VISUALIZING INTRINSIC ISOSURFACE VARIATION DUE TO UNCERTAINTY THROUGH HEAT KERNEL SIGNATURES

by

Nazmus Saquib

A thesis submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Master of Science

CES Program

The University of Utah

THE UNIVERSITY OF UTAH GRADUATE SCHOOL

SUPERVISORY COMMITTEE APPROVAL

of a thesis submitted by

Nazmus Saquib

This thesis has been read by each member of the following supervisory committee and by majority vote has been found to be satisfactory.

	Chair: Mike Kirby
	Christopher R. Johnson
	Suresh Venkatasubramanian

THE UNIVERSITY OF UTAH GRADUATE SCHOOL

FINAL READING APPROVAL

To the Graduate Council of the University of Utah:

I have read the thesis of Nazmus Saquib in its final form and have found that (1) its format, citations, and bibliographic style are consistent and acceptable; (2) its illustrative materials including figures, tables, and charts are in place; and (3) the final manuscript is satisfactory to the Supervisory Committee and is ready for submission to The Graduate School.

Date

Mike Kirby
Chair, Supervisory Committee

Approved for the Major Department

Martin Berzins
Chair/Dean

Approved for the Graduate Council

David Kieda
Dean of The Graduate School

ABSTRACT

It is common to extract isosurfaces from simulation field data to visualize and gain understanding of the underlying physical phenomenon being simulated. As the input parameters of the simulation change, the resulting isosurface varies, and there has been increased interest in quantifying and visualization of these variations as part of the larger interest in uncertainty quantification. In this thesis, we propose an analysis and visualization pipeline for examining the *intrinsic* variation in isosurfaces caused by simulation parameter perturbation. Drawing inspiration from the shape modeling community, we incorporate the use of heat-kernel signatures (HKS) with a simple finite-difference approach for quantifying the degree to which a region (or even a point) on an isosurface has undergone intrinsic change. Coupled with a clustering technique and the use of color maps, our pipeline allows the user to select the level of fidelity with which they wish to evaluate and visualize the amount of intrinsic change. The pipeline is described with a simple example to walk the reader through the different steps, and experimental validation of parameter choices in the pipeline is provided to justify our design. Then we present canonical and simulation examples to demonstrate the pipeline's use in different applications.

CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Numerical simulation as a tool for solving science and engineering problems has become ubiquitous. With the growth of simulation science, there has arisen a renewed emphasis on Validation, Verification and Uncertainty Quantification (VVUQ) within the Computational Science and Engineering (CS&E) community. Simulation scientists are interested not only in quantifying errors that come as a consequence of their modeling and discretization choices, but also in quantifying the uncertainty in the solution (and subsequent derived quantities) dictated by other simulation choices such as the simulation input parameters (*e.g.* material parameters in a structural mechanics simulation). There is a corresponding need for visualization techniques that both specialize in visualizing the output of numerical simulations and are designed to aid the simulation scientists in answering their uncertainty quantification questions. Isosurface extraction (level-sets of a function evaluated at a particular value) is a common visualization technique used for understanding the underlying phenomena expressed by a numerical simulation. From the perspective of simulation uncertainty quantification, a common question posed by simulation scientists is: *given a set of input parameters and given an isosurface extracted from a field generated from a specific setting of the parameters, what is the variation in the isosurface due to parameter perturbations?* Here, we primarily aim to address the geometric variation of isosurfaces.

The visualization science community has taken some approaches in the last decade or so that attempt to answer the above question, as described in Chapter 2. Some of these methods are successful in answering parts of our design criteria for a visualization pipeline that addresses the question above.

In our desired visualization system and related investigation, we aim to meet the following goals and requirements:

- Characterize the *intrinsic* change in isosurfaces due to perturbation. As discussed in more detail in Chapter 2, in a broad sense, intrinsic variations involve changes in the

metric structure of an object (in our case, the isosurface) versus extrinsic changes which deal with the embedding of the object in Euclidean space (8). Our work complements and extends recent work in uncertainty quantification for isosurfaces that has focused primarily on *extrinsic* changes in isosurfaces.

- Identifying an accurate way to generate an isosurface from a simulation, such that the calculations done in the later stages to quantify and visualize the changes in isosurfaces are consistent and not misleading. The choices made in each step of the pipeline should also be verified to establish this robustness criteria further.

- An effective way to communicate the intrinsic quantitative change in isosurfaces by means of visualization. Several works have addressed the problem of visualizing the change or the comparative difference between two or more surfaces (refer to chapter 2). However, our pipeline should complement the current methods by visualizing an entity that will reveal *intrinsic* changes in isosurfaces produced from a simulation due to a perturbation.

- The investigation should choose a metric that will quantify the intrinsic change in isosurfaces and will meet the requirements stated in the previous points. The metric should be robust and informative, and at the same time efficient to calculate.

In summary. our main goal is to visualize a local, quantitative and robust measure of intrinsic variation between isosurfaces. The visualization pipeline should also take into account the following considerations.

- The pipeline should not take more than a *reasonable* amount of time to calculate and produce the visualization. Here, by *reasonable* we mean a comparative timescale according to the precision and complexity of the simulation output.

- The visualization should employ effective and easy means of showing the change in isosurface that is easy for a simulation scientist to perceive. In our case, we focus on effective color mapping using clustering to denote the areas that change. We justify our choice of clustering that will convey the most amount of information to the user.

There are some limitations that we must take into account when proposing a solution for the stated problem.

- We assume that the parameter perturbation does not cause any topology change in the isosurface. For example, a sphere may evole into a bulged ellipse, but our pipeline cannot handle scenarios where the isosurface changes from a sphere to a hollow cylinder. This restriction is posed mainly due to our investigation on intrinsic variation explicitly. This will be further explained in Chapter 2.

- We also assume that a correspondence between the two (initial and perturbed) isosurfaces exist, and it can be computed trivially. The mapping between the two surfaces needs to exist because we would like to quantify pointwise intrinsic change between them.

In order to meet the stated goals and criteria for a visualization pipeline, we propose the following pipeline. The key idea of this work is to construct a signature (essentially a function sampled to generate a high dimensional vector) at each point of the isosurface with the property that this signature captures the intrinsic geometry around that point. By comparing the signatures from corresponding points on different isosurfaces using a finite difference approach, we can then quantify the change in shape between the surfaces at that point. This then allows us to identify points (and regions) where large changes have occurred. The signature we use is the *heat-kernel-signature* (HKS) (51), first developed in the shape modeling community. Once this is done, we develop a framework based on clustering and the use of colormaps to visualize the regions that have different change characteristics. The relevant choices of *tunable* parameters (and particular methods that were chosen from a pool of candidates) in this pipeline are validated using experiments to address the goals stated above. To evaluate such a pipeline's effectiveness and usefulness, we demonstrate it on some real world applications where the visualization will shed some light on answering a science question.

Our overall analysis pipeline starts with a given parametrized procedure for generating isosurfaces. We demonstrate step by step how to extract a high quality mesh, compute robust signatures, and then visualize the difference as described above.

The thesis is structured as follows. In Chapter 2, we present the relevant previous work. We focus our review on two areas: understanding the position of our work in light of the current uncertainty visualization literature and on relevant concepts from shape analysis upon which this work is built. In Chapter 3 we provide a brief review of the mathematical concepts used in our work. We then proceed in Chapter 4 to present

our analysis and visualization pipeline. We demonstrate our pipeline by walking the reader through a canonical example in detail. The section is complemented by a series of experiments to verify and validate the choice of parameters in the pipeline. In Chapter 5 we provide examples of using our pipeline for understanding simulation results. We summarize and conclude in Chapter 6.

CHAPTER 2

RELEVANT WORK

In this chapter, we review the two relevant research fields that are related to our problem formulation and solution. The current research developments in Uncertainty Visualization are described, along with the development of intrinsic shape signatures and shape matching algorithms in the Shape Analysis community.

2.1 Uncertainty Visualization

Uncertainty visualization has been cited as a key new challenge for visualization research(24). Uncertain data that arise from physical measurements have been addressed using fuzzy sets (56). (32) also addressed this issue for ensemble data sets. Texture and/or color are employed to denote uncertainty through a modification of a direct volume rendering technique in (14). Other proposed methods to visualize surface uncertainty include *fat surfaces* (37), likelihood and confidence maps (38), and point primitives for rendering uncertain isosurfaces (19).

Special attention has been paid to isosurfaces generated from uncertain scalar fields. In the visualization community, the concept of *Level Crossing Probability* (LCP) was introduced to quantify and visualize uncertain scalar fields (43), where they introduced a model for uncertain spatial data and the corresponding spatial distribution of uncertain isocontours. As an extension of this method, a probabilistic version of the marching cubes algorithm was proposed in (44), where the authors take account of scalar field in which the data points are possibly correlated. Pfaffelmoser *et al.*(40) proposed the *isosurface-first-crossing-probability* (IFCP) algorithm as an efficient way to calculate the probability incrementally along a ray cast through a correlated random variable field.

All of these works ask a fundamental and central question: how does the isosurface geometry (both extrinsic and instrinsic properties) change as an error or a perturbation is introduced in the input scalar field? The question has been partially answered by both (43) and (40) by building mathematical models of uncertainty arising due to such pertur-

bations. (43) defines numerical condition for isocontours, that describes the sensitivity of isocontours to perturbations in the scalar field. (40) has shown how such sensitivity can be calculated and rendered. However, certain aspects of the fundamental question asked above have not been answered yet. For example, although effects of perturbations in the scalar field has been studied, a study of isosurface sensitivity due to a parameter perturbation in the simulation has not been done. Moreover, the investigations that are carried out so far has taken an extrinsic approach when dealing with isosurface geometry. Although there are some advantages of using an extrinsic measure to capture isosurface sensitivity, certain advantages are endowed if an intrinsic measure is used, which will be discussed in Section 2.1.2. The works done so far do not contribute towards an understanding of the pointwise difference map between surfaces, that may tie a loose end on quantifying the surface difference or sensitivity accurately. A pointwise correspondence may provide useful insights for an accurate quantification of the isocontour uncertainty arising due to a perturbation. To our knowledge, an attempt has not been taken so far to address the issues we mention above.

2.1.1 Comparative Visualization of Surfaces

Our problem of quantifying variation in isosurfaces due to a perturbation in a parameter in the simulation results in comparative surface visualization, as the variations induced by the perturbation causes a change in the shape of the isosurface. Our end goal is to effectively visualize the changes that occur during this process. Many efforts have been taken in the visualization community to compare and visualize the difference (or similarity) between surfaces. We review a few different categories of techniques that are available in the research literature.

- In the area of shape and image retrieval, several methods have been proposed to quantify surface difference, e.g. (54),(29),(36). However, many of these measures are global in nature. Global signatures such as the Hausdorff distance are used in these comparisons. We note that the visualization of local features may yield important insights that may not be very obvious from global comparisons.

- To visualize local changes in shapes, methods like comparative local curvature maps (17) are proposed. In order to visualize pointwise difference between shapes, some methods establish different kinds of correspondence between surfaces, e.g. (34),(30).

However, there are only a very few that establish pointwise correspondence between surfaces to effectively quantify the difference in shape, e.g. (41).

- Instead of comparing any two isosurfaces, a group of isosurfaces can be compared in an information theoretic sense, such as (10). Such algorithms have applications in finding better isovalues and grouping similar isosurfaces together.

- There are many efforts taken to tackle the problem from the point of view of perception. One way is to render the surfaces transparently and overlay them in visualization, e.g the work done by Tory *et al.* (52) and Johnson and Sanderson (25). This helps provide an understanding of the contextual information. However, these do not explicitly show crucial information about the change in properties like curvature. Such a task is often left to the user. To improve on these techniques, textures have been used to improve the shape perception of transparent surfaces. The work done by Interrrante *et al.* (21) show directions of the principal curvature on a surface. Bair and House (2) used grid textures to aid in the perception of surface shape. Alternatively, Weigle *et al.* (55) and Busking *et al.* (11) use *constructive solid geometry* (CSG) operations to solve the perception problem of inside/outside classification.

Although all of these methods are useful perception wise, they do not provide an accurate pointwise difference map to understand exact intrinsic variations in surfaces. For our problem formulation, it is important to visualize the difference in such level to accurately find out the effect of parameter perturbation on isosurfaces.

2.1.2 Intrinsic versus Extrinsic Variation

In a broad sense, intrinsic variations involve changes in the metric structure of an object (in our case, the isosurface) versus extrinsic changes which deal with the embedding of the object in Euclidean space (8). In other words, when studying extrinsic variation between shapes or surfaces, the ambient space surrounding the object of interest is explicitly used to calculate relationships between the underlying surface properties. For an intrinsic measure, the geometric properties are calculated individually based on the metric structure of the shapes and compared through some correspondence between them.

Traditionally, the shape similarity problem has been handled in shape modeling community, either implicitly or explicitly, from an extrinsic point of view (for example, (27),

(22)). All the previous work described on the comparative visualization of surfaces also rely on extrinsic measures according to the definition of extrinsic variation we have given above. For our problem formulation, we need a robust way to characterize any change in surfaces through pointwise correspondence. As noted before, global extrinsic measures like Hausdorff distance do not provide such quantification. A local difference measure like (41) does find pointwise correspondence and visualize the difference between intersecting surfaces by solving a Laplace equation in the space bound by the surfaces. So, such measures only provide us with extrinsic variations. The comparative visualization based on CSG operations ((55), (11)) rely on the ambient space in between the intersecting surfaces, thus this class of methods are also extrinsic in nature.

Although extrinsic variations have been used to characterize shape differences for a long time, recently the shape modeling community has shown a growing interest in intrinsic measures. There are a few flexibilities that are naturally introduced by the virtue of this point of view.

Intrinsic similarity was explored in Elad and Kimmel's paper (16), and since then there has been a rising interest in the shape modeling community to define instrinsic variation based shape signatures.

The main advantage of employing an intrinsic signature to study shapes is this measure's insensitivity to deformations that can be approximated by isometries, since isometries maintain the metric structure of the shape. Extrinsic variation measures are unsuitable for analyzing non-rigid objects that have high range of flexibility. Intrinsic variations, on the other hand, are perfectly suitable in such situations. Isosurfaces that undergo such deformations due to a parameter perturbation cannot be easily analyzed with any extrinsic measure. Figure 2.1 shows a few typical shapes that can be used as an example of the superiority of an intrinsic measure when the shapes are highly flexible.

Moreover, some of the extrinsic methods described above work only in the case of intersecting surfaces. Often, parameter perturbations cause an isosurface to transform in such a way (e.g. translation) that these methods cannot be used effectively. Methods like (41) only seem to work with closed set of surfaces in order to calculate correspondence. These issues arise due to the nature of the extrinsic point of view.

Intrinsic measures can also help in identifying and grouping isosurfaces that are similar in metric structure and shape. Achieving this will let us find parameter clusters or ranges in the simulation that give rise to different behaviors of isosurface. Although this is one

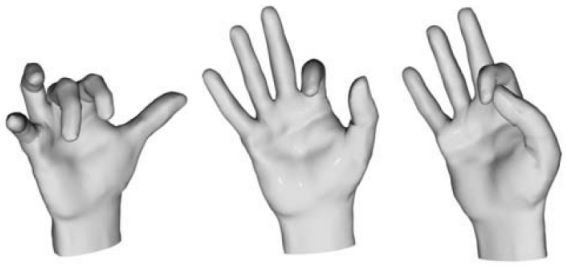


Figure 2.1. An example of comparison between the advantages of intrinsic and extrinsic measure. The right and middle shapes are suitable for comparison using an extrinsic measure as the fingers touch and create a topology change, so no intrinsic measure can be used. However, in the general case (left and middle shapes) where the shapes have gone through quite a lot of deformations (but not a topology change), intrinsic measure is the right way to quantify variations. Picture courtesy of (8).

of our future goals, we note that it is often quite hard to group a set of surfaces using only extrinsic measures.

Therefore, in this paper we introduce a pipeline that relies on an intrinsic measure, the HKS, to quantify the difference between a base surface and a perturbed surface. This approach will complement the extrinsic study done in UQ visualization community regarding quantification of isosurface sensitivity with input perturbations, and will take advantage of the intrinsic point of view to handle isosurface geometry effectively. Moreover, we introduce pointwise correspondence between surfaces and tie this correspondence with our intrinsic sensivity measure to answer the central question formulated in Chapter 1.

2.2 Shape Matching and Correspondence

Shape analysis is a vast area spanning disciplines such as graphics, vision, geometric modeling, computational geometry, and structural biology. Shape matching research is important in structure detection, symmetry matching, feature points selection etc. problems, just to name a few. In any such application, shapes are defined to be similar if they have an isometric transformation between them. Thus the main research problem in the shape matching community revolves around developing shape signatures that are invariant under isometric transformations. The signatures developed so far can be broadly classified into two categories: those that are invariant under rigid motion, and those that are invariant under non-rigid motion. For rigid-motion invariant methods, local point signatures are taken into account by many, whereas for non-rigid motion, global shape

matching algorithms have been proposed. While it is impossible to survey the numerous approaches to shape analysis in the literature, we focus here on specific methods for constructing *invariants* or *signatures* for shape.

A key idea in shape analysis is generating signatures for a shape that remain invariant under "natural" transformations, and thus capture intrinsic aspects of the shape. For example, there have been numerous approaches to defining signatures that are invariant under rigid transformations (rotations/translations/reflections). These include producing neighbor shape distribution representations (12; 6), spin images (23) or shape context (6).

Shape invariants for non-rigid motions (for example, the flexing of joints) have also been investigated. Integral invariant signatures suitable for global shape matching were proposed by (33). A popular technique to capture intrinsic geometry and suppress topological noise due to geodesic distance based signatures is to use the Laplace-Beltrami operator on the manifold. This technique was introduced in *e.g.*, (28), (48) and more recently in (51) where the heat-kernel signature (HKS) was introduced. A scale invariant version was proposed in (9). The HKS is intrinsic and isometry-invariant, thus two isometric shapes would have the same HKS. HKS is multi-scale in nature, and hence both local and global features are detected using this signature. HKS has been proven useful in finding significant features on a shape, and partial and global matching of deformable shapes.

Finding correspondence between two similar shapes (in both extrinsic and intrinsic sense) has been a core focus of research in the shape research community. We describe the key categorization of calculating correspondence here, along with their suitability for our stated problem's solution.

For our stated problem, a preferable intrinsic method of correspondence should satisfy one or more of the following criteria

- The method is as much robust as possible to big deformations,
- A correspondence is defined for all the elements (vertices or faces) of the shape. This is called dense correspondence (as opposed to sparse correspondence).
- It can be probably a partial correspondence method (see the figure 2.2 below for example) instead of full correspondence. An associated problem is that partial correspondence is usually calculated for a set of feature points only.

Figure 2.2. An example of the partial correspondence problem. The goal is to match the similar regions between the two shapes while ignoring the parts in green and blue. Picture courtesy of (53).

- If possible, then the method can produce group correspondence. The group correspondence problem is to find what structures/parts are common to a group of shapes, and which parts can be characterized as not belonging to the group. Such correspondence methods can be useful if we are trying to characterize a family of isosurfaces.

- One (future) extension can be time-varying registration. Time-varying registration is the registration of (possibly continuously deforming) surfaces that are acquired over several frames. This sort of goes with the idea of finding correspondence in a family of isosurfaces produced due to the change of parameter(s).

Many of these methods use the Laplace-Beltrami shape descriptor to find correspondence. Usually these are similarity based correspondence, i.e. in most cases they end up solving a minimization problem. Some of the methods that make use of this direction end up finding only sparse correspondence due to the heavy computation time for optimization. If we consider a method that finds partial correspondence (or forcing a partial correspondence algorithm to find full correspondence), then two possible approaches can be:

1. A series of candidate correspondences is computed and votes are cast on the pairwise assignment that constitute each candidate. At the end, highest number of votes for certain correspondence emerges.

2. Graph-based approach: feature points on a shape can be thought of nodes in a graph, where every pair of nodes is connected with an edge whose weight is proportional to some geometric quantity (e.g. distance between the nodes). Some methods use this formulation to solve subgraph isomorphism and keep the nodes that are not removed during the editing of the graph.

In our problem statement, we have mentioned that we would like to visualize the difference between the isosurfaces for each point present in the base surface for comparison. From that point of view, we now discuss the problems of incorporating any of the above correspondence method.

- While finding full correspondence, some methods cannot assign a correspondence to every point.
- Some methods rely more on statistics and only give approximations.
- Some of the methods described above are quite expensive to calculate.

Given the shortcomings of the available correspondence method, we may have to resort to other customized methods that suit our needs.

CHAPTER 3

MATHEMATICAL BACKGROUND

We will assume without loss of generality that we are given the results of a numerical simulation as a real-valued function $s(\mathbf{x})$ over a domain $\Omega \subset \mathbb{R}^3$. Let us assume our simulation is a function of (at least) one real-valued parameter α to which we want to quantify variation in the output due to perturbation of our parameter. For some small perturbation of α, let $\mathcal{M}_1$ and $\mathcal{M}_2$ denote the "exact" isosurfaces that exist with the data for α_1 and α_2; let $\mathcal{T}(\mathcal{M}_1)$ and $\mathcal{T}(\mathcal{M}_2)$ denote isosurface triangulation extracted from the simulation output respectively. The vertices of the tessellation are assumed to lie on the manifold that the tessellation is approximating; further refinement of the tessellation produces a more accurate representation of the manifold. In this work, we assume that the variation in the field due to perturbation of our parameter does not cause gross topological changes of our isosurface.

3.1 Laplace-Beltrami operator and HKS

Heat Kernels: We define the amount of heat at time t at a point $\mathbf{x} \in \mathcal{M} \subset \mathbb{R}^3$ on a compact Riemannian manifold $\mathcal{M}$ without boundaries as $u(\mathbf{x}, t) : \mathcal{M} \times \mathbb{R}^+ \to \mathbb{R}^+$. Assuming we have an initial heat distribution $f : \mathcal{M} \to \mathbb{R}^+$ at time $t = 0$, $u(\mathbf{x}, t)$ is then a solution of the heat equation

$$\frac{\partial u}{\partial t} = \nabla^2_{\mathcal{M}} u$$

where $\nabla^2_{\mathcal{M}}$ denotes the Laplace-Beltrami (LB) operator (a generalization of the Laplacian to manifolds where derivative are in manifold coordinates), and the condition

$$\lim_{t \to 0} u(\mathbf{x}, t) = f(\mathbf{x})$$

is satisfied. The heat operator H_t is defined as $u(\mathbf{x}, t) = (H_t f)(\mathbf{x})$ where $H_t : L^2 \to L^2$ with L^2 being the space of all smooth, square integrable functions on $\mathcal{M}$.

This operator is directly related to the solution of the heat equation. The solution to the heat equation can be defined in terms of a *heat kernel*. The heat kernel $k_t^{\mathcal{M}}(\mathbf{x}, \mathbf{y}) : \mathbb{R}^+ \times \mathcal{M} \times \mathcal{M} \to \mathbb{R}^+$ on $\mathcal{M}$ for $\mathbf{x}, \mathbf{y} \in \mathcal{M}$ is a function such that for all $f \in L^2$, $t > 0$:

$$u(\mathbf{x}, t) = \int_{\mathcal{M}} k_t^{\mathcal{M}}(\mathbf{x}, \mathbf{y}) f(\mathbf{y})\, dA.$$

The physical intuition of the heat kernel(18; 20) is that it gives the amount of heat transferred from a point $\mathbf{x}$ on $\mathcal{M}$ to point $\mathbf{y}$ on $\mathcal{M}$ in time t. These ideas are well described in, *e.g.*, (18) and (20). For a compact $\mathcal{M}$, the heat kernel has the following eigendecomposition:

$$k_t^{\mathcal{M}}(\mathbf{x}, \mathbf{y}) = \sum_{i=0}^{\infty} e^{-\lambda_i t} \phi_i(\mathbf{x}) \phi_i(\mathbf{y})$$

where λ_i is the i^{th} eigenvalue and ϕ_i is the i^{th} eigenfunction of the LB operator.

The heat kernel has some essential properties that makes it useful in shape analysis. It is symmetric, *i.e.* $k_t^{\mathcal{M}}(\mathbf{x}, \mathbf{y}) = k_t^{\mathcal{M}}(\mathbf{y}, \mathbf{x})$. Other important properties include isometric invariance, multi-scalability and stability, as proved in (51).

Heat-Kernel Signature: The heat-kernel signature (HKS) was first proposed by Sun *et al.* (51) as a local shape descriptor. It is the diagonal of the heat kernel $k_t^{\mathcal{M}}(\mathbf{x}, \mathbf{y})$

$$HKS(\mathbf{x}) : \mathbb{R}^+ \to \mathbb{R},\ HKS(\mathbf{x}, t) = k_t^{\mathcal{M}}(\mathbf{x}, \mathbf{x}).$$

The HKS is invariant under isometric deformations of $\mathcal{M}$ and is lossless: the HKS vectors for all points on a surface uniquely define the surface (up to isometry). The computation of the HKS relies on the computation of the eigenvalues and eigenfunctions of the LB operator, which is a well studied topic with many efficient algorithms available in the literature.

Representation: At any point $\mathbf{x} \in \mathcal{M}$, the HKS is a function of t. In practice, we will represent the HKS by sampling it at a finite set of values of t. The resulting object can be viewed as either a functional approximation or a vector (we will switch viewpoints as appropriate). In either case, we will visualize the HKS as a plot of $HKS(x, t)$ versus t, and use different curves to represent different values of x. Figure 3.1 illustrates two HKS vectors for two different points on a dimpled sphere (see Figure 4.1 for an example) – the blue line denoting a point on the sphere away from a dimpled area, and the red line denoting a point on the dimple itself.

In order to calculate the HKS on a tessellated surface, the discrete LB operator needs to be constructed over the tessellation $\mathcal{T}(\mathcal{M})$. In practice, tessellations may not be very

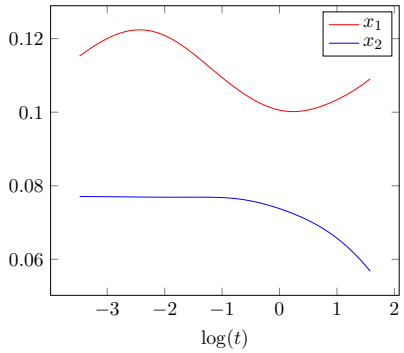


Figure 3.1. Visualizing the HKS.

smooth, depending on the type of isosurface extraction technique employed. In order to increase the accuracy of computation, we require a tessellation that respects the curvature of $\mathcal{M}$ (45).

3.2 Discretization Schemes

In this section, we describe the formation of different discretizations of the LB operator.

Graph Laplacians: An important line of research involves using a *graph* Laplacian as an approximation for the LB operator. In general, the idea behind the graph Laplacian discretization involves creating a weighted graph based on neighborhood information of each vertex, $\mathbf{x}_1, ..., \mathbf{x}_k$ for k points, from $\mathcal{T}(\mathcal{M})$.

There are two variations on building such graphs from $\mathcal{T}(\mathcal{M})$. One is to use a ϵ-neighborhood, where nodes (vertices) i and j are declared connected if

$$\|\mathbf{x}_i - \mathbf{x}_j\|^2 < \epsilon, \epsilon \in \mathbb{R},$$

and the norm is usually the standard Euclidean norm. This is geometrically motivated and intuitive. Another way to choose neighbors is to take the m nearest neighbors (for some m) in the 1-ring neighborhood (or within a Euclidean distance cutoff), which is less geometrically intuitive. After an adjacency graph of size $k \times k$ is created from either of the methods, a $k \times k$ weight matrix W is formed by assigning weights in the (i, j) entry if nodes i and j are connected (according to the adjacency matrix).

Once W is calculated, a diagonal weight matrix D is formed by summing the rows (or columns, since W is symmetric) of W,

$$D_{ii} = \sum_j W_{ji}.$$

The discrete Laplacian matrix L is then given by

$$L = D - W.$$

It is a symmetric, positive semi-definite matrix that can be thought of as an operator on functions defined on vertices of the graph.

Belkin *et al.* (3) were among the first to use the graph Laplacian as a discrete approximation for the LB operator on the manifold. Belkin *et al.*(4) also propose a differently weighted graph Laplacian (the mesh Laplacian). Other weighting schemes include the cotangent approximation (42). The Laplacian can also be estimated from point cloud data directly (without meshing) (5).

In order to understand the different weighting schemes, here we describe the theory behind the above approximation methods. Let $M = (V, E, F)$ be a triangular mesh with n vertices, where V is the set of vertices, E is the set of edges and F is the set of faces. The vertices are represented by the cartesian coordinates $\boldsymbol{v_i} = (x_i, y_i, z_i)$. The heart of the idea of the discrete Laplacian stems from *differential* coordinates of a vertex $\boldsymbol{v_i}$, from hereby called δ-coordinates. The δ-coordinates of a vertex $\boldsymbol{v_i}$ is defined as the difference between the absolute coordinates of $\boldsymbol{v_i}$ and the center of mass of the immediate neighbors.

$$\delta_i = (\delta_i^{(x)}, \delta_i^{(y)}, \delta_i^{(z)}) = \boldsymbol{v_i} - 1/d_i \sum_{j \in N(i)} \boldsymbol{v_j}$$

where $N(i) = \{j|(i,j) \in E\}$ and $d_i = |N(i)|$ is the number of immediate neighbors of i^th vertex. The neighbors can be chosen according the methods described above.

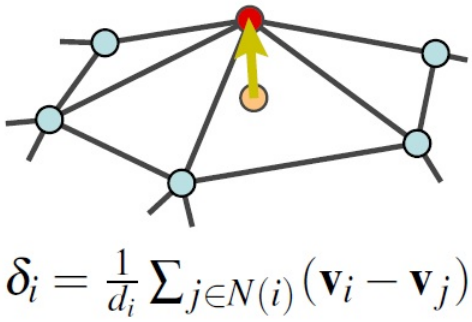


Figure 3.2. *Differential* coordinate is formed from the local cartesian coordinates (50).

The Laplacian matrix is built as a transformation of the local coordinates to the *differential* coordinates system. Let A be the adjacency (connectivity) matrix of the given mesh M.

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

Let D be the diagonal matrix such that $D_{ii} = d_i$. Then the Laplacian can be defined as the transformation matrix that transforms the absolute coordinates to relative coordinates.

$$L = I - D^{-1}A$$

In the summary formulation given at the beginning of the section, we have shown the symmetric version of the Laplacian. The symmetric version is found by multiplying the D matrix with the discrete Laplacian above.

$$DL = L_s = D - A,$$

where

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

In other words, if we apply the matrix L_s on the vector of x, y and z coordinates of V, then we will obtain the differential coordinates. $L_s\boldsymbol{x} = D\delta^{(x)}$, $L_s\boldsymbol{y} = D\delta^{(y)}$ and $L_s\boldsymbol{z} = D\delta^{(z)}$. The matrix L_s (or L) is called the graph Laplacian (that we have defined at the beginning of the section). In this simple derivation, we have used a weighting scheme of 1 or 0 (i.e. if a vertex is considered a neighbor then 1 is multiplied with with the difference between the current vector and the neighbor vector). In other words, we can rewrite the differential coordinate of the i^{th} vertex as

$$\delta_i = 1/d_i \sum_{j \in N(i)} w_{ij}(\boldsymbol{v_i} - \boldsymbol{v_j})$$

where w_{ij} is the associated weight of the differential coordinates.

There are many different weight schemes (other than the simple uniform weighting approach) proposed that uses additional information for faster convergence and more accurate representation of the discrete Laplace-Beltrami operator. A *cotangent* weight scheme is proposed by Meyer *et al.*(**?**) and Pinkall and Polthier (42) that uses angle information between the edges.

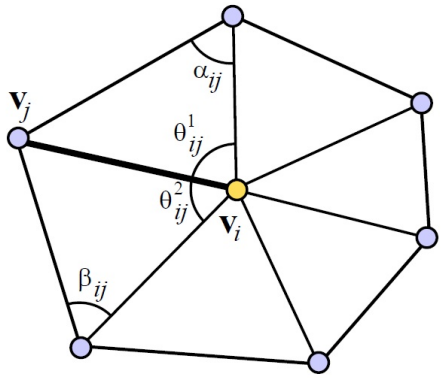


Figure 3.3. The angles used to calculate the cotangent weights for the differential coordinates (50).

$$\delta_i = 1/|\omega_i| \sum_{j \in N(i)} 1/2(cot(\alpha_{ij}) + cot(\beta_{ij}))(\boldsymbol{v_i} - \boldsymbol{v_j})$$

where $|\omega_i|$ is the size of the Voronoi cell of i and α_{ij} and β_{ij} are the angles opposite to the edge (i,j). These weights approximate the mean-curvature normals, as shown by the authors. This has certain advantages and disadvantages. The advantages are that the differential coordinates now only have normal components. With the uniform weighting scheme, we would get tangential components too. These geometry aware weights thus reflect the mean-curvature information in a succint but descriptive way.

There are other weighting schemes proposed that takes account of faster convergence, inexpensive calculations and geometry aware accuracy. The Finite Element Method (FEM) provides a better convergence and accuracy at a low cost. This is essentially a different weighting scheme that is more complicated to calculate compared to the above methods.

Finite Element Methods: A first-order finite element method (FEM) approximates the LB operator by calculating linear weights for each vertex in $\mathcal{T}$ and later using them to calculate a solution for each triangle. Suppose the two-dimensional surface $\mathcal{M}$ is twice differentiable $\left(C^2\right)$ and we have a parameterization of $\mathcal{M}$ such that: $X(u) = \{x_1(u), x_2(u), x_3(u) : u = (u^1, u^2) \in D\}$ for some planar domain D. The LB operator acting on a function F is defined on this parametrized surface X as

$$\nabla^2_X F = \frac{1}{|g|^{1/2}} \sum_{i,j=1}^{2} \frac{\partial}{\partial u^i} \left(|g|^{1/2} g^{ij} \frac{\partial F}{\partial u^j} \right)$$

where the inner products g^{ij} denotes the Riemannian metric tensor.

$$F(x,t) = \sum_{i=1}^{N_T} F_i(x,t)$$

where N_T is the total number of triangles in our discretization. It is shown (13) that after calculating linear weights w_i for each i^{th} vertex from its neighboring triangles, the LB operator is given by a form

$$\hat{\nabla}^2 F(p) = \sum_{i=1}^{m} w_i (F(p_i) - F(p))$$

where $p_1, .., p_m$ are the m neighboring vertices of the vertex p.

The approximation to the LB operator is generated by *assembling* (in the FEM sense) the discretization of the individual parametrized surfaces into a global linear system. Details of such a derivation can be found in (13).

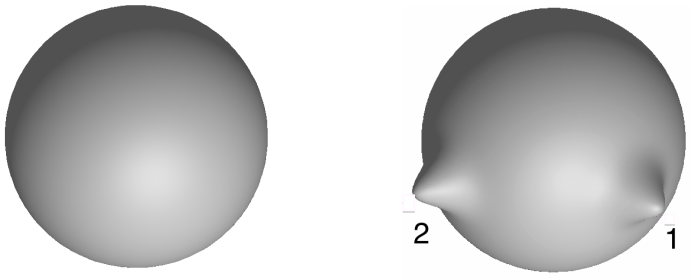
CHAPTER 4

PROPOSED FRAMEWORK

With the mathematical formulation of HKS and implementation techniques described previously, in this chapter we present our proposed pipeline which demonstrates how intrinsic changes in curvature information can be captured by the HKS and visualized using data clustering techniques. Several empirical choices have been made in the pipeline that we justify using experimental evidence. Furthermore, we discuss the limitations and constraints imposed by the design decisions made for this framework.

4.1 Pipeline Description

The proposed pipeline (see Figure 4.2) has several steps that we discuss in the following sections. As a working example that will be used throughout this section, we will demonstrate our method on two shapes: one represents our "base" surface and the second one represents our perturbed surface. The shapes are a sphere and the same sphere with two "dimples" that represent perturbation by a parameter α. These shapes are shown in Figure 4.1.



(a) Unperturbed Shape: Sphere

(b)Perturbed Shape: A sphere with dimples.

Figure 4.1. Unperturbed and perturbed shapes that will be used as a working example throughout this section.

These spheres demonstrate a hypethetical situation where a perturbation has been made to the original sphere shape. In this example, we assume that a sphere has been produced as a zero level-set of a simulation's output scalar field, and upon intriducing a perturbation α in the simulation, the zero level-set changes to the perturbed shape. In reality, we have only created a perturbation in the mesh using a custom tool that will be explained in the next section. Although this is not an accurate portrayal of the problem statement, we have chosen this simple example for demonstration.

Figure 4.2 shows the conceptual flowchart of our proposed framework.

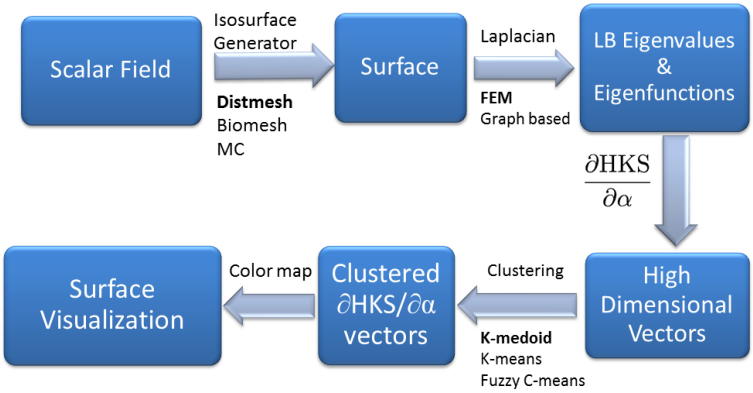


Figure 4.2. A conceptual flowchart of our proposed pipeline.

In the following subsections, we will describe each step of the pipeline, along with the design decisions made for each.

4.1.1 Isosurface Generation

Isosurfaces can be generated from 3D scalar fields by various methods, and each approach has their advantages. Our pipeline relies on computing the eigenstructure of the (discrete) Laplace-Beltrami operator on the surface. Therefore, it is important that the meshing approach be *curvature-sensitive*, providing more resolution in regions of higher curvature. However, in order for the approach to be efficient, it should be *adaptive*, generating a coarser mesh in regions of low curvature.

The Marching Cubes approach (31) is one of the most popular and ubiquitous iso-surface meshing algorithms due to both its simplicity and its computational efficiency. However, it is not by construction sensitive to curvature, and requires significant over-sampling to extract surfaces with reasonable error in high curvature regions. In our

pipeline, the accuracy of the computation of HKS depends on the accuracy of eigenvalues and eigenfunctions of the discrete LB operator, which in turn depend on how close we can get to the true representation of the surface in critical curvature regions - regions where we have high curvature or regions where changes from a high to low curvature occur. We require more resolution at those regions in order to capture the curvature information properly. The resolution of the mesh can be increased uniformly by applying the marching cubes algorithm on a highly sampled 3D scalar field. However, we would like to save computation time at the places of low curvature compared to the places of high curvature. Thus it is better to use a meshing system that assigns a general tesselation to the regions of low curvature (each edge is given a minimum length to ensure that the regions are not too coarse) and puts a finer tesselation otherwise.

Approaches like DistMesh (39) and Biomesh (35) use force-based relaxation methods and sizing fields to adapt the meshing to the local "shape" of the manifold. These "curvature-sensitive" methods are designed to give, for a set number of vertices, nearly optimal vertex positions on the surface such that the corresponding triangulation of the surface captures high-curvature regions.

In the DistMesh algorithm, the geometry (shape of the region) is described using a signed distance function that is negative inside the region. For mesh generation, an iterative technique is employed by treating the initial mesh as a truss structure. The meshpoints are considered nodes in the truss. The algorithm solves for equilibrium at each iteration, assuming a force-displacement function for the bars in the truss. The forces move the nodes and a Delaunay triangulation algorithm adjusts the edges in the process. Usually, such force based simulation produces very high quality meshes. Biomesh uses a particle based simulation where a sizing field calculation maitains the proper distribution of the particles on the surface of the mesh. This method also produces curvature sensitive meshing.

Although curvature-sensitive meshing approaches like Biomesh and DistMesh give more accurate results in this pipeline, as long as the Marching Cubes method is run at sufficiently high resolution, it can be used as well.

4.1.2 Laplacian Approximation

Once we have a mesh that represents the isosurface, we compute the eigenstructure of the induced Laplace-Beltrami (LB) operator. As discussed in Section 3.2, there are two main approaches to computing the eigenpairs of the operator: the graph Laplacian which

is a discrete approach and finite element method (FEM) which is considered a continuous approach. As has been shown by Reuter *et al.* (45), the FEM approach is superior to the graph-based approach as it attempts to build a numerical approximation of the continuous LB operator directly rather than indirectly (by considering a tessellation to merely be a graph). In this way, the discretized LB operator generated by the FEM method is designed (by construction) to converge to the continuous LB operator.

An experimental validation is provided in section 4.2 that demonstrates the accuracy of FEM compared to the Graph Laplacian.

4.1.3 HKS Computation

Given the eigenpairs of the Laplace-Beltrami operator, we can now compute the HKS vectors. These are defined at each point at a particular time t as

$$\mathcal{H}(x,t) = \sum_{i=0}^{\infty} e^{-\lambda_i t}\phi_i(x)^2.$$

In practice, the energy of the spectrum dissipates after the first few hundred eigenvalues (sorted in decreasing order). This phenomenon is illustrated in Figure 4.3. The two groups of curves represent HKS vectors at two points on the dimpled sphere (one on the dimple, and one elsewhere). The curves for varying number of eigenpairs are superimposed: as we can see, increasing the number of eigenpairs used does not significantly change the curve behavior. While the FEM-based Laplacian gives us stable results, HKS vectors based on the graph Laplacian do not converge as quickly. Based upon the experiment in figure 4.3, we use 300 eigenpairs for all our results.

For practical use of the HKS vectors, two more approximations are necessary, and are described in the original paper by Sun *et al.*(51). Each eigenfunction is a function of the position x and the time parameter t. In order to represent the eigenfunctions effectively, we store it as a sequence of sampled values at specific values of t. Since the coefficients $\exp(-\lambda_i t)$ decay exponentially with t, we sample t on a logarithmic scale, so that we can capture effects at long-range (large t) as well as short range. In our experiments, we use $t = 100$ samples (as in (51)).

4.1.4 Constructing Correspondence

As discussed in section 2.2, there are many correspondence methods that can be useful in our framework. However, it is hard to find any method that can construct intrinsic correspondence between two surfaces that work for open surfaces. Moreover,

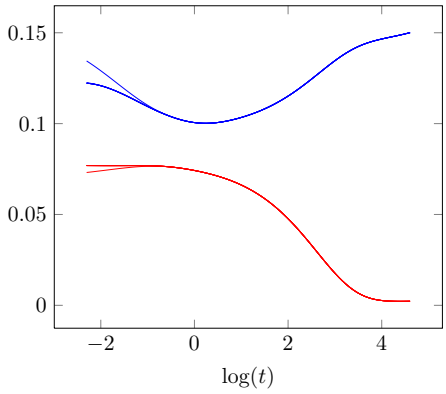


Figure 4.3. HKS vectors at two points on dimpled sphere. The corresponding vectors for varying numbers of eigenpairs (ranging between 10 and 500) are superimposed.

many correspondence methods, as pointed out before, are expensive to calculate. In light of these issues, we have decided to come up with a correpondence method that lets us take advantage of the nature of the particular signature (HKS) we have chosen.

Our approach captures *intrinsic* changes between isosurfaces by measuring the change in HKS vectors between *corresponding points* on the two surfaces. It is indifferent to the particular method that is chosen for this purpose.

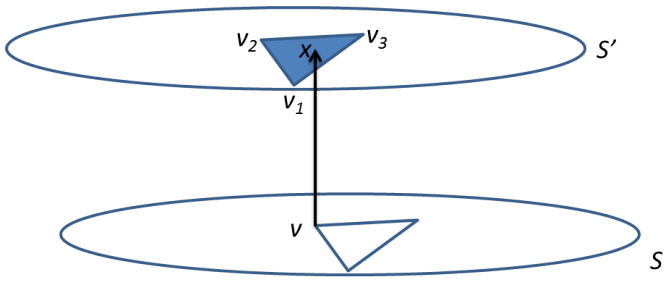


Figure 4.4. The correspondence for each vertex is found by shooting normal rays from the base surface's vertices. The HKS value at the intersected point is found by barycentric interpolation, so that the vectors $HKS(v)$ and $HKS(x)$ can be compared.

In the realm of isosurface correspondence, we possess additional information about the two surfaces (that they are generated by varying a control parameter). This can be exploited to compute more accurate correspondences as follows. Assume that we have generated two isosurfaces $\mathcal{S}, \mathcal{S}'$ from two values of the control parameter α, and our goal is to generate a correspondence between *vertices* on $\mathcal{S}$ and *points* on $\mathcal{S}'$ so that we can

measure the change in the HKS vectors. Under the assumption that the perturbation caused by α is not large and does not cause topological changes, we can shoot a normal ray from a vertex v of $\mathcal{S}$ and find its intersection with a triangle on $\mathcal{S}'$.

Let this triangle have vertices v_1, v_2, v_3 and let the point of intersection be x. Since the surfaces are generated via trilinear approximation (consistent with FEM), it follows that $HKS(x)$ can be expressed as a convex combination of the $HKS(v_i)$ using the barycentric representation of x. We can now compare $HKS(v)$ and $HKS(x)$. This guarantees a 1-1 correspondence between points on the two surfaces, and does not require both meshes to have the same number of points.

4.1.5 Intrinsic UQ

We now have HKS vectors assigned to corresponding points on the two surfaces and can compare them to quantify the local difference in shape. Given a point $x \in \mathcal{S}$ and its corresponding point $x' \in \mathcal{S}'$, we compute a difference

$$\Delta HKS(x) = HKS(x) - HKS(x').$$

As discussed in (51), we cannot compare these vectors directly: as mentioned earlier, the exponential decay in the coefficients as t increases means that higher-order effects will be swamped by the low-order effects (for small t). The solution they propose is to compute for each t the sum

$$w(t) = \sum_x HKS(x)[t]$$

and then construct a diagonal matrix W with $W(t,t) = w(t)$. We now compute

$$\Delta HKS(x)^\top = \tilde{\Delta} HKS(x)^\top W.$$

To form an approximation of the derivative, one would then scale by the differences in the perturbation parameter. As this merely provides a global magnitude scaling, we omit this change of scaling in our examples. If one were to do comparisons between different choices of perturbation as part of the VVUQ process, the scaling factor would need to be appropriately considered. Hence in this work, the resulting vector $\Delta HKS(x)$ is our quantification of the intrinsic shape uncertainty introduced by the change of parameter.

4.1.6 Visualizing the Uncertainty

With each point x on a surface we can now associate a (100-dimensional) vector $\Delta HKS(x)$ that quantifies the local surface change with respect to α. We can visualize

this change by grouping the vectors into similar sets based on their distance from each other. The simplest approach is simply to bin the norms of the vectors, and assign each bin a different color in a linear scale. Unfortunately, this approach fails to capture significant differences in shape variability.

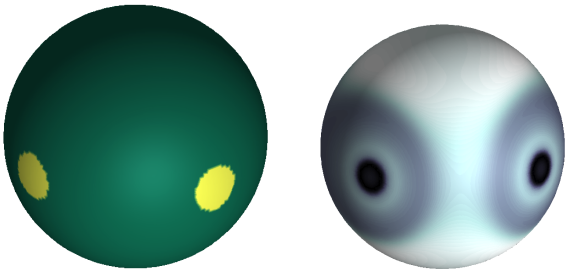
A more general approach is to cluster the difference vectors and assign different colors to each cluster. We employ two different clustering strategies, depending on whether vectors are assigned to single clusters (a hard clustering) or may "share" their membership across multiple clusters (a soft clustering). While our approach is agnostic to the particular clustering method used, we use these two methods to illustrate the different visualizations that may be obtained.

We can endow the space of the $\Delta HKS(x)$ vectors with a norm to define the distance between them. While using an ℓ_2 norm is most natural, this choice assumes that the individual coordinates (representing shape characteristics at different time scales) can be square-summed as distances. A more reasonable choice of norm is the ℓ_1 norm, for which the induced distance between two vectors is merely the sum of their absolute differences:

$$d(\Delta HKS(x), \Delta HKS(y)) = \sum_t |\Delta HKS(x)[t] - \Delta HKS(y)[t]|.$$

For computing a hard clustering, we use the k-medoids algorithm(26). This algorithm is an analog of k-means with the difference being that the cluster center is computed as the *median* of the points assigned to a cluster, as opposed to the mean. Once a cluster center is computed, points are reassigned to their nearest cluster center and the iterative process repeats. Here the median of a set of points is the point that minimizes the sum of distances to the set of points. For ℓ_1 spaces, this is merely the point formed by taking coordinate-wise medians in each dimension: $c = \text{median}(p_1, p_2, \ldots p_n)$ if $c[t] = \text{median}(p_1[t], p_2[t], \ldots, p_n[t])$.

For computing a *soft* clustering, where each point can assign a fraction of its membership to different clusters, we use the fuzzy c-means algorithm(15; 7). We experimented with other approaches, including deterministic annealing (46), and found this to be the most effective method. We also point out that a key deficiency of the above methods is the need to know the number of clusters. If this information is not available, hierarchical agglomerative clustering (HAC) may also be used to obtain a family of clusterings with different numbers of clusters.



(a) k-medoids (b) Fuzzy c-means

Figure 4.5. Visualizing HKS vectors using different clustering mechanisms.

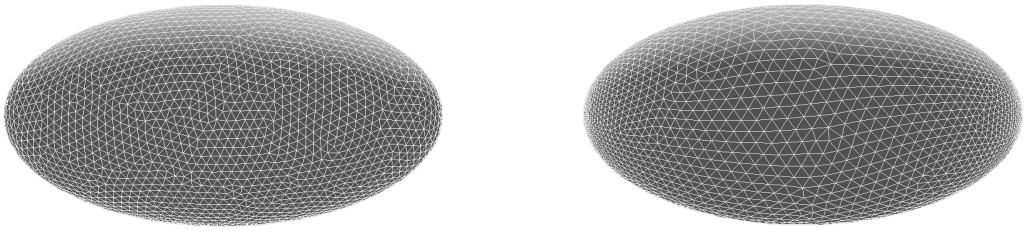
4.2 Justification of Parameter Choices

As one of our desired goals to present a concrete pipeline, we present experimental results that justify the different choices made in the design phase of the pipeline. Here, we compare different isosurfacing methods and investigate which one is better for the purpose of calculating the HKS - our choice of UQ metric. Then we move on to the two different schemes to create the Graph Laplacian and compare them to state which one is more suitable to use for our purpose. Next, we comapre the two prevalent methods of calculating Laplacians and recommend FEM as our choice in the pipeline. The time scale chosen for calculating HKS is validated next. The effect of mesh resolution on the accuracy of calculating the signature is then presented through some experiments. Next, we present the factors that govern the accuracy and reliability of eigenfunctions and eigenvalues of the Laplacian that is used to calculate the HKS. Finally we demonstrate, through visual and quantifiable experiments, the problems associated with a few other choices of clustering methods other than the ones we have recommended for the pipeline.

4.2.1 Comparing curvature sensitive and uniform meshing w.r.t HKS

Visually, the different methods of producing isosurface meshes may give similar looking meshes with similar triangle counts on a shape which has variations in curvature. We highlight this point by meshing an ellipsoidal shape as shown in Figure 4.6. Without being explicitly encouraged to examine high-curvature regions, many viewers would consider both triangulations to be “good” representations of the surface. However, the differences in the quality of the triangulations with respect to capturing curvature are more pronounced when we examine the variations in the HKS vectors. In figures 4.7 and 4.8 we illustrate this by looking at the difference between HKS vectors generated for

our sphere example and plotting a histogram of these magnitudes. Note that the HKS vectors here were produced using an FEM-based approximation of the Laplace-Beltrami operator described in Section 3.2. Ideally, the histogram for a sphere should be a spike at zero, since all HKS vectors should be identical. We see that the mesh produced by DistMesh approximates such a spike, but the mesh produced by Marching Cubes does not. Furthermore, the mesh produced by Marching Cubes has noisier HKS vectors, as we can see in Figure 4.7.



(a) Marching Cubes (b) DistMesh

Figure 4.6. Meshes produced by Marching Cubes and DistMesh

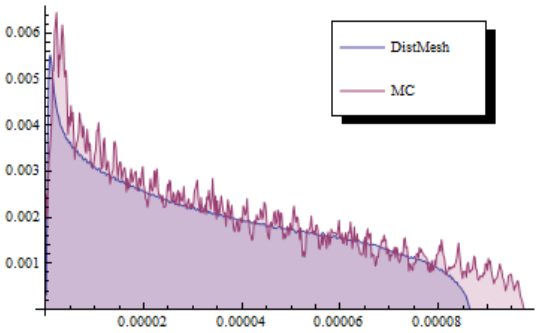


Figure 4.7. Marching Cubes and DistMesh compared on an ellipsoid.

4.2.2 Comparing Neighborhood Selection algorithms for Graph Laplacians

As stated in chapter 3, there are mainly two ways of choosing the nearest neighbor of a point to calculate the differential coordinate at that point, which is eventually used to build the Laplacian matrix. If one decides to use the Graph Laplacians methods to calculate the HKS, we would like to show experiments that compare the convergence and

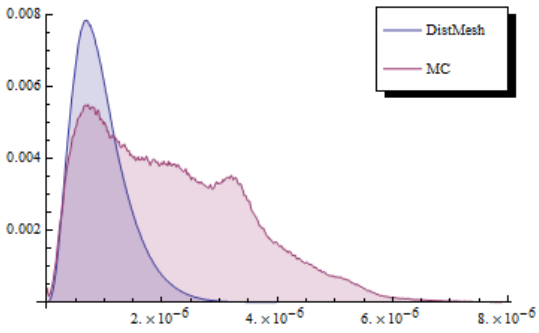


Figure 4.8. Marching Cubes and DistMesh compared on a sphere.

computation cost of these two methods. In our experiments, we find that the nearest neighbor scheme suggested by (3) with a reasonable number of neighbors is the best way to calcualte an accurate discrete Laplacian.

As mentioned in section 3.2, a method to choose the number of nearest neighbors is to use the 1-ring neighborhood of every vertex. However, it is advisable to use a method that is geometrically intuitive. This is why many researchers suggest the ϵ-neighborhood technique. (3) introduces a hybrid of these two methods. They choose k nearest neighbors that are sorted by their Euclidean distance from the vertex of interest. In this section, we compare the two geometrically motivated methods and establish that the scheme suggested in (3) is indeed a better choice.

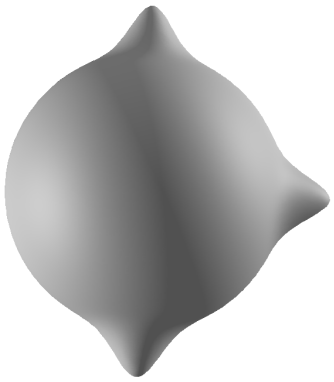
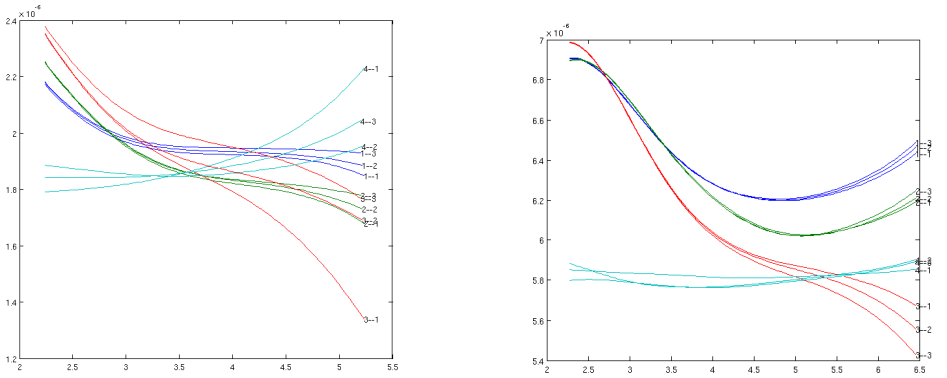


Figure 4.9. Dimpled sphere used for the experiment in section 4.2.2.

For these experiments, we have created three dimples on a sphere (figure 4.9), and nine points (three for each dimple, which are approximately chosen at similar spots on



(a) ϵ-neighborhood (b) nearest neighbor criteria defined in (3)

Figure 4.10. Compare HKS signature produced by three points on four dimples on a sphere.

the dimples) are observed using their HKS signature. The HKS signatures are at first calculated by the ϵ-neighborhood scheme with a radius equal to 15 units (where the sphere has a radius of 50 units). Then the HKS are again calculated using 50 nearest neighbor points (sorted by the Euclidean distance) for each point. The lines are colored according to the points chosen, e.g. the red lines denote the three points that are approximately at the same position on the three dimples, and so on. Another set of points are chosen from the surface of the sphere where there are no dimples, for comparison with the points residing on the dimple.

As shown in figure 4.10, the nearest neighbor scheme provides a cleaner and recognizable pattern of HKS vectors, whereas the ϵ-neighborhood provides a bit less recognizable trend. We would expect to see the red, blue and green lines grouped together similarly in the early stage before they start differentiating themselves due to the position of the dimples.

4.2.3 Comparing Graph Laplacian and FEM

In order to compare GL and FEM methods of calculating the HKS, we create histograms of the norm of difference vectors between each possible pair of HKS vectors on two geometric shapes - a sphere and an ellipsoid. For a sphere, since the HKS vectors are theoretically identical, we expect to see a spike in the histogram (just one mode, theoretically just a vertical line that looks like a "spike" shape). Whereas, for an ellipsoid, we expect to see two modes in the distribution, since the ends and the middle

of the ellipsoid have different curvatures.

Figure 4.11 and Figure 4.12 illustrate the difference in the HKS vector difference magnitude histogram for the ellipsoid and sphere. We see that the graph Laplacian does not approximate the ideal "spike" as well as the FEM-based method.

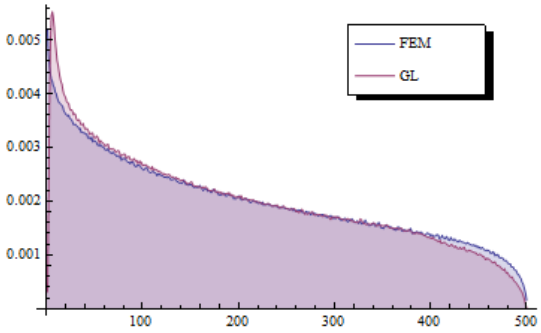


Figure 4.11. FEM and GL compared on an ellipsoid.

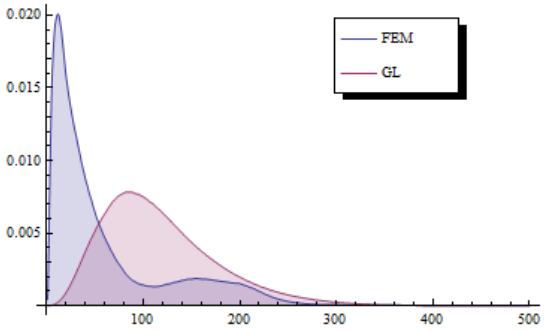


Figure 4.12. FEM and GL compared on a sphere.

This proves that the FEM method is much more reliable than the GL method and produces much cleaner and accurate HKS.

4.2.4 Justifying the Time Scale and Number of Eigenvalues

In our pipeline, we have chosen a logarithmic formula to decide the time scale (that is directly related to the geodesic distance) for calculating the HKS. Also, the number of eigenvalues required to calculate the HKS vectors is chosen as 300. Both of these choices are justified in section 4.1.3. Increasing the number of eigenpairs to calcualte HKS will

not change its accuracy drastically, and it will merely be a burden on the computational load of the pipeline.

4.2.5 The Effect of Mesh Resolution

We have demonstrated in 4.2.1 through histograms of HKS norms that curvature sensitive meshing is desirable when producing isosurfaces. Another parameter that is of natural interest is the mesh resolution. In this section, we demonstrate the effect of mesh resolution in curvature sensitive, uniform and nonuniform meshing of isosurfaces.

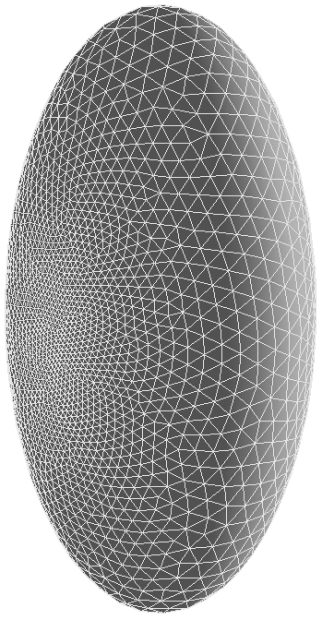
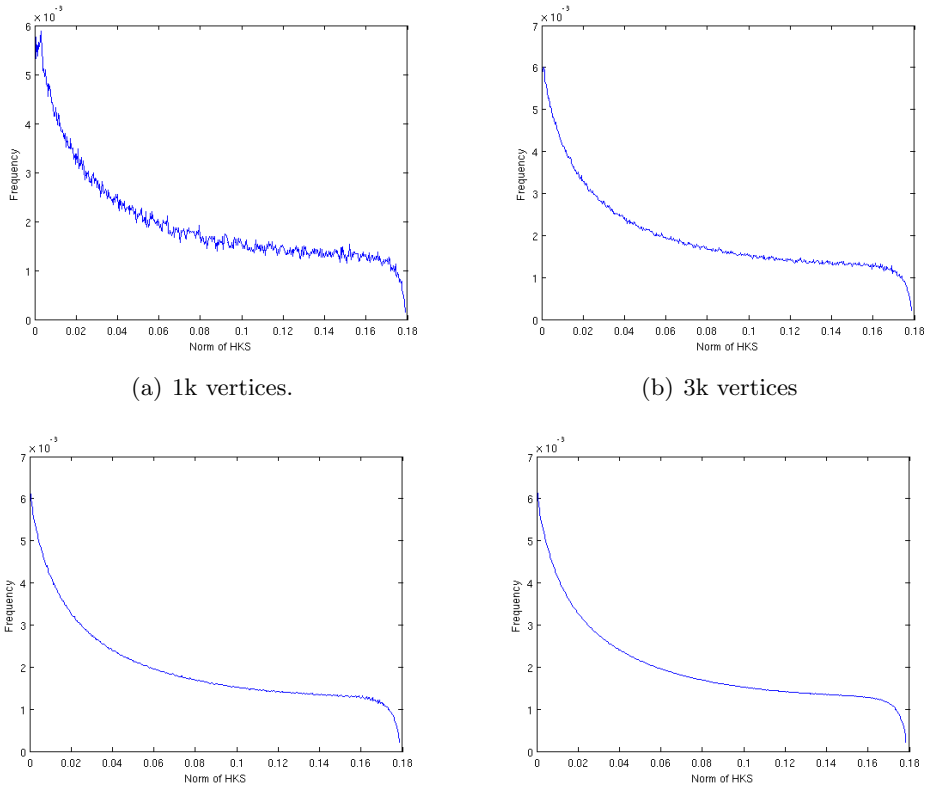


Figure 4.13. Nonuniform meshing using DistMesh that is not curvature sensitive.

In this series of experiments, we have created three kinds of ellipsoids using DistMesh. They are shown in figure 4.6 and 4.13. The ellipsoids in these figures were produced with different (but identical across the three different schemes) number of vertices to understand the effect of mesh resolution when HKS is calculated on these ellipsoids. We calculate the norm of the difference of all HKS vectors an ellipsoid and develop a histogram of all the values. As expected (seen on 4.2.1), we should get a histogram shape that looks sort of like a bimodal distribution.

In figure 4.14, we notice that as we increase the number of vertices, the histograms are much smoother. In fact, an almost smooth version is achieved when the number of



(a) 1k vertices.

(b) 3k vertices

(c) 12k vertices

(d) 50k vertices

Figure 4.14. Histograms for curvature sensitive meshing.

vertices is 3000. The bimodal shape is distinct in these figures too.

In figure 4.15, we notice that the bimodal shape is not very distinct in the histograms. In fact, it is hard to understand that this is a histogram of norm of the differences between all HKS vectors of an ellipsoid. Although the shape and outline of the histogram becomes smoother as we increase the number of vertices, we have clearly established that nonuniform meshing with no sense of geometry can produce poor quality HKS vectors.

In figure 4.16, we demonstrate the histogram for uniform meshing using DistMesh. We notice that the bimodal distribution becomes more noticeable and distinct as we increase the number of vertices on the ellipsoid.

Overall, from these three sets of experiments we can conclude that high meshing resolution can give a better insight on the nature of HKS and the change in the isosurface. However, a sense of geometry of the isosurface that can be used to govern the density of the mesh is crucial too.

4.2.6 The Behavior of Eigenfunctions and Eigenvalues with Perturbations

There are some well-known results (1) that show that the eigenfunctions of the Laplace-Beltrami operator on a unit sphere are spherical harmonic functions. However, there are no general results available in the PDE and Applied Mathematics (theory) literature that shed light on how the eigenfunctions behave on perturbed shapes.

In this subsection, we study the behavior of eigenvalues and eigenfunctions of the LB operator on a unit sphere isosurface. We introduce perturbation in the sphere in the form of a dimple. We then calulate the eigenpairs of all the meshes and plot them together on a chart. To faciliate this study, a GUI interface was built that allows the user to select regions in the sphere where the dimple was to be made. we increase the size of the dimple and visually investigate whether there are significant changes in the behavior of these eigenfunctions.

Figure 4.17 shows one such chart. The rows denote an evolution of a sphere to a large dimpled sphere. The dimple is created at the bottom. The columns denote the five largest eigenvalues (sorted from larger to smaller) of the LB operator calculated on the sphere meshes. The spherical harmonics are quite evident in the chart. We also notice that there is no significant change in the behavior of these spherical harmonics as a dimple is introduced (see the evolution of the harmonics in each column).

Next, we plot the first 300 eigenvalues for each dimpled sphere together (in figure 4.18). They almost superimpose on each other. This shows little deviation from the original eigenpairs (on a sphere) as a large dimple is introduced in the sphere.

Now, we create an eigenfunction chart once again (figure 4.19), but this time we choose the eigenfunctions corresponding to eigenvalues that are farther apart. We take the 1st, 4th, 9th, 16th and 25th eigenvalues from the descending sorted list of eigenvalues. We again observe that the spherical harmonics behavior of eigenfunctions remain intact. In some cases, they are more pronounced, however no definite conclusion can be drawn from this.

The experiments here suggest that perturbations that do not cause drastic change in the shape of the isosurface have similar eigenpairs for the LB operator. This is why other LB operator based signatures, for example the Global Point Signature (GPS) (48) may not work well as our choice of metric for quantifying intrinsic variation. The GPS is defined as

$$GPS(x) = (\frac{\phi_1(x)}{\sqrt{\lambda_1}}, \frac{\phi_2(x)}{\sqrt{\lambda_2}}, ..., \frac{\phi_i(x)}{\sqrt{\lambda_i}})$$

If the eigenpairs are close to each other for a perturbation, then the differences between them are not quite pronounced with this signature since the eigenfunctions are merely scaled by the square roots of the eigenvalues. Instead, HKS takes a multi-scale approach and it is sensitive to smaller perturbations due to its construction (as it is a summation of squared eigenfunctions that are multiplied with an exponentially weighted fall-off of eigenvalues to ensure capturing the contribution from smaller to larger change of scale).

4.2.7 Hard and Soft Clustering - Is there any superior method?

There are several clustering techniques that we have experimented with, to find out if there is a superior method that will work well on HKS vectors. Here, we present a few visualizations made with different clustering techniques, applied on the same shape for same perturbations.

As described in chapter 4, we have used k-medoid and fuzzy c-means in our pipeline to produce the example results. There are few other methods that were eligible candidates for this purpose. One limitation of using k-medoid is that the number of clusters needs to be given beforehand. Deterministic annealing is a method that will choose its own number of clusters, so this was a natural fit for testing with our pipeline.

Deterministic Annealing: The algorithm (47; 46) arises from a statistical physics analogy of cooling an ensemble with a predetermined gradient in order to see how different elements in the ensemble have clustered together. The ensemble is then heated with a different lower temperature and the process is continued until the temperature falls below some threshold.

In order to apply the analogy to high dimenstional vectors, we define the maximum number of codevectors (cluster centers) and start with one codevector only. This initial codevector is the centroid of all vectors. It is updated for the initial distribution of temperature (the initial temperature is an input to the algorithm) using a formula that is derived from an analog of Boltzmann Statistics. The points are checked for "phase transition" and if there were any such transition among the vectors, a new codevector is introduced (i.e. a new cluster emerges). The temperature is lowered and the process is repeated until a convergence test is successful.

We have implemented this algorithm and tried it on standard shapes such as ellipse and spheres with small and bigger dimple perturbations. Figure 4.20 shows a few

visualizations of $\Delta HKS(x)$ vectors clustered with the deterministic annealing algorithm. The perturbations are visualized over the original shape as wireframe meshes. 4.20(a) shows that the $\Delta HKS(x)$ vectors in the perturbation area were not clustered properly. Rather, it looks like the area where some changes are being rendered are probably noise from the clustering.

4.20(b) and 4.20(c) show better results when the perturbations on the isosurfaces were bigger. Although the whole region of perturbation could not be caught, the region where the bigger protrusions occured have been distinguished.

In summary, the deterministic annealing algorithm, although sophisticated and useful in certain ways, was not the right choice for our UQ metric - the $\Delta HKS(x)$ vectors.

MDS: The Multi-Dimensional Scaling algorithms are a set of popular algorithms for dimensionality reduction. Given a high dimensional vectors, it can collapse the dataset by reducing the number of irrelevant dimensions. We have used two kinds of MDS on our $\Delta HKS(x)$ vectors to reduce them to three dimensional datasets. The idea was to treat each dimension as one color channel, and visualize each 3D vector on each point using a color map.

Figure 4.21 shows the visualizations after applying the classical MDS and Universal MDS on a perturbed ellipse - the same one used in figure 4.20(b). As seen from the figures, although the perturbed regions are distinguished from the rest of the shape, the noise is too much for this to be an effective visualization that can convey meaningful information about the perturbed area. Thus, MDS was not a good choice of visualization for our UQ metric either.

4.3 Summary of Experiments

In the previous section, we have detailed a few experiments that justify the parameter and other choices made in the pipeline. Here is a summary of the conclusions made from each experiment.

1. It is recommended to use curvature sensitive meshing when calculating HKS and $\Delta HKS(x)$ (our chosen UQ metric) as opposed to Marching Cubes or uniform meshing.

2. The k-nearest neighbor scheme (where vertices are declared neighbors if they are within a certain radius) gives cleaner and better HKS vectors (for a reasonable k) when compared to the ϵ-ball method.

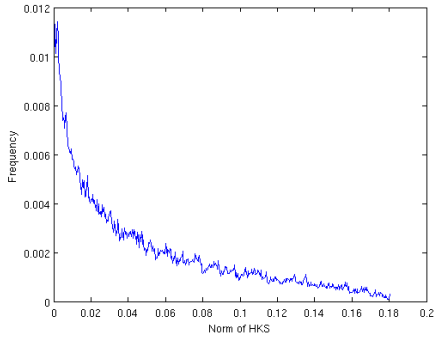
3. The FEM method to construct the Laplace-Beltrami operator, when compared to the Graph Laplacian, gives more accurate and cleaner HKS.

4. In order to account for contribution at higher time scales in HKS, the logarithmic sampling of HKS is a good choice for our pipeline. Moreover, we have decided to use the first 300 largest eigenvalues to calculate HKS in our pipeline. This is a justified choice given the experimental results that demonstrate the negligible change in the values of the HKS vectors for higher number of eigenvalues.

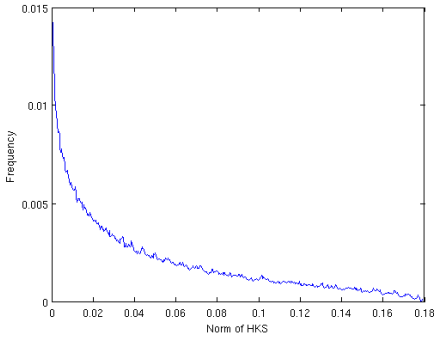
5. Higher mesh resolution, despite the uniform or nonuniform (or curvature sensitive) meshing, gives more accurate HKS. Thus, a higher resolution is desired if curvature sensitive meshing could not be employed. Precautions should be taken in general when LB operator eigenpairs are calculated on any mesh, as the experiments show that the nature of meshing is very important to produce accurate visualization from the pipeline.

6. HKS is a better UQ metric when it comes to quantifying intrinsic variation, as shown by the experimental data that eigenpairs do not change in magnitude very much when perturbations are introduced. Other existing shape matching signatures do not scale the eigenpairs to take account of smaller changes in them.

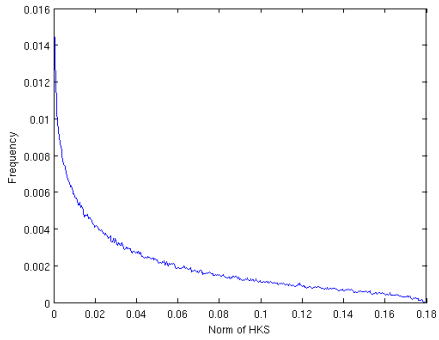
7. Our investigation on different clustering methods to visualize the $\Delta HKS(x)$ vectors shows that it is better to use k-medoid algorithm. Other algorithms, while having certain other advantages, do not cluster the $\Delta HKS(x)$ vectors well enough.



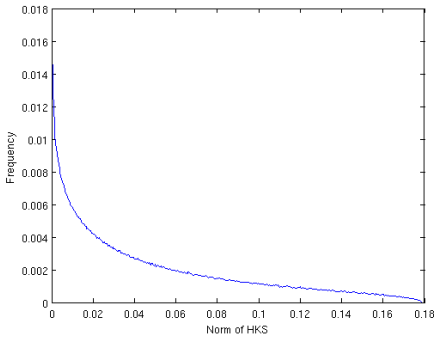
(a) 1k vertices.



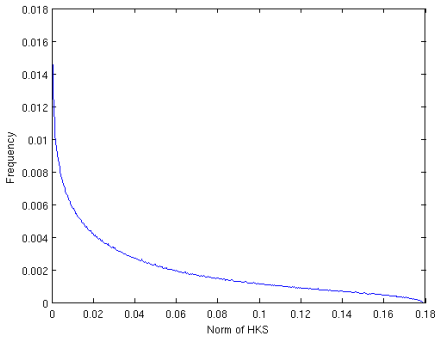
(b) 3k vertices



(c) 5k vertices

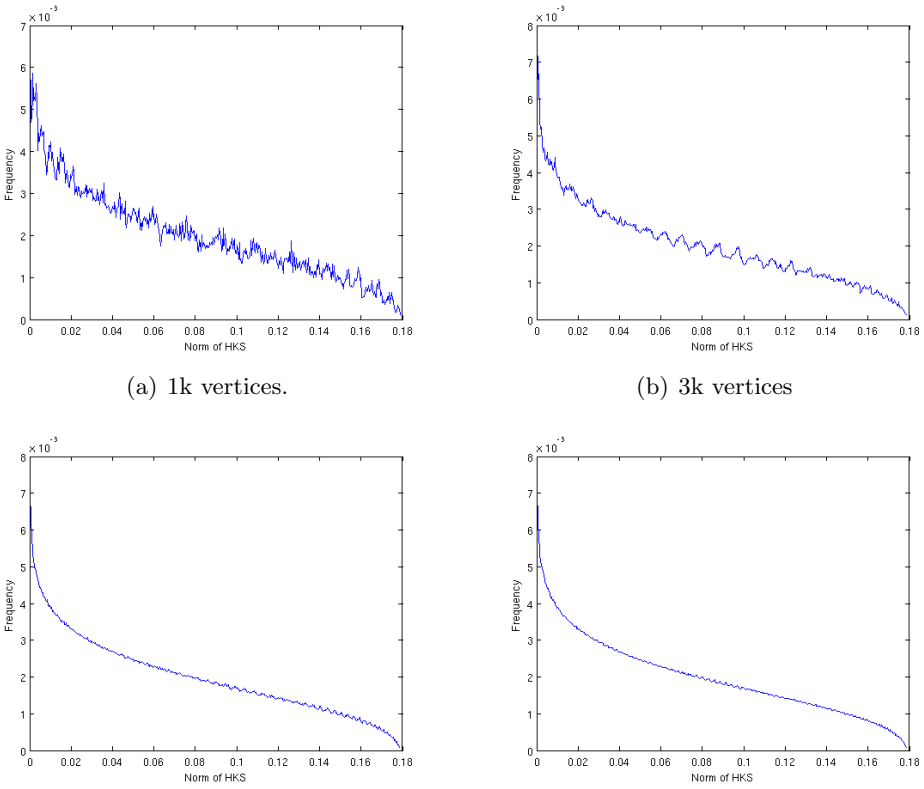


(d) 12k vertices



(e) 22k vertices

Figure 4.15. Histograms for nonuniform meshing (figure 4.13) in DistMesh.



(a) 1k vertices.

(b) 3k vertices

(c) 12k vertices

(d) 22k vertices

Figure 4.16. HKS norm dIfference histograms for uniform meshing using DistMesh.

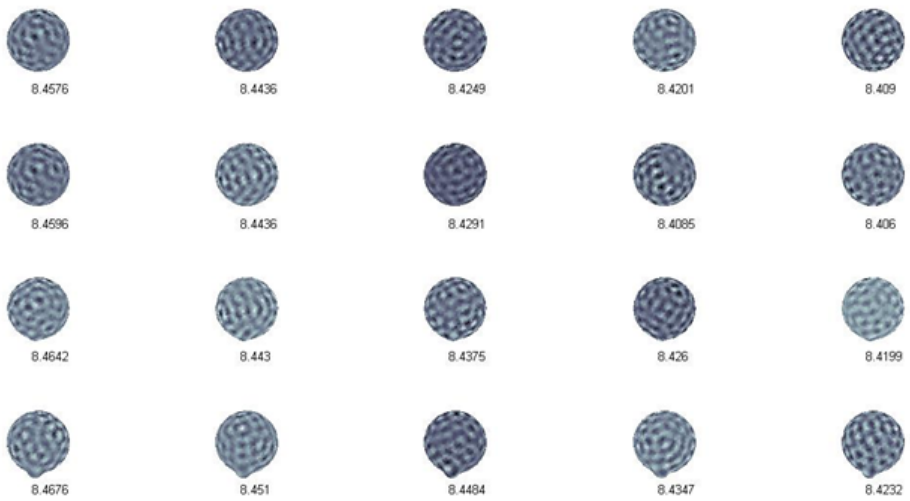


Figure 4.17. Perturbation of a sphere (first row) to a large dimple (last row). Colormap is based on the eigenfunction for the corresponding eigenvalues shown at the bottom of each sphere. Here the first 5 eigenvalues (of largest magnitude) are shown for each row.

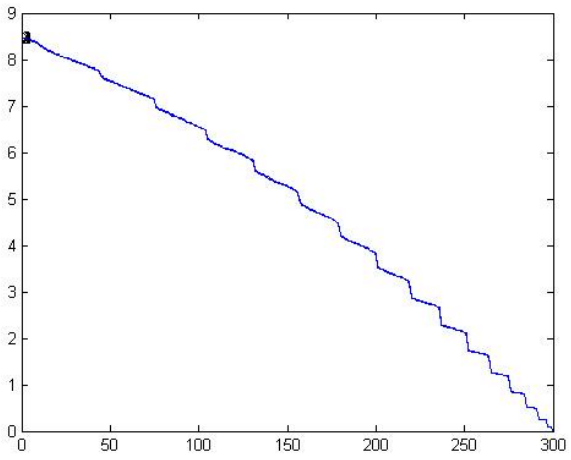


Figure 4.18. The first 300 eigenvalues are shown for each set of perturbation. As seen from the graph, they are very similar

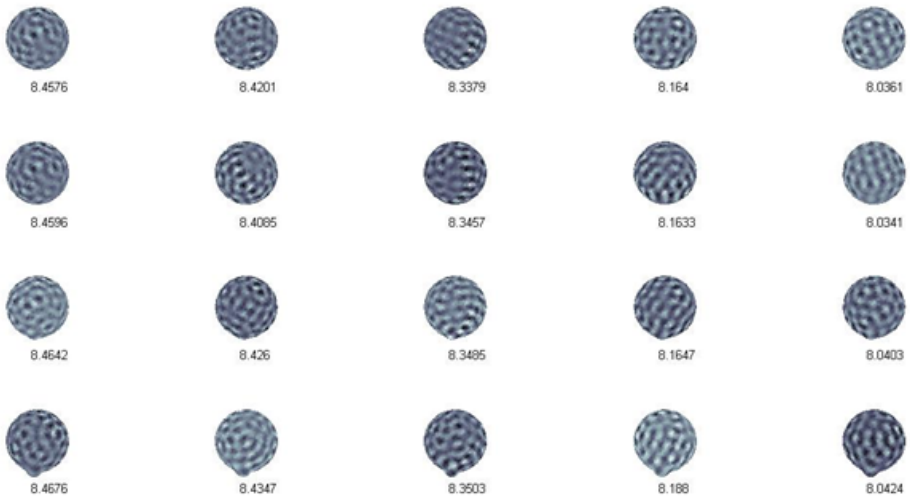
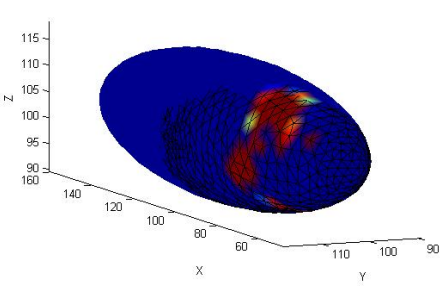
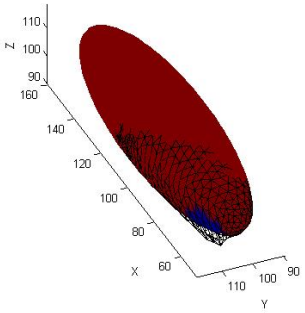


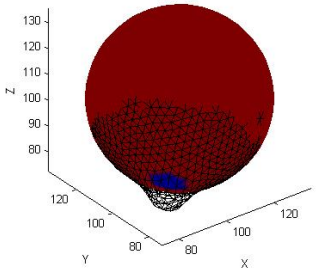
Figure 4.19. A similar eigenfunction chart, this time the eigenvalues are chosen to be further apart from each other (squared distance, i.e. evals(1,4,9,16,25) for each row.



(a) A smaller perturbation on an ellipse.

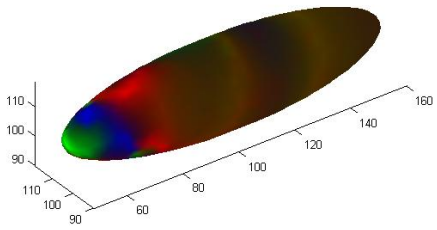


(b) Bigger perturbation

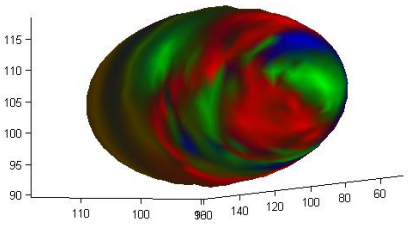


(c) Perturbation on a sphere

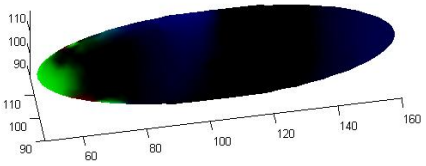
Figure 4.20. Deterministic Annealing clustering algorithm applied to $\Delta HKS(x)$ vectors.



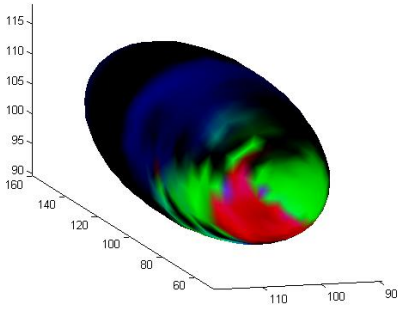
(a) Classical MDS applied to $\Delta HKS(x)$ vectors (view 1).



(b) Viewpoint of the perturbed area



(c) Universal MDS



(d) Viewpoint of the perturbation area (Universal MDS)

Figure 4.21. MDS applied to $\Delta HKS(x)$ vectors.

CHAPTER 5

APPLICATIONS

In this chapter, we present an empirical evaluation of our visualization pipeline. We start with a procedure (that may be of independent interest) to deform a mesh so as to create different surfaces with given correspondences. We then use this process to test the fidelity and sensitivity of the HKS vectors under different kinds of deformation. We complement this with a study on isosurfaces generated from simulation data, using our pipeline to visualize intrinsic changes in the surfaces.

5.1 Validation of Pipeline

To provide validation of our pipeline, we have devised a means of taking one isosurface and causing "depressions" into or out of the surface. We call our tool a "dimple maker". This procedure allows us precise control over which vertices correspond, so that we do not introduce noise when comparing the HKS vectors for two points that should correspond. The meshes of shapes with dimples in the previous sections were created using this tool.

Our problem formulation states that we have two isosurfaces for which we will calculate the point-wise heat kernel signature. Without loss of generality, we introduce another way to look at the same problem. Assume we are given a tessellation $\mathcal{T}(M_1)$, and we would like to produce $\mathcal{T}(\mathcal{M}_2)$ from it by a perturbation. Note that originally we have stated that a scalar parameter α in a simulation governs the perturbation. However, we can directly make a perturbation to the mesh $\mathcal{T}(\mathcal{M}_1)$ that allows us to produce a tessellated $\mathcal{M}_2$ from it while preserving vertex-wise correspondence.

This tool can be used to generate physical perturbations in meshes to produce new shapes. A scalar parameter α governs the amount of perturbation, while the shape of the perturbation is determined by a particle dynamics simulation. The tool, essentially a mesh dimple maker, is easily reproducible and useful for any shape perturbation studies. The need for using particle dynamics to create dimples comes from the fact that we want our perturbation to be insensitive to mesh refinement. A systematic and deterministic

dynamics simulation ensures that despite the meshing resolution, all the vertices in the mesh will be influenced by the same force function, hence the edges in the meshes have no effect on the perturbation whatsoever.

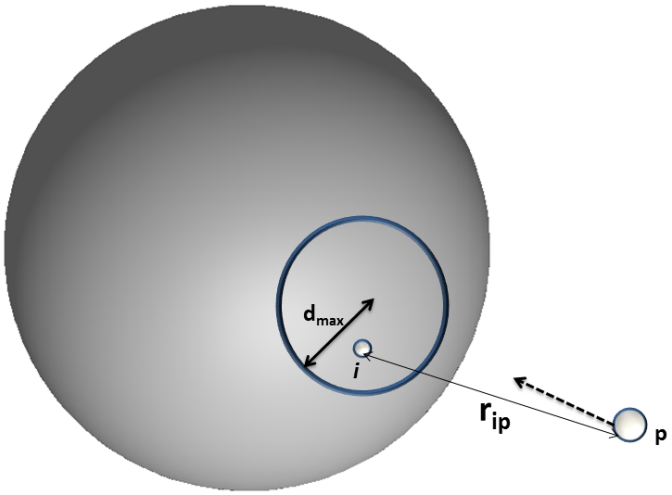


Figure 5.1. A projectile particle p is shot at the sphere. Only the vertices in the circled region are made to be affected, i.e. the circled region will be the dimpled region. The decay parameter d decides (for each particle) how strong the force of attraction should be. The force decreases as we move farther away from the center of the stated circle. Each particle i is influenced by the projectile p's charge, and as the projectile moves towards the sphere, the vertices are pushed in or out depending on their sign of charge. The dynamics simulation is run for a few steps to create a dimple on the sphere.

The idea is to treat the given tessellation's vertices as a point cloud and influence the point cloud by a projectile particle. Each particle is assigned a charge and unit mass, and based on the projectile particle's charge, they are either repelled or attracted towards it. The force function is used to calculate the velocities and subsequent positions of the particle in each iteration of the simulation, with $\mathbf{v}_i$ (the velocity of particle i) computed as

$$\frac{d\mathbf{v}_i}{dt} = \frac{\alpha q_i q_p e^{-d_i}}{||\mathbf{r}_{ip}||^2}\hat{\mathbf{r}}_{ip}$$

where q_i is the charge on the i^{th} particle, q_p is the charge of the projectile, $\mathbf{r}_{ip}$ is the vector from particle i to the projectile, and α is a scalar parameter that is tunable in our system, essentially controlling the amount of perturbation in each iteration. d_i is a decay parameter that is a function of the distance of the i^{th} particle from the projectile particle. This gives us the flexibility to govern which particles will move by how much.

Particles do not exert force on one another; their only motion is via the the projectile. This is done to preserve the shape of the original mesh and localize the regions where we want our perturbations to occur. Once the velocities are found by numerical integration, the positions of the particles (vertices) are updated as

$$\vec{x}_{i+1} = \vec{x}_i + \vec{v}_i \cdot dt + \frac{1}{2}\dot{\vec{v}}dt^2.$$

Using different trajectories and magnitudes for the projectile, many kinds of dimples (push-ins or push-outs) can be made using this tool. Example dimpled shapes can be seen in chaper 4 figure 4.1 and the isosurface examples demonstrated in the later sections.

5.2 Canonical Examples

We now present experiments that demonstrate the fidelity of the HKS vectors to location and scale of deformations, as well as the effectiveness of the overall pipeline. We use synthetic data with deformations generated using the dimple maker described above. All isosurfaces in this section and the subsequent section were generated using DistMesh (39) unless otherwise specified.

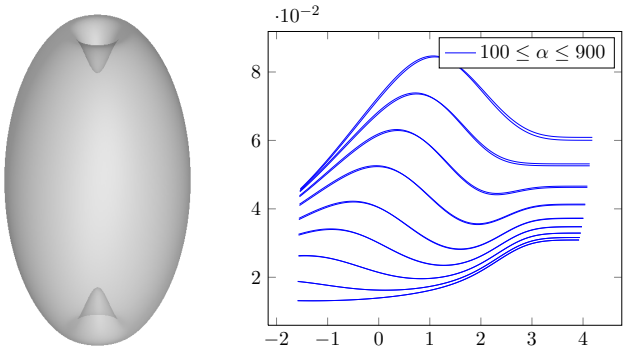
Experiment 1: Ellipse with varying depressions.

The dimple maker tool can be used to examine how robust the HKS is in capturing intrinsic changes of curvature. In Figure 5.2(a) we show an ellipse with two dimples (at the two "poles"). The dimples are created using different values of the force parameter α. The resulting HKS vectors are shown in Figure 5.2(b). Note the steady change in the vector as the deformation increases. We also note that since the two dimples are deformed identically, their HKS curves are almost perfectly aligned: this is further validation that identical deformations will generate very similar curves.

Experiment 2: Sphere with multiple dimples.

We now look at the behavior of our pipeline under multiple deformations with different curvature. To simulate this, we take a sphere and create eight dimples, each of which is then "pushed in" at the top to create regions of different curvature. The resulting figure is shown in Figure 5.3(a).

Running our pipeline on this shape (in comparison with a reference sphere) yields a collection of results for different choices of the number of clusters in the visualization. Three such visualizations are shown in Figure 5.3. Initially, the only change is the (short-range) deformation caused by the dimples. As the number of clusters increase, the central band variation is detected as well, and finally the individual dimples are



(a) Ellipse (b) Resulting sequence of HKS vectors

Figure 5.2. An ellipse with two dimples of varying magnitude: $\alpha = 100, \ldots, 900$.

identified. Since each "push-in" dimple has the same geometry, the clustering correctly places them in a single cluster (indicated by color) even though they are separated on the mesh. This further validates the strength of our pipeline in distinguishing important geometric changes in isosurfaces.

Experiment 3: Sphere with moving dimples.

We now investigate the behavior of the curves as the deformations move around on the surface. We generate a sphere with four numbered dimples (depicted in Figure 5.4(a)). In this experiment, we will move dimple 4 towards dimple 2, while keeping the other two dimples fixed. The two shifted configurations are shown in Figures 5.4(b),5.4(c). Figure 5.4(d) illustrates the resulting HKS vectors for the four dimples. Notice that dimples 1 and 3 are symmetrically placed, and so see the same "view" of the surface. They therefore have almost identical HKS vectors in all configurations. Moreover, initially all four dimples have a symmetric view of each other and generate almost identical HKS vectors.

If we now consider dimple 4 (in orange), it is symmetric to dimple 2 in the second configuration and is different in the third configuration, which is borne out by the HKS vectors. Further, the pair of dimples $1, 3$ have a slightly different view of the shape in configurations 2 and 3, which is reflected in the slight difference in their HKS vectors between configurations.

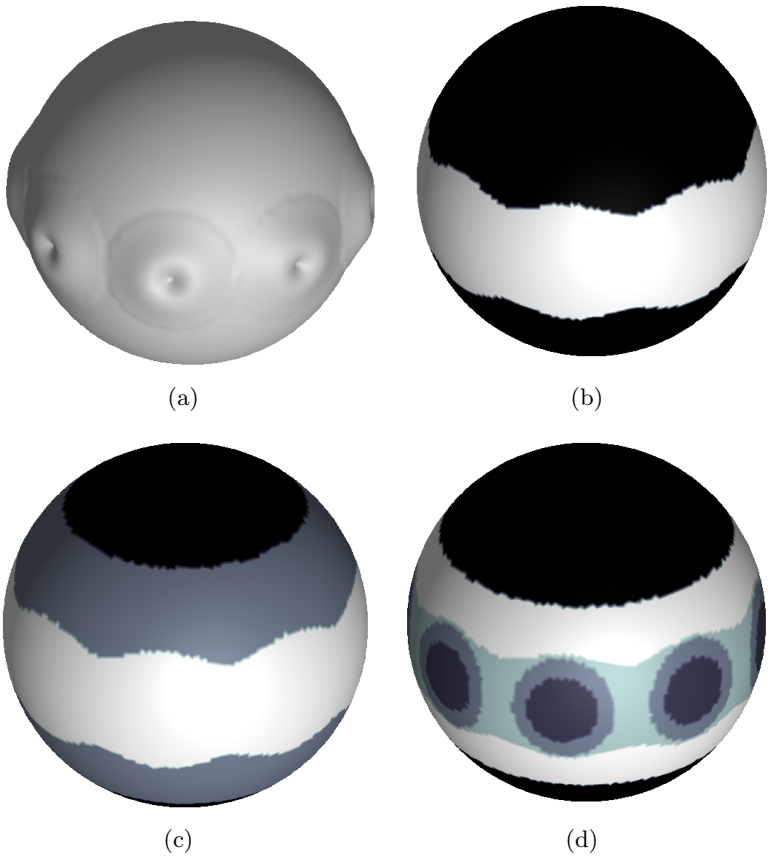


Figure 5.3. (a) A sphere with eight dimples, each of which is pushed out and then the tip is pushed in. (b) K-medoid clustering with two clusters, (c) three clusters and (d) five clusters.

5.3 Real-World Applications

We now demonstrate our pipeline on isosurfaces generated from simulation results. The examples are taken from molecular dynamics simulations. In all cases, we use the procedure described in Section **??** to generate correspondences.

Experiment 1: MD-potentials: closed isosurface.

We demonstrate our pipeline capturing the changes in isosurfaces generated from Born-Mayer potential fields used in molecular dynamics (MD) simulations (49). In this example, we extract two isosurfaces generated by changing the spatial configuration of a three atom system. In Figure 5.5(a), we show a rendering of the two isosurfaces. In Figures 5.5(b) and 5.5(c), we show a two and three k-medoid clustering of the HKS differences. This example demonstrates that our pipeline provides a quantifiable evaluation of the intrinsic change in isosurfaces of the potential (energy) field generated due to

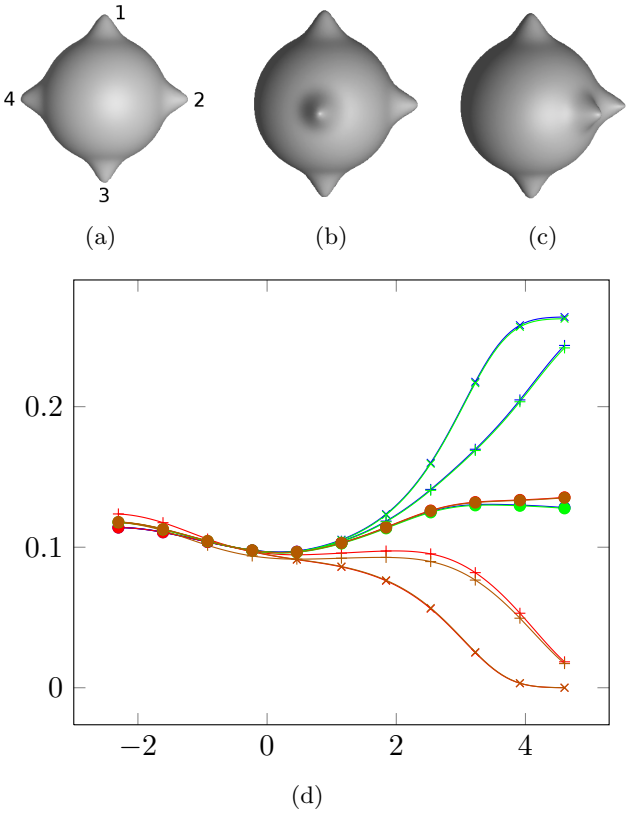
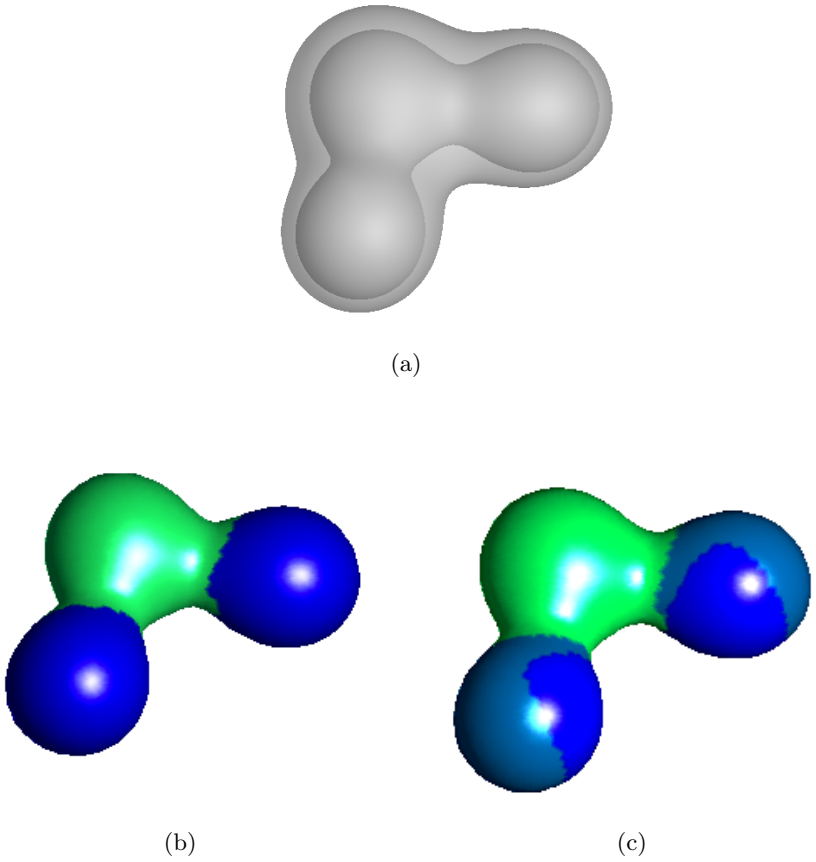


Figure 5.4. (a), (b) and (c) A sphere with four dimples, the fourth dimple "moves" along the equator. (d) HKS vectors for the above configurations. Configurations indicated by tick shapes (a) = •, (b) = ×, (c) = +, and dimples indicated by color (1 = blue, 2 = red, 3 = green, 4 = orange)

perturbation in configuration.

Experiment 2: MD-potentials: open isosurface.

To demonstrate that our pipeline does not require that the isosurfaces being examined be closed, we consider the same MD simulation as above with a different configuration of the atoms and a different isovalue of the potential. In Figure 5.6(a), we show a rendering of the two isosurfaces. In Figure 5.6(b), we show a three k-medoid clustering of the HKS difference. Because curvature-sensitive meshing of open surfaces is a challenging open research area (39), we generate our isosurfaces with a heavily-refined Marching Cubes lattice. Note that the third cluster has pointed out the sharp edges at the end, where the outer isosurface was more curved. This was examined with close scrutiny later, although a normal view does not convey such minute change in curvature. Again, our pipeline clearly expresses in a quantifiable way the intrinsic change due to a change in the atomic



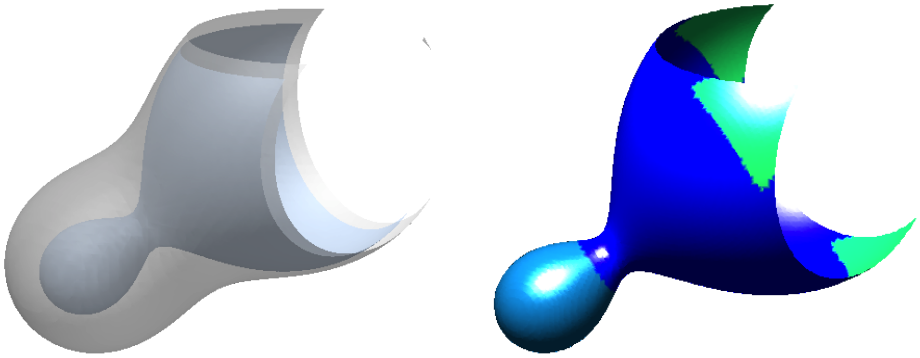
(a)

(b) (c)

Figure 5.5. (a) MD potential surfaces rendered together. (b) end result of running our pipeline on the two given meshes with two clusters and (c) three clusters.

configuration. This helps a simulation scientist easily identify areas of potential change that may not be obvious when designing crystal structures (49).

Capturing such intrinsic isosurface variation is helpful in molecular dynamics applications as the scientist can easily track changes in the isopotential structure of a system of atoms (e.g. protein molecules in biology) when a crucial parameter in the simulation is perturbed. Tracking both extrinsic and intrinsic changes can provide a complete scenario of parameter pertubation. Our pipline have been used in the above example to provide a robust way to understand the intrinsic change in the structure of the isopotential surfaces when a parameter was changed in a molecular dynamics simulation.



(a) (b)

Figure 5.6. (a) Open MD potential surfaces rendered together. (b) end result of running our pipeline on the two given meshes with three clusters.

CHAPTER 6

DISCUSSION

In this thesis, we proposed an analysis and visualization pipeline for examining the intrinsic variation in isosurfaces caused by simulation parameter perturbation. The choice of parameters in the pipeline was validated by a series of experiments. We have demonstrated our pipeline using some cannonical and real world examples. The examples cover a range of tests and challenges (open and closed surfaces, robustness of HKS-our choice of shape signature, capturing multiple deformations on the isosurface using clustering etc) to evaluate the limits of the pipeline. Within the given limitations (that were stated in the introduction), our visualization pipeline performed well on all the examples.

Several previous works (*e.g.* (40; 43)) have focused on the *extrinsic* variations that can be observed in isosurfaces generated from random variable fields. This work presents a complementary perspective in that it provides a quantifiable way of understanding the *intrinsic* variations in isosurface shape (such as change in curvature) that cannot be easily inferred through mere comparative visualization or extrinsic-change quantification. Combined with extrinsic measures (as future work), a more complete understanding of the impact of parameter variation within simulations on isosurfaces can occur. In this sense, our work provides the next natural step needed in the uncertainty quantification and visualization of isosurfaces generated through parameter variation. Additional future work includes extending the pipeline to handle variations due to multiple parameters and to handle properly (mathematically) correspondence in the case of large isosurface deformation.

REFERENCES

[1] K. Atkinson and W. Han, *Spherical Harmonics and Approximations on the Unit Sphere: An Introduction*, Lecture Notes in Mathematics, Springer, 2012.

[2] A. Bair and D. House, *Grid with a view: Optimal texturing for perception of layered surface shape*, IEEE Transactions on Visualization and Computer Graphics, 13 (2007), pp. 1656–1663.

[3] M. Belkin and P. Niyogi, *Laplacian eigenmaps for dimensionality reduction and data representation*, Neural Comput., 15 (2003), pp. 1373–1396.

[4] M. Belkin, J. Sun, and Y. Wang, *Discrete laplace operator on meshed surfaces.*, in Symposium on Computational Geometry, M. Teillaud and E. Welzl, eds., ACM, 2008, pp. 278–287.

[5] M. Belkin, J. Sun, and Y. Wang, *Constructing laplace operator from point clouds in rd*, in SODA '09: Proceedings of the Nineteenth Annual ACM -SIAM Symposium on Discrete Algorithms, Philadelphia, PA, USA, 2009, Society for Industrial and Applied Mathematics, pp. 1031–1040.

[6] S. Belongie, J. Malik, and J. Puzicha, *Shape context: A new descriptor for shape matching and object recognition*, in In NIPS, 2000, pp. 831–837.

[7] J. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, 1981.

[8] A. Bronstein, M. Bronstein, and R. Kimmel, *Rock, paper, and scissors: extrinsic vs. intrinsic similarity of non-rigid shapes*, in Proceedings of the International Conference on Computer Vision (ICCV), 2007.

[9] M. Bronstein and I. Kokkinos, *Scale-invariant heat kernel signatures for non-rigid shape recognition*, CVPR, (2010), pp. 1704–1711.

[10] S. Bruckner and T. Mller, *Isosurface similarity maps*, 2010.

[11] S. Busking, C. P. Botha, L. Ferrarini, J. Milles, and F. H. Post, *Image-based rendering of intersecting surfaces for dynamic comparative visualization*, Vis. Comput., 27 (2011), pp. 347–363.

[12] C. S. Chua and R. Jarvis, *Point signatures: A new representation for 3d object recognition*, International Journal of Computer Vision, 25 (1997), pp. 63–85.

[13] M. Chung and J. Taylor, *Diffusion smoothing on brain surface via finite element method*, in Biomedical Imaging: Nano to Macro, 2004. IEEE International Symposium on, april 2004, pp. 432 – 435 Vol. 1.

[14] S. Djurcilov, K. Kim, P. Lermusianax, and A. Pang, *Visualizing scalar volumetric data with uncertainty*, Computers and Graphics, 26 (2002), pp. 239–248.

[15] J. Dunn, *A fuzzy relative of the Isodata process and its use in detecting compact, well-separated clusters*, Journal of Cybernetics, 3 (1973), pp. 32–57.

[16] A. Elad and R. Kimmel, *On bending invariant signatures for surfaces*, IEEE Trans. Pattern Anal. Mach. Intell., 25 (2003), pp. 1285–1295.

[17] T. Gatzke, C. Grimm, M. Garland, and S. Zelinka, *Curvature maps for local shape comparison*, in Proceedings of the International Conference on Shape Modeling and Applications 2005, SMI '05, Washington, DC, USA, 2005, IEEE Computer Society, pp. 246–255.

[18] A. Grigor'yan, *Heat kernels on weighted manifolds and applications*, Contemporary Mathematics, 398 (2006), pp. 93–193.

[19] G. Grigoryan and P. Rheingans, *Point-based probabilistic surfaces to show surface uncertainty*, IEEE Trans. Visualization and Computer Graphics, 10 (2004), pp. 564–573.

[20] E. Hsu, *Stochastic analysis on manifolds*, AMS, (2002).

[21] V. Interrante, H. Fuchs, and S. M. Pizer, *Conveying the 3d shape of smoothly curving transparent surfaces via texture*, IEEE Transactions on Visualization and Computer Graphics, 3 (1997), pp. 98–117.

[22] D. W. Jacobs, D. Weinshall, and Y. Gdalyahu, *Class representation and image retrieval with non-metric distances*, 1998.

[23] A. Johnson, *Spin Images: A Representation for 3-D Surface Matching*, PhD thesis, Robotics Institute, Carnegie Mellon University, 1997.

[24] C. Johnson and A. Sanderson, *A next step: Visualizing errors and uncertainty*, IEEE Computer Graphics and Applications, 23 (2003).

[25] C. Johnson and A. Sanderson, *A next step: Visualizing errors and uncertainty*, Computer Graphics and Applications, IEEE, 23 (2003), pp. 6 – 10.

[26] L. Kaufman and P. Rousseeuw, *Clustering by means of medoids*, in Statistical Data Analysis Based on the L1-Norm and Related Methods, Y. Dodge, ed., North-Holland, 1987, pp. 405–416.

[27] L. J. Latecki and R. Lakämper, *Shape similarity measure based on correspondence of visual parts*, IEEE Trans. Pattern Anal. Mach. Intell., 22 (2000), pp. 1185–1190.

[28] B. Levy, *Laplace-beltrami eigenfunctions towards an algorithm that "understands" geometry*, in Shape Modeling and Applications, 2006. SMI 2006. IEEE International Conference on, june 2006, p. 13.

[29] X. Li, Y. He, X. Gu, and H. Qin, *Curves-on-surfaces: A general shape comparison framework*, in In Proceedings of International Conference on Shape Modeling and Applications (SMI 2006, 2006.

[30] I. S. Lim, S. Sami, and D. Thalmann, *Colored visualization of shape differences between bones*, in Proceedings of the 16th IEEE conference on Computer-based medical systems, CBMS'03, Washington, DC, USA, 2003, IEEE Computer Society, pp. 273–278.

[31] W. E. Lorensen and H. E. Cline, *Marching cubes: A high resolution 3d surface construction algorithm*, in SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques, New York, NY, USA, 1987, ACM Press, pp. 163–169.

[32] A. Luo, D. Cao, and A. Pang, *Visualizing spatial distribution data sets*, in Proc. Symp. Data Visualization 2003 (VISSYM '03), 2003, pp. 29–38.

[33] S. Manay, B.-W. Hong, A. Yezzi, and S. Soatto, *Integral invariant signatures*, in Computer Vision - ECCV 2004, T. Pajdla and J. Matas, eds., vol. 3024 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2004, pp. 87–99.

[34] T. Masuda, S. Imazu, S. Auethavekiat, T. Furuya, K. Kawakami, and K. Ikeuchi, *Shape difference visualization for ancient bronze mirrors through 3d range images.*, Journal of Visualization and Computer Animation, 14 (2003), pp. 183–196.

[35] M. Meyer, R. Kirby, and R. Whitaker, *Topology, accuracy, and quality of isosurface meshes using dynamic particles*, Visualization and Computer Graphics, IEEE Transactions on, 13 (2007), pp. 1704–1711.

[36] P. Miranda, R. da S. Torres, and A. Falcao, *Tsd: A shape descriptor based on a distribution of tensor scale local orientation*, in Computer Graphics and Image Processing, 2005. SIBGRAPI 2005. 18th Brazilian Symposium on, oct. 2005, pp. 139 – 146.

[37] A. Pang, C. Wittenbrink, and S. Lodha, *Approaches to uncertainty visualization*, The Visual Computer, 13 (1997), pp. 370–390.

[38] M. Pauly, N. Mitra, and L. Guibas, *Uncertainty and variability in point cloud surface data*, in Proc. Eurographics Symp. Point Based Graphics, 2004, pp. 77–84.

[39] P. Persson, *Mesh Generation for Implicit Geometries*, PhD thesis, MIT, 2004.

[40] T. Pfaffelmoser, M. Reitinger, and R. Westermann, *Visualizing the positional and geometrical variability of isosurfaces in uncertain scalar fields*, Computer Graphics Forum, 30 (2011), pp. 951–960.

[41] E. Pichon, , E. P. A, D. N. B, and M. N. A, *A laplace equation approach for shape comparison*, 2006.

[42] U. Pinkall and K. Polthier, *Computing discrete minimal surfaces and their conjugates*, Experimental Mathematics, 2 (1993), pp. 15–36.

[43] K. Pothkow and H.-C. Hege, *Positional uncertainty of isocontours: Condition analysis and probabilistic measures*, Visualization and Computer Graphics, IEEE Transactions on, 17 (2011), pp. 1393 –1406.

[44] K. Pöthkow, B. Weber, and H.-C. Hege, *Probabilistic marching cubes*, Computer Graphics Forum, 30 (2011), pp. 931–940.

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

[46] K. Rose, *Deterministic annealing, clustering, and optimization*, PhD thesis, California Institute of Technology, 1991.

[47] K. Rose, *Deterministic annealing for clustering, compression, classification, regression, and related optimization problems*, in Proceedings of the IEEE, 1998, pp. 2210–2239.

[48] R. Rustamov, *Laplace-beltrami eigenfunctions for deformation invariant shape representation*, in Proceedings of the fifth Eurographics symposium on Geometry processing, Eurographics Association, 2007, pp. 225–233.

[49] N. Saquib, W. Kossler, and M. Deady, *Calculation of muon depth distributions in thin film single crystals*, Physics Procedia, 30 (2012), pp. 42–45.

[50] O. Sorkine, *Laplacian mesh processing*, PhD thesis, Tel Aviv University, 2006.

[51] J. Sun, M. Ovsjanikov, and L. Guibas, *A concise and provably informative multi-scale signature based on heat diffusion*, in Proceedings of the Symposium on Geometry Processing, SGP '09, Aire-la-Ville, Switzerland, Switzerland, 2009, Eurographics Association, pp. 1383–1392.

[52] M. Tory, T. Mller, and M. S. Atkins, *Visualization of time-varying mri data for ms lesion analysis*, in Proceedings of SPIE Medical Imaging, 2001, pp. 590–598.

[53] O. van Kaick, H. Zhang, G. Hamarneh, and D. Cohen-Or, *A survey on shape correspondence*, Computer Graphics Forum, 30 (2011), pp. 1681–1707.

[54] R. C. Veltkamp, *Shape matching: Similarity measures and algorithms*, 2001, pp. 188–197.

[55] C. Weigle and R. M. T. II, *Visualizing intersecting surfaces with nested-surface techniques*, in 16th IEEE Visualization Conference (VIS 2005), 23, 2005, pp. 64–64. http://www.odysci.com/article/1010112988846207.

[56] L. Zadeh, *Fuzzy sets*, Information Control, 8 (1965).