SHAPE CORRESPONDENCES:
LOCAL TO GLOBAL

by
Luming Liang
A thesis submitted to the Faculty and the Board of Trustees of the Colorado School of Mines in partial fulfillment of the requirements for the degree of Doctor of Philosophy (Computer Science).

Golden, Colorado
Date ______________________

Signed: ______________________  
Luming Liang

Signed: ______________________  
Dr. Andrzej Szymczak  
Thesis Advisor

Golden, Colorado
Date ______________________

Signed: ______________________  
Dr. Randy Haupt  
Professor and Head  
Department of Electrical Engineering & Computer Science
ABSTRACT

Finding correspondences between two or more shapes is a fundamental and still unsolved problem in computer graphics and computer vision. Typically, one is interested in finding correspondence between similar objects (e.g. shapes representing different four-legged animals) or deformed versions of the same object (e.g. model of a human in different poses). The problem often suffers from ambiguities, which are brought about by shape symmetry, point slippage, edge stretching and shrinking. Most approaches to shape correspondence put restrictions on the deformation model: for example, matching techniques tailored for near isometric, area preserving or articulated deformations have been developed. Ideally, one would like to design an optimization based approach that would produce an optimal correspondence subject to constraints on the deformation model. However, setting up an optimization problem that can reliably provide a high quality solution and, at the same time, is computationally tractable, has been a major challenge.

The correspondence problem solutions are often broken into three stages:

- extract salient features in the input shapes;
- perform rough matching of the salient features using descriptors;
- globally register two shapes based on the rough matching.

We propose several new contributions to different stages of this framework.

First, we design a local shape descriptor based on the classical Spin Image. Our descriptor (Spin Contour) is essentially the contour of the original Spin Image. It provides considerably higher quality matching results while making comparisons between the descriptors more efficient.

Second, we introduce the *Geodesic Spin Contour*, a variant of the Spin Contour suitable for non-rigid near-isometric shape matching by replacing the Euclidean-based spin coordinates
with geodesic-based coordinates. This descriptor compares favourably with state-of-the-art local shape descriptors when for matching shapes deformed in a near-isometric manner. The Geodesic Spin Contour is suitable for partial matching, i.e. matching shapes with missing parts.

Third, we develop a fully automatic surface registration scheme. This method matches near-isometric shapes by globally minimizing the geodesic distance differences between pairs of features.

Finally, we extend the Iterative Closest Point (ICP) scheme to nonrigid non-isometric registration. Instead of using 1-1 mapping, we use many-many mapping to recover the nontrivial underlying deformation.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>iii</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>ix</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>xiv</td>
</tr>
<tr>
<td>ACKNOWLEDGMENTS</td>
<td>xv</td>
</tr>
<tr>
<td>CHAPTER 1 INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Problem statement</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Challenge 1: Semantically meaningful mapping</td>
<td>2</td>
</tr>
<tr>
<td>1.3 Challenge 2: High computational cost</td>
<td>4</td>
</tr>
<tr>
<td>1.4 Thesis overview</td>
<td>5</td>
</tr>
<tr>
<td>CHAPTER 2 BACKGROUND AND PREVIOUS RESEARCH</td>
<td>7</td>
</tr>
<tr>
<td>2.1 Shape descriptor</td>
<td>7</td>
</tr>
<tr>
<td>2.1.1 Shape descriptors for rigid matching</td>
<td>8</td>
</tr>
<tr>
<td>2.1.2 Shape descriptors for nonrigid matching</td>
<td>8</td>
</tr>
<tr>
<td>2.2 Deformation models</td>
<td>10</td>
</tr>
<tr>
<td>CHAPTER 3 SPIN CONTOUR</td>
<td>15</td>
</tr>
<tr>
<td>3.1 Review of the Spin Image</td>
<td>15</td>
</tr>
<tr>
<td>3.2 Definition</td>
<td>16</td>
</tr>
<tr>
<td>3.3 Central point direction</td>
<td>18</td>
</tr>
<tr>
<td>3.4 Matching spin contours</td>
<td>19</td>
</tr>
<tr>
<td>3.5 Comparison</td>
<td>19</td>
</tr>
</tbody>
</table>
4.6.2 Limitations .................................................. 56

CHAPTER 5 FULLY AUTOMATIC ISOMETRIC SHAPE REGISTRATION ........ 59

5.1 Problem formulation .......................................... 59

5.2 Methodology .................................................... 60

5.2.1 Furthest point sampling .................................... 60

5.2.2 Initial correspondence trim ................................. 61

5.2.3 Optimization on the coarse level ......................... 61

5.2.4 Local optimization of the correspondences ............... 63

5.2.5 Refinement .................................................... 63

5.3 Evaluation ...................................................... 65

5.3.1 Time ......................................................... 66

5.3.2 Accuracy ..................................................... 67

CHAPTER 6 NONRIGID ITERATIVE CLOSEST POINTS ................. 68

6.1 Summary ....................................................... 68

6.2 Assumptions and Limitations ................................ 68

6.3 Overall procedure ............................................ 70

6.4 Distance energy ............................................... 70

6.5 Potential energy ............................................... 70

6.6 Combinatorial Laplacian ...................................... 71

6.6.1 Geometry-based Laplacian ................................. 72

6.7 Energy minimization .......................................... 72

6.8 Correspondence search ....................................... 73

6.9 Evaluation ...................................................... 75
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1.1</td>
<td>Sparse (b) and dense (c) mapping between two human silhouette curves with different sampling rates (a).</td>
<td>2</td>
</tr>
<tr>
<td>Figure 1.2</td>
<td>Dense correspondences (c) found between source (a) and target (b) camel meshes. Black lines represent source-to-target correspondences, red lines represent target-to-source correspondences.</td>
<td>3</td>
</tr>
<tr>
<td>Figure 1.3</td>
<td>Symmetric ambiguity.</td>
<td>3</td>
</tr>
<tr>
<td>Figure 1.4</td>
<td>Dense correspondences (c) found between source (a) and target (b) vertebrae meshes. Black lines represent source-to-target correspondences, red lines represent target-to-source correspondences.</td>
<td>4</td>
</tr>
<tr>
<td>Figure 3.1</td>
<td>The cylindrical coordinate ((\alpha, \beta)) of (p_i) relative to oriented point (p).</td>
<td>16</td>
</tr>
<tr>
<td>Figure 3.2</td>
<td>The contribution of surface triangle (\triangle abc) to a raster pixel ((\alpha, \beta)) in the face-based spin image of oriented point (p): the shaded area (defg). Note that in some cases, edges of the intersection may be curves not straight line segments.</td>
<td>17</td>
</tr>
<tr>
<td>Figure 3.3</td>
<td>Face-based spin image (b) and spin contour (d) at the position located at the pig’s eye (a). The final step in obtaining the spin contour is to connect the discrete points in (c) to form (e). (d) is the intermediate stage of the contour building (after connecting the upper part of the contour).</td>
<td>18</td>
</tr>
<tr>
<td>Figure 3.4</td>
<td>The fandisk model with dense (a) and sparse (b) sampling.</td>
<td>20</td>
</tr>
<tr>
<td>Figure 3.5</td>
<td>Feature matching between the sparse and dense fandisk models. (a) and (c) show matching results of face-based spin image. (b) and (d) show matching results of spin contour. The point directions are set to face normals in (a) and (b), to the new point direction proposed in section 3.2 in (c) and (d). Black lines indicate correct matches, while red lines indicate incorrect matches.</td>
<td>20</td>
</tr>
<tr>
<td>Figure 3.6</td>
<td>The uneven sampled denser horse model in the right is resampled to an evenly sampled sparser horse model in the left.</td>
<td>21</td>
</tr>
</tbody>
</table>
Feature matching between the sparse and dense horse models. (a) and (c) show matching results of face-based spin image. (b) and (d) show matching results of spin contour. The point directions are set to face normals in (a) and (b), to the new point direction proposed in section 3.2 in (c) and (d). Black lines indicate correct matches, while red lines indicate incorrect matches.

Recall versus 1-precision curves on sparse and dense Fandisk models (a) and horse models (b).

Casting model. From left to right: original model, noisy level 0.1mr, 0.2mr and 0.3mr.

Recall vs 1-Precision curves of the casting matching.

Summed distances error curves of the casting matching.

Matching results using spin contour in different noise levels: (a) 0.1mr; (b) 0.2mr; (c) 0.3mr. Black lines show correct matching, red lines indicate wrong matchings.

Recall-precision curves of the Stanford 3D scan matching in different noise levels: (a) 0.1mr; (b) 0.2mr; (c) 0.3mr. Black lines show correct matching, red lines indicate wrong matchings.

Spacetime stereo model and scene.

Recall-precision curves of the spacetime stereo data: (a) raw data; (b) sub-sampled models.

Crane data and its Geodesic spin contours.

Results for the crane dataset. (a) Geodesic distance from the vertex $v$ shown as the grey ball; (b) average geodesic distance; (c) spin image of $v$; (d) spin contour of $v$; (e) geodesic spin image of $v$ and (f) geodesic spin contour of $v$.

Geodesic spin images (b)(c) and geodesic spin contours (d)(e) of corresponding points on a pair of near-isometric centaur models (a). (b) and (d) are computed from the point (gray ball) on the left model; while (c) and (e) are computed from the point on the right model.
Figure 4.4 Geodesic spin contour (d) at the position located at the left hand of the Crane data set (a). The final step in obtaining the geodesic spin contour is to connect the discrete points in (b) to form (d). (e) is the intermediate stage of the contour building (after connecting the upper part of the contour). The process of filling the gaps in the contour is illustrated in the bottom row. We represent the contour as a set of samples on a rectangular grid (shown as pixels here).

Figure 4.5 (a) Crane data; (b) average geodesic distance; (c) local average geodesic distance; (d) full geodesic spin contour; (e) partial geodesic spin contour and (f) partial geodesic spin contour (with local agd) at a point on left hand indicated by the gray dot on (a).

Figure 4.6 Two surfaces from the horse gallop dataset with color-coded descriptor distances to the dotted point on the source (right) mesh plotted: (a) Heat Kernel Signature, (b) Wave Kernel Signature and (c) Geodesic spin Contour.

Figure 4.7 Results for surfaces shown in Figure 4.6, with the tail removed from the source surface (always shown on the right): (a) Heat Kernel Signature, (b) Wave Kernel Signature and (c) Geodesic spin Contour.

Figure 4.8 Robustness and invariance of Geodesic Spin Contour. Geodesic spin contours at the same location of (c) base model, (d) denser model and (e) partial model.

Figure 4.9 Vertex matching using different descriptors on full meshes.

Figure 4.10 The hit rate (a) and average geodesic error (b) for full mesh matching shown in Figure 4.9.

Figure 4.11 Vertex matching using different descriptors on partial data.

Figure 4.12 The hit rate (a) and average geodesic error (b) for partial mesh matching shown in Figure 4.11.

Figure 4.13 The hit rate (b) and average geodesic error (c) for partial mesh matching (a).

Figure 4.14 TOSCA wolf model dense to coarse matching (a) with corresponding hit rates (b) and average geodesic errors (c).

Figure 4.15 TOSCA wolf partial model0 to full model1 matching (column 1) with corresponding hit rates (column 2) and average geodesic errors (column 3).
Figure 4.16 Hit rates (a) and average geodesic errors of TOSCA wolf models. The number 20, 40, 60 denote the number of missing vertices.

Figure 4.17 The average hit rates (a) and average average geodesic errors (b) of TOSCA centaur model matchings.

Figure 4.18 TOSCA centaur model matchings (column 1) with corresponding hit rates (column 2) and average geodesic errors (column 3).

Figure 4.19 TOSCA centaur model matchings (column 1) with corresponding hit rates (column 2) and average geodesic errors (column 3).

Figure 4.20 TOSCA centaur model matchings (column 1) with corresponding hit rates (column 2) and average geodesic errors (column 3).

Figure 4.21 TOSCA centaur model matchings (column 1) with corresponding hit rates (column 2) and average geodesic errors (column 3).

Figure 4.22 TOSCA centaur model matchings (column 1) with corresponding hit rates (column 2) and average geodesic errors (column 3).

Figure 5.1 Furthest sampling on the TOSCA wolf model: 0 and 2.

Figure 5.2 Possible feature correspondences according to the $L_1$ distances between Geodesic Spin Contour of features.

Figure 5.3 Initial coarse match result on the TOSCA wolf model: 0 and 2.

Figure 5.4 Optimized coarse matching on the TOSCA wolf model: 0 and 2.

Figure 5.5 Coarse-to-fine match of the TOSCA wolf model: 0 and 2.

Figure 5.6 The time consumed in adding different number of new matches.

Figure 5.7 Geodesic error between the founded match and the ground truth.

Figure 6.1 The source and target surfaces.

Figure 6.2 Definitions of angles $\alpha_{ij}$ and $\beta_{ij}$.

Figure 6.3 The effect of two-step multiple correspondence search. In (b) and (e), the black lines and the red lines represent the forward correspondences and the inverse correspondences. The source and the target surfaces are shown in Figure 6.1.
Figure 6.4 Comparisons with other methods. .................................................. 77
Figure 6.5 Bidirectional closest point search. .................................................. 78
Figure 6.6 Comparisons with Coherent Point Drift and its improved variant. ... 80
Figure 6.7 Results shown with textures. .......................................................... 81
Figure 6.8 Results shown with textures. .......................................................... 82
Figure 6.9 Results comparison between edge-based and Laplacian-based potential energy with two-multiple correspondence. The pink surfaces are deformed models. .......................................................... 83
Figure 6.10 A saddle point in the source mesh. ............................................... 83
Figure 6.11 First three shape variation components. The blue shape is the average model $P_{mean}$, the pink shapes are three eigen models which are created by adding three PCs to $P_{mean}$, respectively. .................................................. 86
Figure 6.12 Vertebrae models before registration. (a) shows the template model. (b) and (c) show overlapping views of 14 models. .................................................. 87
Figure 6.13 First five shape variation components. The blue shape is the average model $P_{mean}$, the pink shapes are three eigen models which are created by adding three PCs to $P_{mean}$, respectively. .................................................. 87
Figure 6.14 First three shape variation components got by using an interactive fitting method. The blue shape is the average model $P_{mean}$, the pink shapes are three eigen models which are created by adding three PCs to $P_{mean}$, respectively. .................................................. 88
Figure 6.15 24 feature points shown on the source mesh with different views. .... 88
Figure 6.16 Feature points shown on source, target and fitted mesh. ............... 89
LIST OF TABLES

Table 3.1  Feature matching between the original and noisy casting models. A black line indicates a correct match, while a red line indicates a wrong match.  . 24

Table 6.1  Comparisons with other methods with different inputs.  . 79

Table 6.2  Distances between landmarks before and after registration using different methods.  . 85
ACKNOWLEDGMENTS

There are many people whose support has helped bring this thesis to completion.

I would like to thank my advisor Dr. Andrzej Szymczak for his constant encouragement and generous help throughout these years. He is not only a passionate and wise researcher with great insight to the nature of problems, but also a nice teacher with full patience to his students. He helped me survive in the dark days when I was close to quitting the program.

I would like to thank my graduate committee for helping me go through this research process. Their comments and feedbacks greatly improve the quality of this work.

Last, but certainly not least, I would like to thank my family: my wife Hui and my parents. They provide me consistent supports in every aspects through these years. I cannot imagine the days without them.
CHAPTER 1
INTRODUCTION

There are numerous problems that depend on shape correspondence. Examples within computer graphics include scan alignment [1, 2], shape animation [3], mesh segmentation [4, 5], surface editing, shape interpolation and texture transfer. In biomechanics, correspondences between bones of different people can lead to probabilistic shape models that can be used to drive simulations [6–8]. In robotics, shapes correspondences can link objects that a robot encounters and objects in its database, and help it navigate through new environments [9].

1.1 Problem statement

The correspondence finding problem can be formulated as follows. Given shapes $S_1$, $S_2$, $S_3$, ..., $S_n$, find a meaningful relation $R$, or mapping between their elements [10]. When elements $a$ and $b$ of two shapes $S_i$ and $S_j$ are related to each other i.e.,

$$(a, b) \in R,$$

$$a \in S_i, b \in S_j,$$

(1.1)

$(a, b)$ is defined as a valid correspondence. The shapes can be represented in the form of point sets, curves, meshes, images, or parameterized manifolds. Shape elements are differently defined according to the granularity. One could find a very sparse set of correspondences that only maps features to features between two shapes, as shown in Figure 1.1b. In contrast, one could also find relations between vertex sets of these two shapes, as shown in Figure 1.1c. In this context, the shape elements are vertices; while in the former case, they are the extracted feature points. In practice, one-to-one correspondences are often too restrictive, especially when the mapping between two shapes is far from isometric or input shapes have incomparable sampling rates. At this point, one-to-many or many-to-many correspondences are required. The dense mapping shown in Figure 1.1c is an example of the one-to-many correspondence set. Here, the source curve (blue) has 128 vertices and the target curve (red)
has 64 vertices.

**Figure 1.1:** Sparse (b) and dense (c) mapping between two human silhouette curves with different sampling rates (a).

### 1.2 Challenge 1: Semantically meaningful mapping

Defining a meaningful mapping between shapes is highly application dependent. Good correspondences are differently defined with different criteria. For example, rigid alignment of 3D scans is often based on proximity based metrics. However, local features have been used to improve the performance [1]. Proximity alone becomes inadequate to register shapes with large deformations. More sophisticated search criteria are required. For example, the articulation of the motion [11–13] is a reasonable constraint when matching robotics arms in different poses. In this task, points on the shapes are grouped into likely articulated parts during the pairing. When the underlying deformation is unknown and large, the problem is still unsolved. Possible correspondence ambiguities make this problem more complex.

Ambiguities in mapping two shapes are commonly encountered in different applications. As stated in [9], two major geometric sources of ambiguity are global ambiguities and local ambiguities.

Global ambiguities often can be found between shapes that have intrinsic symmetries. For example, animals with four legs often have left-right symmetries. Both these legs have identical features (surface normals, curvatures, spin images). This fact makes the mapping
tend to be semantically incorrect, see the example shown in Figure 1.2. Vertices on one of the front legs of the source camel are mapped to vertices on both of front legs of the target camel. These correspondences are found using the method presented in [8].

Another example of the global ambiguity is shown in Figure 1.3, the matching results are symmetrically flips of the correct results.

Local ambiguities are even more common than the global ones. For example, slippage of correspondences along flat areas is difficult to avoid. Sometimes, lack of the prior knowledge about the deformation is also a source of introducing local ambiguities. Figure 1.4 illustrates such kind of ambiguities. Apparent correspondence inconsistencies show up between the area
of the tip of the right transverse process and the remaining places, as shown in Figure 1.4c. These correspondences are found using the method presented in [8].

Figure 1.4: Dense correspondences (c) found between source (a) and target (b) vertebrae meshes. Black lines represent source-to-target correspondences, red lines represent target-to-source correspondences.

1.3 Challenge 2: High computational cost

The size of 3D data is growing with advances in acquisition techniques. Typical 3D shapes (point sets or triangular meshes) consist of thousands of vertices. To find vertex-wise correspondences between two surfaces each with \( n \) vertices, one needs to explore a space of size superexponential in \( n \), and find a most meaningful and consistent set of matchings. Consistency of the matching is a strong restriction to the problem, and most definitions of consistency make the registration become NP-hard [14].

A common approach to simplify the problem is to use simple-to-compute maps as correspondences. For example, ICP-like (iterative closest point) methods [15, 16] use closest points as best matches. These matches, despite being imperfect, can be used to determine a correspondence map between two surfaces, assuming they are close enough.

Fréchet distance can be used as a matching criterion between shapes. However, even in two dimensions, computing this distance is NP-hard [14]. Alt and Buchin [17] give a practical relaxation of the problem, leading to weak Fréchet distance. This distance can be computed in polynomial time for a pair of parametrized triangulated surfaces. Zeng et al. [18] cast surface registration as a graph-matching problem. Since the high-order graph matching used
in their paper is NP-hard and nonconvex, they use the flexible dual-decomposition technique to find a sub-optimal result.

Another class of methods use a multiresolution approach to match surfaces [19–21]. They find correspondences at coarser level, and gradually add more correspondences. However, choosing a consistent coarse correspondence is not a simple task.

1.4 Thesis overview

The main objective of this thesis is to explore solutions to the above-mentioned challenges. The goal of this research is to find a meaningful dense mapping between two shapes in a tractable time.

In Chapter 2, I briefly review the background and related work.

In Chapter 3, I propose a local shape descriptor for rigid registration, named spin contour. This descriptor is essentially the outer contour of spin image, a classical shape descriptor. An efficient algorithm for computing the spin contour is also discussed in this chapter.

In Chapter 4, I extend spin contour to the nonrigid case by replacing the Euclidean-based spin coordinates with geodesic-based coordinates. This leads to the geodesic spin contour descriptor for isometric shape matching.

In Chapter 5, I describe a fully automatic registration scheme for two near-isometric shapes. This scheme uses coarse-to-dense mapping to reduce the computational cost.

In Chapter 6, I propose a Nonrigid Iterative Closest Points method which use 1-to-many correspondences in the registration. This method is an extension to the classical Iterative Closest Point (ICP) Method.

I conclude the thesis and explore possible future work in Chapter 7.

Chapters in this thesis have also been published in, submitted to, or going to be submitted soon to several conferences and journals:


• Luming Liang and Andrzej Szymczak. Velocities of Vector Field Features, to be submitted soon.

• Luming Liang and Andrzej Szymczak. Fully Automatic Shape Registration of Near-isometric Surfaces, to be submitted soon.
CHAPTER 2
BACKGROUND AND PREVIOUS RESEARCH

The problem of finding shape correspondences has a long history in computer graphics and computer vision. Many methods have been proposed to solve this problem with different contexts. Since the dimensionality of the search space is always quite high, many methods [11, 12, 19, 22] confine the search space by extracting feature points and matching them according to local shape descriptors. After this rough mapping, they generate dense correspondences using some optimization scheme. Here in this chapter, I will briefly review some shape descriptors and related global consistent matching methods.

2.1 Shape descriptor

Many shape matching and search engines rely on estimates of point correspondences to control the shape registration process or evaluation of similarity metrics. These estimates are often obtained using shape descriptors.

A shape descriptor can be local or global. A global shape descriptor is computed for the entire object. Invariant histograms and shape distributions are examples of this kind of descriptors. These descriptors are often used in shape retrieval but are not suitable for shape registration.

Local descriptors encode the shape of a neighborhood of a point. For example, point normals and discrete curvatures can be estimated from a small region around a surface point and can be used as simple shape descriptors. In this thesis, I focus only on local descriptors, since they are more suitable in shape registration.

Local descriptors encode the shape of a neighborhood of a point. For example, point normals and discrete curvatures can be estimated from a small region around a surface point and can be used as simple shape descriptors. In this thesis, I focus only on local descriptors, since they are more suitable in shape registration.

Local shape descriptors are typically designed to be invariant under some kind of transformations. Therefore, local descriptors can be categorized into two types: rigid and nonrigid, which are invariant under rigid (rotation and translation) and some kind of nonrigid, for example, isometric, transformations.
2.1.1 Shape descriptors for rigid matching

Rigid motion does not change the shape: it only affects the location and orientation of a shape in space. Rigid shape descriptors are always object-centered, which means that the descriptors are computed using object-related coordinates.

Campbell and Flynn [23] identify salient regions on 3D surfaces by using shape index and curvedness. Chen and Bhanu [24] define a descriptor that is characterized by its centroid, its local surface type and a 2D histogram. These descriptors are computed only on feature points and then used in a voting-based ICP procedure. Inspired by Scale Invariant Feature Transform (SIFT) [25], the most successful 2d local feature descriptor in the image recognition, Tombari et al. [26] propose the Signature of Histograms of Orientations (SHOT) descriptor. It is computed based on a stable estimation of the local reference frame (normal and tangent plane) to decrease the influence of noise on the result. However, its quality may be degraded by the uneven sampling.

Besides descriptors that are rotation and translation invariant, invariance under scale can be considered as another design requirement. Such descriptors were recently proposed for range images [27–30]. A scale-invariant descriptor was first proposed by Novatnack and Nishino for registering range images with different sizes [27, 28], and then was used in a range images registration scheme [29]. In addition, Bariya et al. [30] extend these ideas to extract a set of multi-scale geometric features for buildings based scale-space representation for range images.

A thorough review of shape descriptors can be found in a review by Kaick et al. [10]. The spin image [31] as well as its multiresolution [32] and face-based [33] variants are local shape descriptors. Spin images have been a useful tool in rigid registration.

2.1.2 Shape descriptors for nonrigid matching

Nonrigid deformations are more complex than rigid ones. Since the dimension of the nonrigid deformation is high, many researchers constrain the deformation in some ways, for
example, assume the motion is articulated or the deformations are isometric.

For articulated shape registration, the shape is assumed to consist of a number of rigid parts meeting at joints. Under this assumption, the spin image with local support [11] and multiresolution curvature [22] are suitable to finding candidate matches.

Another common assumption is that the deformation is an isometry, or that it is close to an isometry. Isometry is defined as a distance-preserving map between two spaces. In the surface registration case, the isometric deformation assumption means the geodesic distances between any two points on the source mesh must be as same as the geodesic distances between their corresponding points on the target mesh. To recover the isometric deformation, the shape descriptor must be invariant under isometric transform. The heat kernel signature (HKS) [34–37] is a local shape descriptor, which captures the dissipation of heat from a point onto the rest of the surface over time. It encodes the local to global information around a point on the surface in a vector. Since the heat diffusion on a surface is fully controlled by the heat kernel that is invariant under isometric deformations, the heat kernel signature is isometry invariant and has been successfully applied to isometric shape matching [34]. HKS performs well also for close to isometric deformations.

The heat kernel signature is a spectral shape analysis method. This group of methods use the spectrum (eigenvalues or eigenvectors) of the Laplace-Beltrami operator to analyze shapes. Beside HKS, Global point signature (GPS) [38] is also belongs to this group. This descriptor encodes scaled eigenfunctions of the Laplace-Beltrami operator computed at a point. It is weaker than HKS because it cannot be used on partial shape matching whereas HKS can.

Another similar idea is Wave kernel signature (WKS). The WKS represents the average probability of measuring a quantum mechanical particle at a point on the surface [39]. This descriptor is also invariant under isometric deformations.
2.2 Deformation models

When computing correspondence between two shapes, one typically makes assumptions about the space of allowed deformations. In this section, we briefly review most common assumptions.

**Rigid alignment**: The rigidity of the shape greatly reduces the dimensionality of the search space. Early work related to 3D registration focused on rigid cases. A 3D rigid transformation can be represented as a superposition of a rotation and a translation. Rigid transformation between a pair of shapes can be determined using the Iterative Closest Point (ICP) algorithm [15, 40]. Clearly, such a rigid transformation defines a correspondence between the two shapes. The ICP algorithm works by iterating the following three steps:

1. Correspondence determination: select points on the source surface (source points). Pair each source point with its closest point (target point) on the target surface.

2. Error minimization: compute a rigid transformation $T$ that minimizes the mean square error between the source points (transformed with $T$) and their corresponding target points.

3. Update: apply transformation $T$ to the source surface; stop if $T$ is close to the identity transformation.

ICP is a local optimization algorithm and it generally requires close initialization (i.e. that the input source and target surfaces are close to being aligned) to converge. If initialization is close enough, the method is robust and usually converges monotonically.

Several methods have been developed to prevent the ICP algorithm from being trapped at local minima, especially when the input data contains outliers. For example, the RANSAC algorithm [41] is designed to robustly match two shapes corrupted by noise. This method randomly chooses 3 points from both the source and the target shape and computes a transformation matrix from them. The transformation matrix is scored based on the number of points on the transformed source shape that are close to the target shape. If that number
is large enough (greater than some predefined threshold), the transform matrix is accepted.
Otherwise, the point triplets are selected again until an acceptable transformation is found.

The original RANSAC algorithm has been enhanced in a number of ways. Irani and Raghavan [42] introduce randomization in the verification stage. Instead of counting over all possible points, they randomly choose some number of points to test whether the transform matrix is acceptable or not. Generally, this leads to a smaller number of points to test at the verification stage and therefore speeds up the algorithm.

Aiger et al. use 4-point [43] congruent sets instead of triplets to find candidate transformations in the RANSAC algorithm. This extension reduces the number of trials in finding an acceptable rigid transform matrix.

Articulated alignment: Many shapes are articulated, i.e. can be approximated with high accuracy by a number of rigid parts connecting at joints. In particular, humans and most machines and animals are articulated shapes.

Chang and Zwicker’s method [12] for articulated shape registration determines and matches the rigid parts ('bones' in the paper) via clustering the estimated local transformations. Some other articulated registration methods [11, 13, 22, 44, 45] are based on similar ideas.

Isometric alignment: In the context of shape matching, isometric deformations are understood as deformations that preserve the geodesic distances on surfaces up to a certain error. To find as isometric as possible mapping between two surfaces, many methods attempt to minimize the isometric distortion of an unknown map \( M \) from the source to the target surface [46]. The isometric distortion can be defined by \( E = \sum |g(s_i, s_j) - g'(M(s_i), M(s_j))| \), where \( g \) and \( g' \) are geodesic distances on the source and target surfaces and \( s_i \) are samples on the source surface.

The Generalized Multidimensional Scaling (GMDS) method, introduced by Bronstein et al. [46, 47], proposes to solve this problem by optimally embedding one surface into another. This method requires solving a difficult non-convex optimization problem. Like ICP, this method may possibly be trapped at local minima.
The Heat Kernel Map (HKM), based on the Heat kernel signature [34], is proposed to match near-isometric surfaces in [48]. This method shows that the full mapping can be defined by a single correspondence. The disadvantage is that their assumption to the isometry is too strict.

Methods proposed in [20, 49] build upon the fact that isometries are low-dimensional subspaces of conformal maps. Three correspondences uniquely determine a conformal map between the source and target surfaces. Lipman et al. [49] use a voting scheme to select a high quality correspondence. Kim et al. [20] blend a set of maps in a total low-distortion criterion.

Another kind of isometric matching algorithms take ideas from RANSAC. Tevs et al. [50] explicitly detects the uncertainties in the shape matching. This method extracts a consistent subset in the full matching graph using a RANSAC-like sampling algorithm. New matches are gradually added to the set in the order of decreasing estimated probability of being correct. This algorithm is further improved by introducing a planned landmark selection (PLANSAC) [51], which means the algorithm determines which samples are best to test in order to find the most reasonable correspondence subset.

Other nonrigid alignments:

Some nonrigid deformations are not articulated or isometric. Numerous extensions of the ICP algorithm to the general non-rigid case have been proposed. Examples of novel ideas related to the correspondence determination step include fuzzy correspondencies [52], Expectation Maximization based iterated closest point (EM-ICP) [53] and a method based on a priori knowledge [54]. By altering the error minimization step, one can develop nonrigid registration algorithms. The key is to change the optimization problem in a way that makes non-rigid transformations feasible, for example as described in [16, 52, 54]. In [55], a non-Euclidean distance which depends on a unit normal difference is used as the distance between two surfaces. This accelerates convergence, but makes the optimization problem non-quadratic, and thus increases the computational complexity.
Instead of finding a 1-1 mapping between point sets of a shape, some methods look for many-to-many correspondences to avoid local minima [52, 53, 56-60]. The basic motivation of doing so is that matching single points is not as robust as matching clusters of points. Specifically, when there are some assumptions on motion of points, like rigidity [53, 56] or articulated motion [11–13, 44, 45, 61], matching patches or grouped points rather than single points is always the right way to go.

The earliest multiple-correspondences-based scheme is introduced by Rangarajan et al. [56], where a correspondence matrix $M_{ij}$ is maintained and updated during each iteration. Here, $i, j$ are indices of feature points of the source and the target models and $M_{ij} = 1$ when point $i$ in the source model corresponds to point $j$ in the target model. Both correspondences and transformations are found by iteratively minimizing a cost function. The multiple correspondences are represented by a technique named softassign, which relaxes the binary correspondence matrix $M_{ij}$ to be continuous valued. The softassign scheme is further improved by Granger and Pennec [53] into Multi-scale Expectation-Maximization ICP method (EM-ICP for short). Although this method is an extension of the rigid ICP, the underlying idea can be easily carried over to the nonrigid setting.

In 2003, Chui and Rangarajan propose the TPS-RPM method that uses thin-plate spline as a representation of spatial deformation. The authors also show that this method is as same as EM for Gaussian Mixture Model (GMM) [57]. Recently, Myronenko and Song [59] present the coherent point drift method that fits the GMM centroids (the source point set) to the data (the target point set) via maximizing the likelihood. This method can be used to recover deformations of feature point sets, especially if large subsets of the source and target point sets can be obtained from one another by applying a smooth deformation. However, it is not clear how to make it equally effective for surfaces, particularly in cases for which extracting consistent features is hard. Fitting the vertex set of the source surface to the vertex set of the target surface using the coherent point drift approach generally leads to poor results if the surfaces differ in vertex density. Jian and Vemuri [58, 60] use GMMs to represent
point sets and then cast the registration problem as a Gaussian mixture alignment problem. They show that many of the formerly proposed methods including ICP [15] can be viewed as special cases in their framework. Another contribution of their work is that they leverage the closed-form expression for the L2 distance between two Gaussian mixtures and therefore come up with an efficient implementation. There is a common assumption to these density-based registration algorithms, that shapes to be registered should have comparable sampling rates. This is because these methods interpret the shapes (point sets in their application) as samples drawn from close to compatible probability distributions on the source and target surfaces [60]. The problem of matching shapes is reduced to matching probability density functions. Therefore, the performance of these methods deteriorates if the sampling rates between models have large differences.
CHAPTER 3
SPIN CONTOUR

This chapter mainly consists of a paper submitted to Journal of Machine Vision and Applications.

3.1 Review of the Spin Image

Spin image [31, 62] is a powerful shape descriptor, useful in point set or surface registration. However, the usage of spin images is hampered by issues such as sensitivity to noise and sampling rate and time-consuming matching process.

We propose a novel spin image based local surface descriptor named spin contour to alleviate these problems. This descriptor is not an image but a 2d point set. Comparisons show that the spin contour is robust to noise and sampling differences. The matching time is also improved over spin images.

A spin image proposed by Johnson and Hebert [31, 63] is created for an oriented point, a point with a normal vector, which defines an object-centered coordinate system. For a particular oriented point $P$ with normal $n$, the spin image is obtained by first transforming the other samples to spin image coordinates by applying the mapping $S : R^3 \rightarrow R^2$ defined by

$$S(Q) = (\sqrt{||Q - P||^2 - (n \cdot (Q - P))^2}, n \cdot (Q - P)).$$

Intuitively, the first spin coordinate, often denoted by $\alpha$, is the distance from $Q$ to the line passing through $P$ and parallel to $n$. The second coordinate $\beta$, is the signed distance from $Q$ to the plane passing through $P$ and perpendicular to $n$, as shown in Figure 3.1. The spin image is obtained by splitting spin coordinate space into a number of bins (pixels) and making pixel intensities proportional to number of samples in the bin.
From the definition, it is not difficult to see that the spin image is highly dependent on the positions of surface vertices. To alleviate this disadvantage, Carmichael et al. [33] propose a face-based spin image. Two implementations of the face-based spin image are introduced: approximation and exact method. In the approximation method, a number of samples are pre-picked on each face, where the number is proportional to the area of this face. Then, the face-based spin image is approximated by projecting these samples to the spin coordinate space. Instead of pre-picking of samples, the exact method directly maps each face to the spin coordinate space. The intensity of each pixel in the image are not a bilinear splat of point counts, but the accumulated area of all mesh elements projected to the space bounded by particular ranges of $\alpha$ and $\beta$.

The computation is briefly illustrated in Figure 3.2. When computing the intensity of pixel $(\alpha, \beta)$ in the spin image with respect to an oriented point $(p, n)$ ($p$ denotes the location and $n$ denotes the normal vector), one first find the volume bounded by two planes corresponding to $\beta \pm 0.5$ and two cylinders corresponding to $\alpha \pm 0.5$. Any mesh element that insects with this volume, like $\triangle abc$, has nonzero contribution to the pixel intensity. This contribution is the area of the part of shape that stays in the volume, polygon $defg$ in Figure 3.2.

### 3.2 Definition

As shown by Carmichael et al. [33], the face-based spin image is more robust than the traditional spin image for surfaces that have different sampling rates. However, its computation is more expensive. In particular, the intersection area computation (Figure 3.2)
The contribution of surface triangle $\triangle abc$ to a raster pixel $(\alpha, \beta)$ in the face-based spin image of oriented point $p$: the shaded area $defg$. Note that in some cases, edges of the intersection may be curves not straight line segments.

is complex and slow. Our experiments show that the set of the influenced pixels (with the amount of influence disregarded) has a better resolving power than actual pixel values in the face-based spin image. This statement will be validated in Section 4. The spin contour is a representation of the set of influenced pixels.

**Definition 1:** For a 3D oriented point $P$ with direction $n$, we define the spin contour $C$ as the boundary of the image of the surface under the spin coordinate mapping $S$ defined in Section 2.

While computing the contour spin image precisely would be complex, there is a simple way to approximate it using the following algorithm. First, determine the range of the spin coordinate $\beta$. Then, pick a number of samples in that range. For each sample $\beta$ and triangle $\Delta$ of the mesh, compute the intersection of $\Delta$ with the plane $n \cdot (x - p) = \beta$ and determine the range $I_{\beta, \Delta}$ of $\alpha$ values for the intersection. The range is empty if the intersection is empty and otherwise is equal to the interval $[l, u]$, with $l$ and $u$ such that the intersection is tightly contained between cylinders of radii $l$ and $u$ around the line passing through $P$ and parallel to $n$.

Since every range is either empty or is a closed interval, the union of all ranges $I_{\beta, \Delta}$ over all triangles $\Delta$ can be uniquely expressed as a finite union of disjoint closed intervals. For each endpoint $\alpha$ of one of these intervals, we insert the point $(\alpha, \beta)$ into the spin contour, as shown in Figure 3.3c. The contour points are then connected form a number of closed curves (see Figure 3.3e). If $I_\beta$ is the range (union of intervals of $\alpha$) for a given $\beta$ (denoted
by black points in a certain row in Figure 3.3f), we add samples in \( I_\beta \setminus I_{\beta-\delta \beta} \) (red points in Figure 3.3g) and \( I_\beta \setminus I_{\beta+\delta \beta} \) (blue points in Figure 3.3h). These point insertions close the spin contour.

![Images](image_url)

**Figure 3.3:** Face-based spin image (b) and spin contour (d) at the position located at the pig’s eye (a). The final step in obtaining the spin contour is to connect the discrete points in (c) to form (e). (d) is the intermediate stage of the contour building (after connecting the upper part of the contour).

### 3.3 Central point direction

Spin image is traditionally computed using the normal estimate as the vector \( n \) [63]. Since surface normals are prone to noise and local sampling variations, they are far from unambiguous in practice. Surface normal is not the only possible choice for this purpose: Any uniquely defined direction can be chosen instead. In this paper, for a point \( P \) on the surface we use the vector extending from \( P \) to the center of mass of the surface as the vector \( n \). In practice, \( n \) is computed using the following formula:
\[ n = \frac{\sum_\Delta \text{area}(\Delta) \text{center}(\Delta)}{\sum_\Delta \text{area}(\Delta)} - P. \] 

(3.1)

This direction tends to be stabler under resampling or noise than local normal vector estimates.

The central point direction has a straightforward local extension to Equation 3.1:

\[ n = \frac{\sum_{\Delta \in B(P,r)} \text{area}(\Delta) \text{center}(\Delta \in B(P,r))}{\sum_\Delta \text{area}(\Delta)} - P, \] 

(3.2)

where, \( B(P,r) \) is a ball in with center at \( P \) and \( r \) as its radius. Equation 3.2 is useful when the input data has clutter or is partial.

### 3.4 Matching spin contours

To compare spin contours, we use the symmetrized and normalized version of the \( L^1 \) distance as the similarity metric. For two spin contours \( X \) and \( Y \), the metric is defined by

\[ L^1(X,Y) = \frac{1}{|X| + |Y|} \left( \sum_{x \in X} d(x,Y) + \sum_{y \in Y} d(y,X) \right), \]

where \( d(a,A) \) is the distance between a point \( a \) and its closest point in the set \( A \).

### 3.5 Comparison

We quantitatively compare the spin contour with several state-of-the-art descriptors using recall vs 1-precision as suggested by Tombari et al. [26]. This criterion is common in comparing descriptors after been proposed by Mikolajczyk and Schmid [64].

We discuss two sets of experiments under precision-recall measurements. The first set is designed to evaluate the recognition ability of descriptors for unevenly sampled models. The second set is aimed at the registration of noisy models.

### 3.6 Unevenly sampled models

Many methods such as the traditional spin image [63], SHOT descriptor [26] and the improved spin image [65] do not take the uneven sampling issue into account. Face-based
spin image [33] is proposed to alleviate this problem. Here, we show that the spin contour has better performance than the face-based spin image. Note that uneven sampling also influences the estimated normal direction, which affects the quality of descriptors based on normal direction estimates. Here, we show that the direction $n$ computed using the formula in Section 3.2 leads to better results than the stable local reference frame estimation method proposed by Tombari et al. [26].

The first pair of input meshes are two fandisk models, as shown in Figure 3.4. The sparse model contains 370 triangles and the dense one contains 12946 triangles. We select 1094 point pairs that are closest in the space as the correct matches. These points are randomly selected from the surface triangles with a uniform distribution.

![Figure 3.4: The fandisk model with dense (a) and sparse (b) sampling.](image)

![Figure 3.5: Feature matching between the sparse and dense fandisk models. (a) and (c) show matching results of face-based spin image [33]. (b) and (d) show matching results of spin contour. The point directions are set to face normals in (a) and (b), to the new point direction proposed in section 3.2 in (c) and (d). Black lines indicate correct matches, while red lines indicate incorrect matches.](image)

We compute different descriptors for this features on both models. We choose to compare the spin contour with the face-based spin image [33], since this method also claims to deal
with the uneven sampling matching problem. We use the same set of parameters for both methods: the support length is 100%, the final image size is $60 \times 120$. For spin contour, this size means we sample the spin space with 120 $\beta$ values and 60 $\alpha$ values to build the contour. The point directions are set to face normals (Figure 3.5a and Figure 3.5b), and the central directions (Figure 3.5c and Figure 3.5d) proposed in section 3.2. Here, the face is the one where the point is sampled. From now on, we consistently use ”FSI-fn” to denote face-based spin image with face normals as point directions and ”FSI-cd” to denote face-based spin image with central point directions; ”SC-fn” and ”SC-cd” for spin contour with face normals and central point directions. In Figure 3.5, each selected feature on the sparse model is connected to the feature that has a closest descriptor on the dense model.

Another pair of unevenly sampled models we experimented with are two horses shown in Figure 3.6. In Figure 3.7, which uses smooth shading, little difference between the sparse and dense models can be seen.

![Figure 3.6: The uneven sampled denser horse model in the right is resampled to an evenly sampled sparser horse model in the left.](image)

We pick a set of feature points in both horse models and compute spin contour as well as face-based spin image using normals and central point directions. Unlike the fandisk case, most points have very different face normals in different horse models. 432 out of 595 points we picked have an angle larger than 0.1 rad between two face normals on two models. If using the third axis of the unambiguous local reference frame [26] as the stabler normal estimations [65], we find that 548 out of 595 points we picked have an angle larger than 0.1 rad between two this kind of normal direction on two models. Here, the local support
radius used to compute this kind of normal is set to 30× mesh resolution of the sparser horse. Therefore, we will not compare to the SHOT descriptor [26] and Improved Spin Image [65] in this set of example. Only 10 points have an angle larger than 0.1 rad between the central point directions we estimate. Results are shown in Figure 3.7.

![Figure 3.7: Feature matching between the sparse and dense horse models. (a) and (c) show matching results of face-based spin image [33]. (b) and (d) show matching results of spin contour. The point directions are set to face normals in (a) and (b), to the new point direction proposed in section 3.2 in (c) and (d). Black lines indicate correct matches, while red lines indicate incorrect matches.](image)

The Recall versus 1-Precision curves shown in Figure 3.8 illustrate the robustness of the spin contour under different sampling rates. Spin contour generates better point matches than the face-based spin image [33] no matter what \( n \) is used.

![Figure 3.8: Recall versus 1-precision curves on sparse and dense Fandisk models (a) and horse models (b).](image)

### 3.7 Noisy models

We add random noise with uniform distribution to the casting model in the range of \((0, 0.3\times \text{mr})\), where \(\text{mr}\) is the mesh resolution, as shown in Figure 3.9. We register the original
mesh to three noisy mesh with different noise levels. We extract random feature points on both meshes and compute different shape descriptors on these features.

In this section, we compare the spin contour with the traditional spin image [66] (SI), the face-based spin image [33] (FSI), the improved spin image [65] (ISI) and SHOT descriptor (SHOT) [26]. The size of all spin images (include spin contour) is $60 \times 120$ and the support length is the size of the bounding box of the model. For the SHOT descriptor, we set the local radius to be 30 times mesh resolution, and the number of local shape bin for each point to be 6. We use the same set of parameters to estimate the unambiguous local reference frame for SHOT descriptor [26] and the improved spin image [65].

![Figure 3.9: Casting model. From left to right: original model, noisy level 0.1mr, 0.2mr and 0.3mr.](image)

The matching results are shown in Table Table 3.1. Here, we connect features on the original mesh to the best match on the noisy mesh according to a particular descriptor. The corresponding 1-precision vs recall curves are shown in Figure 3.10. "fn" means we just use face normals as point directions, while "cd" means using central point directions. Notice that the decrease in recognition ability is quite slow for the spin contour as the noise level increases.

We plot the summed distances between incorrect matched pairs with different noise levels in Figure 3.11, which clearly shows that the spin contour outperforms the face-based spin image with central direction in this case.
Table 3.1: Feature matching between the original and noisy casting models. A black line indicates a correct match, while a red line indicates a wrong match.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1mr</td>
<td><img src="a" alt="Image" /> noise 0.1mr</td>
<td><img src="b" alt="Image" /> noise 0.2mr</td>
<td><img src="c" alt="Image" /> noise 0.3mr</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 3.10: Recall vs 1-Precision curves of the casting matching.

3.8 Model-scene matching

To further test the power of spin contour, we compare its performances on model-scene data sets provided by Dr. Tombari in Computer Vision Lab, University of Bologna. This set of data contains 45 scenes, 6 models and clutter objects, which is publicly available at http://vision.deis.unibo.it/research/78-cvlab/80-shot. All models used in this data set are obtained from the stanford 3D scanning repository, which are also publicly available at https://graphics.stanford.edu/data/3Dscanrep/. Several papers also used this data set for
evaluations [26]. The ground-truth transforms between each models and each scenes are given by the authors of this data set.

We build up our experiments according to the suggestions in [26]. First, we add random noise of amplitudes 10%mr, 20%mr and 30%mr (mr is the mesh resolution) to the scenes, respectively, and randomly sample both the models and the scene with a lot of points on the surfaces. Then, we choose almost 500 pairs of samples from both the model and the scene, the distances between a pair of points after applying the ground-truth transforms are below 10% of the mesh resolution. Finally, we choose approximately 1000 more points are other parts of the scene as clutters.

Since the models are occluded with each other in the scene, using the size of bounding box as the support length is not applicable. In this experiment, we choose 10% of the bounding box size as the support length of spin image, face-based spin image, improved spin image and spin contour, which is comparable to 15× mesh resolution. This value is suggested as a suitable choice of the radius of SHOT descriptor by its author [26].

The matching results of the spin contour can be found in Figure 3.12. Corresponding recall-precision curves are shown in Figure 3.13. One may observe that in this experiment, the performance of face-based spin image built using face normals as the axis and the original

Figure 3.11: Summed distances error curves of the casting matching.
spin image is almost the same. This fact is due to the models are finely sampled that have equal sampling rates everywhere. SHOT descriptor always takes the second place in all three levels of noise contamination, better than traditional spin images, which is consistent with the results reported in [26].

Note that the local version of central point direction is used in this experiment. Here, we choose the radius \( r \) in Equation 3.2 as 10% of the bounding box size. A value between 5% to 15% of the bounding box size does not change the results too much in this set of model-scene matchings.
3.9 Spacetime stereo matching

To further explore the abilities and limitations of spin contours, we compare its performance on a set of spacetime stereo data provided by Dr. Tombari in Computer Vision Lab, University of Bologna. This data set is publicly available at http://vision.deis.unibo.it/research/78-cvlab/80-shot. A typical pair of the model and the scene are shown in Figure 3.14. One may observe that these scans contain many holes that hamper the recognition. Another aspect of this data set is that the surfaces are very smooth and the sampling rate is high. As stated in [26], this data set is better to reveal the descriptiveness of a shape descriptor, because these scans have less noise and fewer salient features.

![Figure 3.14: Spacetime stereo model and scene.](image)

We design two experiments on this data set. In the first experiment, we compare different descriptors on matching original models and scenes. We set the radius of the local reference frame in SHOT descriptor computation as 15 times the mesh resolution and we set the support lengths of various spin images as well as spin contour comparable to this value. Results shown in Figure 3.15a illustrate that SHOT descriptor outperforms spin contour. This is because these models and related scenes have the same sampling rates. Another observation is that the spin contour using face normals as axis is better than the spin contour using central point directions as axis. This is because the data is very smooth.

The second experiment is designed to match a scene to models with different sampling rates. The original Mario model shown in Figure 3.14 has 45205 vertices. We create two other Mario models with different resolution levels: coarser ones with 9977 and 5452 vertices,
respectively. This time, we only compare three different methods: SHOT, face-based spin image and spin contour. To focus on comparing descriptiveness, we use a same set of local reference frame proposed in [26]. The axis used for the face-based spin image and spin contour is set to the axis corresponding to the third eigenvector of the SHOT local reference frame. Results shown in Figure 3.15b show that the spin contour captures is more invariant with respect to the sampling rate than the other descriptors.

3.10 Space and Time analysis

Another merit of the spin contour is the reduction of the matching time. To accelerate the computation of the $L_1$ distance between two point sets, one could build nearest neighbor search data structures [67] for these two point sets. In addition, the number of contour points shown in Figure 3.3e is 1314, whereas the number of image pixels in corresponding spin image shown in Figure 3.3b is $300 \times 600 = 180,000$, which is approximately 150 times larger than the number of contour points.

The time complexity of the crosscorrelation between two images with $n \times n$ pixels is $O(n^2)$, however, the time complexity of computing $L_1$ distances between the corresponding
contours is $O(m \log m)$, where $m$ is the number of contour points. Typically, $m \approx 4n$, so the time complexity of the $L_1$ distance computation is $(n \log n)$ in practice.

### 3.11 Conclusion and future work

The spin contour can be viewed as an information extraction from the face-based spin image. This extraction accelerates both the computational time and the matching time of the face-based spin image. In addition, the spin contour with the central point direction estimation improve the recognition ability for unevenly sampled and noisy models.

Our results show that replacing frame based on normal vector estimate with one based on the central point direction can greatly improve the descriptor quality. The natural local extension of this direction can be successfully used in the cases of cluttered scene or partial data.

Recording the contour of the face-based spin image may not be the only method to improve the recognition ability. Other sophisticated features may be extracted from the spin image or face-based spin image to increase its quality. Finally, it would be interesting to develop a variant of spin image optimized for non-rigid deformations. This could involve developing a more sophisticated metric for comparing spin contours that is invariant under certain types of deformations.
CHAPTER 4
GEODESIC SPIN CONTOUR

This chapter mainly consists of a paper accepted by International Conference of Shape Modeling International 2014 (special issue of Elsevier Journal of Computer & Graphics, accepted for publication).

4.1 Summary

We propose a novel shape descriptor named \textit{geodesic spin contour}. Geodesic spin contour is similar to the classical spin image, but it replaces spin coordinates with quantities invariant under isometric transformations: geodesic distance and average geodesic distance. This makes it applicable to matching near-isometric shapes. By choosing a small support length, one can obtain geodesic spin contours that describe the local geometry of the shape, and therefore are suitable for surfaces with missing data. Experimental results show that the geodesic spin contour provides an effective way to match features on near isometric shapes.

4.2 Introduction

Local shape descriptors are an important component of many computer graphics, computer vision and geometry processing applications [10, 39]. For example, a descriptor can be used to match points on two different shapes [39], and these matches can be used in a more sophisticated registration framework [19, 21]. Descriptors can also be used in shape segmentation. For example, points with similar descriptors can be clustered into the same category [35]. Another potential usage of local shape descriptors is retrieval [35]. A global shape descriptor could be build from local descriptors using the bag-of-features technique. Finally, one can use local descriptors to segment meshes [68] and detect stable regions [69, 70].

A local shape descriptor should ideally be descriptive and robust. Descriptiveness means that points on the shape whose neighborhoods significantly differ should have significantly
different descriptors. Robustness means resistance to data contamination or imperfections, such as noise, inconsistent sampling or missing information. Typically, evaluation of a descriptor involves measurements of both properties [10, 34, 39].

Spin image is a classical local shape descriptor in rigid registration [31, 62, 63, 66]. Given a directed point on the shape, one can build a local spin coordinate system and map other parts of the shape into this space. This descriptor is invariant under rigid transformations such as rotation and translation. However, it is not invariant under nonrigid transformations.

Descriptors based on the spectral decomposition of Laplace-Beltrami operators [34, 35, 39, 72] have been gaining growing attentions in these years. The most notable examples in this descriptor family are Heat Kernel Signature (HKS) [34] and Wave Kernel Signature (WKS) [39]. These descriptors are computationally efficient and are invariant under isometric deformation.
In this paper, we built upon prior work on the spin image descriptor to construct a similar descriptor suitable for near isometric deformations. To achieve our goal, we replace the spin coordinates with isometry-invariant quantities. Also, we use a sampled version of the contour, or the boundary of the image of the surface under our mapping, as the descriptor. We call this descriptor *geodesic spin contour*. Examples of geodesic spin contours for different points on a 3D shape are shown in Figure 4.1. To compare the geodesic spin contours, we use the $L^1$ distance between the sets of contour samples.

The focus of this paper is on partial near-isometric matching. Let us stress that this is different from a full registration framework (as described in, e.g. [11, 19–22, 46, 49, 50, 73]). Matching points on two shapes using the descriptor we developed can be the first step of a fully automatic registration algorithm. Our task is also different from the shape retrieval (as described in [74, 75]). The descriptors proposed for shape retrieval is a single descriptor for a whole shape, however, our descriptor is designed for a particular point on the shape. In [32], the descriptor used for shape retrieval is classified as global shape descriptor and the descriptor used for point-wise registration is classified as local shape descriptor.

Most local near-isometric matching descriptors such as the spectral based methods, e.g. Heat Kernel Signature (HKS) [34] and Wave Kernel Signature (WKS) [39], are vulnerable to missing data. This is because they are based on spectral decomposition of the Laplace-Beltrami operator, which can change considerably when a part of the shape is removed. In contrast, our descriptor is based on geodesic distances. We extend the idea of the spin image, which represents a relation between two quantities, to create a new local descriptor. Instead of using the image, we use the contour of the image as the descriptor.

The structure of the paper is as follows. In Section 2, we briefly review the related prior work. We define our geodesic spin contour descriptor in Section 3. We illustrate the recognition ability of the geodesic spin contour in Section 4. We provide additional insights and discuss potential extensions in Section 5.
4.3 Related work

Many local shape descriptors have been proposed for near-isometric local shape registration. Our descriptor is an intrinsic extension of the spin image, a rigid shape descriptor, to the near-isometric case. In this section, we briefly review the development of spin image and also review efforts on generalizing other rigid descriptors to the near-isometric case.

4.3.1 Prior work on spin image

Our descriptor is motivated by the robust and versatile spin image descriptor, has been successfully applied to rigid shape matching problems with incomplete data or to find 3D models in cluttered scenes [31, 62, 63, 66]. Spin image is originally designed for rigid registration. However, by varying the support length, one can produce a set of spin images at one single point on the shape, encoding information from local to global [63]. The local spin image with relatively small support length can be used for nonrigid registration when the underlying transformation can be approximated by locally rigid motion [11–13].

A spin image [31, 63] is created from an oriented point, a point with a normal vector, which defines an object-centered coordinate system. For a particular oriented point $P$ with unit length normal $n$, the spin image is obtained by first transforming sample points on the surface to the spin image coordinates by applying the mapping $S : R^3 \rightarrow R^2$ defined by

\[
S(Q) = (\sqrt{||Q - P||^2 - (n \cdot (Q - P))^2}, n \cdot (Q - P)).
\] (4.1)

Intuitively, the first spin coordinate is the distance from $Q$ to the line through passing through $P$ and parallel to $n$. The second coordinate is the signed distance from $Q$ to the plane passing through $P$ and perpendicular to $n$. The spin image is obtained by splitting the spin coordinate space into a number of bins (pixels) and making pixel intensities proportional to the number of samples in the bin.

Many improvements of the spin image is proposed to extend its usage. Carmichael et al. [33] present a face-based spin image to alleviate the problem caused by uneven sampling rates of the shape. Multiresolution spin image matching [32] allows a faster way to compare
spin images between different points.

4.3.2 Intrinsic generalization

Many works generalize rigid local descriptors or image descriptors to the nonrigid surface matching case [76–79]. Some of these works uses intrinsic coordinates or intrinsic invariants to capture local surface information [76, 78, 79].

Zaharescu et al. [77] propose a mesh feature detector (MeshDOG) by extending the SIFT-like descriptor to 3D and a mesh feature descriptor (MeshHOG) by computing local shape histogram. Gatzeke et al. [76] create the local curvature map based on geodesic fans. This improvements open the door of using geodesics to build intrinsic local coordinate system. A more recent local shape descriptor proposed by Kokkinos et al. [78] extends the shape context to the surface case. Like MeshHOG [77], this generalization is also a local histogram. However, it does not require any local coordinates; instead, it is built upon a geodesic net around a given point. After binning nearby regions, they integrate intrinsic invariant like HKS [34] or Scale-invariant HKS [35] in each bin to form their descriptor. This descriptor performs well on near-isometric shape registration.

Geodesic spin contour is also an intrinsic generalization of a local extrinsic shape descriptor. This makes it similar to the intrinsic shape context [78]. The major difference between our method and intrinsic shape context is that our method focus on the relationship between two intrinsic values, whereas the intrinsic shape context [78] uses information from only one intrinsic value.

4.4 Geodesic spin contour

In this section, we will introduce how to build geodesic spin contour.

4.4.1 Definition

There are two major issues that prevent the spin image from being applicable to near isometric shape matching:
Figure 4.2: Results for the crane dataset [71]. (a) Geodesic distance from the vertex $v$ shown as the grey ball; (b) average geodesic distance; (c) spin image of $v$; (d) spin contour of $v$; (e) geodesic spin image of $v$ and (f) geodesic spin contour of $v$.

1. the local reference frame (include normal direction) at a point is unreliable under the nonrigid deformation, in addition, it is outright extrinsic: rigid rotations of the object can orient the normal at a point in any desired direction, for instance;

2. the spin coordinates are projections of Euclidean distances, which are not invariant under isometric deformations.

We design a descriptor that is similar to the spin image but uses two different coordinates that are close to invariant under near-isometric deformations. The key idea of spin image is to use an image to encode the relation between two invariants under rigid transformation.
Such encoding increases the descriptiveness of the descriptor compare to only using one invariant. Here, we naturally replace the Euclidean distance with geodesic distance to devise a isometric counterpart of the spin image, which is similar to local curvature map [76] and intrinsic shape context [78]. Since we do not have any local tangent plane to project the geodesic path, we must find another invariant under isometric deformations.

Average geodesic distance at a point is the simplest isometric invariant can be computed based on the precomputed geodesics: just requires an area-weighted averaging of geodesic distances from any other points to a given point in a certain radius. We choose to use it as the other invariants because it provides additional information which indicates the intrinsic location of a point in the shape. This relative location information makes the average geodesic distance be used in several isometric shape registration schemes [20, 49].

As shown in Figure 4.2, to compute the geodesic spin contour for a point on the surface, we first compute the geodesic distance (Figure 4.2a) from this point to every other points of the surface. This geodesic distance function replaces the first spin coordinate, i.e. the second coordinate of $S(Q)$ in Equation 4.1. Then, the average geodesic distance (Figure 4.2b) is computed as the first coordinate of $S(Q)$. To sum up, given a point $P$ on a surface, we define its geodesic coordinates in a way similar to the spin coordinates. For any other points $Q$ on the same surface $S$, a mapping $G : S \rightarrow R^2$ that maps the surface into the space where the geodesic spin contour resides is determined as follows

$$G(Q) = (agd(Q), gd(P, Q)).$$

Here, $agd(Q)$ is the average geodesic distance at $Q$ and $gd(P, Q)$ is the geodesic distance between $P$ and $Q$. For simplicity, we define

$$\begin{align*}
\alpha &= agd(Q), \\
\beta &= gd(P, Q)
\end{align*}$$

as the geodesic spin coordinates, and will use this notation throughout the paper.

The geodesic spin image at a given point $P$ is generated in the same way as the standard spin image, by using the geodesic spin coordinates associated with $P$ (Equation (4.2)) instead.
of the standard spin coordinates (Equation (4.1)). Then, one could simply obtain the contour of this image (Figure 4.2f) as the geodesic spin contour (See Figure 4.2e). The contour is defined as the boundary between pixels with zero values and pixels with non-zero values.

We choose to use the geodesic spin contour instead of directly using the geodesic spin image because the difference between two images is often dominated by the (in many cases, incidental) locations of peak intensities. It turns out that image contours are better invariants in most situations. As shown in Figure 4.3, the geodesic spin images have the brightest peak in different places, which makes the cross-correlation coefficients between these two images only 0.33. In contrast, the geodesic spin contours shown in Figure 4.3d and e are much more similar, their $L_1$ distance (after we normalize the range of spin coordinates to $[0, 1]$) is only 0.006. Note that these two models are not exactly isometric, so the correlation coefficient is not 1 and the $L_1$ distance is not 0. The recent work [80] discusses the same phenomenon in the case of rigid shape matching.

We will introduce a direct and fast implementation of geodesic spin contour in the next section. We find that computing the face-based geodesic spin image is not necessary in computing the geodesic spin contour. We will still loop over the faces of the mesh, but only recording and merging the ranges of the geodesic spin coordinates.

4.4.2 Fast implementation

We first precompute geodesic distance functions from a number of samples on the surface using Surazhsky’s algorithm [81]. By averaging these distance functions, we obtain an approximation of the average geodesic distance.

While computing the geodesic spin contour precisely would be complex, there is a simple way to approximate it using the following algorithm. Given a surface point $P$, we first determine the range of the geodesic spin coordinate $\beta$. Then, we pick a number of equispaced samples in that range. For each sample $\beta$ and triangle $\Delta$ of the mesh, we compute the intersection of $\Delta$ with the geodesic isoline, which is defined as a set of surface points $Q$:

$$gd(P, Q) = \beta$$

and determine the range $I_{\beta,\Delta}$ of $\alpha$ values for the intersection. The range is
empty if the intersection is empty and otherwise is equal to the interval $[l, u]$, with $l$ and $u$ equal to the average geodesic distances at the intersection points of the geodesic isoline.
Figure 4.4: Geodesic spin contour (d) at the position located at the left hand of the Crane data set [71] (a). The final step in obtaining the geodesic spin contour is to connect the discrete points in (b) to form (d). (c) is the intermediate stage of the contour building (after connecting the upper part of the contour). The process of filling the gaps in the contour is illustrated in the bottom row. We represent the contour as a set of samples on a rectangular grid (shown as pixels here).
and triangle Δ. For simplicity, we assume that both the geodesic distance and the average geodesic distance functions are linear inside a mesh triangle.

Since every range is either empty or is a closed interval, the union of all ranges $I_{\beta,\Delta}$ over all triangles Δ can be uniquely expressed as a finite union of disjoint closed intervals. For each endpoint $\alpha$ of one of these intervals, we insert the point $(\alpha, \beta)$ into the geodesic spin contour, as shown in Figure 4.4b. The contour points are then connected to form a number of closed curves (see Figure 4.4d). If $I_\beta$ is the range (union of intervals of $\alpha$) for a given $\beta$ (denoted by black points in a certain row in Figure 4.4e), we add samples in $I_\beta \setminus I_{\beta-d}$ (red points in Figure 4.4f) and $I_\beta \setminus I_{\beta+d}$ (blue points in Figure 4.4g), where $d$ is the spacing between samples along the $\beta$ axis. This closes the geodesic spin contour. In what follows, we will represent the geodesic contour as a set of samples (2D points) generated by the procedure described above.

4.4.3 Local geodesic spin contours

Like spin image, geodesic spin contour can be smoothly transformed from local to global by adjusting the support length $L$. The local version of geodesic spin contour describes the geometry of the surface in the geodesic disk of radius $L$ around $P$. In what follows, $L$ will always be specified as the fraction of the geodesic diameter $D$ of the input surface, i.e. the maximum distance between pairs of points on the surface. As shown in Figure 4.5d, the full geodesic spin contour is obtained by using a support length equal to $D$; while a partial geodesic spin contour for support length $0.3D$ shown in Figure 4.5e is obtained by clipping the full geodesic spin contour.

In order to make the local version of geodesic spin contour depend only on the neighborhood of $P$, we replace the global average geodesic distance with a local version defined as the average of distance functions from points no more than $r$ away from $P$. In practice, we
Figure 4.5: (a) Crane data; (b) average geodesic distance; (c) local average geodesic distance; (d) full geodesic spin contour; (e) partial geodesic spin contour and (f) partial geodesic spin contour (with local agd) at a point on left hand indicated by the gray dot on (a).

Figure 4.6: Two surfaces from the horse gallop dataset with color-coded descriptor distances to the dotted point on the source (right) mesh plotted: (a) Heat Kernel Signature, (b) Wave Kernel Signature and (c) Geodesic spin Contour.
approximate this average with the following formula:

$$\text{agd}(P) = \frac{\sum_{Q: gd(P,Q)<r} A(Q) gd(P,Q)}{\sum_{Q: gd(P,Q)<r} A(Q)},$$

where the summation extends over all vertices $Q$ with geodesic distance to $P$ smaller than $r$ and $A(Q)$ denotes sum of areas of triangles incident to a vertex $Q$. Figure 4.5b and Figure 4.5c show the local geodesic distance plotted on the surface with $r = D$ and $r = 0.3D$, where D is the geodesic diameter of the surface. The geodesic spin contour with support length $L = 0.3D$ and using average geodesic distance with local radius $r = 0.3D$ is shown in Figure 4.5f. In general, $L$ and $r$ are not required to be the same.

Local support length and local average geodesic distance give the geodesic spin contour the ability to match near isometric shapes with partial data missing.

### 4.4.4 Similarity measure for geodesic spin contours

To compare two contour point sets, we use the symmetrized and normalized version of the $L_1$ distance as the similarity metric. For two geodesic spin contours $X$ and $Y$, the metric is defined by

$$L_1(X,Y) = \frac{1}{|X| + |Y|} \left( \sum_{x \in X} d(x,Y) + \sum_{y \in Y} d(y,X) \right),$$

where $d(a,A)$ is the distance between a point $a$ and its closest point in the set $A$. 

---

**Figure 4.7:** Results for surfaces shown in Figure 4.6, with the tail removed from the source surface (always shown on the right): (a) Heat Kernel Signature, (b) Wave Kernel Signature and (c) Geodesic spin Contour.
4.4.5 Robustness

We show the robustness of Geodesic Spin Contour in Figure 4.8. A wolf model has been taken from the TOSCA dataset [82] (http://tosca.cs.technion.ac.il/book/resources_data.html) as the base model. Then we create a denser model by applying one step of the Loop subdivision to this base model (Figure 4.8a) and create a partial model by randomly deleting 60 vertices and their incident triangles in the base model (Figure 4.8b). Finally, we compute the geodesic spin contour at a same point (gray point in Figure 4.8a and b). The resulting contours shown in Figure 4.8 are very similar.

Here, the radius for computing the local average geodesic distance is set to $0.15D$, the support length is $0.4D$, where $D$ is the geodesic diameter of the input surface. In the next section, we will thoroughly compare our descriptor to more state-of-the-art methods.

4.5 Evaluation

In this section, we qualitatively and quantitatively compare the geodesic spin contour with other state-of-the-art descriptors.

4.5.1 Qualitative comparison

We start our discussion with a visual comparison of our approach to the Heat Kernel Signature (HKS) [34] and the Wave Kernel Signature (WKS) [39]. We test these methods on a horse gallop dataset (http://people.csail.mit.edu/sumner/research/deftransfer/data.html). Figure 4.6 shows the comparison for complete source and target frames. Results of matching a full frame with another frame with missing data is shown in Figure 4.7. For HKS and WKS, we use the default parameter values suggested by the authors. To get the best performance of HKS according to [34], we compute 300 eigenvalues $\lambda_{i=1..300}$ and eigenvectors. Then we obtain HKS by uniformly sampling 100 points in the logarithmical scale over time interval $[t_{\min}, t_{\max}]$, where $t_{\min} = 4 \ln 10 / \lambda_{300}$ and $t_{\max} = 4 \ln 10 / \lambda_2$. We also use 300 eigenvalues and eigenvectors to compute WKS. The number of evaluations used in WKS is also 100. The variance of the WKS gaussian is set to 43.
Figure 4.8: Robustness and invariance of Geodesic Spin Contour. Geodesic spin contours at the same location of (c) base model, (d) denser model and (e) partial model.

6. Geodesic spin contours are computed using $L = 0.2D$ and $r = 0.1D$.

Figure 4.6 shows color-coded descriptor distance from the base point located at the gray dot on the right (source) horse to any point of the same surface and the left shape. Colder colors indicate smaller descriptor difference (or points with neighborhoods more similar to the neighborhood of the base point). The base point is a non-salient point (the gray dot in Figure 4.6), which is similar to the choice in [39]. One may observe that although WKS and geodesic spin contour encode different information, the corresponding point on the left shape represents a global minimum of the descriptor difference.
Figure 4.7 shows a similar experiment for the partial matching setting. We cut the tail of the source (right) horse in Figure 4.6, and then choose the same location to recompute the descriptors. This leads to significant degradation of the performance of HKS and WKS. The coldest color areas of the distance map on the target (left) mesh shown in Figure 4.7a and b move to the back of the models and include numerous sample points that do not correspond to the point picked on the source mesh. In this case, both HKS and WKS fail to catch the best corresponding point on the target mesh (the gray ball on the left mesh). However, local geodesic spin contour still captures the correct correspondence. The distance color map does not change too much compared to those in Figure 4.6.

This result shows that the geodesic spin contour with local support is suitable for near-isometric partial matching. Quantitative comparison presented in the next section further supports this finding.

4.5.2 Quantitative comparison

We quantitatively compare geodesic spin contour with HKS [34], WKS [39] and spin image [31]. We choose to compare to spin image is because: 1) our method is developed from spin image and 2) spin image with a small support length is still used in nonrigid matching [11]. We use as same parameter settings as the previous experiment in computing HKS, WKS and geodesic spin contour. For spin image, we set the image size to 120 × 120, the support length to 0.2× the diagonal length of the mesh bounding box.

We measure two quantities to compare the capabilities of these descriptors:

1. the hit rate proposed by Aubry et al. [39];
2. the average geodesic errors between the observed matches and the ground truth.

We compute these two quantities in the following steps:

1. Pick vertices on the source mesh by means of furthest point sampling [39];
2. Compute descriptors for these features and for every vertex on the target mesh;
3. Find $k$ best matches on the target mesh for the points selected on the source mesh, where $k \in [1, \frac{N}{100}]$, and $N$ is the number of vertices of the target mesh.

4. Compute hit rates (a hit means that for a vertex $p$ on the source mesh, its ground truth match is present among the $k$ best matches determined by our algorithm) and average geodesic errors (i.e. the average geodesic distance to the true match over all points selected on the source surface and over their $k$ best matches). Both metrics are going to be shown as functions of $100 \times k/N$, i.e. percentage of the total vertex number that can be used as matches for each of the selected points on the source mesh.

Here, a “hit” means the ground-truth match could be found by using a particular shape descriptor.

We choose the same set of meshes shown in Figure 4.6 and pick 50 vertices on the source mesh, see gray dots in Figure 4.9.

We connect the vertices, on the source (right) mesh, that could not be matched to their ground-truth targets to their three best matches in the target mesh, as shown in Figure 4.9. The corresponding hit rate is shown in Figure 4.10a. One may observe that the hit rate of geodesic spin contour is relatively lower than WKS. However, at the points where geodesic spin contour could not find the correct match, the “incorrect” matches found via matching geodesic spin contour are not so far from the ground truth, as shown in areas around the head of the horse in Figure 4.9d. The average geodesic error plotted in Figure 4.10b clearly supports this statement.

To test the capability of partial data matching, we cut the tail of the source horse and match its sampled vertices to a full horse mesh, as shown in Figure 4.11. Here, one may observe that the cutting greatly degrades the performance of WKS. Some vertices on the body of the source horse are matched to the head of the target horse.

Since we use local average geodesic distance and local support length to obtain the geodesic spin contour, the influence of the tail cutting is limited. However, missing data degrades the performances of both WKS and HKS, according to both the hit rate (Figure 4.12a) and the
average geodesic error (Figure 4.12b).

We do the same experiment on crane dataset [71]. We first choose two distinct frames and cut the left hand and the right foot of the human of the source (right) mesh, as shown in Figure 4.13a. Then, we pick 50 vertices on the source mesh using the furthest point sampling. Finally, we perform the matching by computing descriptor distances. We keep using the same set of parameters for this dataset. Both the hit rate curves and the average geodesic error curves shown in Figure 4.13 testify the effectiveness of geodesic spin contour on this dataset.
We have shown the invariance of geodesic spin contour under local topological changes using the TOSCA [82] wolf data set in the previous section. Here, we describe a more in-depth comparison further comparisons using the same model. To test the descriptiveness of a descriptor under local geometric and topological contaminations, we make a denser version of wolf model0 and pick 50 features using furthest sampling on this denser model. Then we find the best matches of these features on wolf model1.

All through this set of comparisons, the parameter used for HKS and WKS is as same as the experiments before. The support length of the geodesic spin contour is \(0.4D\) and the radius of averaging geodesic distances is \(0.15D\), where \(D\) is the geometric diameter.

The denser mesh in Figure 4.14a is created by applying one-step loop subdivision on the original wolf model0, the coarser mesh is wolf model1. Geodesic Spin Contour beats HKS and WKS in both the hit rate (Figure 4.14b) and the average geodesic error (Figure 4.14b).

Besides the sampling rate variation, we make another comparison on matching mesh with local data contamination to a full mesh. We randomly remove 20, 40 and 60 vertices as well as their incident triangles from wolf model0, as shown in Figure 4.15. Geodesic spin contour has slightly lower hit rates compare to WKS in matching full model0 to model1, but much
better average geodesic errors. As the amount of missing data increases, the performance of both WKS and HKS deteriorates, while the performance of the geodesic spin contour remains almost the same. To better illustrate how the data removal affects the performance of the descriptors, we plot all the hit rate curves and average geodesic curves with different amounts of missing data in Figure 4.16. Geodesic spin contour shows the highest stability among three descriptors.

The TOSCA [82] centaur data set consists of six models (centaur0-5) that are close to isometric deformations of each other. As shown in Figure 4.18 to Figure 4.22, we match model 0 with any other models in this dataset. To evaluate partial matching performance,
Figure 4.12: The hit rate (a) and average geodesic error (b) for partial mesh matching shown in Figure 4.11.

Figure 4.13: The hit rate (b) and average geodesic error (c) for partial mesh matching (a).

Figure 4.14: TOSCA wolf model dense to coarse matching (a) with corresponding hit rates (b) and average geodesic errors (c).
we cut the right hand of model 0 and match this partial model to other models. In this experiment, we also pick 50 vertices on the source mesh using the furthest point sampling and then perform the matching by computing descriptor distances. For computing geodesic spin contours, we use $0.2D$ as the support length and $0.15D$ as the radius of computing local average geodesic distance. We build intrinsic shape context (ISC) on Scale-Invariant HKS
Figure 4.16: Hit rates (a) and average geodesic errors of TOSCA wolf models. The number 20, 40, 60 denote the number of missing vertices.

[35]. The time interval of SI-HKS here is [1, 20], the frequency interval is [2, 20]. The radius of ISC is 20, and the bin size is 5 rings × 16 directions.

We plot the average results of the matchings shown in Figure 4.18 to Figure 4.22 in Figure 4.17. Here, one can observe that geodesic spin contour can match the hit rate of WKS and outperform it for the geodesic error measure in the partial matching scenario.

Figure 4.17: The average hit rates (a) and average average geodesic errors (b) of TOSCA centaur model matchings.
Figure 4.18: TOSCA centaur model matchings (column 1) with corresponding hit rates (column 2) and average geodesic errors (column 3).

Figure 4.19: TOSCA centaur model matchings (column 1) with corresponding hit rates (column 2) and average geodesic errors (column 3).

From the results shown in Figure 4.18, Figure 4.19, Figure 4.20, Figure 4.21 and Figure 4.22, one could observe that

- According to the hit rates, geodesic spin contour outperforms ISC on all full and partial model matchings; it outperforms WKS on some full model matchings (model 1 and
model 4) and almost all partial model matchings (except model 5).

- According to the average geodesic errors, geodesic spin contour outperforms WKS on almost all models, including both full model matchings and partial model matchings; ISC performs the best among three descriptors on both full and partial matching of
Figure 4.22: TOSCA centaur model matchings (column 1) with corresponding hit rates (column 2) and average geodesic errors (column 3).

- Cutting the left hand of model 0 largely degrades the matching results of WKS, however, it does not degrade the matching results of geodesic spin contour and ISC too much.

4.6 Conclusion and future work

This paper presents the geodesic spin contour descriptor for matching isometric shapes. This local shape descriptor is constructed by replacing spin coordinates used in spin image [63] with two isometry-invariant quantities. Geodesic spin contour uses the relationship between these quantities to extract the local and global information for a point on a shape. In this paper, we choose geodesic distances and their local averages as these quantities, which is the most straightforward choice. This choice is clearly not the only choice and may even not be the best choice in near-isometric matching. However, the good performance of Geodesic spin contour in nonrigid matching opens the door to improvements based on replacing spin coordinates with other suitable invariants for some specific nonrigid deformations.

At this point, little work has been done on theoretical evaluation of different descriptors: virtually all research on the related topics is driven by experiments. Development of rigorous
theoretical analysis framework for comparing local 3D shape descriptors is an important
topic for future research.

We choose the contour of the image rather than the image itself as the descriptor because
spin contour is more robust and accurate than spin image. In addition, computing $L_1$
distances between two contours is faster and also consumes less memory than matching two
images via cross-correlation.

4.6.1 Parameters

A number of parameters need to be set to generate the geodesic spin contour: image
resolution, support length and the radius for the local geodesic distance computation. The
parameter values influence the matching performance. This is a common problem with all
spin image based descriptors [32, 33, 62] and also to the local intrinsic coordinate based
methods [76, 78]. For partial matching, the most important parameter is the support length,
which is specified as a fraction to of the geometric diameter $D$ all through this paper. Users
could also set it to a fixed value, like the radius setting in intrinsic shape context [78]. For
example, for the TOSCA [82] centaur model, where we set the support length to $0.2D$, the
support length is approximately 72. Intuitively, when the missing part of a mesh is large and
focused in a particular local area (e.g. cut a hand of centaur), the support length should be
smaller. When the missing part is small and is spread to all parts of the shape, the support
length should be larger (see TOSCA [82] wolf comparison). The radius for local average
geodesic distances can be picked based on the same considerations.

4.6.2 Limitations

Geodesic spin contour has several limitations. First, it requires more expensive com-
putation compare to both HKS and WKS. In practice, we first compute geodesic distance
between all pairs of vertices in a mesh. This step is the bottleneck of our method. We run
the code on a MacBook pro with a 2.9GHz Intel Core i7 and 8G 1600MHz DDR3 memory.
All methods are implemented with single threading using c++. For the TOSCA wolf model,
which has 4344 vertices and 8684 faces, it uses approximately 15 minutes to compute all the geodesic distances. Parallel implementation of the geodesic distances may alleviate this problem. The computational time of each geodesic spin contour is negligible compare to the geodesic distance computational time. Still for the TOSCA wolf model, it takes only 0.674ms in average to obtain a contour with support length $L = 0.4D$ (approximately 71.60), radius of computing $AGD = 0.15D$ and image resolution is $120 \times 120$.

HKS and WKS are considerably faster than geodesic spin contour. Full computation time of HKS and WKS on one TOSCA wolf model are 12.52 seconds and 5.73 seconds, respectively, which means HKS and WKS only spend 2.9ms and 1.3ms on one vertex. Intrinsic shape context [78] with radius= 20 and bin= $5 \times 16 = 80$ uses approximately 15 seconds on preprocessing (includes computing signatures and average triangle areas) and 20 minutes in computing the descriptors on all vertices, which means intrinsic shape context spends approximately 46.1ms on one vertex. Thus its total computational complexity is comparable to our descriptor.

Another limitation of the geodesic spin contour is the input data restriction. The geodesic distance is invariant under isometric deformation. However, in practice, one is interested in nearly-isometric rather than perfectly isometric deformations. Two corresponding point pairs on two meshes related by near-isometric deformation may have different geodesic distances and different geodesic shortest paths. In practice, on a triangular mesh, the further the points, the more unreliable the geodesic distance is. Using only the local support may alleviate this problem. Another way to overcome this problem is replacing the geodesic distance with a stabler distance, for example, interior distance [83], inner distance [84, 85], or spectral distance [86].

Geodesic spin contour can be used for matching surfaces deformed so that intrinsically distant parts are fused or nearby parts are separated (e.g. model of a human deformed so that their arms touch). However, such deformations can substantially change the geodesics near the fused or separated parts and therefore geodesic spin contour is not reliable in these
areas. The same is true if large parts of the model are cut: the performance of geodesic contours near the cut boundaries can be low. However, geodesic contour works well when a number of small patches are removed from the model, since this does not substantially affect the geodesic distances (Figure 4.15).
CHAPTER 5
FULLY AUTOMATIC ISOMETRIC SHAPE REGISTRATION

In this chapter, we discuss a fully automatic, optimization based nonrigid shape registration scheme. The underlying deformations between the shapes are assumed to be close to isometric.

5.1 Problem formulation

When the underlying deformation between a pair of shapes \( s \) and \( t \) is isometric, the correspondence problem could be stated as the following optimization problem

\[
\begin{align*}
\text{minimize} & \quad \sum_{(s_i, t_k) \in m, (s_j, t_l) \in m} |g(s_i, s_j) - g(t_k, t_l)|, \\
& = \sum_{i,j,k,l} |g(s_i, s_j) - g(t_k, t_l)| m(i, k) m(j, l), \\
& m(i, k) = \begin{cases} 
0 & \text{if } (i, k) \text{ is a valid correspondence} \\
1 & \text{if } (i, k) \text{ is not a valid correspondence}
\end{cases} \\
\text{subject to} & \quad \sum_{i} m(i, k) = 1, \forall k \\
& \quad \sum_{k} m(i, k) \leq 1, \forall i \\
& \quad m(i, k) = 0 \text{ or } 1.
\end{align*}
\] (5.1)

where \( m \) is an undetermined isometric map, \( s_i \) and \( t_k \) are feature points selected on shapes \( s \) and \( t \), respectively and \( g(a, b) \) is the geodesic distance between point pair \( a \) and \( b \). Roughly speaking, the constraints on \( m(i, k) \) ensure that \( m \) defines a one-to-one mapping between the points on the two shapes, with \( s_i \) corresponding to \( t_k \) if and only if \( m(i, k) = 1 \). The objective function measures the deviation of the correspondence map from isometry.

This problem is known as a quadratic assignment problem [21, 87], and is hard to solve because of both the non-convexity and the binary constraints on the variables.
5.2 Methodology

In practice, the input surfaces may contain tens of thousands of vertices. We solve the matching problem in a coarse-to-fine manner. We first sample the mesh and find correspondences only for salient feature points. Optionally, this step can be left to the user since it involves finding only a small number of correspondences. We then propagate the correspondences to other vertices to reduce complexity.

5.2.1 Furthest point sampling

Given a triangular surface, we pick the salient features using furthest point sampling. We pick \( n \) features using the procedure as follows:

1. arbitrarily pick a vertex on the mesh and add this vertex to the vertex list;
2. find the furthest vertex to the current vertex list and add this vertex to the list;
3. if the size of the list is less than \( n + 1 \), go to step 2;
4. else remove the first vertex in the list, because this vertex is arbitrarily picked.

Figure 5.1: Furthest sampling on the TOSCA wolf model: 0 and 2.

Figure 5.1 illustrates 10 features sampled on two TOSCA wolf models. Note that in general, there may be no natural correspondence between these features. For example, features in the red circles have no corresponding points and, ideally, should be discarded.
as outliers. Features in the blue circles are not located in the same relative positions in two meshes. Furthest point sampling is a simple but effective method to pick a number of landmark features on similar models.

5.2.2 Initial correspondence trim

After picking features from two input meshes, we further reduce the search space by comparing the descriptor distances between feature points.

As shown in Figure 5.2, we pre-select a set of potential correspondences according to the $L_1$ distances between Geodesic Spin Contour of features. If the distance is larger then a threshold, we do not insert this correspondence into the set. The output correspondence will be chosen from this set of correspondences. Clearly, care needs to be taken when selecting the threshold. For example, threshold that is too small may lead to an infeasible optimization problem later on in our procedure.

5.2.3 Optimization on the coarse level

We solve the minimization problem in Equation 5.1 by introducing a 4d map $M$ and reformulate it into
minimize $\sum_{i,j,k,l} |g(s_i, s_j) - g(t_k, t_l)| M(i, j, k, l)$,

subject to $M(i, j, k, l) \geq m(i, k) + m(j, l) - 1$,
$M(i, j, k, l) \geq 0$
$\sum_i m(i, k) \leq 1$,
$\sum_k m(i, k) \leq 1$,
$\sum_i \sum_k m(i, k) \geq \hat{n}$,
$m(i, k) = 0$ if $(s_i, t_k)$ is not in the preselect match set,
$m(i, k) = 0$ or $1$ if $(s_i, t_k)$ is in the preselect match set.

where $\hat{n}$ is the number of valid correspondences I plan to find. In our experiments, we set $\hat{n} = \frac{4}{5} \min(|s|, |t|)$, where $|s|$ and $|t|$ are the number of features picked in meshes $s$ and $t$ (respectively). Basically, this means that 20% of features in the smaller mesh are outliers. Just as in the previous formulation, $m$ represents a mapping between features on the two meshes. Because of the constraints (and the fact that the coefficients of the objective function are positive) $M(i, j, k, l)$ is set to zero if pairs of indices $(i, k)$ and $(j, l)$ represent corresponding features on meshes $s$ and $t$. The set $M$ can represent any set of

A coarse matching obtained using this method is shown in Figure 5.3. Notice that the two outliers in Figure 5.1 (in red circles) have been correctly singled out.

Figure 5.3: Initial coarse match result on the TOSCA wolf model: 0 and 2.
5.2.4  Local optimization of the correspondences

As the selected features may not appear at exactly the corresponding positions in both the shapes, we locally optimize the matchings using a gradient descent based process described in this section.

For example, in Figure 5.3, the location of the two features at the wolf nose is certainly suboptimal. This is because the initial features are \textit{not} in the exact same positions.

The matchings are optimized in a quite straightforward way. The energy function we are going to minimize is

\[
E = \sum_{i,j,k,l} (g(s_i, s_j) - g(t_k, t_l))^2, \tag{5.3}
\]

where \(s_i\) and \(t_k\), \(s_j\) and \(t_l\) are two corresponding feature pairs. We move the feature vertex along the direction of the gradient of \(E\) in small step on the mesh (in practice we choose this step as 2 times the edge length). When we move the features on the source mesh, we keep the target mesh features at the same places, and vice versa.

Therefore, when we move source features, positions of \(s_i\) and \(s_j\) are unknowns; when we move target features, positions of \(t_i\) and \(t_j\) are unknowns.

The gradient at point \(s_i\) can be computed as

\[
\nabla E(s_i) = 2 \nabla g(s_i, \cdot)(g(s_i, s_j) - g(t_k, t_l))
\]

Here, \(\nabla g(s_i, \cdot)\) is the gradient of the geodesic distance function from \(s_i\) on mesh S.

The refined matching is shown in Figure 5.4. Here, we can see that the features on the nose appear at a more consistent locations.

5.2.5  Refinement

At this point, we can use the optimization framework to obtain a matching between coarse features on the two meshes. We gradually add new vertices to the feature list and compute the new feature correspondences. The new correspondences at a finer level are computed
Figure 5.4: Optimized coarse matching on the TOSCA wolf model: 0 and 2.

Based on the original coarser correspondences by solving the following optimization problem:

\[
\begin{align*}
\text{minimize} & \quad \sum_{i,k} |g(s_i, s_j) - g(t_k, t_l)|m(i,k), \\
\text{subject to} & \quad \sum_i m(i,k) \leq 1, \\
& \quad \sum_k m(i,k) \leq 1, \\
& \quad \sum_i \sum_k m(i,k) \geq \hat{n}, \\
& \quad m(i,k) = 0 \text{ or } 1.
\end{align*}
\]

(5.4)

In the above problem, \( m \) represents the unknown matchings between the finer features (just as in the problems used previously, \( m(i,k) = 1 \) if features with indices \( i \) and \( k \) match) and \( (s_j, t_l)s \) are the known matchings in the coarser level. \( \hat{n} \) is the desired number of inlayers still set to \( \frac{4}{3}n \).

After obtaining the result based on Equation 5.4, we also optimize the denser matches using the local gradient descent.

Figure 5.5 shows the coarse-to-fine matching between two TOSCA wolf model. Figure 5.5a, c, e show the matchings before local refinement; Figure 5.5b, d, f show the matchings before local refinement. Here, in each level, 20 new matches are determined based on the old matches. The black lines are correspondences found in the initial coarse matching. The red, green, blue lines are new matches added in level 1, 2 and 3.
Figure 5.5: Coarse-to-fine match of the TOSCA wolf model: 0 and 2.

5.3 Evaluation

We evaluate this method in aspects of both the computational time and accuracy. Geodesic distances between any two vertices of the mesh are pre-computed.
5.3.1 Time

We use IBM Cplex (http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/) to solve all optimization problems. To obtain the final matching we need to:

- find the coarse level matching
- reﬁnemene the coarse matching
- locally optimize of the correspondences.

In most cases, the number of salient features of a shape is less than 10. We choose to preselect 10 features using the furthest sampling. For the wolf models shown in the last section, the coarse level matching time of 10 features is 0.374 seconds. If we increase the number of initial features to 15, the matching process takes 12.31 seconds.

Compared to the initial matching stage, both the reﬁnement stage and the the local optimization take negligible time. The local optimization of 100 matches takes less than 0.001 seconds. When we choose 10 as the initial feature number, the time consumed in the reﬁnement stage is shown in Figure 5.6.

Figure 5.6: The time consumed in adding different number of new matches.
5.3.2 Accuracy

The accuracy is evaluated by the average geodesic error to the ground truth of the found matchings. We plot the average geodesic errors at each stage of the matching in Figure 5.7 for the TOSCA wolf model shown in Figure 5.5.

![Geodesic error between the founded match and the ground truth.](image)

**Figure 5.7:** Geodesic error between the founded match and the ground truth.

As shown in Figure 5.7, the colors are chosen as same as the Figure 5.5. The geodesic errors remain more or less the same when the number of found matches increase. The x-axis of this plot is the number of iterations of the local matching optimization. Iteration 0 represent the state that no local optimization is performed. One can observe that it generally reduces the geodesic errors. Note that some curves stop earlier than others (e.g. the red curve stops at iteration 3), because the local optimization converges after 3 iterations.
CHAPTER 6
NONRIGID ITERATIVE CLOSEST POINTS

Part of the content of this chapter has been published on a short paper in Computer Graphics International 2011 [8].

6.1 Summary

We introduce an ICP-like nonrigid surface registration algorithm based on multiple correspondences. Our method, called the Nonrigid Iterative Closest PointS (NICPS), can be applied to surfaces of arbitrary topology. It does not impose any restrictions on the deformation, e.g. rigidity or articulation. Finally, it does not require parametrization of the input meshes.

Our method is based on an objective function that combines distance and regularization terms. Unlike for the standard ICP, the distance term is determined based on multiple two-way correspondences rather than single one-way correspondences between surfaces. A Laplacian-based regularization term is proposed to take full advantage of multiple two-way correspondences. This term regularizes the surface movement by enforcing vertices to move coherently with their 1-ring neighbors.

The proposed method works well when no global pose differences or significant amount of bending exists in the models, for example, families of similar shapes.

6.2 Assumptions and Limitations

We assume that the source (model) and the target (data) are provided in a similar pose. This enables us to use surface normals to trim false correspondences. Both meshes should represent objects of same kind, like vertebrae of different people. For such input surfaces, our method can work better than other surface registration methods. However, our method is not suitable to recover large articulated motions because we do not impose any particular type
of restrictions on the underlying deformations. Also, it is designed to work with complete surfaces, with no missing information.

The input data for our methods are surface models of human femurs and vertebrae. They are extracted from 3D images captured from CT-scans. The vertebrae surfaces

- are watertight, have no boundary;
- are of genus 1;
- are not related by deformations that can be described as articulated;
- have varying sampling rates over the space, e.g., the template mesh has over 7700 vertices while some specimens only have 4000 vertices; moreover, the surfaces typically have more samples around intrusions than flat parts;
- are similar in shape and pose;
- do not have consistent features that can be reliably determined by an automated algorithm.

Because of these characteristics, many existing methods cannot be directly used on our models.

![Figure 6.1: The source and target surfaces.](a) source (b) source (c) target)
6.3 Overall procedure

In what follows, by $S$ and $T$ we denote the source and target surfaces. In our implementation, both $S$ and $T$ are triangle meshes of generally different connectivity. Our algorithm constructs a sequence of meshes $R_0 = S, R_1, R_2, \ldots, R_n$, of identical connectivities, that tend to get closer to surface $T$. Meshes $R_0, R_1, R_2, \ldots, R_n$ have the same connectivity as the source mesh $S$. In order to obtain $R_{f+1}$ from $R_f$, we solve an optimization problem whose objective function combines the distance term and the energy term. The distance term ensures that the surfaces represent increasingly better approximation of $T$. The energy term attempts to preserve the quality of the mesh. The algorithm terminates when it reaches the maximum number of iterations or $\text{dist}(R_n, T) < \epsilon$, where $\epsilon$ is a user-specified threshold.

6.4 Distance energy

In order to obtain the distance term, we first determine a set of pairs of corresponding points $(p, t_p)$, with $p \in R_f$ and $t_p \in T$.

The distance term is the sum of squared distances between points in all correspondence pairs, i.e.

$$D = \sum ||p - t_p||^2$$  \hspace{1cm} (6.1)

Note that neither $p$ nor $t_p$ are confined to a vertex of the mesh. Using surface-to-surface distance gives better result than using vertex-to-vertex distance.

6.5 Potential energy

A deformation model, that restricts the movement of mesh vertices of the deformed mesh relative to each other, is usually required in the nonrigid registration procedure to prevent the mesh quality from deteriorating. This movement control can be imposed explicitly or implicitly.

An example of an explicit scheme is described in [88] and [89]. They use geometry-constrained diffusion to convolve the deformation field. Then the convolved displacement
field is used to warp the current surface to get a new surface.

In [55, 90–92], implicit deformation models are proposed based on differential geometry. In these methods, the surface is required to be parameterized before one can calculate the Gaussian curvature [90], Gaussian radial basis function [91] or unit normal [55].

These characteristics are used to define the objective function that controls the deformation. However, the requirement to parameterize the input surfaces restricts these technique to genus-0 surface setting. For surfaces of higher genus, parameterization would require cutting the input surfaces in a consistent manner, which is hard to do.

### 6.6 Combinatorial Laplacian

We choose an implicit regularization scheme, which is similar to those described in [1, 16, 93]. We use the following potential energy based on the combinatorial Laplacian operator:

\[
G(x) = \sum_i ||\Delta_i x^{f+1} - \Delta_i x^0||^2,
\]  

(6.2)

where \(x^k\) is the vector obtained by concatenating coordinates of all vertices of \(R_k\), \(x^k_i\) represents the 3D coordinates of the \(i\)-th vertex of \(R_k\) and \(\Delta_i x\) is the sum of vectors running from vertex \(i\) to all of its neighbors along the mesh edges.

In practice, the deformation regularized by using this energy term is different than the deformation obtained using the edge based terms described in [1, 16, 93]:

\[
G'(x) = \sum_{(i,j) \in E} ||L_{ij} x^{f+1} - L_{ij} x^0||^2,
\]

(6.3)

where \(E\) consists of all pairs \((i, j)\) such that the mesh contains an edge connecting \(i\)-th and \(j\)-th vertex and \(L_{ij} x = (x_j - x_i)\) is the vector running along the edge from vertex \(i\) to vertex \(j\).

The intuitive motivation of substituting Equation (6.3) with Equation (6.2) as the regularization of the deformation is that we found in biomedical models, saddle points in one mesh often correspond to convex points in another mesh, as shown in Figure 6.10. This
observation is based on experiments. Such shallow saddle points appear in the model only because of the acquisition error. The traditional potential energy described in Equation (6.3) would attempt to preserve these saddles. From the comparisons that will be shown in following sections, one can see that our energy term allows the mesh to deform more freely. Especially when cooperating with the multiple two-way correspondence search, our energy term will achieve a better result. In what follows, we call our potential energy term (Equation (6.2)) the Laplacian-based energy term, and the term given in Equation (6.3) – the edge-based energy term.

Note that $x^{t+1}$ is unknown, and $x^0$ is known since it represents the source mesh.

6.6.1 Geometry-based Laplacian

The combinatorial Laplacian operator $\Delta_i x$ described in Equation (6.2) does not take the geometry into account. In some cases, especially if the surface sampling is far from regular, it should be replaced with a geometry-aware discretization of the Laplacian. A number of such discretizations are described in [94]. In this work, we use the formulation due to Desbrun et al’s [95], based on the following formula:

$$\Delta_i x = \frac{3}{A_i} \sum_{j \in N_i} \left( \cot \alpha_{ij} + \cot \beta_{ij} \right) \frac{1}{2} (x_j - x_i),$$  \hspace{1cm} (6.4)

where $A_i$ is the total area of all triangles incident to $x_i$, $N_i$ is the set of indices of 1-ring neighbor vertices of $x_i$ and $\alpha_{ij}$ and $\beta_{ij}$ are triangle angles shown in Figure 6.2.

6.7 Energy minimization

The total energy to be minimized is a weighted sum of the distance energy and the potential energy. The distance energy term as well as the potential energy term are quadratic functions, therefore the objective function is quadratic:

$$E = D + \alpha G = x^T A x - 2b^T x + c, \hspace{1cm} (6.5)$$

where $A$, $b$ and $c$ represent its quadratic, linear and constant terms. This objective function $E$ is minimized for $x = A^{-1}b$. This is a sparse linear system that can be solved using the
standard conjugate gradient method. The potential energy term can be further explicitly expressed in terms of the Laplacian:

\[
G = \|\Delta_i(x^f - x^0)\|^2 \\
= |\Delta_i(x^f - x^0)|^T \Delta_i(x^f - x^0) \\
= (x^f - x^0)^T \Delta_i^T \Delta_i (x^f - x^0),
\]

(6.6)

where \(\Delta_i\) is the geometry-based Laplacian operator defined in Equation 6.4, \(x^f\) and \(x^0\) are vertex coordinates in the current mesh and the source mesh, respectively. The quadratic and linear term contributed by \(G\) are \(\Delta_i^T \Delta_i\) and \(2\Delta_i^T \Delta_i x^0\).

The parameter \(\alpha\) is adjusted adaptively during registration. For small values of \(f\), \(R_f\) is far away from \(T\). Thus, \(\alpha\) must be relatively larger to protect the mesh quality under a potentially large deformation. When the deformed surface becomes close to the target, we use a smaller value of \(\alpha\). In practice, we set

\[
\alpha = 100/1.1^f.
\]

(6.7)

Recall that \(f\) is the iteration number. The results are similar if 1.1 in the denominator is replaced by a similar value.

### 6.8 Correspondence search

Possible correspondence pairs are determined at the beginning of each fitting step between current surface \(R_f\) and target surface \(T\). The simplest, but often unreliable way of finding
correspondences between the two surfaces is based on proximity. To increase the reliability, one can use 2-way correspondences [19, 96, 97] or incorporate local normals [1, 55, 88, 93, 98]. By “2-way”, we mean that the correspondences are searched for both from the source to the target (forward correspondence) and from the target to the source (inverse correspondence).

These improvements of the simplest proximity-based scheme are often insufficient when some parts of the source surface are far from the corresponding parts of the target surface: the correspondences may be still unreliable. Filtering correspondences based on normal vector information (i.e. disregarding them if the corresponding points’ normals significantly differ) can be an effective heuristic to remove false correspondences. However, it can potentially lead to problems shown in Figure 6.3(a), where some vertices in the source (blue) mesh are have no correspondences in the target (pink) mesh that pass the normal test. This leads the fitting procedure to a poor result (Figure 6.3(c) and (d)).

The problem is inherited from the basic ICP algorithm. In some cases, nearest correspondences are not correct correspondences. To alleviate this problem, we improve the searching procedure by using more correspondence pairs in the low confidence regions, where the source and the target meshes are far away.

In both forward and inverse correspondence search, instead of finding just one nearest vertex with similar normal information, we search for multiple possible correspondences. $k$ possible correspondences on a mesh $B$ for one vertex $a$ of a mesh $A$ with normal $n_a$ are found as follows.

- **Step 1**: Search for $k$ vertices of $B$ closest to $a$. Among these $k$ vertices, find the one ($b_0$) closest to $a$ that has a similar normal (i.e. such that the angle between its normal and $n_a$ is less than 90). If no such vertex exists, $a$ is regarded as an outlier in the forward correspondence search. Otherwise, let $d$ be the distance between $a$ and $b_0$.

- **Step 2**: Find all vertices of $B$ whose distance to $a$ is smaller than $2d$. For any such vertex $b$ that has normal similar to $n_a$, search the incident triangles to find the closest point $t_a$ to $a$. Use all resulting pairs $(a, t_a)$ as correspondence pairs.
Note that:

- This process is used both to find forward and backward correspondence pairs.

- The factor of $2d$ in step 2 is application dependent. For our data, choosing a value between $2d$ and $3d$ will give similar results.

Intuitively, vertices in a region on the current surface that is far away from the target have more correspondences. Since all of them contribute to the distance term, the deformation field is essentially smoothed in such areas, which is highly desirable since it helps to preserve the mesh quality. When two surfaces become closer to each other, the number of possible correspondences is decreasing and smoothing does not take place.

We show the effect of this scheme in Figure 6.3. Here, “2-way+Normal” stands for using both the 2-way one-to-one correspondence finding and the normal based correspondence trimming. Fitting results are shown in (c) and (d). “2-step multiple” stands for using the two-step multiple correspondence selecting scheme, described in this section. Results are shown in (f) and (g).

In our procedure, we use $k = \frac{50}{1.5^f} + 1$, where $f$ is the number of iterations. When two surfaces are getting closer, we can use fewer correspondences.

6.9 Evaluation

Here, we use the same source and target meshes shown in Figure 6.3 to evaluate the quality of results obtained using different methods. The source mesh is a template mesh created by biomechanical engineers, target meshes are reconstructed from CT-scans of patients.

6.9.1 Comparison to single correspondence schemes

We compare the registration results of our method with methods incorporating single correspondence search in Figure 6.4.

**Andresen2001**: The method presented by Andresen and Nielson [89], where they use their method in mandibular registration [88]. Their method is based on displacement diffusion,
(a) overlapped view and correspondences found by just using the 2-way correspondence search and normal selection in the part where large deformations exist. The 2-way+normal based procedure fails to produce a high quality deformation in the boxed area.

(b) 2-way+Normal
(c) 20 iterations

(d) 2-step Multiple
(e) 20 iterations

Figure 6.3: The effect of two-step multiple correspondence search. In (b) and (e), the black lines and the red lines represent the forward correspondences and the inverse correspondences. The source and the target surfaces are shown in Figure 6.1.
which is an explicit regularization. Displacements between source and target are estimated explicitly by using nearest point search, and then a convolved displacement field is used to update the mesh. In [89], authors state that one does not have to use the crest line, which is used in [88] and [99], a geometrical feature for the initial registration. Therefore, we here just use displacement vectors between nearest point pairs from source to target as the initial displacement field. The time-step of the diffusion stage is chosen as 0.1. For simple and models like mandible and cubes shown in their paper, their method is efficient and accurate, especially when the source and target meshes are similar. However, for complex models like vertebrae, their method fails to give a satisfiable outcome. The reason is that the directly found displacements before diffusion step are often far from reliable. Even after the diffusion step, the displacements are mostly wrong. For complex meshes, like vertebrae shown in this paper or skull models shown in [99], interactively chosen feature lines may greatly improve the registration result.

**Pauly2005:** The method presented by Pauly et al, [1], where a similar framework is used. It is also an energy minimization based scheme. The differences between their method and the proposed method include:

- They use single correspondence rather than two-way multiple correspondences.
- Their distance energy term is weighted with a confidence value, determined by bidirectional closest point search.
- Their potential energy term is area-weighted edge-based energy term.
“Bidirectional closest point search” of [1] means the following: Suppose that a point \( p \) in the source mesh has a correspondence point \( t_p \) in the target. One can use nearest neighbor search to determine an inverse correspondence of \( t_p \) in the source mesh, named \( q \). If \( p \) and \( q \) are close, then the correspondence between \( p \) and \( t_p \) is more likely to be correct. More details can be found in section 4 of [1]. From the result we can see that their method preserves the mesh well, but fails to push the source mesh to the relatively far target. In this case, single correspondence weighted with bidirectional closest point search is not enough to give us correct correspondences. As shown in Figure 6.5(b), the inverse correspondence (black dot) of the corresponding point (red dot) on the target is far away from the source point (blue dot). However, the false correspondence (green dot) of the point at black dot has an even closer inverse correspondence point (purple dot). In this case, the summand of the distance term that related to the false correspondence will be assigned a much higher weight.

We show corresponding \( l_1, l_2 \) and \( l_{\text{max}} \) distances between the current mesh and the target mesh in Table 6.1 with different methods. \( l_1, l_2 \) and \( l_{\text{max}} \) distances between the current mesh \( R \) and the target mesh \( T \) are defined by
\[
l_1(R, T) = \frac{1}{|R|} \int_{R, p \in R} |d(p)|,
\]
\[
l_2(R, T) = \sqrt{\frac{1}{|R|} \int_{R, p \in R} d(p)^2},
\]
\[
l_{\text{max}}(R, T) = \max_{p \in R} d(x_i),
\]
where \( p \) denote the points on \( R \), \(|R|\) denotes the surface area of \( R \). Here, NICPS is our method. All distances are expressed in millimeters.

Table 6.1 compares results obtained by fitting the template mesh to 14 vertebrae shapes used to build statistical shape model (Section 7).

**Table 6.1: Comparisons with other methods with different inputs.**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( l_1 )</td>
<td>( l_2 )</td>
<td>( l_{\text{max}} )</td>
<td>( l_1 )</td>
<td>( l_2 )</td>
</tr>
<tr>
<td>1</td>
<td>0.803</td>
<td>0.997</td>
<td>3.895</td>
<td>0.771</td>
<td>0.916</td>
</tr>
<tr>
<td>2</td>
<td>1.053</td>
<td>1.404</td>
<td>7.335</td>
<td>0.968</td>
<td>1.238</td>
</tr>
<tr>
<td>3</td>
<td>0.713</td>
<td>0.888</td>
<td>3.704</td>
<td>0.733</td>
<td>0.878</td>
</tr>
<tr>
<td>4</td>
<td>1.162</td>
<td>1.595</td>
<td>8.306</td>
<td>1.085</td>
<td>1.394</td>
</tr>
<tr>
<td>5</td>
<td>2.351</td>
<td>3.172</td>
<td>10.416</td>
<td>1.980</td>
<td>2.613</td>
</tr>
<tr>
<td>6</td>
<td>1.509</td>
<td>2.216</td>
<td>10.619</td>
<td>1.464</td>
<td>2.103</td>
</tr>
<tr>
<td>7</td>
<td>1.336</td>
<td>1.964</td>
<td>11.530</td>
<td>0.711</td>
<td>1.236</td>
</tr>
<tr>
<td>8</td>
<td>2.507</td>
<td>3.382</td>
<td>11.113</td>
<td>2.249</td>
<td>3.030</td>
</tr>
<tr>
<td>9</td>
<td>1.601</td>
<td>2.053</td>
<td>8.041</td>
<td>1.200</td>
<td>1.499</td>
</tr>
<tr>
<td>10</td>
<td>2.603</td>
<td>3.440</td>
<td>13.941</td>
<td>2.039</td>
<td>2.824</td>
</tr>
<tr>
<td>11</td>
<td>2.088</td>
<td>2.661</td>
<td>9.652</td>
<td>2.900</td>
<td>3.777</td>
</tr>
<tr>
<td>12</td>
<td>2.144</td>
<td>2.731</td>
<td>9.687</td>
<td>2.355</td>
<td>2.856</td>
</tr>
<tr>
<td>14</td>
<td>3.052</td>
<td>4.269</td>
<td>11.779</td>
<td>2.259</td>
<td>3.068</td>
</tr>
<tr>
<td>average</td>
<td>1.745</td>
<td>2.372</td>
<td>9.478</td>
<td>1.581</td>
<td>2.100</td>
</tr>
</tbody>
</table>

Both Pauly2005 and our method are aimed to decrease the L2 distance between source and target meshes. One can observe that our method achieves better performance: the average L2 distance is approximately 34% of Pauly2005’s.

6.9.2 Comparison to other multiple correspondence schemes

**Coherent Point Drift:** Myronenko and Song [59] developed a point set registration method. We use vertex sets of triangular surfaces as inputs of the algorithm. The parameter of this method is set according to the example source code provided by authors: http://sites.google.com/site/myronenko/research/cpd. The method we choose is the nonrigid
registration with low-rank kernel approximation. In addition, the acceleration part of using FGT to compute matrix-vector products is adopted. All results shown here require no more than 100 iterations to converge.

![Figure 6.6: Comparisons with Coherent Point Drift [59] and its improved variant [60].](image)

From the results shown in Figure 6.6, we see the current point set in coherent point drift method converges to an unsatisfactory state. Here, the source mesh has about 4,000 vertices, and the target has about 7,700 vertices. The sampling rates of surfaces are significantly different, and the vertex sets do not represent consistent features. Average residual errors of this method is also larger than those of our method (See Table Figure 6.6). Other similar cases can be found in comparisons using synthetic data.

**Jian2011**: The point set registration method developed by Jian and Vemuri [60] also takes vertex sets of triangular surfaces as inputs. In addition, a set of control point is used to accelerate the algorithm. As shown in Figure 6.6, the model is elongated. This phenomena is also due to different sampling rates over two surfaces. The code we use to perform the comparisons is provided by authors: [http://code.google.com/p/gmmreg/](http://code.google.com/p/gmmreg/).

To see the distortions after registration, we have put stripe textures on the deformed surfaces as shown in Figure 6.7 and Figure 6.8. The texture coordinates used here are z- (column 1), y- (column 2) and x-coordinates (column 3) of the source mesh, respectively. The apparent distortion for our technique is slightly higher than for some other methods, which is a consequence of more aggressive pursuit of the fit quality goal.
Figure 6.7: Results shown with textures.
Figure 6.8: Results shown with textures.
6.9.3 Edge-based energy vs Laplacian-based energy

If we incorporate our two-way multiple correspondence search scheme into Pauly2005 to substitute the bidirectional closest point search scheme, we will get results shown in Figure 6.9a, where an uncomfortable distortion in the deformed mesh is visually apparent.

![Figure 6.9: Results comparison between edge-based and Laplacian-based potential energy with two-multiple correspondence. The pink surfaces are deformed models.](image)

(a) Edge-based  (b) Laplacian-based

However, if we use Laplacian-based potential energy rather than edge-based potential energy, the result looks much better.

The difference is due to the fact that the correspondences of saddle points in the the source mesh are not saddle in the target. See Figure 6.10. Edge-based potential energy will try to preserve this saddle, however, the Laplacian-based will not. Generally, Laplacian-based potential energy allows the mesh to deform more freely.

![Figure 6.10: A saddle point in the source mesh.](image)
6.10 Feature preservation

24 feature points are manually picked by biomechanical engineers on both source and target meshes, as shown in Figure 6.16a and c.

The feature points identified on each specimen mesh were chosen in correspondence with anatomical landmarks (Figure 6.15). These included points at the following locations: the most superior and posterior margin of the spinous process (0), the most inferior and posterior margin of the spinous process (1), the most lateral margins of the left (2) and right (3) superior facet surfaces, the most lateral points (but midpoints in the superior-inferior direction) of the left (4) and right (5) transverse processes, the most inferior margins of the left (6) and right (8) inferior facet surfaces, the most medial and superior margins of the left (9) and right (10) superior facet surfaces, the most distal (7) and proximal (17) margins of a midsagittal line along the posterior wall of the vertebral foramen, a series of uniformly spaced points on the most distal circumference of the body (18, 19, 20, 21, 22, 23), a series of uniformly spaced points on the most proximal circumference of the body (11, 12, 13, 14, 15, 16).

From the results, we observe that most features are fitted to corresponding ones on target model except feature point 0 and feature point 21. See Figure 6.16b and d. Distances between corresponding landmarks are shown in Table 6.2. Our method achieves the shortest average distance.

6.11 Application: statistical shape analysis

We use our registration method to analyze the shape variation of a family of femur models and a family of vertebrae models. Similar approaches can be found in [99, 100]. We applied our procedure to compute compatible triangulations of a set of 14 vertebrae and 14 femur models.
Table 6.2: Distances between landmarks before and after registration using different methods.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>9.5863</td>
<td>10.8456</td>
<td>9.32940</td>
<td>6.51063</td>
<td>10.6626</td>
<td>5.53484</td>
</tr>
<tr>
<td>1</td>
<td>9.29295</td>
<td>3.30022</td>
<td>8.31313</td>
<td>3.34858</td>
<td>10.0866</td>
<td>2.51325</td>
</tr>
<tr>
<td>2</td>
<td>5.51388</td>
<td>4.08575</td>
<td>3.80643</td>
<td>2.74631</td>
<td>1.10903</td>
<td>4.40684</td>
</tr>
<tr>
<td>3</td>
<td>6.94648</td>
<td>3.28229</td>
<td>2.74631</td>
<td>1.10903</td>
<td>4.40684</td>
<td>3.49391</td>
</tr>
<tr>
<td>4</td>
<td>7.90560</td>
<td>3.72635</td>
<td>4.19258</td>
<td>1.67472</td>
<td>4.66790</td>
<td>2.18246</td>
</tr>
<tr>
<td>5</td>
<td>3.09942</td>
<td>1.67350</td>
<td>2.38713</td>
<td>2.17449</td>
<td>4.50588</td>
<td>1.89824</td>
</tr>
<tr>
<td>6</td>
<td>4.35649</td>
<td>4.96940</td>
<td>4.12022</td>
<td>1.91627</td>
<td>6.70526</td>
<td>3.49391</td>
</tr>
<tr>
<td>7</td>
<td>3.95392</td>
<td>1.97403</td>
<td>3.80601</td>
<td>5.30688</td>
<td>7.05678</td>
<td>3.17574</td>
</tr>
<tr>
<td>8</td>
<td>2.92400</td>
<td>3.55421</td>
<td>3.14298</td>
<td>4.16167</td>
<td>3.73214</td>
<td>3.31785</td>
</tr>
<tr>
<td>9</td>
<td>6.50798</td>
<td>5.56568</td>
<td>3.59548</td>
<td>4.31211</td>
<td>3.49265</td>
<td>4.61843</td>
</tr>
<tr>
<td>10</td>
<td>3.36709</td>
<td>4.32855</td>
<td>2.90664</td>
<td>4.30973</td>
<td>5.09522</td>
<td>3.30413</td>
</tr>
<tr>
<td>11</td>
<td>2.14916</td>
<td>2.46599</td>
<td>1.57927</td>
<td>3.73072</td>
<td>4.32255</td>
<td>1.68942</td>
</tr>
<tr>
<td>12</td>
<td>5.70937</td>
<td>5.32796</td>
<td>6.14953</td>
<td>8.22429</td>
<td>8.63105</td>
<td>5.77087</td>
</tr>
<tr>
<td>13</td>
<td>2.64051</td>
<td>2.73187</td>
<td>1.45411</td>
<td>2.38454</td>
<td>7.24908</td>
<td>1.04047</td>
</tr>
<tr>
<td>14</td>
<td>3.93723</td>
<td>4.17767</td>
<td>3.88454</td>
<td>2.51152</td>
<td>5.47829</td>
<td>3.05711</td>
</tr>
<tr>
<td>16</td>
<td>5.81798</td>
<td>7.58011</td>
<td>6.39554</td>
<td>3.55704</td>
<td>9.67503</td>
<td>5.02726</td>
</tr>
<tr>
<td>17</td>
<td>6.19745</td>
<td>4.22991</td>
<td>3.86540</td>
<td>2.37615</td>
<td>9.80193</td>
<td>3.19152</td>
</tr>
<tr>
<td>18</td>
<td>7.87452</td>
<td>3.26287</td>
<td>3.09552</td>
<td>7.46175</td>
<td>9.82274</td>
<td>3.12538</td>
</tr>
<tr>
<td>19</td>
<td>5.56326</td>
<td>3.66574</td>
<td>2.50066</td>
<td>8.54122</td>
<td>7.97243</td>
<td>2.53802</td>
</tr>
<tr>
<td>20</td>
<td>7.72364</td>
<td>3.97502</td>
<td>3.61909</td>
<td>4.80908</td>
<td>7.08929</td>
<td>2.38924</td>
</tr>
<tr>
<td>21</td>
<td>4.89901</td>
<td>3.06382</td>
<td>1.99577</td>
<td>2.88418</td>
<td>8.62967</td>
<td>1.62487</td>
</tr>
<tr>
<td>22</td>
<td>7.59178</td>
<td>4.62163</td>
<td>3.71842</td>
<td>2.15563</td>
<td>7.79401</td>
<td>2.86062</td>
</tr>
<tr>
<td>23</td>
<td>5.76564</td>
<td>4.29529</td>
<td>4.0401</td>
<td>4.1083</td>
<td>7.10477</td>
<td>3.38939</td>
</tr>
</tbody>
</table>

6.11.1 Femur models

First, we choose one of the 14 input femur models $P_0$ as the base mesh. To bring the meshes into poses similar to $P_0$’s, we first roughly align them manually using a simple user interface and then run the rigid ICP algorithm to improve the alignment.

Then, we deform the base mesh to each of 14 meshes $P_i$ ($i = 0, 1, \ldots, 13$) obtaining a new sequence of models $P_i'$ ($i = 0, 1, \ldots, 13$) (note that $P'_0 = P_0$). $P_i'$ is geometrically close to $P_i$. Moreover, all meshes $P_i'$ have the same connectivity as the base mesh $P_0$. The average shape $P_{mean}$ can be computed by simply averaging the vertex coordinates over all of the deformed models (Figure 6.11).

For each deformed model $P_i'$ ($i = 1, 2, \ldots, 13$), we define the $3n$-dimensional shape vector $S_i = P_i' - P_{mean}$, where $n$ is the number of vertices of the base mesh $P_0$. This set of shape vectors $S_i$ spans a space of the input shapes. We perform principal component analysis (PCA) on the resulting set of shape vectors to find predominant shape variations (eigenshapes).
The first three principal components (PC) of variations captured approximately 93.9% (69.8% for the first PC, 18.5% for the second PC and 5.6% for the third PC) of the variance of the data. Their effects to the shape changing are illustrated in Figure 6.11.

![Figure 6.11: First three shape variation components. The blue shape is the average model $P_{\text{mean}}$, the pink shapes are three eigen models which are created by adding three PCs to $P_{\text{mean}}$, respectively.](image)

One can see that

- the first PC controls the length;
- the second PC controls the twist;
- the third PC controls the bend.

### 6.11.2 Vertebrae models

For the vertebrae models, we fit a template model to 14 vertebrae models (Figure 6.12) and use the same procedure to construct PCA models. In this case, the template mesh is a highly regular mesh constructed manually [6]. We proceed in the same way as in Section 6.11.1.

The variation of vertebrae is more complex than that of femurs. In particular, first three principal components of variations capture symmetric aspects of the shapes, as shown in Figure 6.13.

The same task can also be accomplished by using an interactive fitting methods [6]. The results are shown in Figure 6.14. Note that the eigenshapes are similar, although not the same (since they are based on different correspondence information). Our fitting procedure requires a minimum amount of user input (just to roughly align the input shapes) and therefore make shape analysis less dependent on human labor.
Figure 6.12: Vertebrae models before registration. (a) shows the template model. (b) and (c) show overlapping views of 14 models.

Figure 6.13: First five shape variation components. The blue shape is the average model $P_{\text{mean}}$, the pink shapes are three eigen models which are created by adding three PCs to $P_{\text{mean}}$, respectively.
Figure 6.14: First three shape variation components got by using an interactive fitting method. The blue shape is the average model $P_{\text{mean}}$, the pink shapes are three eigen models which are created by adding three PCs to $P_{\text{mean}}$, respectively.

(a) 1st PC  (b) 2nd PC  (c) 3rd PC

Figure 6.15: 24 feature points shown on the source mesh with different views.

(a) front view  (b) side view  (c) side view  (d) top view

6.12 Conclusions and limitations

This paper is an extension to our 4-page short paper [8]. We propose a two-way multiple correspondence search scheme for nonrigid surface registration. To take full advantage of
this scheme, one should adopt a relatively looser potential energy term, for example, the Laplacian-based potential energy term. The scheme does not rely on parameterizations and can be applied to families of complex models with relatively high shape variabilities, such as vertebrae.

Intuitively, multiple correspondences act like a smoothing filter, with the amount of smoothing proportional to local distance between the source and the target surface. All correspondences for a vertex of the source mesh are essentially “averaged” by the potential energy term. This is exactly what is desired in nonrigid shape registrations: ideally, if the distance between the source and the target model is large, one should focus on large scale geometric features when computing the deformation. Smaller scale features should influence the deformation on later stages. This observation has been exploited in coarse-to-fine multigrid deformation such as [44].

Figure 6.16: Feature points shown on source, target and fitted mesh.
The proposed method is limited in several aspects:

- The computational time is longer than traditional methods. In the example shown in the comparison section, average running times for 10 iterations of Pauly2005’s method, Andresen2001’s method and ours are 153, 227 and 379 seconds. The multiple correspondence scheme is approximately one and half times slower than only using one-to-one correspondences.

- It is not clear about how many correspondences are suitable in each step of the registration. The problem is similar to the problem of coarse-to-fine strategy used by Gilles et al.[44], where the cluster size is manually decreased.

- The current version of multiple correspondence search algorithm cannot distinguish similar patterns. For example, our method may deform two horse legs into one or swap them.

Besides, there are a number of parameters that need be chosen for different types of inputs. Among them, the weight and form of potential energy function are crucial. Intuitively, the more correspondences we use, a more flexible potential energy is required. In addition, correspondence trimming method can be further improved. Besides trimming correspondences according to normal differences, more sophisticated criteria, such as multi-level curvatures used in Huang et al’s method [22]and spin images used by Chang and Zwicker [12], can be used.

The main goal of this work is to substitute the manual fitting by biomechanics engineers [6], and then perform a more objective statistical shape analysis. The shape variation information is then used to build specimen-specific Finite Element (FE) models [6, 7]. Since the vertebrae models are manifolds, using Principle Geodesic Analysis (PGA) [101] instead of PCA may yield more meaningful results.
In this thesis, we presented several contributions to shape matching and registration. First, we discussed the spin contour (Chapter 3) and the geodesic spin contour (Chapter 4). Both are local shape descriptors that can concisely describe neighborhoods of points on a shape. Our experiments show that they provide high quality results while being robust with respect to different types of data contamination. We also present two global matching methods, the fully automatic isometric matching (Chapter 5) and nonrigid iterative closest points (Chapter 6).

An approach that is commonly taken in practical applications is to choose a suitable type of deformation (consistent with the application) and then apply algorithms suitable for this kind of deformation to perform shape matching or analysis. For example, when registering the 3D laser scan data, one often focuses on rigid transformations. In this scenario, spin contour is better choice over the traditional spin image. When registering two animated meshes, geodesic spin contour and the fully automatic shape matching can be used to recover the near-isometric deformations between these meshes. When the registration is performed between two human vertebrae, the underlying deformation is complicated: neither rigid, nor near-isometric. In such cases, the nonrigid iterative closest points is suitable.

Shape registration is still an open question. All new methods proposed in this thesis could be improved and extended. First, the common idea behind the spin contour and the geodesic spin contour is that image contour has better descriptiveness than image itself. At this point, our evaluation is purely experimental. It would be interesting to develop a rigorous theoretic analysis of different spin image based descriptors. Such analysis may lead to deeper understanding of the information content of the spin image and help further improve its performance.
Second, besides triangular mesh data, spin image has been used on matching images and vector fields [102][103]. Spin contour and geodesic spin contour have potential to improve the matching performance of spin image for data other than 3D meshes.

Third, many parameters need to be tuned for different set of data. This problem is common to all state-of-the-art shape matching algorithms. We would like to work on a unified parameter optimization framework for our algorithms.

Finally, current global shape matching algorithm still cannot consistently handle global symmetric flip problem. Since we only use geodesic distance difference as the major part of our objective function, our method suffers from the same problem. A possible future direction is to investigate how to automatically find and reject symmetrically flipped correspondences.
REFERENCES CITED


