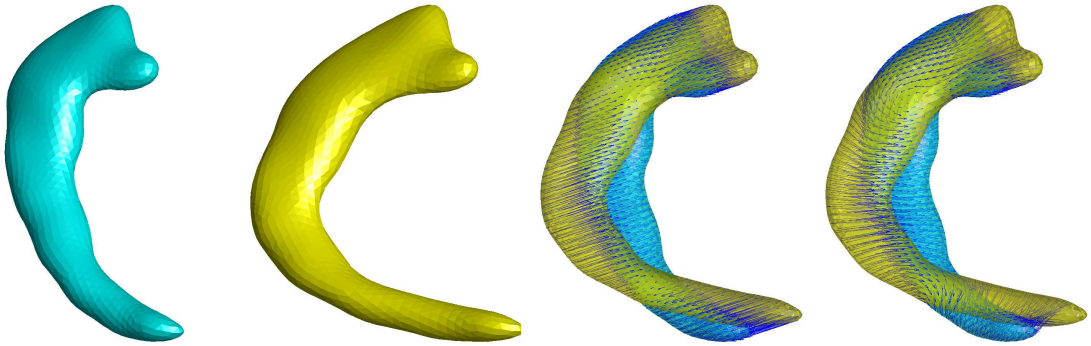


Figure 4.6: Comparison of our method with ICP method using synthetic data. (a) presents the original surface and (b) is obtained by bending and stretching the shape from the upper and lower ends. (c) shows the result of mapping these two shapes. (d) shows the result of ICP rigid registration result. The ICP method register the shape from one side and therefore this method cannot generate accurate result for bending deformation. (e) and (f) present the results of point-to-point mapping from the original surface to the deformed one using our method and ICP method, respectively. Because the result of non-rigid ICP depends on rigid ICP, the result is not accurate. Our method can detect the deformation accurately. In order to show that our method can handle both global and local deformation simultaneously, we make a bump on the deformed surface as presented in (g). The result of mapping (a) to (g) is presented in (h).

In order to quantitatively evaluate the capabilities of these methods in localizing the point-to-point correspondence, we use the following metric:

$$A = 1 - \frac{\sum_{i=1}^{m} |d_i - d_i^O|}{\sum_{i=1}^{m} |d_i^O|}, \tag{4.14}$$



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

Figure 4.7: Comparison of our method with Shi et al.'s method [52] using synthetic data. (a) presents the original surface and (b) is obtained by bending and stretching the shape from the upper and lower ends. (c) shows the result of mapping these two shapes using our method. (d) shows the result of Shi et al.'s approach. The lower tip of the shape is not aligned correctly using Shi et al.'s method while our method can align all the points accurately.

where d_i is the distance between corresponding points calculated using either methods, d_i^O is the known ground truth distance, m is the number of all nodes and i is the index of nodes. The experiments demonstrate that the average outcome for our method is 91.4% while it is 85.8% for Shi et al.'s method. This number is 75.6% for ICP method. Therefore, our method can find the point-to-point correspondence better than the other two methods for complex deformation. In this experiment we use 10 iterations and 12 non-zero eigenvalues and eigenvectors to do the alignments. For 1500 mesh vertices, our method takes 67 seconds while Shi et al.'s approach takes 94 minutes. In [52], they mentioned that their execution time for 1000 triangle faces with approximately 500 vertices, is 20 minutes.

Table 4.3: Comparison among our method, Shi et al.'s method and non-rigid ICP method.

Capabilities	Our Method	Shi et al.'s Method	Non-rigid ICP
No Rigid Registration	✓	✓	
Local Deformation	✓	✓	✓
Average Accuracy: A	91.4%	85.8%	75.6%
Computation	$<$ 120s	$>$ 20m	$>$ 60s

Table 4.3 demonstrates the comparison between our method, Shi et al.'s method, and non-rigid ICP method. As mentioned before, non-rigid ICP method requires rigid registration before aligning two surfaces while Shi et al.'s and our methods do not have this requirement. All the methods can localize the deformation of the surface but our method has the best average accuracy based on metric defined in Equation 4.14. The computational time for our method is considerably less than Shi et al.'s method and similar to non-rigid ICP. Therefore, our method has the best features to align two surfaces.

4.4.2 Applications on Real Patient Imaging Data

Alzeimer Data

Alzheimer disease (AD) is a brain mis-functionality that is caused by the loss of neurons and neural volume. Hippocampus is vulnerable to damage in the early stage of Alzheimer. Volumetric longitudinal studies using MR images show hippocampal atrophy during time in comparison to healthy cases. In this study we show the point-to-point deformation for an Alzheimer case. We employ 10 AD and 10 healthy cases, which have longitudinal study for one year to track and compare the deformation of hippocampi. The cases are downloaded from the hippocampal study from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database (adni.loni.usc.edu) and are segmented using FreeSurfer software. Then, the 3D objects and meshes with 3000 vertices are generated. In this study, we use the first 10 eigenvectors to align the surfaces. Figures 4.8a and 4.8b show a sample of AD case for the baseline and after one year. The deformation in the tail part can be detected visually. The deformation mapping is shown by using the blue arrows in Figure 4.8c . It needs to be mentioned that the deformation mapping is down-sampled by five in order to better visualize the results. As can be seen, our method can detect the deformation accurately. Figure 4.8d shows the 6th eigenvector before and after alignment. Columns A and

B show the eigenvector for the baseline surface before and after alignment, respectively. Column C shows the targeted surface eigenvector. It is noted that our method can match and align the eigenvectors. The color map shows the scaled value of the eigenvector.

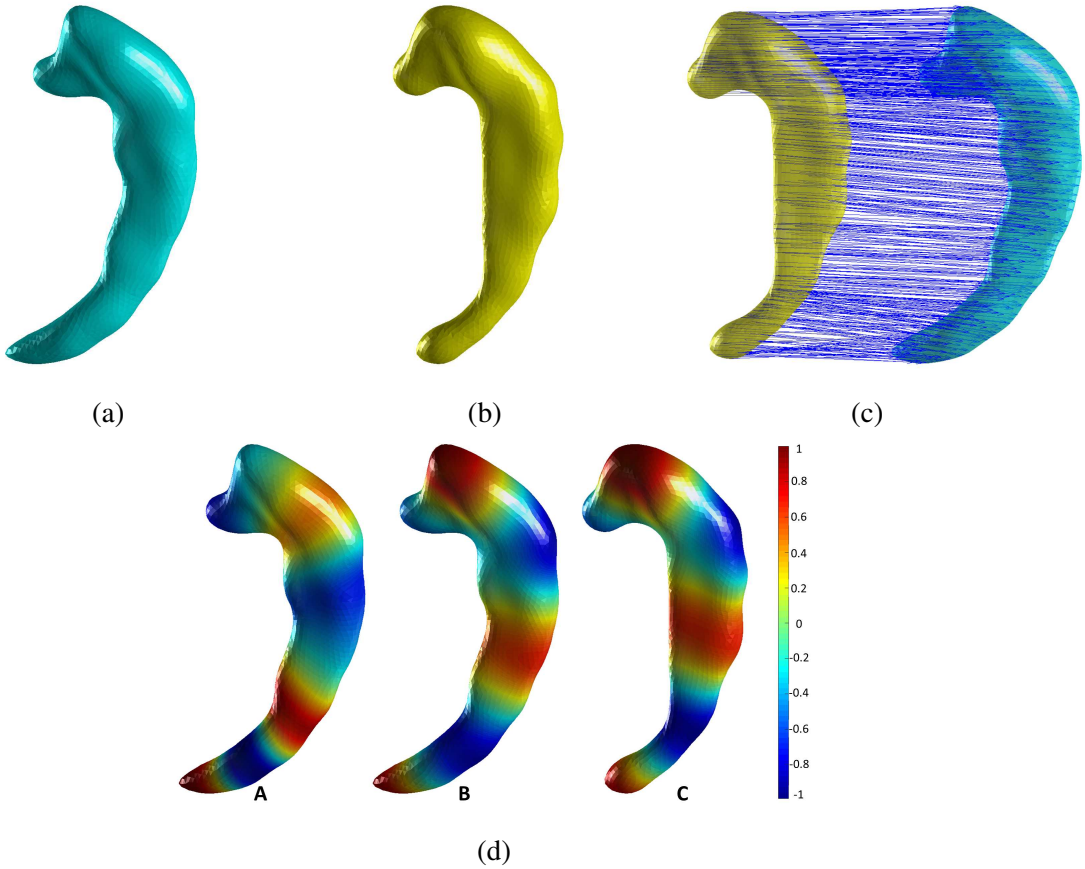


Figure 4.8: (a) shows the baseline hippocampus and (b) shows the hippocampus for the same subject after one year. (c) presents the result of mapping the baseline hippocampus to the one after one year. (d) presents the 6th eigenvector before and after mapping. Column A shows the eigenvector before alignment and column B shows the eigenvector after alignment. Column C shows the 6th eigenvector for the target surface.

In order to show the variation of eigenvalues of the manifolds before and after alignment, we list the 2nd to 10th non-zero eigenvalues of baseline hippocampus (before and after mapping) and hippocampus after one year in Table 4.4. The eigenvalues are normalized by the first nonzero one to remove the scale factor. It can be seen that after

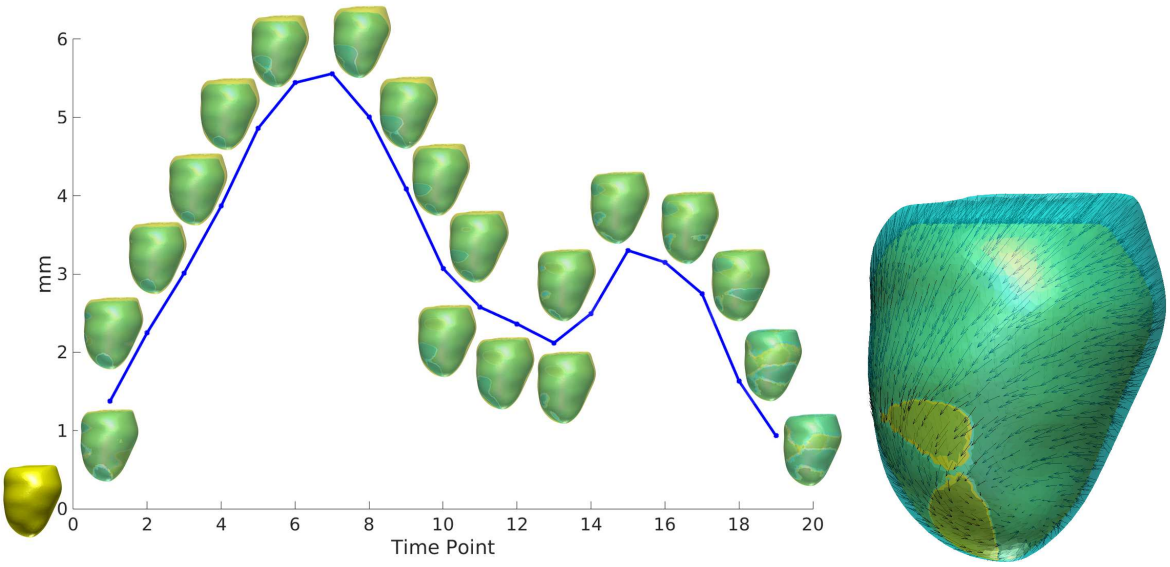
applying the spectrum alignment algorithm, the eigenvalues of the source manifold have changed to well align with the target ones.

Table 4.4: The result of aligning eigenvalues from the baseline hippocampus to one after one year (target) using the same case as in Figure 4.8.

Manifold	$\lambda_i/\lambda_1, i \in [2, 10]$
Baseline	3.40, 7.39, 10.66, 15.36, 17.04, 21.50, 23.13, 24.94, 29.77
Target	4.17, 8.65, 11.42, 16.23, 18.97, 23.18, 26.19, 30.53, 32.38
Aligned	4.17, 8.65, 11.42, 16.23 , 18.96, 23.17, 26.19, 30.49, 32.39

Cardiac Data

We use the Sunnybrook Cardiac Data [38] for this experiment. The data is acquired from the 3D left ventricle during cardiac cycles from end diastolic to end systolic, and



(a) (b)

Figure 4.9: (a) The result of mapping the first time point shape to all other shapes and then computing the mean average of the distance. The yellow surface is the left ventricle in diastolic state. The other shapes shows the contracted left ventricle toward systolic state overlaid on the yellow surface. (b) The result of mapping the first time point surface (cyan) to the 7th one (yellow) which has the most deformation according to the plot in (a).

then, back to end diastolic cycle. For this study, we use five cases for each category: heart failure with infarction, heart failure without infarction, left ventricle (LV) hypertrophy, and healthy. Meshes with 5000 vertices are generated. 10 non-zero eigenvalues and the corresponding eigenvectors are used for alignment. We map the surface for the end diastolic (first time point) to all other 19 time points. Then, the mean averages of the displacements for all the nodes are calculated and used to generate a plot of surface displacement for 19 different time points. Figure 4.9a shows the result of mapping these surfaces using our method and the mean average plot for a sample case. The first shape close to the origin of the axes is the diastolic shape and the rest of the shapes are the target ones. As can be seen, the results follow the heart beating pattern. Figure 4.9b shows the overlay of the first time point surface (cyan) to the 7th one (yellow) which has the most mean displacement according to the average plot. The results of mapping these two surfaces are also presented in this figure by using arrows, which connect the corresponding points and show the point-to-point deformation mapping. One can see that our method can detect the contraction and the tangential turning deformation of the heart.

This method can be used in different applications in cardiac study. The first application is longitudinal study of a subject. In this study, the 3D images of the subject are generated for more than one heart beating full cycles. Using these models, the deformation plot for mean average of displacement can be generated. Physicians usually look at more than one cycles to exam the abnormality of the heart. By using our method, the abnormality can be detected via the mean average plot and then using the point-to-point deformation mapping to find out the specific abnormal time points. Therefore, the abnormal area of the ventricle may be detected. Figure 4.10a shows three heart beating cycles and the one time point which is marked by a red circle in the 2nd cycle showing abnormality according to its location. As can be seen, the abnormality can be detected using our method. We detect this abnormality by creating a bump (Figure 4.10c) on a heart surface (Figure 4.10b). Then,

our method can generate the displacement color map and the abnormal region can be also located (Figure 4.10d).

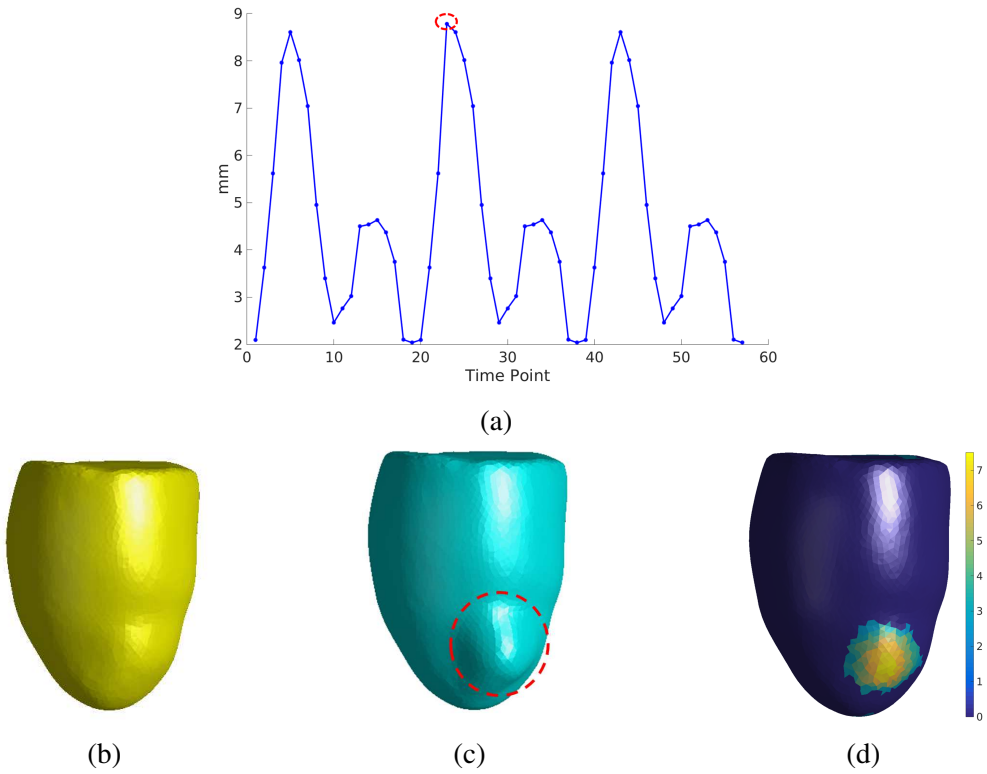


Figure 4.10: The longitudinal study for a subject using our method. By studying more than one cycle, the abnormal beat can be detected using our method and the abnormal area of the left ventricle can be identified. (a) shows 3 heart beat cycles and the abnormal time point is marked by a red circle. We create this abnormality by creating a bump on a heart surface. (b) shows the original heart. (c) shows the deformed surface. The deformed area is marked by a red circle. (d) shows the displacement color map generated by our method to detect the abnormal region.

The second application is the cross-subject analysis through temporal alignment. The same time point of a heart beat cycle for two different subjects can be identified. For instance, in one subject, the largest heart contraction may happen at the 7th time point, but in another subject, the largest heart contraction may be detected in 8th time point. Using our method, different time points in different subjects can be aligned according to the heart beating cycle through temporal alignment. Figure 4.11 shows this procedure. Using

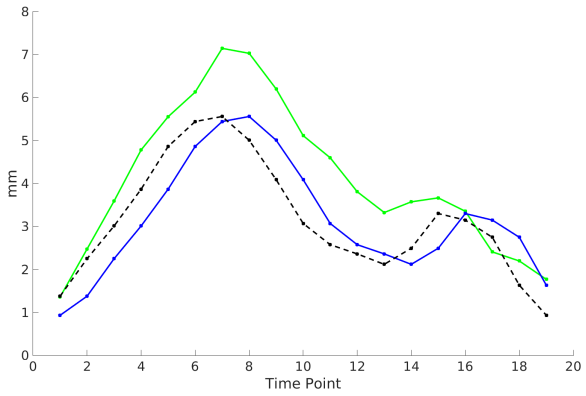


Figure 4.11: The blue and green curves show the mean average mappings for two different cases. The black dashed curve shows the blue curve after alignment.

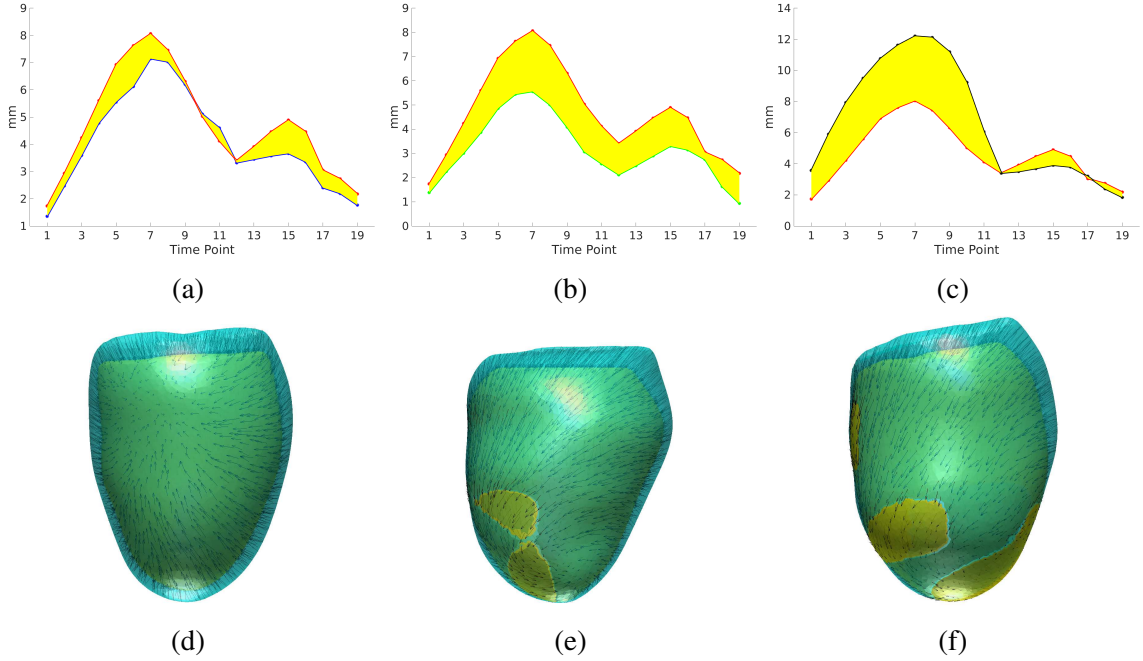
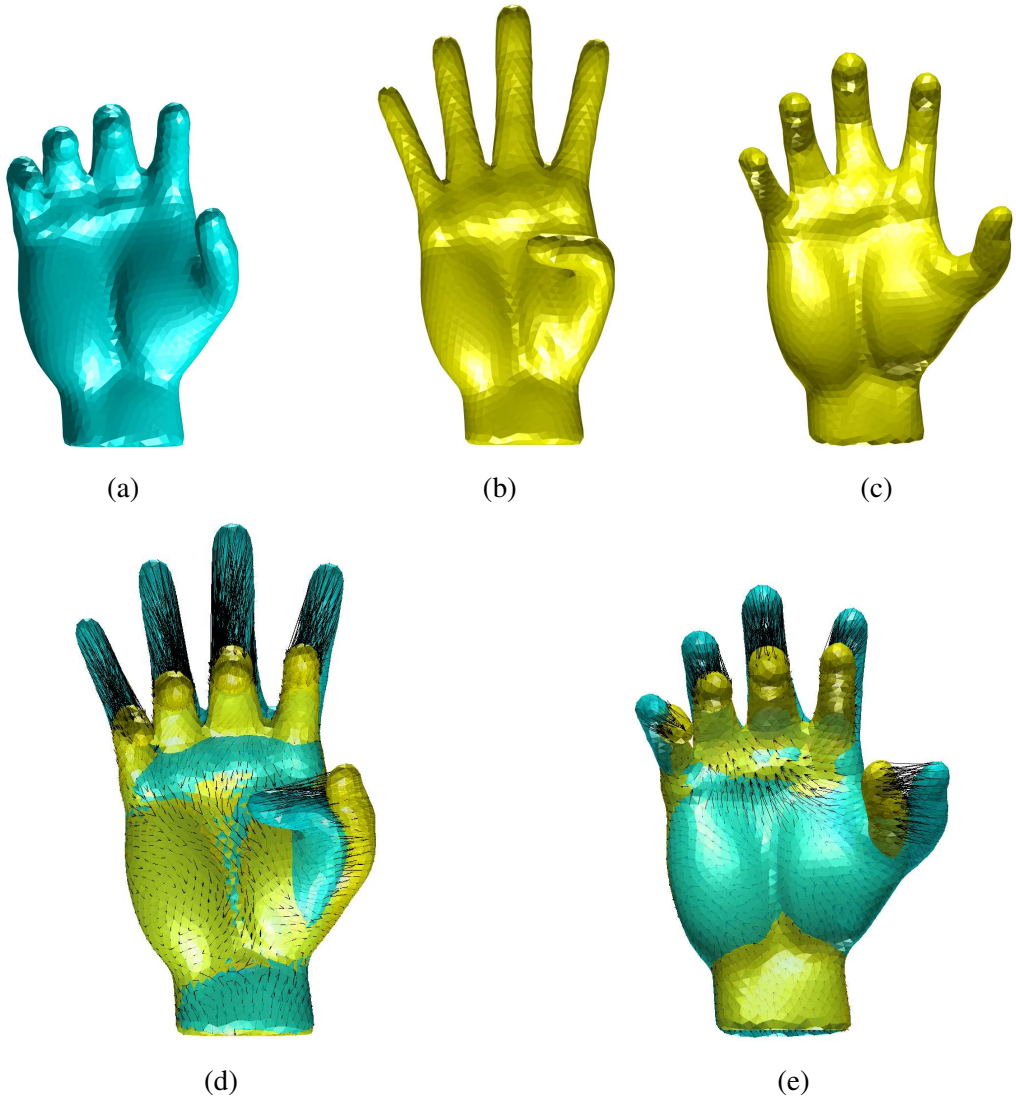


Figure 4.12: The result of comparing the mean average of five temporally aligned healthy cases, to a healthy and two patient cases. In all plots, the red curves show the average plot generated by calculating the average of five aligned healthy cases' mean average mappings. (a) The blue curve shows a healthy case. (b) The green curve shows a diseased heart case. (c) The black curve shows a hypertrophy case. One can use the difference between two curves (yellow area) to accurately distinguish the healthy from patient cases. (d), (e), and (f) show the displacement mapping of the left ventricle from the diastolic to systolic state for normal, diseased, and hypertrophy cases showed in (a), (b), and (c), respectively.

this alignment, we can align different healthy cases and then use the aligned mean average mapping to generate a standard mean average mapping. This mapping can be used to compare with normal and patient cases in order to categorize the abnormal cases from the normal ones by using the difference indicator between this mapping and a target one. In Figure 4.12, the red plot shows the standard mean average plot generated by calculating the average of mean average mappings from temporally aligned five healthy cases. Figure 4.12a shows a healthy case in a blue curve, as compared with the standard mean average mapping. Figure 4.12b shows a diseased case in a green curve when temporally aligned to the standard mean average mapping and Figure 4.12c shows a LV hypertrophy case in a black curve. Yellow area shows the difference between two curves. As can be seen, the difference area in healthy case is much less than that in the patient cases. Using difference area, one can accurately categorize healthy cases from patient cases. Figures 4.12d, 4.12e, and 4.12f show the displacement mapping of the left ventricle from the diastolic to systolic state for normal, diseased, and hypertrophy cases showed in Figures 4.12a, 4.12b, and 4.12c, respectively.

4.4.3 Application on Hand Data

In order to demonstrate that our method can be used on non-medical applications as well, we use data of different hand gestures and create 3000 uniform vertices on the surfaces. Figures 4.13a, 4.13b, and 4.13c show different gestures of the hand. We employ the shape in Figure 4.13a as the original surface and then align it to the shapes in Figures 4.13b, and 4.13c. Figures 4.13d and 4.13e show the results of this alignment using our method. The original and target surfaces are overlaid and the point-to-point alignment is shown using the arrows that connect the corresponding points. As can be seen, different parts of



(a) (b) (c)

(d) (e)

Figure 4.13: The results of aligning different hand gestures. (a) shows the original surface. (b), and (c) show the target surfaces. (d) and (e) show the results of aligning original surface to the surfaces demonstrated in (b) and (c) respectively. The results show that our method can detect the point-to-point correspondence between two surfaces correctly.

the hand are aligned correctly and our method can detect the point-to-point correspondence accurately.

Note that, for searching for proper eigenvectors, in addition to sign ambiguity, there are some cases that the order of the eigenvectors switches. In addition, by using large number of eigenvectors, numerically it is possible for near multiplicities of eigenvectors to

cause the eigen-spaces to split in different directions [52]. In such cases, the eigenvectors matching becomes difficult. In our algorithm, we mainly focus on resolving the sign ambiguities. The order switching of eigenvectors and detection of high dimensional multiple eigenvectors will be our future research work.

Chapter 5: 3D SHAPE REGISTRATION TASKS TO A SCALABLE SCIENTIFIC WORKFLOW SYSTEM

5.1 Introduction

In this chapter, we employ and enhance a scientific workflow system named DATAVIEW (www.dataview.org) to integrate our methods mentioned in previous chapters. Using this workflow system, the user can execute all the tasks using their web browsers without installing any software such as Python or MATLAB on their computer. The user can also run all the tasks in parallel in order to speed up the processing time for multiple cases.

5.2 Related Work

Scientific workflow systems have become a popular tool since they allow the definition algorithms and applications as a collection of functional blocks. The composition of these blocks can be done graphically, by dragging and dropping functional modules which are linked together with pipelines that represent inputs and outputs. Scientific workflow systems offer users a better "picture" and understanding of how an application is designed by making its dataflow structure explicit. Thus the workflow contains information about what each component would do rather than how it will achieve it, along with information about how to connect components. Somehow, scientific workflows are a representation that may be part of formal software design documentation.

Many workflow environments have emerged over the last few years. Taverna is a workflow editor environment developed to support "in silico" experiments for bioinformaticians for the MyGrid project [34]. It facilitates converting standalone workflows into web-services and sharing workflows manually in the myExperiment [16], an online repository of workflows that enables users the discovery and distribution of Taverna workflows.

A new server version of Taverna is deployed using which user can run their workflows on the server machine.

Similar to Taverna, Kepler [1] provides a graphical interface with the capability of invoking Web services. Kepler integrates disparate software components, such as merging "R" scripts with compiled "C" code. This workflow system can be run on the server as well which gives users the facility of running their code remotely on the server machine. For sharing the workflows users need to export their workflows and then share it via a central repository, the Kepler Component Repository [22]. For using all of the above platforms, users need to install these tools on their local machines before they actually can use the workflow system [3]. Therefore, the computers with limited resources cannot run complex code and also the users should have the knowledge of installing the software on their computers. The Galaxy platform is an open source web-based platform that allows users to create, execute, and share their workflows in the purpose of data intensive bioinformatics research. The benefit of using a web-based workflow system is that the only requirement for using this platform is a web browser. They also extend the Galaxy framework to the grid computing infrastructure for running bioinformatics applications [21]. The mentioned workflow platforms are mostly used for field of bioinformatics [5]. Some works uses Taverna [55, 32] and Kepler [62] for medical image purpose but as mentioned before these toolkits need to be installed on the user computer. Also for sharing the workflows, the user should upload them manually through their online repository. In [30], they implement a workbench based workflow system for biomedical imaging in the field of FiberFlow system purposes. For using this platform user need to install it on their computer as well. Also, they propose to employ grids for some computation-intensive tasks but they do not use it in their system.

5.2.1 DATAVIEW workflow system

In this work, we employ the DATAVIEW system to integrate our methods. We choose this system because it is web-based and flexible to be improved to integrate software tools such as MATLAB and Python. It is implemented using Java and MySQL server. It provides a web-based interface for uploading, and downloading the data products. Also using the browser, the data products and the tasks that are already implemented in the system can be employed by the user to design, save, share, and run their workflows. The user can drag and drop the input of the workflow, the output stub and the tasks that are built in as part of the system to design their own workflows. By connecting the data products as inputs to the input port of the tasks; outputs of the tasks to the output stub or input ports of other tasks, the dataflow of the workflows are specified. Figure 5.7 presents the user interface that can be deployed to design the workflow. As can be seen, the left panel contains the data products that are uploaded to the system. The right panel contains the built-in tasks, and the saved workflows that can be reused again. This option is very helpful to avoid designing the same workflow. For using DATAVIEW for our purpose, some parts of this system needs to be enhanced which are our major contributions in this work and can be summarized as follows:

- **The software tools which are required to be used are integrated into the system.**
- **We supplement this platform with grid computing.** This is done in order to centralize and speed up the processing power of this workflow system. Using grid we can leverage the powerful data parallel modes of workflow enactment, execute multiple tasks simultaneously, access to the robust and fault-tolerant interface which can handle various error conditions commonly. Also employing a secure environment of the grid, the data and results of processing are protected with user credential. By us-

ing this web-based workflow system which utilizes grid, different users can design, save, share, run, and check results or error of their workflows in parallel. The two mentioned parts are the components of the Task Manager section of the system. We will explain these parts in more detail in the next section.

5.3 System Implementation

As mentioned before, for utilizing the web-based DATAVIEW system for our purpose, we enhance and add some sections. The changes are in the following parts: the Task Manager, and the Data Product Manager that will be described in more detail in this Section.

5.3.1 Data Product Manager

The data products in this workflow system are mostly file type such as Analyze format which is one of the standard formats for storing 3D medical images, 3D object files such as obj and ply, MAT-Files which are the format for saving and reading the results for MATLAB tool, and text files for grid processing. The text files that are used for grid processing gives user the capability of writing the name of all input files in text file which facilitate performing the same processing on all inputs in parallel.

When a user register the data product using the interface, the compatibility of the data product with the structure of workflow is checked by workflow manager. In case of being compatible, the data is transferred to the storage elements that can be grid or the DATAVIEW server machine and is identified with a unique ID assigned by the user. The transportation is performed with secure protocols such as sFTP. The specification of the data like type, path and name is then saved in the database by workflow management in an XML based language. The XML representation for a sample of the file is depicted in

```
<dataProduct name="user given name">
  <description>"Filename"</description>
  <type>File</type>
  <data>
    <file>
      <FileName>"Filename" </FileName>
      <FilePath>>"File Path on Grid or Server" </FilePath>
      <FileType>"File Type"</FileType>
      <FileStorage>"Grid" or "Server"</FileStorage>
    </file>
  </data>
</dataProduct>
```

Figure 5.1: The XML presentation of the File type data product.

Figure 5.1. As the file is registered in the database, file name, path, user given name, type of the file, the location, and the storage type of the file is saved in the database. This will be helpful for addressing the file in implementation section.

5.3.2 Task Manager

The tasks in this work are mostly script-based tasks. The scripts can be located in the DATAVIEW server machine or in the grid. The tasks are saved in the database in the XML format. The XML presentation of the script based tasks contains some features including the specification of input and output ports, the location of the script, the specification of the host on which the script is executed on it. A template of the task representation can be seen in Figure 5.2.

Based on the specification of the workflow, tasks can be executed at the DATAVIEW server, or in a grid node. The scripts are designed in a way that it generates an error text file in case any error occurs during running the task. After the completion of the task, the task manager checks the error text file. If no error exists in the text file, results will be generated in the corresponding ports and the specification of the result files will be saved in the database. Otherwise, for each output port, the corresponding error will be generated. For grid-based workflow tasks, we employ the Wayne State grid. This infrastructure utilizes

```
<workflowSpec>
 <workflow name="Workflow Name">
  <workflowInterface>
   <workflowDescription>"workflow Description"</workflowDescription>
   <inputPorts>
    <inputPort>
     <portID>"ID of the port"</portID>
     <portName>"Name of the port"</portName>
     <portType>"Type of the port"</portType>
     <portDescription>"Description of the port"</portDescription>
    </inputPort>
    … </inputPorts>
   <outputPorts>
    <outputPort> The same as input ports</outputPort>
    … </outputPorts>
  </workflowInterface>
  <workflowBody>
   <taskComponent taskType="Type of the task">
    <executable>"The specification of the script"</executable>
    <appName>"Application name"</appName>
    <apptime>"Approximate time for running the task"</apptime>
    <taskInvocation>
     <operatingSystem>"Can be Linux, Windows, etc"</operatingSystem>
     <invocationMode>"Remote or local"</invocationMode>
     <invocationAuthentication>
      <hostName>"The IP of the host"</hostName>
      <userName>"user"</userName>
      <password>"pass"</password>
     </invocationAuthentication>
    </taskInvocation>
   </taskComponent>
  </workflowBody>
 </workflow>
</workflowSpec>
```

Figure 5.2: The XML presentation of the task.

the PBS Pro software for full-featured workload management and job scheduling. The processing of running job on grid is depicted in Figure 5.3.

The input of the tasks can either be the file that is uploaded on the grid by user or the output files of other tasks. Using the input files and the task specification, the workflow task

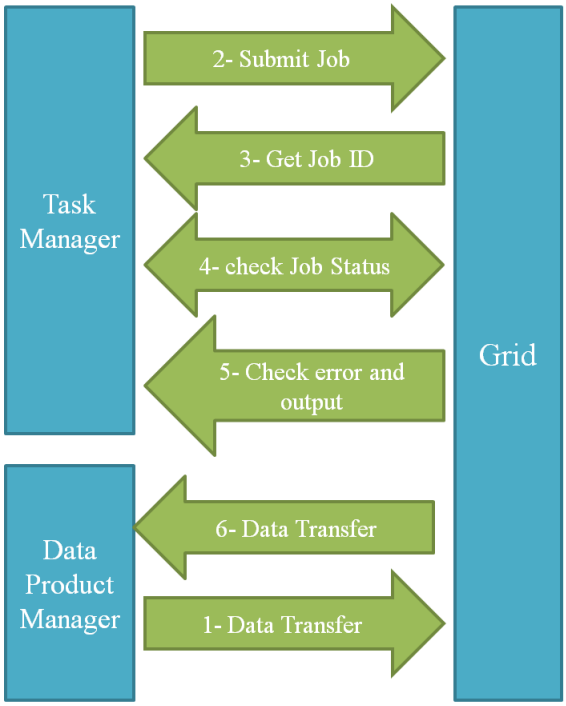


Figure 5.3: The procedure of submitting a job to the grid, check the status of the job and check the error and output files. This part is the new implementation for task manager and data product section for the purpose of submitting a job to the grid environment.

manager submits the task to a grid node and gets a unique job ID. This ID will be held for future reference and monitoring the status of the job. The task manager periodically polls the jobs for their status, until either the job is completed, or an error is generated. It should be mentioned that as can be seen in Figure 5.2, we design a parameter in the specification of grid-based workflow tasks using which the approximate time for executing the task is defined. This option is very helpful especially for the tasks that may take a long time to be executed. Using this option, the task manager waits for the time specified and then checks the status of the task periodically until the completion of the job. After the completion of the job, the task manager copies the error and output text file generated by the job to the DATAVIEW server machine using the sFTP protocol and check the error text. In case of the completion of the task without any error the output files specification are exported from

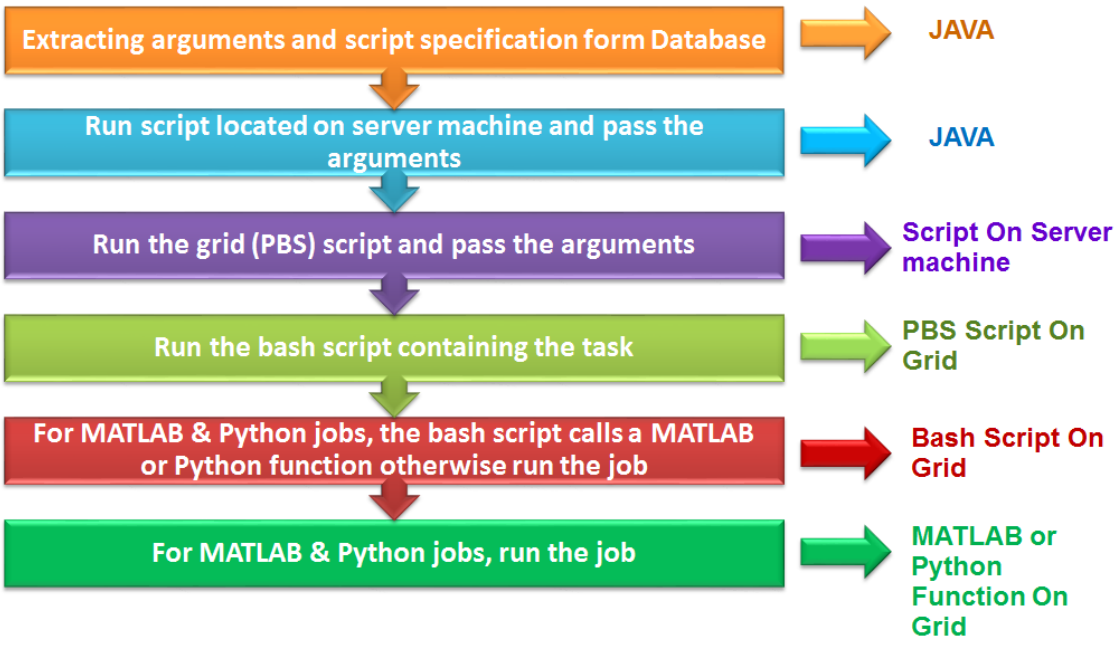


Figure 5.4: The invocation procedure for running the code in grid.

output text file. The name of the files and their locations are sent to the database to register them as a data product. These data products can be used for next tasks or as outputs of the whole system. It should be mentioned that only the text files contains the name of the output files is transferred to the DATAVIEW server machine not the actual files. This helps the efficiency of the processing.

The invocation flow for running the tasks in grid using the workflow system can be described as follows and depicted in Figure 5.5. The input files are uploaded to the system by user and the path of files is saved in the database. When a workflow is designed and run, the workflow manager which is implemented in Java, load the specification of the data products from the database. These data products are the input files of the workflow designed by the user. The workflow system also contains some pre-built tasks which are implemented before and the user deploys them for designing and running their workflow. The specification of the tasks is also uploaded from the database.

Therefore, in the first step, the location of the scripts for running the task is uploaded from the database to the system. Then the script using the designed data products as arguments is called by Java. This script is located on the server machine and it calls a job script on a grid machine using the assigned data product.

In the second step, as the name and address of the script and the arguments which are the specification of data products are already exported from the database in previous section, these data will be pushed to the script located on the DATAVIEW server machine.

In the third step, using the ssh method we employ the qsub command to run a job in a grid environment. The job script as can be seen in the third rectangle is a PBS script that explains some specification of the jobs. These specifications are:

- The name of the queue using which the job will run.

- The number of the cpus and nodes that job would use to run the code.

- The path of the output folder and error folder. These folders would be used to copy the output and error text file that the grid generates automatically while running the code.

- email of the algorithm developer who runs the job to report the completion of the job.

Some other options can be specified in the job script that we did not use in this work and can be read in the PBS help file for more information. In addition to the specification mentioned above, the PBS script specifies the name of the script which contains the codes for executing the job. It also determines the data path.

In the forth step, The data and the scripts are copied to a temporary folder by grid and the script would be executed and the results will be copied back in the folder specified in the job script. After completing the job, the PBS script checks the availability of the results and in case of not generating the results, an error text generated that will be returned to the DATAVIEW server machine. Also, the path and the name of the output files are

returned to the server machine as a text file. In addition to these error files, the error text file that depicts the error of the tools is generated automatically in the grid error folder. This folder is specified in the grid PBS script as mentioned before.

For integrating MATLAB and Python in our system, the bash script file on grid call the MATLAB or Python function. The input arguments, which are pushed to this script from previous script, are integrated in the function. Figure 5.5 shows a sample workflow of the codes for MATLAB jobs.

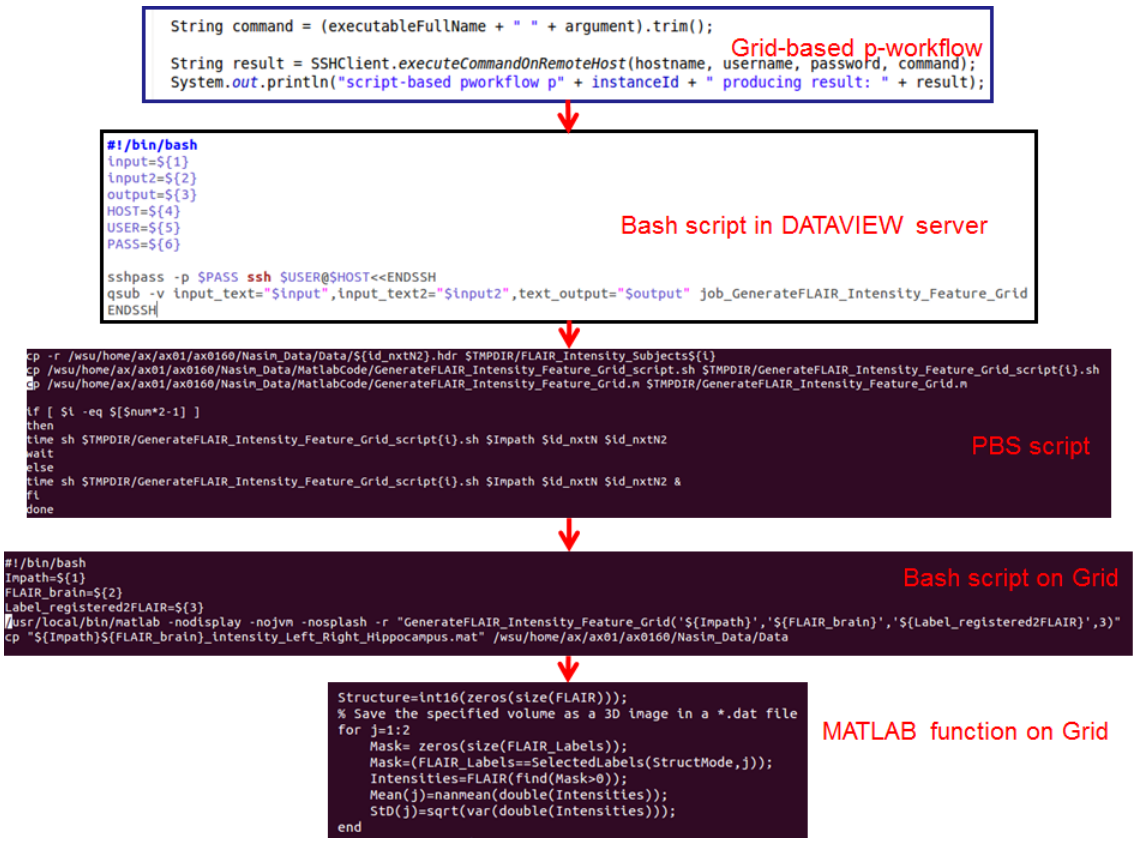


Figure 5.5: The procedure for running MATLAB code in grid.

For running the tasks in the grid environment, we implemented a new technique using which up to 64 task runs can be executed in parallel for one single task. For instance, 64 Registration task instances can be run in parallel in one unique job. The maximum number of the cores in Wayne Grid nodes is 64. That is the reason that we can run at most

64 tasks in parallel. This technique is helpful in running the same task for 64 different cases in parallel. If we run these tasks in separate jobs, the system is required to wait for each job in queue and thus the performance of the processing would decrease considerably. In this kind of processing, the input names are written in a text file and the output names are generated as a text file too in which the name of the result files and the path of them are specified. Output files are also generated and registered to the database. The intermediate data are saved for both grid and DATAVIEW server based primitive workflows for fault and error monitoring. In case of error detection, the intermediate results can be employed to check the reason of raising the error.

5.4 Experimental Study

The workflow processing for the our methods are depicted in Figure 5.6. The green rectangles are the inputs of the workflow that are uploaded to the platform. The blue rectangles are the actors or tasks that execute on the input data products. The yellow rectangles are the outputs of the workflow. The formats of the input data products are text files which has the names of the input images. The second layer includes actors for calculating the spectrum of the shape. The third layer include the actor that perform our method and generate MATLAB file that will be used to visualize the deformation of the shape. The last yellow layer is the results of the processing as a MATLAB file and also an error file that shows the error generated from executing the job. The two actor layers are written in Python.

The designed platform can execute more complicated workflows. Figure 5.7 shows different type of registration for 2D images for medical purposes. In order to do this workflow, we integrated BET skull striping [53] for skull striping, FREESURFER [42] for segmentation of different structures of 2D images, FLIRT [19] for registration, and MATLAB

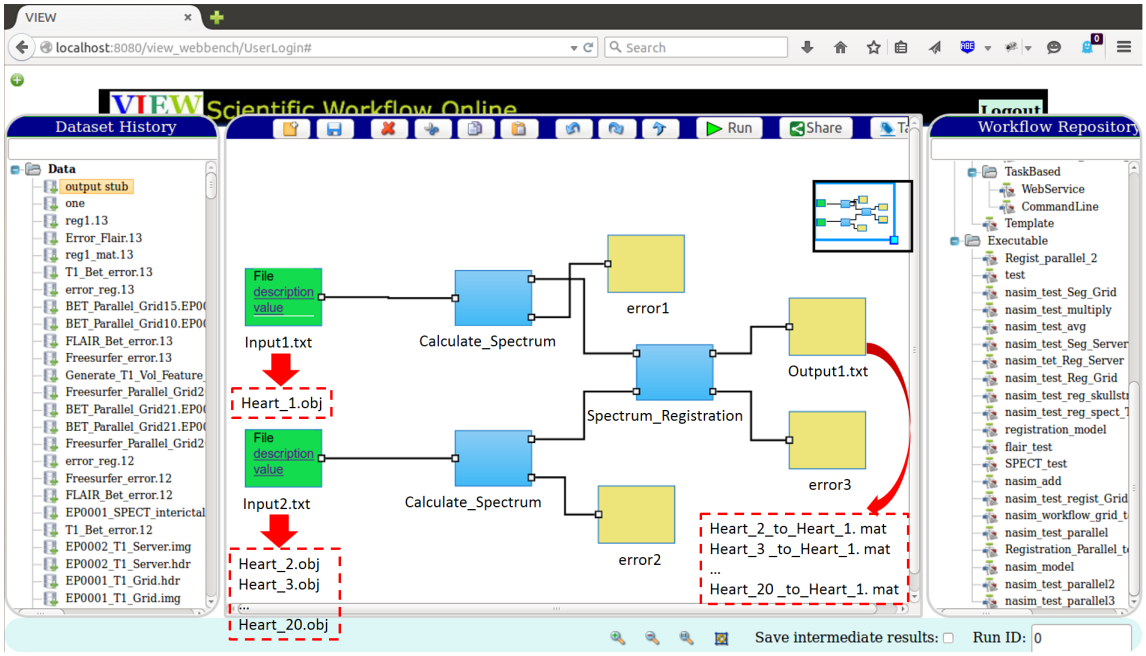


Figure 5.6: The scientific workflow for our method running on grid in parallel. The input and output files are text files. The tasks can be run for up to 64 cases in parallel. The green rectangles are the inputs of the workflow. They can be selected from the left panel of the platform. The blue rectangles are the task agents which can be dragged and dropped from the right panel of the platform. The yellow rectangles are the outputs of the workflow. They can be chosen from the left side as output stub and renamed by users.

for feature extraction tasks. By employing different type of tools for different tasks, we can compare their results and choose the best one for each task.

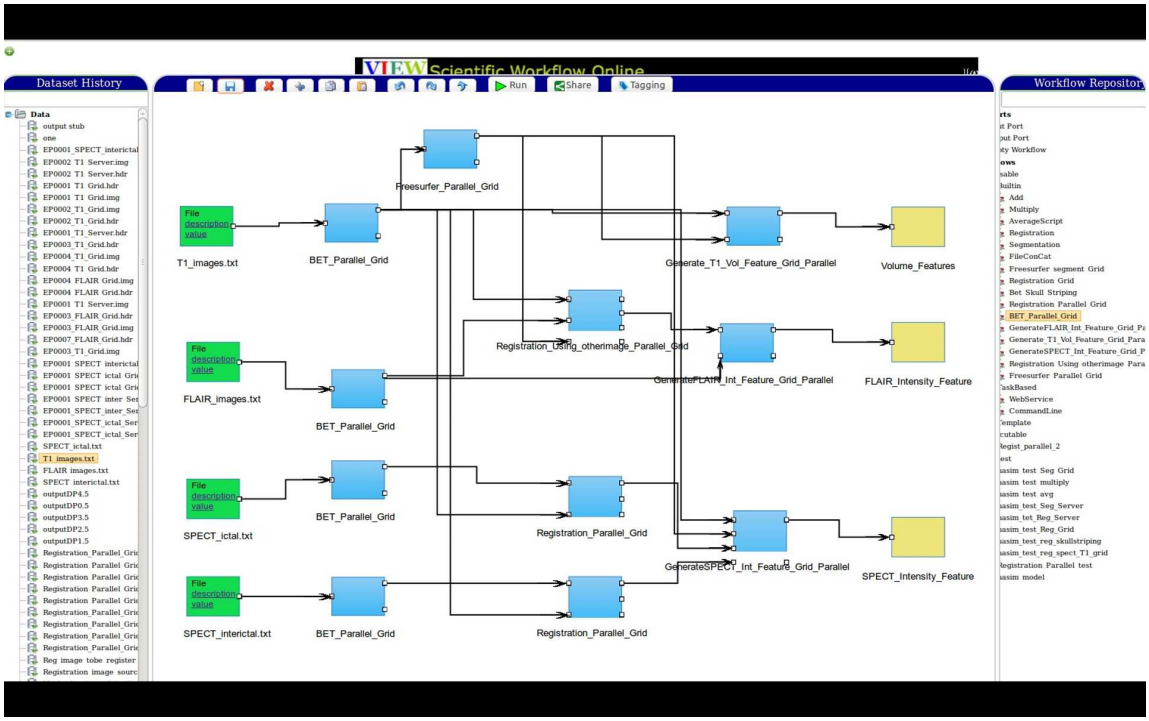


Figure 5.7: A more complicated sample of workflow for epilepsy diagnosis.

Chapter 6: CONCLUSION

In this work, we have introduced two new method based on our developed spectrum variation theorems for localizing and quantifying deformations and finding the point to point correspondence between two surface shapes. We have also, integrated these methods in a workflow system called DataView system to give the users the capability of running their codes using their web-browsers. The main contributions include:

- We have proved that the eigenvalues of the spectrum constitute an analytic function of a scale function defined on the Riemann metric. The deformation is an integral of the derivative of the scale function. The theorem applies both continuous analytic and discrete cases, therefore, warrants an algorithm to align the shape spectra of discrete shapes represented with triangle meshes. Given two closed triangle meshes, the spectra can be aligned from one to another with a scale function defined on each vertex. Our extensive experiment results have demonstrated that the proposed method can detect the global and local deformation of shapes. Our applications to real epilepsy and Alzheimer data have shown its potential in clinical diagnosis. Furthermore, the comparisons to non-rigid ICP method and voxel-based method have indicated that our method has more advantages.

- We have introduced a new method based on shape spectrum to find the point-to-point correspondence between two surfaces undergoing global and local deformations. We employ both eigenvalues and eigenvectors of the surfaces in the alignment. Our method can localize the non-isometric deformation of the surface and find the displacement map for all the vertices. Because we use certain feature points instead of all the vertices to align the eigenvectors, our method is considerably more efficient than existing methods. We have applied our method to both synthetic and real data, and the results confirm the advantage and accuracy of our method. We have

also compared our method with ICP and an spectral-based methods and showed the advantages of our method over them.

- We employed and enhanced a web-based scientific workflow system named DATAVIEW to integrate our methods. We also extend this system to utilize grid for processing the data in parallel and thus speed up the performance of the system. For implementing the extension parts we added some sections to the data product and task manager part of the system. Using this workflow system, users can design, save, run, and share their workflow using their web-browsers without the need of installing any software and regardless of the power of their computers. Using Grid the same task can be executed on up to 64 different cases which will increase the performance of the system enormously. Another benefit of this system is that the same tasks can be implemented using different tools and the results of them can be compared for further decision making.

SELECTED PUBLICATIONS

1- H. Hamidian, Z. Zhong, F. Fotouhi, and J. Hua, ”Surface Registration with Eigenvalues and Eigenvectors”, Accepted with Minor Revision in IEEE Transactions on Visualization and Computer Graphics, Jan 2019.

2- H. Hamidian, J. Hu, Z. Zhong, and J. Hua, “Quantifying Shape Deformations by Variation of Geometric Spectrum,” In Proceedings of the International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI), Vol. 9902, pp. 150-157, 2016.

3- J. Hu, H. Hamidian, Z. Zhong and J. Hua, “Visualizing Shape Deformations with Variation of Geometric Spectrum,” IEEE Transactions on Visualization and Computer Graphics (IEEE Vis ’16), Vol. 23, No. 1, pp. 721-730, 2017.

4- H. Hamidian, J. Hu, F. Fotouhi, H. Soltanian-Zadeh, Jing Hua, “Quantifying and Visualizing Shape Deformations by Variation of Geometric Spectrum”, presented in the First WSU-HFHS Inter-institutional Symposium on Imaging in Motion and Beyond, January 22, 2016.

5- H. Hamidian, S. Lu, S. Rana, F. Fotouhi, H. Soltanian-Zadeh, “Adapting Medical Image Processing Tasks to a Scalable Scientific Workflow System,” IEEE World Congress on Services conference, pp. 385 – 392, 2014.

6- M.R. Nazem-Zadeh, J.M. Schwalb, H. Bagher-Ebadian, H. Hamidian, A. R. Akhondi-Asl, K. Jafari-Khouzani, and H.Soltanian-Zadeh, “Lateralization of Temporal Lobe Epilepsy using a Novel Uncertainty Analysis of MR Diffusion in Hippocampus, Cingulum, and Fornix, and Hippocampal Volume and FLAIR Intensity,” J. of the neurological sciences, vol.342 (1), pp. 152-161, 2014.

7- S. Maraka, Q. Jiang, K. Jafari-Khouzani, L. Li, S. Malik, H. Hamidian, T. Zhang, M. Lu, H. Soltanian-Zadeh, M. Chopp, P.D. Mitsias, “Degree of Corticospinal Tract Dam-

age Correlates With Motor Function After Stroke," Annals of Clinical and Translational Neurology, vol. 1, No. 11, pp. 891-899, 2014.

REFERENCES

[1] I. Altintas, C. Berkley, E. Jaeger, M. Jones, B. Ludscher, and S. Mock. Kepler: An extensible system for design and execution of scientific workflows. In *16th Intl. Conf. on Scientific and Statistical Database Management*, 2004.

[2] R. Bakkestrom, N. L. Christensen, E. Wolsk, A. Banke, J. S. Dahl, M. J. Andersen, F. Gustafsson, C. Hassager, and J. E. Moller. Layer-specific deformation analysis in severe aortic valve stenosis, primary mitral valve regurgitation, and healthy individuals validated against invasive hemodynamic measurements of heart function. *Echocardiography*, pages 1–9, 2018.

[3] A. Barker and J. Hemert. Scientific workflow: A survey and research directions. *Parallel Processing and Applied Mathematics*, 4967:746–753, 2008.

[4] P. Besl and N. McKay. A method for registration of 3-d shapes. ieee transactions on pattern analysis and machine intelligence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2), 1992.

[5] M. Borcz, R. Kluszczyński, K. Skonieczna, T. Grzybowski, and P. Bała. Processing the biomedical data on the grid using the unicore workflow system. *Euro-Par 2012: Parallel Processing Workshops*, 7640:263–272, 2013.

[6] A. Bronstein, M. Bronstein, and R. Kimmel. Generalized multidimensional scaling: a framework for isometry-invariant partial surface matching. *Proc Natl Acad Sci U S A.*, 103(5):1168–1172, 2006.

[7] J. J. Caban, P. Rheingans, and T. Yoo. An evaluation of visualization techniques to illustrate statistical deformation models. *Computer Graphics Forum*, 30(3):821–830, 2011.

[8] L. Cosmo, M. Panine, A. Rampini, M. Ovsjanikov, M. M. Bronstein, and E. Rodolà. Isospectralization, or how to hear shape, style, and correspondence. *CoRR*, abs/1811.11465, 2018.

[9] H. Hamidian, J. Hu, Z. Zhong, and J. Hua. Quantifying shape deformations by variation of geometric spectrum. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016*, pages 150–157, 2016.

[10] J. Han, P. Yin, Y. He, and F. Gu. Enhanced icp for the registration of large-scale 3D environment models: An experimental study. *Sensors*, 16(2), 2016.

[11] M. Hermann, A. C. Schunke, T. Schultz, and R. Klein. A visual analytics approach to study anatomic covariation. In *IEEE PacificVis*, pages 161–168, 2014.

[12] M. Hermann, A. C. Schunke, T. Schultz, and R. Klein. Accurate interactive visualization of large deformations and variability in biomedical image ensembles. *IEEE Trans. on Visualization and Computer Graphics*, 22(1):708–717, 2016.

[13] J. Hu, H. Hamidian, Z. Zhong, and J. Hua. Visualizing shape deformations with variation of geometric spectrum. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):721–730, 2017.

[14] J. Hu and J. Hua. Salient spectral geometric features for shape matching and retrieval. *The Visual Computer*, 25(5):667–675, 2009.

[15] W. Huizinga, D. Poot, M. Vernooij, G. Roshchupkin, E. Bron, M. Ikram, D. Rueckert, W. Niessen, and S. Klein. A spatio-temporal reference model of the aging brain. *NeuroImage*, 169:11–22, 2018.

[16] D. Hull, K. Wolstencroft, R. Stevens, C. A. Goble, M. R. Pocock, P. Li, and T. Oinn. Taverna: a tool for building and running workflows of services. *Nucleic Acids Research*, 34:729–732, 2006.

[17] K. Jafari-Khouzania, K. Elisevichb, S. Patela, B. Smithc, and H. Soltanian-Zadeh. Flair signal and texture analysis for lateralizing mesial temporal lobe epilepsy. *NeuroImage*, 49(2):1159–1571, 2010.

[18] V. Jain and H. Zhangu. Robust 3D shape correspondence in the spectral domain. In *IEEE International Conference on Shape Modeling and Applications, IEEE Computer Society*, page 19, 2006.

[19] M. Jenkinson and S. Smith. A global optimisation method for robust affine registration of brain images. *Medical Image Analysis*, 5(2):143–156, 2001.

[20] Z. Karni and C. Gotsman. Compression of soft-body animation sequences. *Computers and Graphics*, 28(1):25–34, 2004.

[21] K. Karuna, N. Mangala, C. Janaki, S. Shashi, and C. Subrata. Galaxy workflow integration on garuda grid. In *IEEE 21st International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, pages 194–196, 2012.

[22] G. Kontonatsios, I. Korkontzelos, B. Kolluru, P. Thompson, and S. Ananiadou. Deploying and sharing u-compare workflows as web services. *Journal of Biomedical Semantics*, 4(7), 2013.

[23] E. Konukoglu, B. Glocker, A. Criminisi, and K. M. Pohl. Wesd–weighted spectral distance for measuring shape dissimilarity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(9):2284–2297, 2013.

[24] A. Kovnatsky, M. M. Bronstein, X. Bresson, and P. Vandergheynst. Functional correspondence by matrix completion. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 905–914, 2015.

[25] A. Kovnatsky, M. M. Bronstein, A. M. Bronstein, K. Glashoff, and R. Kimmel. Coupled quasi-harmonic bases. *Computer Graphics Forum*, 32(2):439–448, 2013.

[26] R. Lai, Y. Shi, I. Dinov, T. F. Chan, and A. W. Toga. Laplace-Beltrami nodal counts: a new signature for 3d shape analysis. In *IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, pages 694–697, 2009.

[27] Z. Lai, J. Hu, C. Liu, V. Taimouri, D. Pai, J. Zhu, J. Xu, and J. Hua. Intra-patient supine-prone colon registration in ct colonography using shape spectrum. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2010*, volume 6361, pages 332–339, 2010.

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

[29] S. Li, F. Shi, F. Pu, X. Li, T. Jiang, S. Xie, and Y. Wang. Hippocampal shape analysis of alzheimer disease based on machine learning methods. *AJNR Am J Neuroradiol*, 28(7):1339—-1345, 2007.

[30] C. Lin, S. Lu, X. Fei, A. Chebotko, D. Pai, Z. Lai, F. Fotouhi, and J. Hua. A reference architecture for scientific workflow management systems and the view soa solution. *IEEE Transactions on Services Computing*, 2(1):79–92, 2009.

[31] O. Litany, E. Rodolà, A. M. Bronstein, and M. M. Bronstein. Fully spectral partial shape matching. *Computer Graphics Forum*, 36(2):247–258, 2017.

[32] K. Maheshwari, P. Missier, C. Goble, and J. Montagnat. Medical image processing workflow support on the egee grid with taverna. In *Computer-Based Medical Systems*, 2009.

[33] A. Naitsat, S. Cheng, X. Qu, X. Fan, E. Saucan, and Y. Y. Zeevi. Geometric approach to detecting volumetric changes in medical images. *Journal of Computational and Applied Mathematics*, 329:37–50, 2018.

[34] T. Oinn, M. Addis, J. Ferris, D. Marvin, M. Senger, M. Greenwood, T. Carver, K. Glover, M. Pocock, A. Wipat, and P. Li. Taverna: a tool for the composition and enactment of bioinformatics workflows. *Bioinformatics*, 20(17):3045–3054, 2004.

[35] M. Ovsjanikov, M. Ben-Chen, F. Chazal, and L. Guibas. Analysis and visualization of maps between shapes. *Comput. Graph. Forum*, 32(6):135–145, 2013.

[36] M. Ovsjanikov, M. Ben-Chen, J. Solomon, A. Butscher, and L. Guibas. Functional maps: A flexible representation of maps between shapes. *ACM Trans. Graph.*, 31(4):30:1–30:11, 2012.

[37] N. Peinecke, F.-E. Wolter, and M. Reuter. Laplace spectra as fingerprints for image recognition. *Computer-Aided Design*, 39(6):460–476, 2007.

[38] P. Radau, Y. Lu, K. Connelly, G. Paul, A. Dick, and G. Wright. Evaluation framework for algorithms segmenting short axis cardiac MRI. In *The MIDAS Journal - Cardiac MR Left Ventricle Segmentation Challenge*, 2009.

[39] M. Reuter. *Laplace Spectra for Shape Recognition*. Books on Demand GmbH, 2006.

[40] M. Reuter. Hierarchical shape segmentation and registration via topological features of laplace-beltrami eigenfunctions. *International Journal of Computer Vision*, 89(2):287–308, 2010.

[41] M. Reuter, M. Niethammer, F. Wolter, S. Bouix, and M. Shenton. Global medical shape analysis using the volumetric laplace spectrum. In *International Conference on Cyber worlds, NASA-GEM Workshop*, pages 417–426, 2007.

[42] M. Reuter, N. Schmansky, H. Rosas, and B. Fischl. Within-subject template estimation for unbiased longitudinal image analysis. *Neuroimage*, 61(4):1402–1418, 2012.

[43] M. Reuter, F. Wolter, and N. Peinecke. Laplace-beltrami spectra as "shape-dna" of surfaces and solids. *Computer-Aided Design*, 38(4):342–366, 2006.

[44] M. Reuter, F. Wolter, M. Shenton, and M. Niethammer. Laplace beltrami eigenvalues and topological features of eigenfunctions for statistical shape analysis. *Computer-Aided Design*, 41(10):739–755, 2009.

[45] E. Rodolà, L. Cosmo, M. M. Bronstein, A. Torsello, and D. Cremers. Partial functional correspondence. *Computer Graphics Forum*, 36(1):222–236, 2017.

[46] E. Rodolà, A. Torsello, T. Harada, Y. Kuniyoshi, and D. Cremers. Elastic net constraints for shape matching. In *IEEE International Conference on Computer Vision*, pages 1169–1176, 2013.

[47] R. Rustamov. Laplace-beltrami eigenfunctions for deformation invariant shape representation. In *Fifth Eurographics symposium on Geometry processing*, pages 225–233, 2007.

[48] R. M. Rustamov. Laplace-Beltrami eigenfunctions for deformation invariant shape representation. In *the fifth Eurographics symposium on Geometry processing*, pages 225–233, 2007.

[49] R. M. Rustamov, M. Ovsjanikov, O. Azencot, M. Ben-Chen, F. Chazal, and L. Guibas. Map-based exploration of intrinsic shape differences and variability. *ACM Trans. Graph.*, 32(4):72:1–72:12, 2013.

[50] Y. Shi, R. Lai, R. Gill, D. Pelletier, D. Mohr, N. Sicotte, and A. Toga. Conformal metric optimization on surface (cmos) for deformation and mapping in laplace-beltrami embedding space. In *14th International Conference on Medical Image Computing and Computer-assisted Intervention - Volume Part II*, pages 327–334, 2011.

[51] Y. Shi, R. Lai, and A. W. Toga. Cortical surface reconstruction via unified reeb analysis of geometric and topological outliers in magnetic resonance images. *IEEE Transactions on Medical Imaging*, 32(3):511–530, 2013.

[52] Y. Shi, R. Lai, D. J. J. Wang, D. Pelletier, D. Mohr, N. Sicotte, and A. W. Toga. Metric optimization for surface analysis in the laplace-beltrami embedding space. *IEEE Transaction on medical imaging*, 33(7):1447–1463, 2014.

[53] S. Smith. Fast robust automated brain extraction. *Hum. Brain Mapp*, 17(3):143–155, 2002.

[54] A. Sotiras, C. Davatzikos, and N. Paragios. Deformable medical image registration: A survey. *IEEE Trans Med Imaging*, 32(7):1153–1190, 2013.

[55] W. Tan, R. Madduri, A. Nenadic, S. Soiland-Reyes, D. Sulakhe, I. Foster, and C. Goble. Cagrid workflow toolkit: A taverna based workflow tool for cancer grid. *BMC Bioinformatics*, 11(542), 2010.

[56] D. Tsuzukia, H. Watanabec, I. Danb, and G. Taga. Minr 10/20 system: Quantitative and reproducible cranial landmark setting method for mri based on minimum initial reference points. *Neuroscience Methods*, In Press, 2016.

[57] L. Wang, A. M. Fagan, A. R. Shah, M. F. Beg, J. G. Csernansky, J. C. Morris, and D. M. Holtzman. Csf proteins predict longitudinal hippocampal degeneration in early stage dementia of the alzheimer type. *Alzheimer Dis Assoc Disord*, 26(4):314—-321, 2012.

[58] T. Windheuser, U. Schlickewei, F. R. Schmidt, and D. Cremers. Large-scale integer linear programming for orientation preserving 3d shape matching. *Computer Graphics Forum*, 30(5):1471–1480, 2011.

[59] D.-M. Yan, B. Lévy, Y. Liu, F. Sun, and W. Wang. Isotropic remeshing with fast and exact computation of restricted voronoi diagram. *Computer Graphics Forum*, 28(5):1445–1454, 2009.

[60] C. Zarow, L. Wang, H. Chui, M. Weiner, and J. Csernansky. Mri showsmore severe hippocampal atrophy and shape deformation in hippocampal sclerosis than in alzheimer's disease. *International Journal of Alzheimer's Disease*, (483972), 2011.

[61] W. Zeng, L. M. Lui, L. Shi, D. Wang, W. C. Chu, J. C. Y. Cheng, J. Hua, S.-T. Yau, and X. Gu. Shape analysis of vestibular systems in adolescent idiopathic scoliosis using geodesic spectra. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2010*, pages 538–546, 2010.

[62] J. Zhang. Ontology-driven composition and validation of scientific grid workflows in kepler: a case study of hyperspectral image processing. In *GCC Workshops*, 2006.

[63] Z. Zhong, X. Guo, W. Wang, B. Lévy, F. Sun, Y. Liu, and W. Mao. Particle-based anisotropic surface meshing. *ACM Trans. Graph.*, 32(4):99:1–99:14, 2013.

[64] K. Zhou, J. Synder, B. Guo, and H. Shum. Iso-charts: Stretch-driven mesh parameterization using spectral analysis. In *Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, pages 45–54, 2004.

[65] G. Zou, J. Hu, X. Gu, and J. Hua. Authalic parameterization of general surfaces using lie advection. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2005–2014, 2011.

[66] G. Zou, J. Hua, Z. Lai, X. Gu, and M. Dong. Intrinsic geometric scale space by shape diffusion. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1193–1200, 2009.

ABSTRACT

3D SURFACE REGISTRATION USING GEOMETRIC SPECTRUM OF SHAPES

by

HAJAR HAMIDIAN

August 2019

Advisors: Dr. Jing Hua and Dr. Farshad Fotouhi

Major: Computer Science

Degree: Doctor of Philosophy

Morphometric analysis of 3D surface objects are very important in many biomedical applications and clinical diagnoses. Its critical step lies in shape comparison and registration. Considering that the deformations of most organs such as heart or brain structures are non-isometric, it is very difficult to find the correspondence between the shapes before and after deformation, and therefore, very challenging for diagnosis purposes.

To solve these challenges, we propose two spectral based methods. The first method employs the variation of the eigenvalues of the Laplace-Beltrami operator of the shape and optimize a quadratic equation in order to minimize the distance between two shapes' eigenvalues. This method can determine multi-scale, non-isometric deformations through the variation of Laplace-Beltrami spectrum of two shapes. Given two triangle meshes, the spectra can be varied from one to another with a scale function defined on each vertex. The variation is expressed as a linear interpolation of eigenvalues of the two shapes. In each iteration step, a quadratic programming problem is constructed, based on our derived spectrum variation theorem and smoothness energy constraint, to compute the spectrum variation. The derivative of the scale function is the solution of such a problem. Therefore, the final scale function can be solved by integral of the result from each step, which, in

turn, quantitatively describes non-isometric deformations between two shapes. However, this method can not find the point to point correspondence between two shapes.

Our second method, extends the first method and uses some feature points generated from the eigenvectors of two shapes to minimize the difference between two eigenvectors of the shapes in addition to their eigenvalues. In order to register two surfaces, we map both eigenvalues and eigenvectors of the Laplace-Beltrami of the shapes by optimizing an energy function. The function is defined by the integration of a smooth term to align the eigenvalues and a distance term between the eigenvectors at feature points to align the eigenvectors. The feature points are generated using the static points of certain eigenvectors of the surfaces. By using both the eigenvalues and the eigenvectors on these feature points, the computational efficiency is improved considerably without losing the accuracy in comparison to the approaches that use the eigenvectors for all vertices. The variation of the shape is expressed using a scale function defined at each vertex. Consequently, the total energy function to align the two given surfaces can be defined using the linear interpolation of the scale function derivatives. Through the optimization the energy function, the scale function can be solved and the alignment is achieved. After the alignment, the eigenvectors can be employed to calculate the point to point correspondence of the surfaces. Therefore, the proposed method can accurately define the displacement of the vertices. For both methods, we evaluate them by conducting some experiments on synthetic and real data using hippocampus and heart data. These experiments demonstrate the advantages and accuracy of our methods.

We then integrate our methods to a workflow system named DataView. Using this workflow system, users can design, save, run, and share their workflow using their web-browsers without the need of installing any software and regardless of the power of their computers. We have also integrated Grid to this system therefore the same task can be

executed on up to 64 different cases which will increase the performance of the system enormously.

AUTOBIOGRAPHICAL STATEMENT

Hajar Hamidian is a PhD candidate in Computer Science at Wayne State University. She received her Master's degree (2008) in Electrical Engineering from University of Tehran in Tehran, Iran. She received her Bachelor's degree (2005) in Biomedical Engineering from Shahed University in Tehran, Iran. Her research interests include 3D Visualization, Computational Modeling, 3D Shape Analysis, and Image Processing.