ROBUST CORRESPONDENCE AND RETRIEVAL OF ARTICULATED SHAPES

by

Varun Jain
B.E., Institute of Engineering & Technology
Devi Ahilya University
Indore (India), 2003

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
in the School
of
Computing Science

© Varun Jain 2006
SIMON FRASER UNIVERSITY
Summer 2006

All rights reserved. This work may not be
reproduced in whole or in part, by photocopy
or other means, without the permission of the author.

APPROVAL

Name: Varun Jain

Degree: Master of Science

Title of thesis: Robust Correspondence and Retrieval of Articulated Shapes

Examining Committee: Dr. Arthur (Ted) Kirkpatrick,
Assistant Professor, Computing Science
Chair

Dr. Richard (Hao) Zhang, Senior Supervisor
Assistant Professor, Computing Science

Dr. Greg Mori, Supervisor
Assistant Professor, Computing Science

Dr. Ze-Nian Li, Supervisor
Professor, Computing Science

Dr. Richard Vaughan, SFU Examiner
Assistant Professor, Computing Science

Date Approved: May 26/06


SIMON FRASER UNIVERSITY library

DECLARATION OF PARTIAL COPYRIGHT LICENCE

The author, whose copyright is declared on the title page of this work, has granted to Simon Fraser University the right to lend this thesis, project or extended essay to users of the Simon Fraser University Library, and to make partial or single copies only for such users or in response to a request from the library of any other university, or other educational institution, on its own behalf or for one of its users.

The author has further granted permission to Simon Fraser University to keep or make a digital copy for use in its circulating collection, and, without changing the content, to translate the thesis/project or extended essays, if technically possible, to any medium or format for the purpose of preservation of the digital work.

The author has further agreed that permission for multiple copying of this work for scholarly purposes may be granted by either the author or the Dean of Graduate Studies.

It is understood that copying or publication of this work for financial gain shall not be allowed without the author's written permission.

Permission for public performance, or limited permission for private scholarly use, of any multimedia materials forming part of this work, may have been granted by the author. This information may be found on the separately catalogued multimedia material and in the signed Partial Copyright Licence.

The original Partial Copyright Licence attesting to these terms, and signed by this author, may be found in the original bound copy of this work, retained in the Simon Fraser University Archive.

Simon Fraser University Library
Burnaby, BC, Canada

Abstract

We consider the problem of shape correspondence and retrieval. Although our focus is on articulated shapes, the methods developed are applicable to any shape specified as a contour, in the 2D case, or a surface mesh, in 3D. We propose separate methods for 2D and 3D shape correspondence and retrieval, but the basic idea for both is to characterize shapes using intrinsic measures, defined by geodesic distances between points, to achieve robustness against bending in articulated shapes. In 2D, we design a local, geodesic-based shape descriptor, inspired by the well-known shape context for image correspondence. For 3D shapes, we first transform them into the spectral domain based on geodesic affinities to normalize bending and other common geometric transformations and compute correspondence and retrieval in the new domain. Various techniques to ensure robustness of results and efficiency are proposed. We present numerous experimental results to demonstrate the effectiveness of our approaches.

Reader's Summary

Shape correspondence is an important problem in computer graphics and computer vision. It is usually the first step in graphics applications such as parameterization, texture mapping, shape morphing and other attribute transfer applications. It is also useful for shape retrieval and object alignment, recognition and hence is of interest to both the graphics and vision communities. Shape retrieval in the 2D domain has been well studied in computer vision. With the recent advances in 3D model acquisition technology, 3D models have become ubiquitous, for example, on the internet, and as a result, the need for a system for retrieval of 3D models from databases containing numerous 3D shapes has also gained prominence.

One of the main difficulties faced by any shape correspondence or retrieval algorithm involves matching shapes that differ from each other by various forms of geometric transformations. Even a simple rotation of shapes can cause many algorithms to return incorrect results. Most of the previous work on this problem focuses on shapes that differ from each other by rigid transformations (translation and rotation) and uniform scaling. As expected, these methods do not perform well for shapes with non-rigid transformations, such as bending and non-uniform stretching.

In this work, we consider the problem of correspondence and retrieval of articulated shapes. Along with the usual rigid transformations, such shapes are also expected to undergo non-rigid bending transformation. Although the focus of our work is on articulated shapes, our methods are in general applicable to any kind of shapes. In 2D, the shapes are defined as contours, whereas in 3D, as triangle meshes. The basic idea in both cases is to study the shape in a geodesic manner to capture intrinsic shape information. Shape correspondence is then computed based on this intrinsic information rather than the absolute coordinates of the given points. Geodesic distances between two points on a shape remain invariant when the shape undergoes bending transformation. Hence, such geodesic approach will

make the resulting algorithm robust to shape bending, thus suitable for articulated shapes. In 2D, this can be done by traversing the contour geodesically and gathering representative shape information. We propose a novel shape descriptor for 2D shapes, inspired by the well known notion of *shape context*, that uses such geodesic traversal to attain robustness against bending along with invariance to rigid transformations. Due to the way this descriptor is constructed, we call it *geodesic shape context*.

For 3D shapes, we adopt the spectral correspondence approach. We first perform a spectral transformation on the given 3D mesh to obtain another mesh (spectral embedding) that is normalized against all rigid transformations and uniform scaling. For articulated shapes, we modify the spectral approach to use geodesic point affinities, so that the spectral embedding is also normalized against shape bending. Now, conventional algorithms can be applied in the spectral domain thus achieving invariance to bending. We also identify various problems associated with spectral embeddings and conventional spectral correspondence methods. We suspect that these problems arise due to non-uniform scaling and other factors such as shape noise and degeneracies. We take appropriate measures to mitigate the effect of such problems on the final results.

Acknowledgments

I would like to express my earnest gratitude to the faculty and my fellow students at the Graphics Usability and Visualization lab of the School of Computing Science, for creating a scintillating research environment in the lab and making my experience during my degree, a real treat. I specially thank my supervisor Dr. Richard Zhang, for his invaluable guidance, motivation and readiness to help throughout my stay at the grad school and also for his everlasting enthusiasm towards the subject and his meticulous perfection while evaluating my work.

I am overwhelmed with gratitude for my parents and family for their colossal and unbroken support during all my education years.

I owe special thanks to my friends Ai, Andrew, Chintan, Frank, Ginger, Matt, Nadia, Nilesh and Rajat for their stupendous support and help and for making my education a fun and rewarding experience.

I thank all members of my supervisory committee for their constructive criticism of our work and their insight on various aspects of the thesis. I would also like to thank Heather, Kersti, Val and other staff members at the School of Computing Science for making my stay at Simon Fraser University delightful and memorable.

Contents

List of Figures

Chapter 1

Introduction

Shape recognition is one of the classical problems in the field of computer vision and has been the subject of immense amount of research in the vision community. The aim of this research is to build a recognition system that performs as well as a human. Even after numerous attempts at building an efficient and accurate recognition system, recognition performed by a human is still far better than that performed by a machine. For example, humans can recognize objects from an extremely cluttered scene, or a scene with a great deal of noise, with exceptional accuracy. Humans can even identify a person even if they have not seen him/her from the angle they are looking at this person at the moment. One reason why conventional shape matching systems do not perform as well as a human is because the method used by a human brain to perform recognition is not completely understood and hence is hard to simulate.

A problem closely related to that of shape recognition is shape retrieval. Many instances of the recognition problem are in fact also instances of the retrieval problem. For example, face recognition is nothing but the problem of retrieving a face from a database, that is most similar to the query face. However, recognition is a more general problem and includes other applications such as identifying an object in a cluttered or occluded scene. Another problem related to the shape recognition problem is that of shape correspondence and registration. In many cases, the result of shape correspondence serves as the starting point for a shape recognition system. Many algorithms for shape retrieval also begin with solving the correspondence problem first.

The focus of this thesis is shape correspondence and retrieval. Both shape correspondence and retrieval in 2D has been well studied in the field of computer vision. Research

in the case of 3D shape correspondence and retrieval is however relatively new. Before presenting the details of our work, we briefly describe the two problems we are addressing, the method we adopt and our contributions.

1.1 Background: Problems, Applications and Techniques

We now give a brief description of the problem of shape correspondence and retrieval. We also discuss various relevant applications to motivate our study and give an overview of the general techniques used to solve the problem.

1.1.1 Shape Correspondence and Registration

Shape correspondence is the problem of finding a meaningful matching between the structural elements of two shapes. We consider these shapes to be defined as contours (approximated by polygons) in the case of 2D shapes and surfaces (approximated by triangle meshes) in the 3D case. The structural elements to be matched in our case are the vertices of the two polygons or triangle meshes. Shape registration, on the other hand, refers to finding the spatial transformation between the vertices of one shape and the corresponding vertices of the second shape, that is, a rigid or non-rigid alignment that best aligns the vertices of the two shapes. Shape correspondence and shape registration are interdependent problems in the sense that the output of one is generally used to compute the other.

Applications and General Techniques

Shape correspondence has numerous applications in both computer vision and graphics. A variant of this problem is that of feature point correspondence on images. Here, the problem is of finding a matching between the feature points in two similar images. Such correspondence leads to the computation of registration parameters between the two images. This registration can be used for various applications such as reconstruction of data from different images, similarity measure for image based object retrieval and recognition, etc. Shape correspondence also has applications in the field of medical image analysis, such as matching two images to identify diseased tissue or detect tumors. In this work however, we mainly focus on the applications of shape correspondence in computer graphics. A meaningful shape correspondence is usually the first step in numerous graphics applications such

as constructing animation sequences, shape morphing, parameterization, texture mapping and other attribute transfer problems.

Note that for most of the applications of shape correspondence pertaining to vision and medical image analysis, the major concern is to robustly obtain correspondences between shapes in the presence of noise, clutter, occlusion and other factors. In many instances, in such applications the input images are already well aligned or differ from each other with only rigid or minor non-rigid transformations. In contrast, for graphics applications, noise, clutter etc. are not the major concerns as the two shapes are already well defined. The challenge here is to match shapes that differ from each other from non-rigid transformations such as bending and/or non-uniform stretching or scaling as in applications such as animation, morphing, etc. In particular, we focus on the class of *articulated shapes.* Such shapes are formed of various segments connected to each other at movable joints (for example, humans, scissors, animals, etc.). The most common transformation between such shapes is bending and minor non-uniform stretching. It is, in general, difficult to define such transformations mathematically (for example, using registration parameters). Hence, traditional approaches for non-rigid alignment do not give satisfactory results.

There are in general two types of methods for computing shape correspondence: those based on absolute coordinates (extrinsic) and those based on relative information (intrinsic). Intrinsic methods are in general more effective than extrinsic ones, as the matching computed from intrinsic methods is based on overall shape similarity. We briefly explain the two general approaches below:

1. **Extrinsic methods:** Methods based on the absolute coordinates of the vertices of the two shapes iteratively compute the correspondence and registration between two shapes in an inter-leaved fashion e.g., the well known ICP algorithm [7]. The correspondence is computed using a "closest point" measure which is solely based on the spatial coordinates of the vertices. It can be shown that computing the correspondence and registration that gives the most optimal alignment of the two shapes is NP-hard. Hence, these iterative techniques alternate between the steps of finding correspondence, and finding registration parameters.

2. **Intrinsic methods:** Given two shapes, the general idea here is to represent every point using shape information from the perspective of that point. That is, every point is transformed into a high dimensional feature space where the shape information can

be encoded in the coordinates of the point. Matching is then performed based on the coordinates of the points in this feature space. Such information can be collected in the original space or in a difference domain such as one that is obtained via spectral analysis, which we refer to as the spectral domain. Based on this classification, there are two types of intrinsic techniques:

- *Point descriptor based:* For every point on the two shapes construct a descriptor that encodes shape information relative to that point. Two points are then matched if their descriptors match. The descriptor is constructed in the original spatial domain and is defined in a way that is canonical for the two shapes and is invariant (or robust) to common geometric transformations.
- *Spectral methods:* Again, every point is represented using relative shape information. However, the information collected as is, is not canonical for the two shapes, and possess other undesirable properties such as redundancy, large complexity, non-robustness, etc. Hence, the data is transformed into the spectral domain using principal component analysis (PCA). This not only gives a canonical representation of the data for the two shapes, but also enables us to extract the most important information (principal components).

In general, the descriptor based methods are the most recent and most popular in the literature. This is mainly due to their simplicity and drawbacks associated with the other methods. We give detailed description of the relevant shape correspondence methods and their strengths and weaknesses in Chapter 2.

1.1.2 Shape Retrieval

Shape retrieval is the problem of retrieving shapes from a large database of shapes that are most similar to a given query shape. Hence, shape retrieval can be reduced to the problem of shape similarity, i.e., finding how similar two given shapes are. Generally, shape retrieval systems consist of a user interface, where the user can specify a query shape, and the result returned is an ordered list of shapes in the database, similar to the query shape (ordered by similarity).

Although most research work on this problem is motivated by improvement of the accuracy of retrieval, the user interface part of the system has its own difficulties. For 2D, the query shape can be easily specified by the user via simple drawing(s). However, for

3D shapes, specifying the query shape is non-trivial and is usually done by drawing the 2D image of the shape from various views and then reconstructing the 3D shape from these views. This issue is not the aim for this thesis and we assume that the query is already in the form of a 3D shape that is constructed using some user interface. As for correspondence, we focus on retrieval of articulated shapes in this thesis.

Applications and General Techniques

Shape similarity in 2D has many applications and is one of the most studied problems in the vision community. It is used for image based pattern/object recognition, face recognition, signature verification, character recognition, and so on. From the perspective of computer graphics, however, the shape similarity problem is more important for 3D shapes. With the recent advances in the acquisition and visualization technology for 3D models, there has been a flurry of 3D models available for use to the common user from, for example, the Internet. Many websites [63] provide large databases of 3D models consisting of thousands, even tens of thousands of models. In such situation, the need for a shape retrieval system is dire in order to facilitate users in fields such as entertainment, medical research, machine parts/tools design, etc., to obtain what they require from these databases in an efficient and effective manner.

The major challenge in constructing a shape similarity algorithm is that the algorithm must be sufficiently discriminating to distinguish between different shapes, and at the same time, be robust to minor variations of a shape. Another aspect of the problem is the time and space complexity of the query method as the databases in question can be arbitrarily large. There are two most common techniques used for shape retrieval:

1. **Correspondence based:** To compute similarity between two shapes, the first step in these methods is to construct a correspondence between the vertices of the two shapes and then perform a registration. The similarity measure is then computed as the cost of performing this correspondence and registration [4, 5, 31, 32].

2. **Global shape descriptor based:** In this method, both shapes are first defined using some global shape descriptor. This descriptor usually captures the overall shape of the object. Similarity between the two shapes is then computed as the similarity between their descriptors. Well known examples of such global shape descriptors include the spherical harmonics descriptor [34] and the light field descriptor [13].

The correspondence based method is in general more popular in the case of 2D matching since in many applications, the input data is simply a set of points and no real shape is specified. However, for 3D shape similarity, the second method has been preferred.

1.2 Our Method

We present different methods for 2D and 3D shapes. However, the basic idea for both methods is to use intrinsic measures to obtain a more meaningful representation of the shape. Another commonality between the two methods is the use of geodesic traversal to capture the intrinsic information in order to achieve robustness when handling articulated shapes. For 2D shape correspondence and retrieval, we adopt the point descriptor approach, while for 3D shapes, we use the spectral method. We now describe both methods briefly:

- **2D correspondence and retrieval using geodesic shape context:** A popular way of computing shape matching is to use local shape descriptors. First introduced by Belongie et al. [4, 5], the *shape context* of a shape, at a point, is a histogram that describes the shape relative to that point. This histogram is constructed by dividing the space around the point into appropriately shaped bins and counting the number of points in every bin. The idea behind using such histograms is to match points not just based on their spatial coordinates, but based on coordinates that also contain shape information from the perspective of that point.

  Shape context is the basis for the method we develop for computing shape correspondence in 2D. Shape contexts as described by Belongie et al. [4, 5] are invariant to rigid transformations in the shapes as well as uniform scaling. Our aim is to design a shape descriptor that is additionally robust to shape bending, so that it can be used for articulated shapes. Hence, we identify the property of any shape that remains invariant to bending. Clearly geodesic distances between points do not change when shapes are bent. Thus, we construct bins in a geodesic manner, hence the name "geodesic shape contexts". After constructing these bins, we collect curvatures of the points in every bin to form the descriptor. Note that curvature is not invariant to bending. Hence, we perform appropriate processing to render the descriptor robust to bending. Matching is then computed by comparing the descriptors of the points on the two shapes.

  The matching computed is associated with a correspondence cost, which is the sum

of squared differences between the descriptors of corresponding points. We use this cost as a similarity measure between two shapes, where a lower cost means greater similarity. This similarity measure is then used for the purpose of shape retrieval.

- **3D correspondence and retrieval using spectral embeddings:** The descriptor approach described above can also be used for 3D shapes. However, the computation of geometric properties such as curvature or the principal directions of curvatures becomes non-robust in the case of 3D shapes. These properties are essential for constructing many shape descriptors. For example, curvature is required for constructing geodesic shape contexts, principal directions and surface normals are required by rotation invariant shape contexts [4, 5, 19, 37] and spin images [31, 32] etc. Moreover, many 2D shape correspondence techniques rely on the ordering of the points along the contour to compute correspondence efficiently. They are not applicable to 3D, due to the lack of a canonical ordering of points on a 3D surface. Hence, we decided not to use the conventional descriptor based approaches for 3D correspondence.

  The spectral method of correspondence was first introduced by Umeyama [62] for graph matching and Shapiro and Brady [57] for image matching. The basic idea here is to first compute a spectral embedding of the two shapes, and then perform matching in the spectral domain. The spectral embedding is given by the eigenvectors of the *affinity matrix* of the shape. The affinity matrix contains the affinity between every pair of points on the shape. The advantage of performing such a transformation is that if the affinity is defined to be invariant or robust to a certain class of transformations, then the whole matching process becomes invariant/robust to this class. Hence, we define affinities based on geodesic distances between points to make the matching process invariant to bending. Note that affinities defined in such a way are also invariant to rigid transformations. Although spectral methods have been well studied, we identify certain problems, for example eigenvector switching, with the method that have never been identified and suggest appropriate measures to alleviate these problems.

  Once we have the spectral embedding of two shapes, we can regard these embeddings as new shapes and conventional shape similarity measures can be applied to compute the similarity between these shapes. Hence, traditional global shape descriptor based algorithms can be applied to the embeddings to perform shape retrieval. We study the effect of these algorithms on spectral embeddings and also present other global

descriptors that can be easily obtained from the spectral embedding and used for retrieval.

1.3 Contributions

In this thesis, we present methods for correspondence and retrieval of articulated shapes. Although our method is designed to work well particularly on articulated shapes, it can be used for general shapes that are represented as 2D contours or 3D surface meshes. We present separate methods for 2D and 3D shapes. The basic idea for both methods is to represent the shapes using intrinsic measures and to define the intrinsic measures in a "geodesic" way in order to achieve robustness against bending in articulated shapes. For 2D shapes, we extend the shape contexts of Belongie et al. [4, 5]. Whereas, for 3D, we improve the spectral correspondence method of Shapiro and Brady [57] using various techniques. The following are our contributions in this work:

- **New point descriptor for 2D shape correspondence:** A new descriptor for points on a 2D contour is developed. This descriptor is essentially a "geodesic" version of shape contexts [4, 5]. The descriptor is used for first computing a correspondence between the points of two 2D contours and then defining a measure of similarity between the two contours for the purpose of shape retrieval.

- **Bending invariant spectral embeddings for 3D shapes:** For 3D shape correspondence, we use the spectral method of Shapiro and Brady [57]. The spectral method has previously been studied only for 2D datasets. First, we adapt the method for use with articulated 3D shapes by using geodesic distances to build spectral embeddings. These embeddings can subsequently be used for correspondence and retrieval. However, we show that the naive embeddings used by Shapiro and Brady [57] are non-robust to various factors and we develop the following improvements over their method:

  - We present a new eigenvector scaling scheme for the embeddings and give theoretical arguments to show that the new scheme makes the method robust to difference in the size of given data.
  - We show the presence of reflections (cause by eigenvector switchings) in the spectral embeddings. This negates the common belief that for spectral methods,

the eigenvector order given by the corresponding eigenvalues is suitable for shape correspondence. We provide heuristics to deal with these reflections so as to obtain more robust matching.

- We show the presence of other rigid/non-rigid transformations in the embeddings and suggest the use of non-rigid alignment to attain robustness against them.
- We present a proximity based heuristic to improve the quality of correspondence obtained from the spectral correspondence method. In fact, the heuristic can be used in conjunction with any correspondence algorithm and we also take advantage of it to improve the results of our 2D correspondence method.

- **3D shape retrieval using spectral embeddings:** We apply the spectral embedding framework to the problem of 3D shape retrieval on articulated database and show absolute improvements over conventional global shape descriptor based retrieval algorithms. We also present a another global descriptor and empirically show that it performs better than conventional descriptors for the purpose of retrieval.

1.4 Thesis Organization

The rest of the thesis is organized as follows: in Chapter 2 we discuss previous attempts and methods to solve the correspondence and retrieval problem. Chapter 3 is devoted to 2D shape correspondence and retrieval, where we develop a new shape descriptor for 2D shape matching and retrieval. In Chapter 4, we discuss the problem of 3D shape correspondence and retrieval. We explain in detail the method of spectral correspondence and spectral embeddings and use it for the purpose of matching and retrieval of 3D shapes. We conclude with limitations of our method and possible future work in Chapter 5.

Chapter 2

Previous Work

Several techniques have been developed over the years for the problem of shape correspondence and retrieval. In this Chapter, we review those that are most relevant to our work.

2.1 Shape Correspondence and Registration

A detailed survey of shape correspondence techniques can be found in [64]. As explained briefly in Chapter 1, point correspondence may be computed based on either absolute coordinates (extrinsic methods) or relative information (intrinsic methods), e.g., using weighted graphs. The intrinsic methods can be further classified into spatial domain point descriptor based and spectral domain methods. We now review the literature based on this classification.

The first class of shape correspondence techniques are iterative alignment schemes. These methods iteratively compute a correspondence between the two shapes and a transformation which would transform one shape into another. The correspondence is computed using a "closest point" criteria which is based on the absolute coordinates of the points. The transformation is obtained by optimizing an energy. Such techniques include the well known iterative closest point (ICP) algorithm of Besl and Mckay [7] and its variants [50], which can handle affine transformations. Recent works, most notably the TPS-RPM method of Chui and Rangarajan [14], attempt to incorporate non-rigid deformations into the ICP framework, using thin-plate splines to model the deformation. However, these methods can easily get trapped in bad local minima if the shapes are not approximately aligned initially, since the correspondence, which dictates the optimization, is computed using a Euclidean

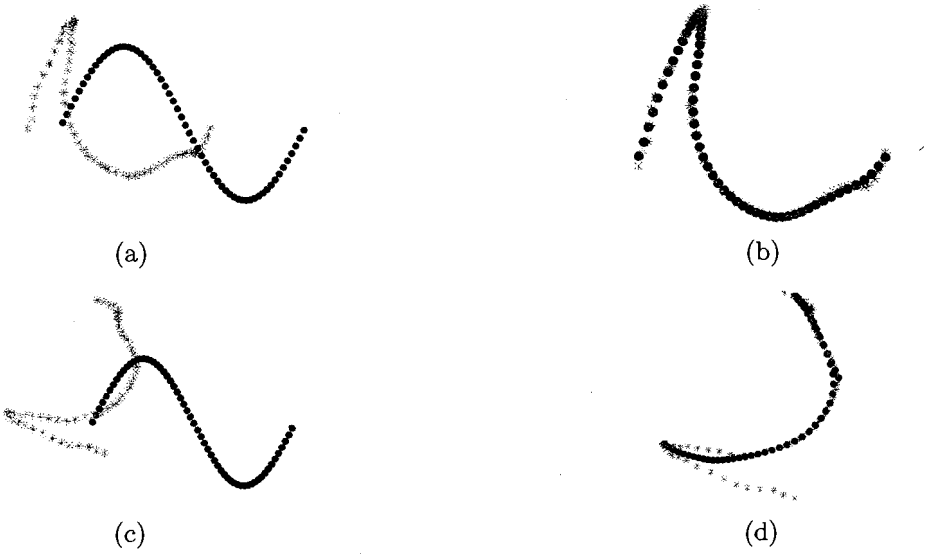


Figure 2.1: TPS-RPM [14] failed by rotation. (a) Two 2D shapes (note that although the shapes differ with non-rigid deformations, they are approximately aligned in terms of rotations. Hence, the "closest point" correspondence is relatively robust); (b) correct matching is obtained; (c) one of the shapes is rotated (now the "closest point" matching is no longer reliable); (d) Incorrect matching is obtained.

closest point method. Thus rotation alone can cause a bad matching, as shown in Figure 2.1.

Sumner et al. [59] and Zayer et al. [69] attempt to alleviate this problem by fixing a small number of feature points on the shapes to be matched. Sumner et al. rely on these feature points as guidance to ICP in order to escape local minima, whereas Zayer et al. use interpolation, based on barycentric coordinates, of the correspondence between the feature points to compute the remaining point correspondences. In both methods, it is imperative that the feature points selected on the two meshes be corresponding in a meaningful way. This can only be done with user assistance as automatic selection and matching of the feature points is equivalent to the correspondence problem we are trying to solve in the first place.

The second class of techniques describe every point on a shape by encoding shape information from the perspective of that point. Point matching is then based on appropriate distances between the descriptors. Well-known descriptors for images include shape contexts

[4, 5] and spin images [31, 32], both utilizing a histogram obtained by binning the space around a point according to the Euclidean metric and collecting point counts. These methods have subsequently been generalized in a straightforward manner to handle 3D point sets [19, 37].

Neither shape contexts nor spin images are invariant to shape bending. For example, the shapes in Figure 2.2 are similar as they differ only by a bending of the lower part. But the bending transformation disturbs the pair-wise distance configuration between the points and hence the shape context or spin image representation for similar points would vary greatly.

Belongie et al [4, 5] also suggest an iterative solution to improve the robustness of shape contexts. They first find a point-to-point correspondence between two shapes and then transform one shape into another using thin plate splines. Then they iterate between these two steps. However, such iterative solutions are known to converge to local minima if the initial correspondence is not reasonable. Also, designing a transformation model that incorporates non-rigid shape deformations is usually difficult.

In addition, shape context [4, 5] is also not invariant to rotations. To fix this problem for 2D shapes, it is suggested that the shape context for a point be computed with respect to the orientation of the tangent of the contour at that point. This requires a robust estimation of tangents and as we noticed in our experiments, the matching obtained with shape contexts can be problematic even with slight variations in tangent estimation. It is well known that robust estimation of tangent plane and principal directions of curvature over 3D shapes is even more difficult.

Recently, Gatzke et al [23] proposed *curvature map*, a shape descriptor based on sampling a number of points on the shape over the geodesic neighborhood of a point and recording the curvature of the shape at these samples. This type of shape descriptor is again not invariant to non-rigid shape deformations and as suggested by our experiments, recording curvature over sample points is non-robust to noise and moderate bending in a shape. Note that Gatzke et al. only describe their descriptor for 3D shape matching. We have implemented and analyzed a 2D version of it.

Liu et al [42] propose a shape descriptor for 2D contours using the spectral properties of a point's immediate neighborhood, effectively performing a local PCA. The local nature of the descriptor achieves better robustness against non-rigid deformations but the descriptor

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

Figure 2.2: Matching of two hand-drawn "L" shapes. (a) Original shapes; (b) matching using 2D shape contexts [4, 5]; (c) matching using 2D curvature map; (d) matching using our descriptor. Note that we use a simple best matching strategy to find the final matching in all cases, as we wish to compare the qualities of the descriptors only.

becomes less descriptive. Therefore, they rely on the second step of their matching algorithm, which computes the matching by formulating and solving a constrained dynamic programming problem, to compensate for any inadequacy in their shape descriptor.

Indeed, this type of formulation heavily relies on the fact that the given points are ordered along the contour. One problem with such an assumption is that, it is not easily applicable to 3D shapes, as there is no obvious ordering of points over a surface. In fact, having the constraint of order preservation while finding the matching greatly simplifies the problem. Scott et al [53] present an algorithm for solving an "order preserving assignment problem". Using this algorithm in our experiments, we noticed that it gives excellent matchings for all the shape descriptors mentioned so far. This shows that considering the points to be ordered while finding the matching places less emphasis on the quality of the descriptor. Order-preserving matching typically involves expensive optimizations, as in Liu et al. [42] and Scott et al. [53]. Since this is often far too expensive for large point sets, these approaches can only be applied to a small number of feature points along the shape; this would then require robust feature detection.

More recent shape signature based methods for 3D shapes include those by Gelfand et al. [24] and Li et al. [40]. Both methods are robust under only rigid transformations. The partial matching scheme of the former do provide a way to detect articulated subparts of a shape and subsequently match them to the subparts of the second shape. However, their work lacks proper discussion and analysis of the performance and robustness of the matching method when there are multiple pose changes in the models or when the models consist of a large number of articulated subparts.

The third class of techniques, spectral shape correspondence, involves first constructing intrinsic (relative) point representations of the two shapes, in the form of weighted graph adjacency or affinity matrices. Elad and Kimmel [17] make use of geodesic proximities to construct bending-invariant surface signatures through multi-dimensional scaling. Application to object classification has been considered, but they do not solve the harder correspondence problem. Given a proximity matrix, a k-dimensional spectral embedding can be computed via principal component analysis (PCA). Shapiro and Brady [57] use L_2 distances between the embedded points to compute a correspondence, while Umeyema [62] chooses the correlation between the embedding coordinates. Both Caelli and Kosinov [9] and Carcassoni and Hancock [10] rely on spectral clustering and cluster correspondence to guide point correspondence. The use of spectral embeddings has traditionally been exploited in the computer vision and machine learning literature. Recently, they have found several applications in geometry processing as well, including mesh segmentation [42], spherical parameterization [26], and surface reconstruction [36].

Common to all the existing spectral correspondence techniques is the premise that the eigenmodes from two similar shapes should match up, according to the magnitude of their corresponding eigenvalues. One of our main observations is that this ordering of the eigenmodes is not always reliable. As the eigenvalues characterize data variance in the direction of the corresponding eigenvectors, eigenmode ordering based on eigenvalues implies ordering by data variance. This may not be appropriate since variance only captures global information and does not reflect the way specific data points would vary. We observe that under shape stretching, certain eigenmodes may be "switched". Failure to resolve such reflections or other non-rigid discrepancies between spectral embeddings will lead to poor matching results. In this thesis, we propose heuristics to handle such switchings in the spectral domain, where the spectral embeddings will be corresponded after alignment using non-rigid ICP based on thin-plate splines.

There are other ways of finding a correspondence without using shape descriptors. Sederberg et al [56] and Zhang [72] present two similar physically based approaches for correspondence of 2D polygons which fit one polygon over another by optimizing certain energy functional and then find a correspondence. Such global optimization is rather expensive. Hence, the method cannot be used for applications such as shape searching, where a shape has to be matched with many other shapes.

2.2 Shape Retrieval

It is quite conceivable that a great deal of prior knowledge is incorporated into the process of human object recognition and classification, perhaps with subpart matching [28, 21] playing an important role. In this thesis however, we focus on purely shape-based approaches. Note that all correspondence methods described in Section 2.1 can be used for shape retrieval, as a similarity measure between two shapes can be defined using the correspondence obtained from these methods. In fact, for 2D shape retrieval, such correspondence based similarity measure is widely used. However, for 3D shapes, global shape descriptors [61] based methods are more popular and a review of such techniques would be the focus of this Section. At a high level, a 3D shape retrieval algorithm either works on the 3D models directly, e.g., [34, 47, 60], or relies on a set of projected images [13, 15] taken from different views. Let us call these the object-space and the image-space approaches, respectively. The latter, e.g., the Light Field Descriptors (LFD) of Chen et al. [13], has a more intuitive appeal to visual perception and thus often result in better benchmark results for retrieval [48], but at the expense of much higher computational cost.

Many of the object-space shape descriptors construct one or a collection of spherical functions, capturing the geometric information contained in a 3D shape in an *extrinsic* manner [48]. These spherical functions represent the distribution of one or more quantities, e.g., distance from points on the shape to the center of mass [65], curvatures [58], areas [3], surface normals [33], etc. The bins are typically parameterized by the sphere radius and angles. The spherical functions are, in most cases, efficient to compute and robust to geometric and topological noise, but they may be sensitive to the choice of sphere center or the bin structures. To align the bins for two shapes properly, these approaches require pre-normalization with respect to translation, rotation, and uniform, e.g., [65, 3, 33], or nonuniform scaling [35]. As an alternative, rotation-invariant measures computed from the spherical functions, e.g., energy norm at various spherical harmonic frequencies [34], can be utilized. However, non-rigid transformations are not handled by these approaches.

As a salient intrinsic geometric measure, surface curvature, as well as the principal curvature directions, have been used for shape characterization and retrieval [68, 58]. These approaches are sensitive to noise (unless smoothing is applied in pre-processing) and non-rigid transformations such as bending. Another intrinsic approach is the use of shape distributions [47], where a histogram of pairwise distances between the vertices of a mesh

defines the shape descriptor. Other form of statistics, e.g. [46], can also be used and bending-invariance can obviously be achieved if geodesic distances are used in this context, but the discriminative power of the histograms is suspect.

The most common approach to handling articulated shapes is via skeletal or other graph characterization of the shapes, e.g., [27, 71], and then apply graph matching [9]. The cost of extracting the skeletons can be high, e.g., when medial-axes are used [71], and the subsequent graph matching is often computationally expensive and the shape descriptor itself is sensitive to topological noise. Our approach also uses a graph-based intrinsic characterization of the shape structures. The spectral embeddings automatically normalize the shapes with respect to rigid-body transformations, uniform scaling, and bending, and they are fast to compute. The resulting shape descriptors provide a more intuitive way of characterizing shapes, compared to shape distribution [47]. In addition, the spectral approach is quite flexible and allows for different choices of graph edge weights and distance computations, which can render the approach more robust against topological noise. The effects of these choices on shape retrieval is studied.

The idea of using spectral embeddings for data analysis is not new and clustering [45, 70] and correspondence analysis [11, 57, 30] are the main applications. Past work that is most relevant to ours is the use of bending-invariant shape signatures by Elad and Kimmel [17]. They work on manifold meshes and compute spectral embeddings using multidimensional scaling (MDS) based on geodesic distances. A more efficient version of MDS is adopted to approximate the true embeddings; this is different from Nyström approximation which we have adopted in this work. They only tested shape retrieval on manifold, isometric shapes, e.g., models obtained by bending a small set of seed shapes. In practice, many 3D shapes are neither manifolds nor isometric to each other, thus a more robust approach, based on more general graphs and distance measures, and a more complete experiment, are called for.

Chapter 3

2D Shape Correspondence and Retrieval

In this Chapter, we explain in detail, our shape descriptor for 2D shapes. The shape descriptor is used for computing correspondence and the correspondence is used for computing similarity measure between two shapes for the purpose of retrieval. We now explain the methods for shape correspondence and retrieval separately.

3.1 Shape Correspondence Using Geodesic Shape Contexts

3.1.1 Overview

Let us first briefly review the given problem, identify goals and outline our method. Given two contours in 2D (approximated by polygons), the problem is to compute a matching between the vertices of the contours. We now design the shape descriptor to be used for computing this matching. While designing the shape descriptor for 2D contours, we consider the following goals:

1. The descriptor should be invariant to ordinary geometric transformations such as rotation, translation and uniform scaling.
2. It should be robust against shape noise.
3. It should be robust to non-rigid deformations; especially shape bending.

4. It should capture the shape information sufficiently, so that the correspondence obtained from a (fast) best matching procedure is adequate. In this way, expensive bipartite-matching (as in Belongie et al. [4, 5]) or dynamic programming formulations (as in Liu et al. [42]) for matching the resulting shape descriptors, can be avoided.

The inspiration for our work is the shape context of Belongie et al [4, 5]. The 2D shape context for a point is constructed by first dividing the plane into concentric circular bins and then collecting the number of points in each bin as the shape descriptor. The reader is referred to [4] and [5] for details regarding shape contexts. We have noticed in our experiments that shape contexts are as such quite robust. However, they fail when confronted with shapes having non-rigid deformations, such as bending. As we are aiming for robustness to bending, it is intuitive to ask the question: what property of a shape remains unchanged when sub-parts of the shape are bent. It is easy to see that the geodesic distance (distance along the contour) between two points on the shape remains constant when shapes undergo bending transformations. Hence, a natural generalization of shape contexts to include bending transformations would be to construct geodesic bins instead of bins according to Euclidean distances.

We perform a geodesic traversal along the shape and collect geometric information, hence the name *geodesic shape contexts*. The information that we collect is the average curvature inside these geodesically formed bins. We refer to this curvature information as the basic geodesic shape context for a point. We then refine this basic descriptor using appropriate techniques to achieve robustness against non-rigid deformation.

3.1.2 Geodesic Shape Descriptor

We now explain in details the construction of the geodesic shape context for every point of the contours and describe the matching procedure.

Basic Geodesic Shape Context

Given two contours S_1 and S_2, we first construct the basic geodesic shape context for every point on both contours. For a point P on a contour, the geodesic shape context is constructed as follows:

1. Divide the geodesic neighborhood of P into bins of non-decreasing lengths ending at point Q furthest away from P along the contour, as shown in Figure 3.1.

2. For each bin, find the average of the curvatures of all the points lying inside the bin.

3. If C_i is the average curvature of the i^{th} bin, then the basic geodesic shape context of the point is defined as the vector $\{C_1, C_2, C_3, \ldots C_n\}$, where n is the number of bins.

Instead of increasing the bin size strictly exponentially, as in the original shape context, we repeat bins of equal size to capture local information better, e.g., in Figure 3.1.

If $\{P_1, P_2, P_3, \ldots\}$ are points along the contour, we calculate the curvature at a given point P_i by fitting a cubic parametric curve $(x(t), y(t))$ to the set of points

$$(P_{i-w}, P_{i-w+1}, \ldots P_i, \ldots P_{i+w-1}, P_{i+w})$$

for some neighborhood size w. Typically, we choose $3 \leq w \leq 5$. The curvature at P_i is then given by:

$$C_{P_i} = \frac{||X'(t) \times X''(t)||}{||X'(t)||^3}, \text{ where } X(t) = \begin{bmatrix} x(t) \\ \\ y(t) \end{bmatrix}.$$

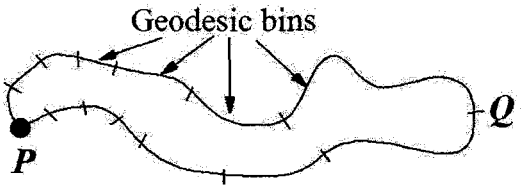


Figure 3.1: The basic geodesic shape context for point P is the vector $\{C_1, C_2, C_3, \ldots C_n\}$ where n is the number of bins and C_i is the average curvature of the points in the i^{th} bin.

Once the descriptor is formed for every point on the two shapes, we can directly compare a point P on S_1 to a point Q on S_2 with the Euclidean metric:

$$DisSim(P, Q) = \sum_{i=1}^{n} ||C_P(i) - C_Q(i)||^2, \tag{3.1}$$

where, $C_P(i)$ and $C_Q(i)$ are the average curvatures of the i^{th} bin for P and Q, respectively.

A match for point P from all points in S_2 is the point which has the minimum dissimilarity with P. In other words:

$$match(P) = \text{argmin}_{Q \in S_2}(DisSim(P, Q)). \tag{3.2}$$

The averaging scheme we use, in contrast to curvature map [23], makes the descriptor robust against surface noise and irregularities. This can be explained from a signal processing perspective. Consider the curvature plot for a contour as a signal. Curvature map samples this signal at regular intervals. It is easy to see that if the samples are not selected carefully, it is quite probable that some features of the signal are not sampled. Taking averages, on the other hand, is equivalent to first convolving this signal with a box filter and then recording samples at regular intervals on the convolved signal. A sample on the convolved signal contains information about its neighborhood which makes the sampling more feature sensitive. At the same time, box filtering also achieves noise removal.

Hard and hysteresis thresholding

Although the averaging scheme renders our descriptor robust against noise and small shape deformations, it does not handle more drastic bending in the shape well, as the curvature of corner vertices can drastically change with bending. To handle this, we propose thresholding of curvatures. Specifically, we give more weight to negative curvatures, corresponding to concavity in a shape, and attenuate the positive curvatures.

The intuition behind this is based on human perception. It is generally believed that humans match shapes based on the similarity of their parts. We simulate this by placing more emphasis on concavities following the "Minima Rule" [28], which stipulates that part boundaries are found at negative curvature minima. This can be accomplished by the following simple thresholding scheme: if C is the curvature at a point, the thresholded curvature is given by:

$$\hat{C} = \begin{cases} C/C', & \text{if } C \geq 0 \\ -(1 - e^{C}), & \text{if } C < 0, \end{cases} \tag{3.3}$$

where, C' is some large constant we select.

However, our experiments suggested that such a hard thresholding can be rather sensitive to slight variations in the shape. We therefore suggest to use a *hysteresis* thresholding scheme instead, which is commonly used for edge detection in image processing [29]. For all our experiments, we have used the following thresholding scheme: if C_i is the curvature at point P_i, $i = 1, \ldots, n$, the thresholded curvature is given by:

$$\hat{C}_i = \begin{cases} C_i/C', & \text{if } C_i \geq 0 \\ -(2 - e^{C_i}), & \text{if } C_i < \tau. \\ -(2 - e^{C_i}), & \text{if } \tau < C_i < 0 \text{ and } C_{i-1}, C_{i+1} < 0 \\ C_i, & \text{otherwise,} \end{cases} \tag{3.4}$$

where, $\tau < 0$ is the threshold for negative curvature.

As before, we attenuate all the positive curvature values. For negative curvatures, in the hard thresholding scheme before, we were amplifying any curvature that is less than zero, which can be sensitive to minor shape variations and noise. In the new scheme, we amplify any negative curvature which is less than the threshold τ. These are the curvatures that we consider as shape features. But for negative curvatures greater than the threshold τ, we only amplify them if we find that they indeed define a feature. We decide whether a negative curvature represents noise or feature by looking at the curvature of its neighbors where the neighborhood size can be increased when appropriate. This soft thresholding generally works well since it uses selective amplification and hence, does not amplify noise. We found from our experiments that setting C' to be the maximum curvature along the contour and τ to be a quarter of the average of all the negative curvatures along the contour generally works well for our correspondence tasks.

To further explain the need for thresholding, consider the shapes given in Figure 3.2. It is evident that the red contour is similar to the blue except that the top portion is bent downwards. Also shown are the plots of the original curvature and their thresholded values. As is clear from the curvature plots, the concavities on both the shapes match well with the thresholded curvatures than with the original curvatures. The latter can be thought of as containing detailed shape information which is non-robust to various shape transformations. This information is conveniently filtered out by our thresholding.

3.1.3 Guiding Heuristic Using Proximity

We can obtain the matching using equation 3.2 defined earlier, just based on the geodesic shape context descriptor. However, due to the global nature of our descriptor, it can be hard to distinguish between nearby points on a shape and points that have similar shape descriptors. To guide the matching process so as to obtain a consistent (with respect to the linear ordering of the contours) matching we propose a proximity heuristic. Note that

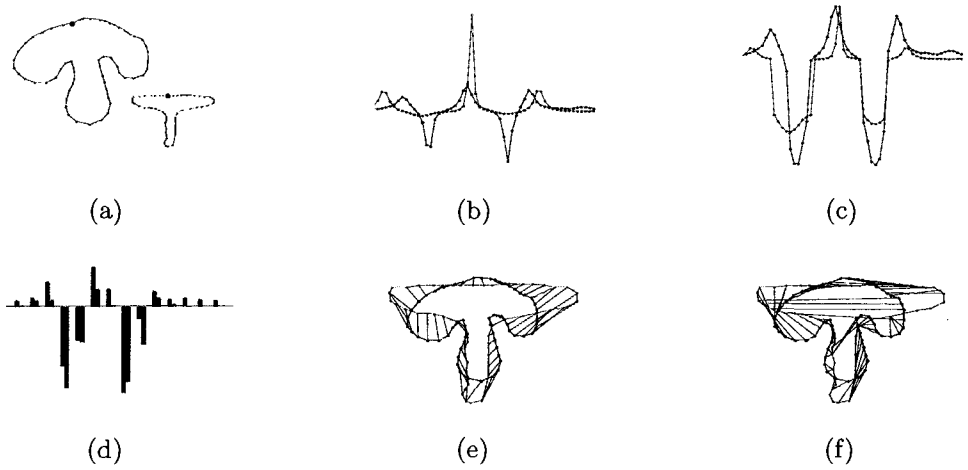


Figure 3.2: (a) Two hand drawn mushroom shapes. (b) Curvature plots starting from the black point along the contour in counter-clockwise direction. (c) Thresholded curvatures. (d) Thresholded curvatures averaged over geodesic bins. (e) Matching obtained from our shape descriptor. (f) Matching obtained from mapping of unthresholded curvatures.

this heuristic is independent of the linear ordering of the contour points and hence is extendable to 3D. In fact we would use the same heuristic in later Chapters to obtain better correspondence for 3D shapes. As our shape descriptor is robust and descriptive, using this simple proximity heuristic already leads to excellent results. We now describe the proximity heuristic.

The proximity guidance scheme relies on one or more "anchor" points, chosen on both shapes, that are well matched by similarity between point shape descriptors alone. The matching of any remaining points is based on both descriptor similarity and the point's proximity to the anchor points. In our experiments, we select only two anchor points per shape. The first point is chosen to be the best matched pair, $a_1^{(1)}$ and $a_2^{(1)}$, on the two shapes using only descriptor similarity. To choose the second point, we add a penalty for choosing a point near the first point. This requirement is aimed at improving the accuracy of the heuristic, since, if the anchor points are selected close to each other, their impact on nearby points would be much less than that on geodesically far points.

Let A_S denote the dissimilarity matrix where the ij^{th} entry is the dissimilarity between

the i^{th} point in S_1 and j^{th} point in S_2. That is,

$$A_S(i,j) = DisSim(i,j).$$

where, $DisSim(i,j)$ is defined in equation 3.1 Let $A_S(a_1^{(1)}, a_2^{(1)})$ be the minimum entry of A_S, where $a_1^{(1)}$ and $a_2^{(1)}$ are points of S_1 and S_2 respectively. $a_1^{(1)}$ and $a_2^{(1)}$ are the first anchor points for the two shapes. We then find the geodesic distance of all points in S_1 from point $a_1^{(1)}$ and the geodesic distance of all points of S_2 from the point $a_2^{(1)}$. Let $dist_1^{(1)}(i)$ be the geodesic distance of i^{th} point of S_1 from $a_1^{(1)}$, and $dist_2^{(1)}(j)$ be the geodesic distance of j^{th} point of S_2 from $a_2^{(1)}$. We define a distance-dissimilarity matrix A_{SD}^1 as follows:

$$A_{SD}^1(i,j) = ||dist_1^{(1)}(i) - dist_2^{(1)}(j)||^2.$$

Now we construct a new dissimilarity matrix for choosing the second anchor. As we would like to prevent the second anchor from being too close to the first one, the new dissimilarity matrix is defined as:

$$A_S^{new}(i,j) = A_S(i,j) - \frac{1}{2}[dist_1^{(1)}(i) + dist_2^{(1)}(j)].$$

The second term on the right hand side is the penalty so that the next best match found is generally not close to the first anchor. We appropriately scale both the terms so that they are of the same order of magnitude. Let $A_S^{new}(a_1^{(2)}, a_2^{(2)})$ be the minimum entry of A_S^{new}. $a_1^{(2)}$ and $a_2^{(2)}$ are the second anchor points for both shapes. Again, if $dist_1^{(2)}(i)$ is the geodesic distance of i^{th} point of S_1 from $a_1^{(2)}$ and $dist_2^{(2)}(j)$ is the geodesic distances of j^{th} point of S_2 from $a_2^{(2)}$ we define the second distance-dissimilarity matrix as:

$$A_{SD}^2(i,j) = ||dist_1^{(2)}(i) - dist_2^{(2)}(j)||^2.$$

Finally, we redefine the dissimilarity between two points P and Q as:

$$DisSim^{new}(P,Q) = A_S(P,Q) + c_1\ A_{SD}^1(P,Q) + c_2\ A_{SD}^2(P,Q), \tag{3.5}$$

where c_1 and c_2 are free parameters which can be set by the user.

Subsequently, the correspondence in equation 3.2 is now redefined as:

$$match(P) = \text{argmin}_{Q \in S_2}(DisSim^{new}(P,Q)). \tag{3.6}$$

This heuristic can be extended to fixing more matching pairs. However, it should be noted that fixing more matching pairs will make the process non-robust against stretching since this heuristic depends completely on geodesic distances which change drastically with stretching. Hence, we restrict ourselves to using only two anchor points. Simply stated, the proximity heuristic is a method to define dissimilarity between two points. The heuristic not only takes into account, the dissimilarity between the descriptors of the points, but also considers the dissimilarity between the proximity of the points from the anchor points.

3.2 Shape Retrieval

We have already described the process of finding a correspondence between two 2D contours S_1 and S_2. Now we use this correspondence to define a similarity measure between the two. This similarity measure will be subsequently used for shape retrieval. The similarity cost between S_1 and S_2 is given by:

$$SimCost(S_1, S_2) = \sum_{P \in S_1} DisSim(P, match(P)) \tag{3.7}$$

where, $DisSim()$ and $match()$ are defined in equations 3.5 and 3.6.

Note that the similarity cost is a cost function, hence, lower the similarity cost between two shapes, more similar the shapes.

3.3 Experimental Results

For the purpose of evaluating our shape descriptor, we report its performance when applied to shape retrieval on two shape databases, and shape correspondence of various contours. For shape retrieval experiments, we have used the Brown database [12] of 95 binary images and Ling et al's [41] articulated shapes database. The later consists of 40 images from 8 different objects. Each object has 5 images articulated to different degrees. Both the databases are illustrated in Figure 3.3. Note that for all our experiments we first extract the contour of the image and then apply our method on this contour. Also, for shapes in Ling et al's [41] database, only the outer contour was extracted.

Figure 3.4 shows some results obtained from our shape searching experiments. The first column in the Figure shows the shape that was used as a query to the database. The next five columns show the top five matches returned from our searching algorithm. Here, the

query shape itself has been excluded from the set of shapes for retrieval. The similarity measure used for this experiment has already been defined in the previous Section. Note the performance of our method on shapes from the articulated shapes database.

Figure 3.7(a) and 3.7(b) show an image representation of the dissimilarity matrices between the images of Brown database and Ling et al.'s database respectively. The ij^{th} entry of the matrix is the dissimilarity between image i and image j. A dark spot represents less dissimilarity and a bright spot represents greater dissimilarity. The images were ordered so that similar images appear together in the matrix. The block diagonal structure of both the matrices show that the dissimilarity, obtained using our shape description, between similar images is low.

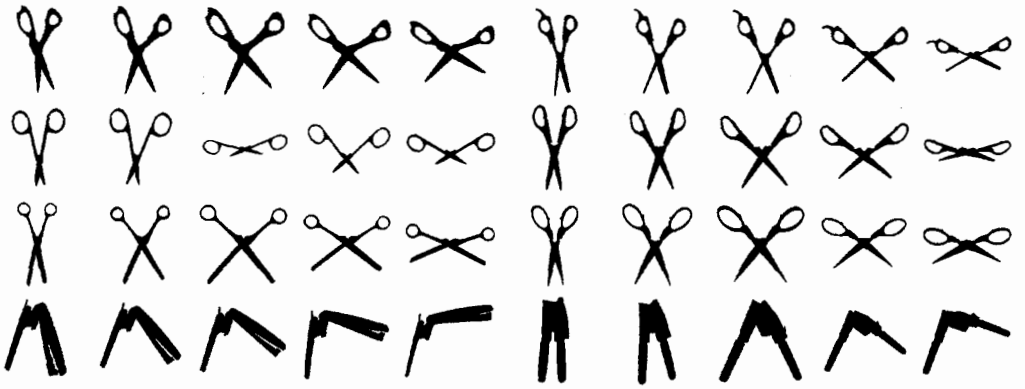
Next we evaluate our descriptor when applied to shape correspondence. Given two contours, we use our shape descriptor to find a point-to-point matching. Matching obtained for different pairs of contours is shown in Figure 3.5 and Figure 3.6. We show matching points by drawing a black line between the two. The first column shows the shapes to be matched. The second and the third columns show the matching obtained from shape contexts [4, 5] and 2D curvature maps respectively. The fourth column shows matching obtained from the our descriptor.

It can be easily seen from these experiments that the shape context [4, 5] descriptor fails to retrieve proper matching when the two shapes differ by some level of bending. For example, the arms and legs of the human shapes in 3.5(b) and 3.6(b) are poorly matched although the torso is matched fairly well. Note that our descriptor is robust against such shape variations, as well as moderate stretching deformations (e.g., Figure 3.6(a)). Figure 3.5(c) and 3.5(d) are contours of images taken from Ling et al's database which have a great deal of articulation. However, our shape descriptor still performs a good job of retrieving the right correspondence. These examples demonstrate that our shape descriptor contains more shape information than shape contexts.

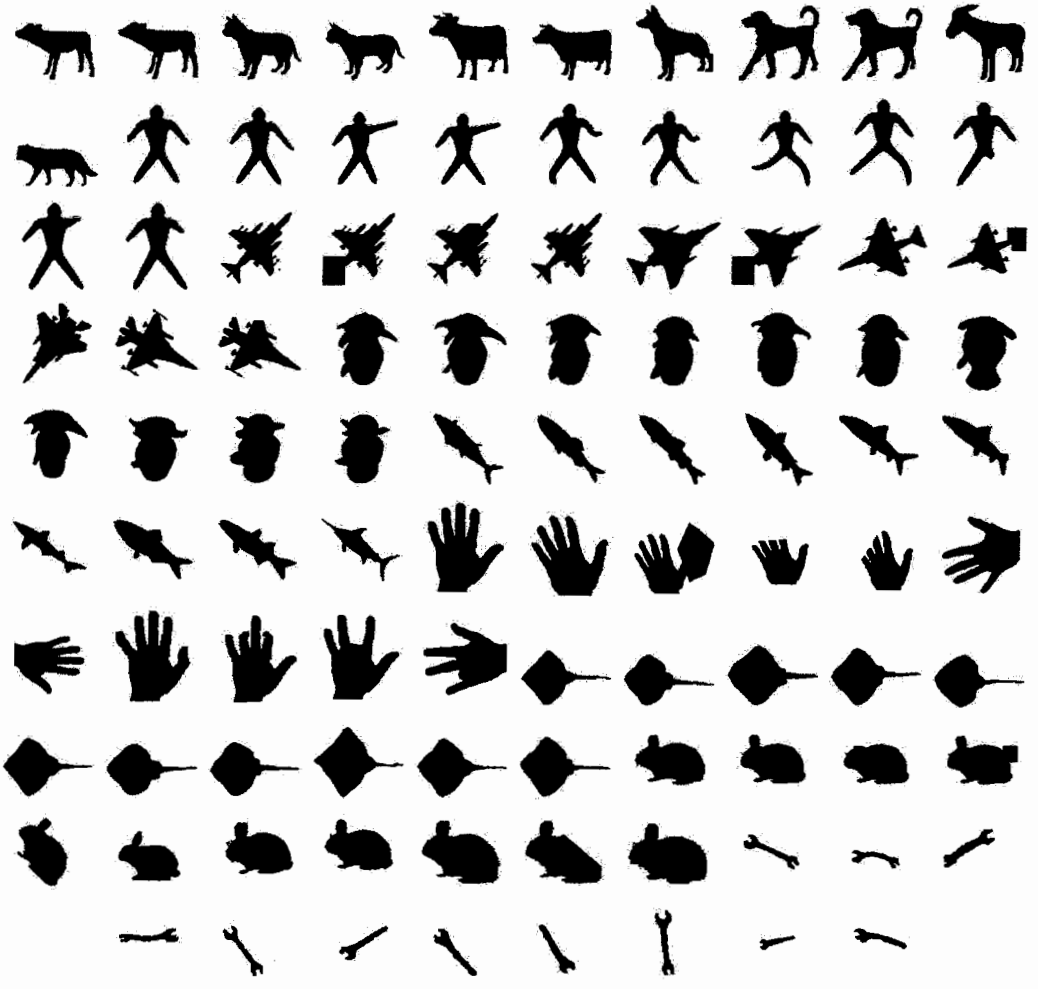
For all the examples shown in this paper, after computing the shape descriptor for every point, the point-to-point matching is computed by a simple best match strategy, that is, point $P \in S_1$ is matched with $Q \in S_2$ if $DisSim(P,Q) = min_{i \in S_2}(DisSim(P,i))$. No extra measures are being taken to force the correspondence to be one-to-one although, as suggested by Figure 3.5, our descriptor performs better in terms of recovering a one-to-one correspondence.

3.4 Limitations

The main limitation of our current method is that it can be applied only to shapes defined as contours. In other cases, especially for computer vision applications, the problem is typically to correspond two point sets, where the points are features marked on two images with no particular shape defined by them. Since our method is based on the calculation of curvature and depends on the proper specification of the geodesic neighborhood of a point, it cannot be applied in such situations. Nevertheless, the focus of our work is on articulated shapes which can be specified in the form of contours. Our method is also inapplicable when the contour can not be appropriately specified, for example, on folded or intersecting shapes. Another limitation of our method is that its extension to 3D shapes is unclear. As the method depends on an ordering of bins in the geodesic neighborhood of a point, it cannot be easily extended to 3D since this ordering is not obvious on the geodesic neighborhood of a point on a 3D surface. One way to define this ordering canonically is by using reference axes, for example, principal directions of curvature at the point. However, calculation of properties such as curvature is always much more difficult and non-robust in the case of 3D shapes relative to 2D. Since the robustness of correspondence depends on the robustness of the calculation of the descriptor, we suspect that descriptor based approaches are not suitable for application to 3D shapes especially when the descriptor is based on physical properties of the shape, simply because of the non-robustness involved in the computation of these properties. This is precisely the reason why we choose to use spectral methods for 3D shape correspondence, presented in the next Chapter, instead of descriptor based methods.



(a) Articulated shape database of Ling et al. [41].



(b) Brown database [12].

Figure 3.3: Image databases used for experiments.

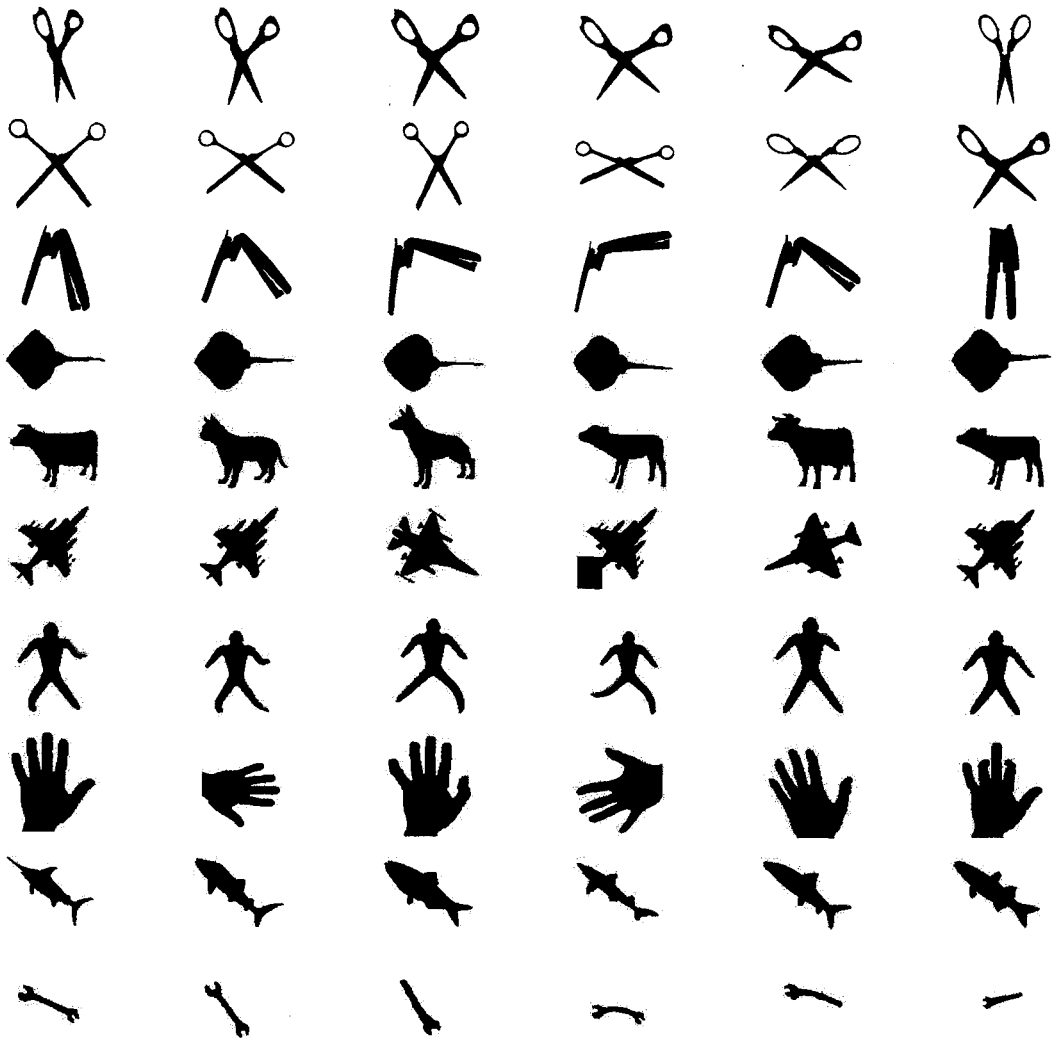


Figure 3.4: Shape searching results: First column shows the input query shapes. The next five columns show the top five matches from the database.

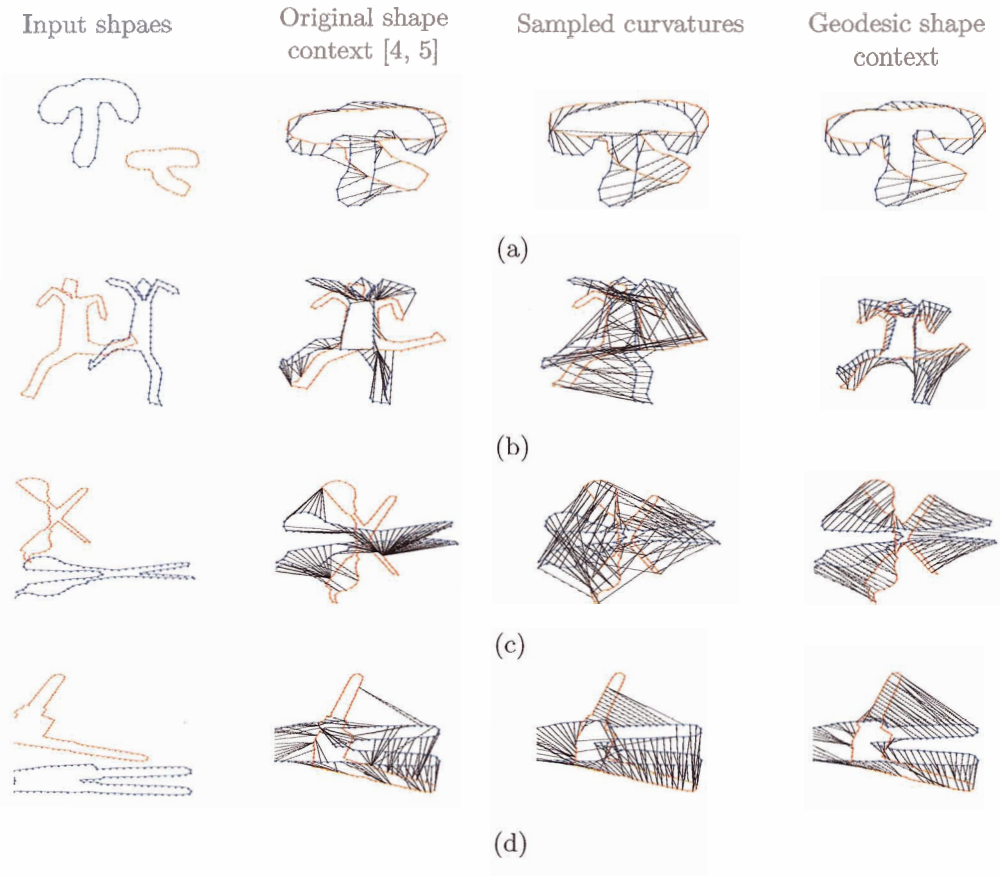


Figure 3.5: Comparison between point correspondence results, where corresponding points are linked by a line segment in black. The leftmost column shows the two shapes to be matched. Then from left to right, correspondence results using the original shape context [4, 5], results using sampled curvatures and results using our descriptor. As we can see, the original shape context performs surprisingly well on the human figures with a great deal of bending, except for a few bad mismatches at the arms and legs; they are not as robust on the "mushroom" shapes. Sampling curvatures, even with robust curvature estimation, does not work well. Our shape descriptor, on the other hand, exhibits a high degree of robustness throughout our experiments.

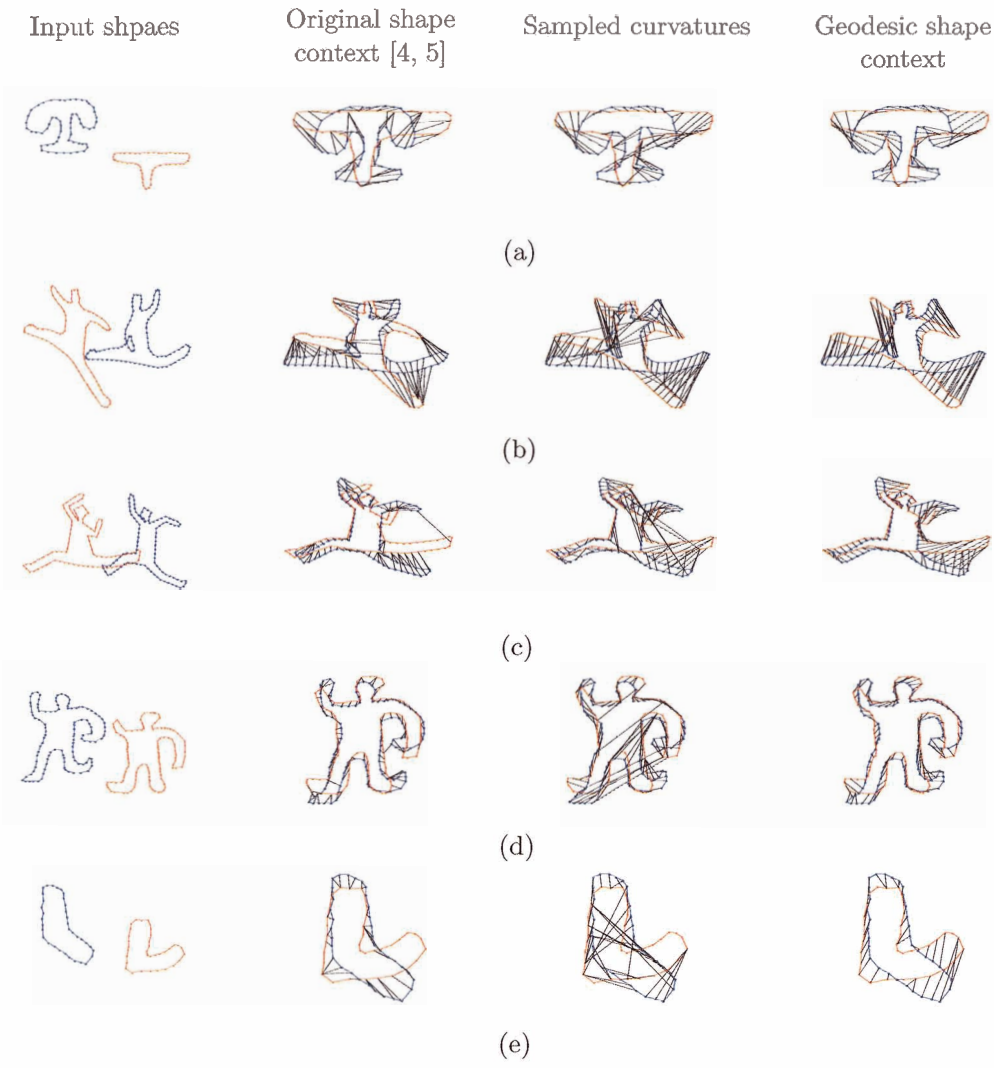
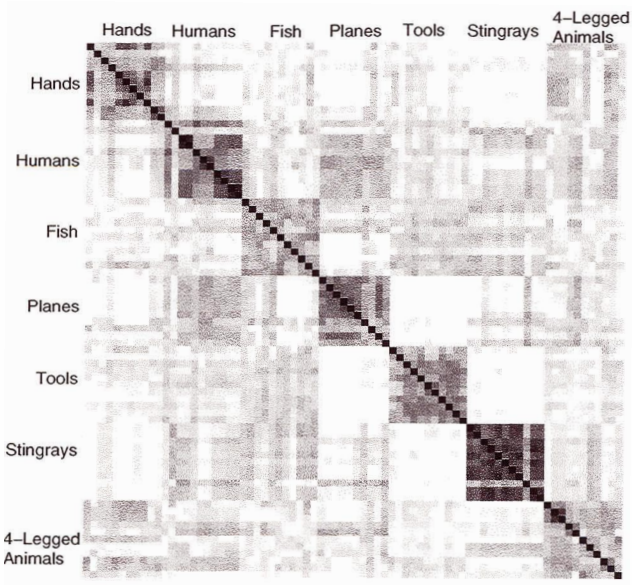
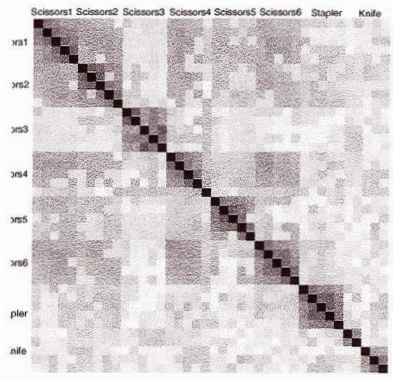


Figure 3.6: More point correspondence results. Corresponding points are linked by a line segment in black. The leftmost column shows the two shapes to be matched. Then from left to right, correspondence results using the original shape context [4, 5], results using sampled curvatures and results using our descriptor.



(a) Dissimilarity matrix for Brown database.



(b) Dissimilarity matrix for Ling et al's database.

Figure 3.7: Image plot of the dissimilarity matrix for the Brown shape database and Ling et al's articulated shape database, obtained using our shape descriptor as a means for shape retrieval. Note that darker spots represent less dissimilarity, hence a better match.

Chapter 4

3D Shape Correspondence and Retrieval

In this Chapter, we explain in details, our method for correspondence and retrieval of 3D shapes. We apply eigenanalysis to the input 3D shapes and use the resulting spectral embeddings for correspondence and retrieval. We first give an overview of the spectral method, then explain the construction and properties of the spectral embeddings. Next we present the application of spectral embeddings to correspondence and retrieval.

4.1 Overview

Let us first give a brief overview of the problem we address and the algorithm we propose. The correspondence problem can be stated as: given two 3D shapes M_1 and M_2, in the form of triangular meshes with n_1 and n_2 vertices, respectively, we wish to compute a correspondence C between the two sets of vertices in M_1 and M_2. That is, $C(i)$ is the vertex in M_2 that best corresponds to vertex i in M_1. Note that the correspondence computed is not required to be bijective. However, in the case where $n_1 = n_2$ and a one-to-one correspondence is sought, we can easily modify our method to meet the goal. The retrieval problem, on the other hand, reduces to finding a similarity measure between two meshes.

The major steps of our approach are as follows: first, we establish an $n_1 \times n_1$ affinity matrix A where A_{ij} is the affinity between vertices i and j of M_1. Similarly, we compute an $n_2 \times n_2$ matrix B, the affinity matrix for M_2. The affinities that we use in our implementation

are based on *geodesic distances* in order to attain invariance to bending. The geodesic distance between any two points on a surface is defined as the minimal length of all surface curves between these points. In other words, it is the shortest distance (along the surface) between these points. Next we find the spectral embeddings $\hat{A}_k$ and $\hat{B}_k$ of the matrices A and B, respectively. These embeddings give k-dimensional coordinates of all the vertices of M_1 and M_2. Hence, we have essentially transformed 3D mesh vertices into points in k-D space. The embeddings are based on the eigenvectors of A and B, properly processed as we describe in Section 4.2. The purpose of transforming the 3D mesh from the spatial domain to the k-D spectral domain is to attain invariance to bending, rigid transformations, and uniform scaling, as well as robustness to difference in mesh sizes, for example.

Once we have the two k-D meshes that are already properly normalized, we use these embeddings to compute correspondence and retrieval. For correspondence, we use iterative alignment to robustly align them and then obtain a correspondence via best matching based on the L_2 distance. As we shall explain in Section 4.2, rigid alignment is insufficient to deal with moderate stretchings in the shapes to be matched. Hence, we modify the well-known iterative closest point (ICP) algorithm to include non-rigid transformations. Specifically, we use thin plate splines to model non-rigid transformations and compute a registration between the meshes in the spectral domain. As mentioned earlier, this registration is not required to be one-to-one, but can be forced to be bijective by using the Hungarian algorithm for bipartite matching. For retrieval, we compute global descriptors on the k-D embeddings and use the distance between the descriptors as the similarity measure between the shapes.

4.2 Spectral Embedding

In general, intrinsic point representations can be obtained via pairwise point proximities, specified by a symmetric *affinity matrix* $A = \{a_{ij}\}$, where $a_{ij} \geq 0$ characterizes the similarity or simply the graph adjacency [9, 62] between points i and j. One may view the affinity matrix A as a data vector whose n columns (or rows) represent n-dimensional data points.

The most common proximity measure used to define relationship between points for shape matching is the Euclidean distance [11, 10, 54, 57], which implies invariance to rotation and translation. For mesh correspondence and retrieval, we use geodesic distances between the mesh vertices, computed via fast marching [17], to include invariance to bending as well. Invariance to uniform scaling is achieved by mapping the geodesic proximities into the

interval $[0, 1]$ using a *scale-dependent, positive semi-definite* kernel function. In this thesis, we use Gaussian kernels which is a common choice for spectral correspondence [11, 10, 54, 57].

Although the point proximities contain a great deal of shape information, without a proper point mapping, one cannot compare such representations for two data sets directly. Also, the size of the data sets, or the dimensionality of the point representations, may not be the same. Last but not the least, the high dimensional representations may contain a great deal of redundancy, resulting in unnecessarily high computational cost. These observations naturally lead us to consider transforming two data sets, respectively, into some information-preserving subspaces that share the same low dimensionality. This can be accomplished through principal component analysis (PCA) on the affinity data.

4.2.1 Principal Component Analysis (PCA)

Given the data (affinity) matrix $A \in \mathbf{R}^{n \times n}$, we first compute its principal components $\mathbf{u_1}, \ldots, \mathbf{u_n}$, which are the *normalized* eigenvectors of the autocorrelation matrix $R = AA^T$. Since A is symmetric, $R = A^2$ and $\mathbf{u_1}, \ldots, \mathbf{u_n}$ are simply the eigenvectors of the affinity matrix A. Let $\lambda_1, \ldots, \lambda_n$ be the corresponding eigenvalues of A and suppose that $\lambda_1 \geq \ldots \geq \lambda_n$. Projecting the data matrix onto the first k principal components yield

$$\hat{A}_k = A^T U_k = U_k \Lambda_k^T, \tag{4.1}$$

where $U_k = [\mathbf{u_1}|\ldots|\mathbf{u_k}]$, $\Lambda_k = diag(\lambda_1, \ldots, \lambda_k)$, and the columns of $\hat{A}_k$ represent a k-dimensional *spectral embedding* of the data points. Note that a point permutation induces the same permutation of the embedded coordinates but leaves the spectrum invariant.

In the case of mesh spectral embeddings, the data points are mesh vertices. A spectral embedding associates with each mesh vertex a k-dimensional coordinate. In the 3D spectral domain, one can visualize the embedding of a mesh M by rendering a mesh whose connectivity is the same as M and whose vertices are given by the embedding coordinates, as shown in Figure 4.1.

Although $\hat{A}_k$ gives a provably best k-dimensional approximation of A (in terms of the Frobenius norm) [16], it may not be suitable for matching. The more important requirement is for the projection axes, derived from the principal components, to be compatible between two data sets.

Figure 4.1: A human mesh (left) and its 3D spectral embedding, constructed using the second, third, and fourth eigenvectors. This particular choice of the eigenvectors is explained in Section 4.3.

4.2.2 Eigenvalue Scaling

Given two affinity matrices A and B characterizing two shapes, possibly in different scales, a scale-dependent kernel can normalize the affinity values in A and B. If the number of vertices, n_A and n_B, in the two (mesh) shapes differ however, we first need to truncate both spectral embeddings to the same dimension $k \leq \min\{n_A, n_B\}$. In addition, since we normalize each eigenvector, the cardinality of a data set affects the magnitude of the entries in its eigenvectors, which in turn affects the embeddings.

Correspondence algorithms that use unscaled eigenmodes [11, 10, 57] as spectral embeddings are common. Shapiro and Brady [57] first suggest a scaling of the the eigenmodes by eigenvalues, as in equation (4.1), but did not elaborate. Caelli and Kosinov [9] scale the eigenmodes using *squared* eigenvalues and then project the resulting embeddings onto the unit k-sphere for matching graphs of different vertex counts. Evidently, proper scaling of the eigenmodes is a crucial normalization step. None of these approaches adequately resolves the discrepancies in the scales of the principal components (or embeddings) due to difference in the cardinality of the data sets. We propose to scale the principal components by the *square root* of the eigenvalues, yielding projections

$$\hat{A}_k = U_k \Lambda_k^{\frac{1}{2}} \qquad \text{and} \qquad \hat{B}_k = V_k \Gamma_k^{\frac{1}{2}}, \tag{4.2}$$

where $A = U\Lambda U^T$ and $B = V\Gamma V^T$ are the eigenvalue decompositions of A and B, respectively, with $U_k, V_k, \Lambda_k, \Gamma_k$ defined as in equation (4.1). Spectral embeddings of this form are

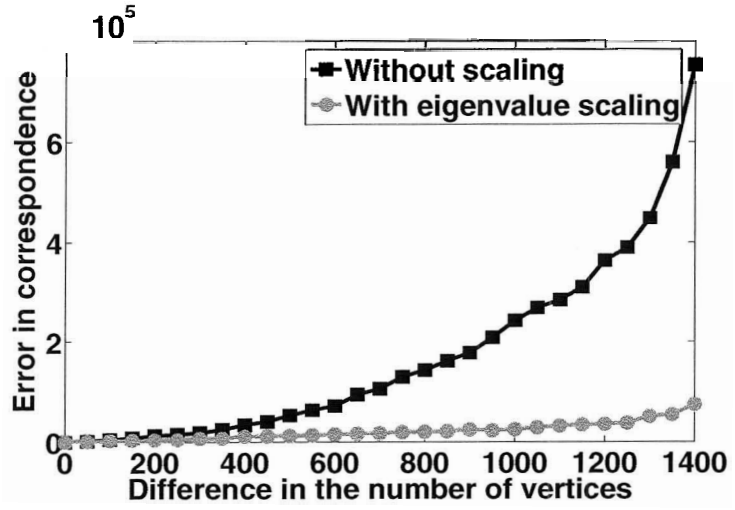


Figure 4.2: Effect of eigenvalue scaling on correspondence, based on the correspondence error plots.

well known in the spectral clustering literature [45].

Justifications: Consider the vector of projections $\hat{\mathbf{a}}_i$ from set A. We can estimate the scale of these projections by $s_{A,i} = ||\hat{\mathbf{a}}_i||^2/n_A$. From equation (4.2), we have

$$\hat{\mathbf{a}}_i = \sqrt{\lambda_i}\mathbf{u_i} \text{ and } \hat{\mathbf{b}}_\mathbf{i} = \sqrt{\gamma_\mathbf{i}}\mathbf{v_i}.$$

It follows that $s_{A,i} = \lambda_i/n_A$ and $s_{B,i} = \gamma_i/n_B$. With the affinity matrices having unit diagonal elements, signaling that a point has maximal affinity to itself, we have

$$s_{A,i} = \frac{\lambda_i}{trace(A)} = \frac{\lambda_i}{\sum_{j=1}^{n_A}\lambda_j} \text{ and } s_{B,i} = \frac{\gamma_i}{\sum_{j=1}^{n_B}\gamma_j}.$$

We do not normalize these scales to some constant, since they represent data variations along the projection axes and thus contain shape information. We only wish to remove the effect of different data size; this is achieved by normalizing the eigenvalues, which represent data variations.

Another justification for equation (4.2) is that the dot-product matrices $\hat{A}_k\hat{A}_k^T$ and $\hat{B}_k\hat{B}_k^T$ are respectively the best rank-k approximations, in Frobenius norms, of A and B [16], which are already normalized to scale. Using the same argument, we see that the dot product

matrices resulting from eigenmode scaling with eigenvalues themselves [57] become best rank-k approximations of the autocorrelation matrices $AA^T = A^2$ and B^2, respectively, whose entries do depend on the size of the data sets.

Experimental results: The effectiveness of our eigenvalue scaling scheme is shown in Figure 4.2, where we plot the correspondence errors in the case of scaled versus unscaled spectral embeddings. The correspondence error is measured as the total geodesic distance $\sum_{i=1}^{n} g(v_i, v_i')$, where n is the number of vertices to be matched, v_i is the vertex corresponding to i that is computed by an algorithm, and v_i' is the ground-truth match for i. The plots are against the difference in the number of vertices of the two meshes to be matched. The correspondence algorithm used is our own and it is described in Section 4.3. In producing the two plots, the only difference is whether the eigenmodes are scaled.

Measuring error for dense correspondence is not easy since the ground-truth correspondence is impractical to establish manually. In our evaluation, we first construct successively decimated copies of the same 3D mesh using the QSlim mesh decimation program of Garland [22]. Next we use our algorithm to find correspondences between the decimated copies and the original mesh and measure correspondence errors. The ground-truth can be trivially established since QSlim retains the positions of undecimated vertices.

A motivation based on Kernel-PCA: It turns out that the eigenvalue scaling scheme we have proposed can also be motivated in the framework of Kernel-PCA. It can be shown that the embedding of mesh vertices obtained using our scheme is essentially the same as their projection obtained using Kernel-PCA, where, the kernel matrix is defined by a Gaussian. We give details of Kernel-PCA for completeness, and show its relationship to our method in the next Section.

4.2.3 Kernel-PCA and Spectral Embeddings

Given a point set $X = x_1, \ldots, x_n$, where, $x_i \in \mathbf{R}^g$ is a g-D point, recall that standard PCA attempts to extract the principal components of the data by computing the eigenvectors of the covariance matrix:

$$C = \frac{1}{n} \sum_{i=1}^{n} x_i x_i^T.$$

Here, we assume that the points set X is centered to the origin, that is, $\sum_{i=1}^{n} x_i = 0$. The principal components are given by $V = [v_1|v_2|\ldots|v_g]$, where, $\{v_k|1 \leq k \leq g\}$ is given

by $\lambda_k v_k = C v_k$, and λ_k is the corresponding eigenvalue. These eigenvectors define a new coordinate system which is an orthogonal transformation of the canonical coordinate system of $\mathbf{R}^g$. The leading eigenvectors are more important principal component directions along which the point projections have larger variations. If the variations along the trailing eigenvectors are sufficiently small, it is reasonable to discard the corresponding projections, effectively reducing the data dimension without losing much information. This process is known as dimensionality reduction and is one of the main applications of PCA.

Kernel PCA (KPCA) [51], an extension to the standard PCA, first applies to X a generally non-linear mapping $\phi : \mathbf{R}^g \rightarrow \mathcal{F}$, where $\mathcal{F}$ is referred to as the *feature space*. Then the standard PCA is carried out in $\mathcal{F}$ on the point set $\phi(X) = \{\phi(x_i) | x_i \in X\}$. Since $\mathcal{F}$ may have a very high, possibly infinite, dimensionality, the non-linear properties of the data X can be "unfolded" into linear ones. Thus algorithms that work on linear structures, e.g., PCA, can be effectively applied in $\mathcal{F}$.

The mapping ϕ is never explicitly given, but implicitly specified by the inner products between the data and encoded in a *kernel matrix* $K \in \mathbf{R}^{n \times n}$, where $K_{ij} = k(x_i, x_j) = \phi(x_i) \cdot \phi(x_j)$. Algorithms that run in the feature space based only on inner products can be efficiently implemented in the original space by replacing inner products by the kernel function k. The most commonly used kernel is the Gaussian radial basis function [51]

$$k(z_i, z_j) = e^{-\frac{d_{ij}^2}{2\sigma^2}}, \; d_{ij} = ||z_i - z_j||^2. \tag{4.3}$$

We wish to show in this Section that spectral embeddings and KPCA are infact the same except that the kernel function is now the Gaussian of geodesic distances instead of the Euclidean distances in equation 4.3. That is, the feature space $\mathcal{F}$ in the case of spectral embeddings is such that inner products are defined by the Gaussian of geodesic distances in the original space. This leads us to provide yet another justification for the eigenvector scaling scheme we proposed in the previous Section. We first give a brief description of KPCA (details can be found in [51]) and show the relationship between KPCA and spectral embeddings. To perform PCA in the feature space, the new covariance matrix is given by:

$$\hat{C} = \frac{1}{n} \sum_{i=1}^{n} \phi(x_i) \phi(x_i)^T. \tag{4.4}$$

The principal components of the points in feature space are then given by $\{\hat{v}_k | 1 \leq k \leq h\}$, where h is the dimensionality of the feature space and,

$$\hat{\lambda}_k \hat{v}_k = \hat{C} \hat{v}_k. \tag{4.5}$$

From equations 4.4 and 4.5 we get $\hat{\lambda}_k \hat{v}_k = \frac{1}{n}\sum_{i=1}^{n}(\phi(x_i).\hat{v}_k)\phi(x_i)$. This implies that the principal components $\hat{v}_k$ can be represented by a linear combination of $\phi(x_i)$'s. That is,

$$\hat{v}_k = \sum_{i=1}^{n} \alpha_{ki}\phi(x_i). \tag{4.6}$$

Now from equations 4.4 and 4.5 we get:

$$\begin{aligned}
\hat{\lambda}_k \hat{v}_k &= \frac{1}{n}(\sum_{i=1}^{n} \phi(x_i)\phi(x_i)^T)\hat{v}_k \\
\Rightarrow \hat{\lambda}_k \hat{v}_k \phi(x_l) &= \frac{1}{n}(\sum_{i=1}^{n} \phi(x_i)\phi(x_i)^T)\hat{v}_k.\phi(x_l) && \text{(for some } l\text{).} \\
\Rightarrow \hat{\lambda}_k \sum_{j=1}^{n} \alpha_{kj}(\phi(x_j).\phi(x_l)) &= \frac{1}{n}(\sum_{i=1}^{n} \phi(x_i)\phi(x_i)^T)\sum_{j=1}^{n} \alpha_{kj}\phi(x_j).\phi(x_l) && \text{(after substituting for } \hat{v}_k \text{ from equation 4.6).} \\
\Rightarrow n\hat{\lambda}_k A\alpha_k &= A^2\alpha_k.
\end{aligned}$$

In the last equation, $A \in \mathbf{R}^{n\times n}$ is a matrix such that, $A_{ij} = \phi(x_i).\phi(x_j)$, and α_k is the vector $(\alpha_{k1}, \alpha_{k2}, \ldots, \alpha_{kn})$. A is essentially the kernel matrix. If A is defined such that it is symmetric, the last equation becomes equivalent to:

$$n\hat{\lambda}_k\alpha_k = A\alpha_k. \tag{4.7}$$

Now the principal component of the data point $\phi(x_i)$ is given by their projection on the principal component axes $\hat{v}_k$. That is,

$$\begin{aligned}
\hat{\phi}(x_i)_k = \hat{v}_k.\phi(x) &= \sum_{j=1}^{n} \alpha_{kj}(\phi(x_j).\phi(x_i)) \\
&= \sum_{j=1}^{n} \alpha_{kj}A_{ij} \\
&= A_i\alpha_k \qquad (A_i \text{ is the } i^{th} \text{ row of } A).
\end{aligned}$$

Hence, the projection of $\phi(x_i)$ along all the principal components $\{\hat{v}_k | 1 \leq k \leq n\}$ is given by the vector:

$$\hat{\phi}(x_i) = (A_i\alpha_1, A_i\alpha_2, \ldots, A_i\alpha_n).$$

Hence, the projection of the whole dataset $\phi(X)$ is given by the matrix:

$$\begin{aligned}\hat{\phi}(X) =& [A\alpha_1|A\alpha_2|\dots|A\alpha_n] \\ & [n\hat{\lambda}_1\alpha_1|n\hat{\lambda}_2\alpha_2|\dots|n\hat{\lambda}_n\alpha_n] \quad \text{(from equation 4.7).}\end{aligned}$$

Note that the principal components $\hat{v}_k$ need to be of unit length. This gives a normalization condition for α_k's:

$$\begin{aligned}&\hat{v}_k.\hat{v}_k = 1 \\ \Rightarrow &\sum_{i,j=1}^{n} \alpha_{ki}\alpha_{kj}(\phi(x_i)\phi(x_j)) = 1 \qquad \text{(after substituting for } \hat{v}_k \text{ from equation 4.6).} \\ \Rightarrow &\sum_{i,j=1}^{n} \alpha_{ki}\alpha_{kj}A_{ij} = 1 \\ \Rightarrow &\alpha_k.A\alpha_k = 1 \\ \Rightarrow &n\hat{\lambda}_k(\alpha_k.\alpha_k) = 1.\end{aligned}$$

Thus, we need to divide all α_k's by $\sqrt{n\hat{\lambda}_k}$. Now we re-write the equation for the projection $\hat{\phi}(X)$ after performing this normalization:

$$\hat{\phi}(X) = [\sqrt{n\hat{\lambda}_1}\alpha_1|\sqrt{n\hat{\lambda}_2}\alpha_2|\dots|\sqrt{n\hat{\lambda}_n}\alpha_n]. \tag{4.8}$$

Note from equation 4.7 that α_k's are the eigenvectors of the kernel A and $n\hat{\lambda}_k$'s are the corresponding eigenvalues. Now the relation between spectral embeddings and KPCA is quite clear. The projection $\hat{\phi}(X)$ is exactly the spectral embedding of the point set X, where we consider that the kernel matrix A is defined using a Gaussian of geodesic distances and hence is the same as the affinity matrix. Also, our scaling scheme has emerged naturally under the KPCA framework, hence, providing another justification for scaling the eigenvectors with the square root of the eigenvalues.

One more detail here is that equation 4.4 assumes that the data set $\phi(X)$ is centered at the origin which may not be true. Hence, the kernel matrix is not centralized. However, it is easy to compute the centralized matrix from the uncentralized version. Another option is to exclude the first eigenvector from the projection, which we adopt as it further reduces the dimensionality.

4.2.4 Non-robustness of Eigenmodes

Eigenmode switching: Perturbation theory predicts that when eigenvalues move close to each other, the corresponding eigenvectors may switch order [25]. We have observed that such switching can occur early in the eigenvalue order, e.g., between 4 and 8, even when the two shapes being matched are perceptually similar. But there is no general pattern of eigenvalue clustering that is sufficiently reliable to detect the switchings. As switching of two coordinates, the eigenvectors, induces a reflection in the spectral domain, spectral correspondence based on the L_2 distance measure or correlations [57, 62], even with the aid of clustering [9, 10], can fail.

For a visual illustration of eigenmode switching, we color-plot the eigenvectors in MATLAB, where the entries in an eigenvector are used as indices into the color map. To enhance our illustration, we nonlinearly warp the color map. As shown in Figure 4.3, given in the color plate, two similar shapes have compatible eigenmodes, reflected by consistent color plots, only up to the 4^{th} eigenvalue. The 5^{th} and 6^{th} eigenmodes are switched and color patterns for the next a few eigenvectors, those exhibiting higher-frequency color variations, do not exhibit any discernible patterns. Evidently, correspondence analysis using eigenvalue orderings to pair up eigenmodes beyond the 4^{th} one would be hard to justify in this case. Moreover, since the eigenvalue represents the variation of the data along the projection axis, an eigenvector can be made to appear at any position in the spectrum by stretching or shrinking the data along that axis. Such stretching can be easily found in real world shapes.

Relevance to eigenvalue scaling: One interesting point to note is that as the magnitude of the eigenvalues of the geodesic affinity matrices exhibit rapid decay, as shown in the caption of Figure 4.3, eigenvalue scaling has the effect of rapidly attenuating the effects of higher-frequency eigenvectors. This would be quite appropriate since these eigenvectors are less reliable to use for correspondence analysis. As a side effect, the resulting correspondence algorithm will be less sensitive to the number of eigenmodes chosen. In previous works, e.g., [10], some heuristic has to be adopted to determine the proper dimensionality to use.

Sign flips: Besides reflections induced by switching, other transformations in the spectral domain also need to be handled to achieve robust spectral correspondence. One such transformation is due to the arbitrary determination of the signs of the eigenvectors by the numerical eigenvalue program, as already noted in previous works [9, 57]. Note that this is another form of reflection. Caelli and Kosinov [9] propose to use a dominant sign

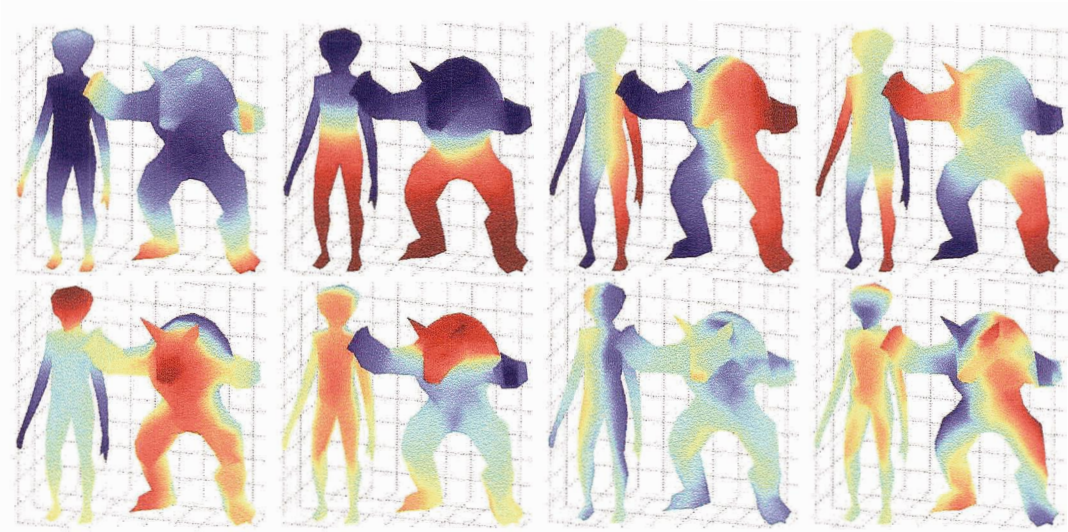


Figure 4.3: Eigenvector plots for two shapes, both with 252 vertices. The first 8 eigenvalues are [205.6, 11.4, 4.7, 3.8, 1.8, 0.4, 0.26, 0.1] and [201.9, 10.9, 6.3, 3.4, 1.8, 1.2, 0.31, 0.25], respectively.

correction, always ensuring that there are more positive entries in each eigenvector. This is highly unreliable however since in practice, most eigenvectors have about the same number of positive and negative entries. Shapiro and Brady [57] use a greedy approach to correct one sign at a time by optimizing for a correspondence cost. In the presence of eigenmode switchings, this approach is not robust either.

Other transformations: Consider the spectral embeddings of two similar human meshes using the 2^{nd}, 3^{rd} and 4^{th} eigenvectors (this particular choice of the eigenvectors is explained in Section 4.3), as shown in Figure 4.4(a). Ideally, the embeddings would be perfectly aligned. However, a rotational difference in the embeddings is clearly visible. In addition, there are also other discrepancies of a non-rigid nature. Another example is given in Figure 4.4(b), where the second (light gray) mesh is merely a scaled version (scaled along the x direction) of the first (dark gray) mesh. But there again is a rotation in the embedding. We believe that such transformations in the spectral domain, as well as eigenmode switchings, are the result of non-uniform stretching in the shapes. Obviously, a matching algorithm must be able to deal with all these transformations in order to operate robustly.

(a) Two similar human meshes with their spectral embeddings differ by a rotation and some stretching.

(b) A human mesh and its stretched version. Their spectral embeddings differ by a rotation.

Figure 4.4: Stretching in spatial domain induces rotation and non-rigid transformations in the spectral domain.

4.2.5 Possible Remedies for Non-robustness of Eigenmodes

We made several attempts to resolve the problem of eigenvector switching and sign flips in the spectral embeddings. In this Section, we would like to briefly discuss some of these attempts. There are essentially two ways to fix this problem: either un-switch the eigenvectors to obtain reflections-free embeddings, or acknowledge that the embeddings could have reflections and design algorithms that are invariant to these reflections. The first method requires us to find the ordering of the eigenvectors that best aligns the two embeddings. This can be mathematically expressed as:

$$\text{minimize}_{(M,P)} ||M\hat{A}_k P - \hat{B}_k||$$

where, $\hat{A}_k$ and $\hat{B}_k$ are the two $n_1 \times k$ and $n_2 \times k$ embeddings defined in equation 4.2, P

is a $k \times k$ selector matrix whose entries are from $\{-1, 0, 1\}$ with exactly one non-zero entry per row and column, and M is a $n_2 \times n_1$ correspondence matrix whose entries are either 0 or 1, with exactly a one 1 in every row. Here, P is the matrix that selects a particular ordering and sign configuration of the eigenvectors (columns) in $\hat{A}_k$, and M defines the correspondence between the embedding coordinates (rows) in $\hat{A}_k$ and $\hat{B}_k$. This optimization is a generalization to the graph-isomorphism problem, in which, $\hat{A}_k$ and $\hat{B}_k$ become square adjacency matrices representing two graphs and M and P become permutation matrices such that $M = P^T$. Since, the size of matrix P is small as $k << \min(n_1, n_2)$, we might consider the solution of exhaustively searching for best possible P. Given P, M can be established by using the simple "best matching" scheme in $O(n_1 n_2)$ time. Exhaustively going through all P's results in a time complexity of $O(k!2^k n_1 n_2)$ since there are $k!$ possible orderings of k eigenvectors and 2^k sign configurations for every ordering. Unfortunately, this complexity is tractable only for very small values of k. Hence, we look into heuristic minimization of the given problem to reduce the complexity. We discuss some heuristics in Section 4.2.6.

The second method of dealing with reflections in the spectral embeddings is to design algorithms invariant to such reflections. Along this direction, we attempted the following two approaches:

- **Reflection invariant point descriptors:** Once we have the spectral embeddings of two shapes, one way to compute correspondence is to match them using shape descriptors, e.g., shape context. Since the embeddings are invariant to bending, the shape descriptor used will be invariant to bending. However, now we have to use a *reflection invariant shape descriptor* to account for the reflections in the embeddings. Shape contexts by themselves do not have such property since the bins made by angular spatial division will report different point counts if the shapes are reflected. Hence, our aim would be to construct a spatial binning scheme in which the point count in a bin does not change if the shape is reflected. One such binning strategy would be to construct only spherical bins. Such binning is not only invariant to reflections but also to rotations. Moreover, we need to deal only with reflections caused by switching of coordinate axes or flipping the sign of coordinate axes, but such binning is invariant to any arbitrary reflection about the center of the spherical shells. It is generally believed that as invariance of a shape descriptor (towards more transformations) increases, its

descriptive power decreases. Hence, ideally we would like to have a descriptor which is invariant only to the class of reflections we are concerned with and at the same time is sufficiently descriptive.

- **Symmetric polynomial transformation:** We have transformed the shapes from the spatial domain to spectral domain in order to achieve invariance to bending and other transformations. In the same spirit, another way of dealing with reflections in spectral embeddings is to transform them in a domain that is invariant to these reflections. Symmetric polynomials is one such candidate. Given a set of numbers $X = \{x_1, x_2, \ldots, x_k\}$, the k symmetric polynomials for this set are defined as:

$$
\begin{aligned}
s_1 &= \sum_{i=1}^{k} x_i \\
s_2 &= - \sum_{1 \le (i_1, i_2) \le k} x_{i_1} x_{i_2} \\
&\vdots \\
s_k &= (-1)^{k-1} \sum_{1 \le (i_1, i_2, \ldots, i_k) \le k} x_{i_1} x_{i_2} \ldots x_{i_k}
\end{aligned}
$$

  Symmetric polynomials are nothing but the coefficients of a polynomial of degree $(k-1)$, whose roots are given by the elements in set X. It is well known that there is a one-to-one mapping between the coefficients of a polynomial and its roots, that is, for a particular set of coefficients, there will be a unique set of roots and vice versa. Also, symmetric polynomial formulation is invariant to reordering the numbers in the original set (for example, the value of $s_1 \ldots s_k$ remain the same if say x_1 and x_3 are swapped). We use this property and transform the spectral embedding points into corresponding symmetric polynomials, thus achieving invariance to eigenvector switching. That is, for the $(n_1 \times k)$ embedding matrix $\hat{A}_k$, the $(n_1 \times k)$ symmetric polynomial matrix $S_{\hat{A}_k}$ is such that the i^{th} row of $S_{\hat{A}_k}$ contains the k symmetric polynomials obtained from the k numbers in the i^{th} row of $\hat{A}_k$. Similarly, $S_{\hat{B}_k}$ is obtained from embedding $\hat{B}_k$, and matching is performed using $S_{\hat{A}_k}$ and $S_{\hat{B}_k}$. Invariance to sign-flips of the eigenvectors is achieved by taking the absolute values of the entries in $\hat{A}_k$ and $\hat{B}_k$.

Although both reflection invariant shape descriptors and symmetric polynomials are elegant ways of dealing with reflections in the spectral embeddings, our experiments show

that they do not perform satisfactorily when used for correspondence. We suspect two reasons for such results: first, making the embeddings invariant to reflections also makes the resulting representation less descriptive, second, both the formulations, though invariant to reflections, are susceptible to other rigid/non-rigid transformations in the embeddings explained earlier. Experimentally, we found that heuristically finding the right ordering and sign configuration of the eigenvectors achieves better results than using reflection invariant formulations.

4.2.6 Heuristic Eigenvector Reordering

Several heuristics can be proposed in order to efficiently reorder the eigenvectors of two spectral embeddings. In this Section, we present a few of them:

Greedy Reordering: Let us first consider a low dimensional embedding, e.g., with only two eigenvectors. We exhaustively find the best possible ordering and signs of these few eigenvectors. Now we incrementally add one eigenvector at a time and at each step, compute the best possible position and sign of the new eigenvector. This results in $O(k^2)$ possibilities to compare, greatly reducing the time complexity from $O(k!2^k n_1 n_2)$ for exhaustive search, to $O(k^2 n_1 n_2)$.

Pairwise Flipping: Although, theoretically, the eigenvectors of two affinity matrices may differ by an arbitrary reordering, in practice, for sufficiently similar shapes, we noticed that only consecutive eigenvectors exchange positions. Hence, the best ordering may be found by flipping pairs of consecutive eigenvectors. More specifically, at every iteration, we identify a pair of consecutive eigenvectors that, if flipped, will reduce the correspondence cost the most. This pair is then flipped and next pair is searched for. Note that if an arbitrary reordering is given, then this flipping is more likely to result in a local minima, when compared to the greedy approach. However, the complexity of this heuristic is $O(fkn_1n_2)$, where, f is the number of pairs that were flipped. In practice, this number is expected to be small. To deal with sign flips in eigenvectors, we take the absolute value of the eigenvector coefficients while computing the right ordering. Once the correct ordering is found, the correct sign configuration can be computed using other methods, for example, the greedy approach of Shapiro and Brady [57].

4.3 Spectral Shape Correspondence Algorithm

Observe that iterative alignment techniques, e.g., [14], can work quite well when the initial shapes are approximately aligned, while spectral embedding can automatically remove the effects of rigid-body transformations, uniform scaling, and shape bending. Hence, a natural approach would be to perform non-rigid alignment in the spectral domain before computing the matching. The only obstacle now is to handle reflections caused by eigenvector switching and sign flips, as they can introduce large discrepancies into the initial configurations of the shapes to be matched. As explained earlier, we deal with this problem by heuristically finding the right ordering and sign configuration of the eigenvectors. Due to the rapid decay of eigenvalues and eigenvalue scaling, we never find it necessary to use more than $k = 6$ eigenvectors to arrive at a satisfactory mesh correspondence. So k is always small and even the exhaustive solution for eigenvector reordering can be used.

Once we have aligned the two spectral embeddings properly, we attempt to transform one embedding into another. Due to the presence of non-rigid deformations in the spectral domain, we modify the original rigid ICP algorithm [7] by replacing its transformation model with the use of thin-plate splines. Thin-plate splines are well-known and have been applied to model non-rigid transformations before [6, 14] in the context of 2D shape registration. A brief overview of this technique is given in the Appendix A.

Now consider two 3D meshes M_1 and M_2 with n_1 and n_2 vertices, respectively. Without loss of generality, assume that $n_1 \leq n_2$. Let us describe each step of our spectral correspondence algorithm in details below. Experimental results for shape correspondence are given in Section 4.6.

1. **Geodesic affinities:** Construct Gaussian affinity matrix A where

$$A_{ij} = e^{\frac{-d_{ij}^2}{2\sigma_{M_1}^2}} \tag{4.9}$$

   where d_{ij} is the geodesic distance between vertex i and j in M_1. The Gaussian kernel width σ_{M_1} is set to be the maximum geodesic distance between any two vertices in M_1. The performance of our method is relatively invariant to the choice of σ_{M_1} as long as it is set to a sufficiently large value. Similarly, we construct B, the Gaussian affinity matrix for mesh M_2.

2. **Spectral embeddings:** The affinity matrices A and B are eigenvalue decomposed

and the resulting spectrum are truncated to k. Each of the k eigenvectors is scaled with the square root of its corresponding eigenvalue. These steps have already been described in details in Section 4.2.1 and 4.2.2.

We have already stated the reason for ignoring the first eigenvector of the affinity matrix in Section 4.2.3. The following is another justification for the same. Note that if the Gaussian width is sufficiently large, the row-sums of the affinity matrix are almost constant. As a result, the first eigenvector of the matrix will be close to a constant vector and can be safely ignored. From now on, we denote by $\hat{A} \in \mathbf{R}^{n_1 \times (k-1)}$ and $\hat{B} \in \mathbf{R}^{n_2 \times (k-1)}$, as first defined in Equation 4.2, the $(k-1)$-dimensional embeddings of M_1 and M_2, respectively, where the first eigenvector is disregarded. $\hat{A}$ and $\hat{B}$ are essentially $n_1 \times (k-1)$ and $n_2 \times (k-1)$ matrices where the i^{th} rows of $\hat{A}$ and $\hat{B}$ are the $(k-1)$-dimensional spectral embedding coordinates of the i^{th} vertices of meshes M_1 and M_2 respectively. In all our experiments, we have used $k = 5$ or 6 hence giving a 4 or 5-dimensional spectral embedding after disregarding the first eigenvector.

3. **Eigenvector reordering and sign correction:** We keep the ordering and signs of the eigenvectors of one mesh, e.g., M_1, fixed. With either the exhaustive search or any heuristic, we need to compute the cost of a correspondence, which we describe below. First, we obtain a best matching C based simply on the L_2 metric; other metric, such as the Chi-square or Mahalanobis distance is also possible. Specifically, for a vertex $v_i^{M_1}$ of mesh M_1, $v_{C(i)}^{M_2}$ is the corresponding vertex of M_2, where

$$C(i) = \text{argmin}_j \|\hat{A}_i - \hat{B}_j\|. \tag{4.10}$$

Here $\hat{A}_i$ and $\hat{B}_j$ denote the spectral embedding coordinates of vertex i in mesh M_1 and vertex j in mesh M_2 respectively (i.e. the i^{th} and the j^{th} row of the matrices $\hat{A}$ and $\hat{B}$ respectively). The cost of the correspondence C is given by the sum:

$$cost(C) = \sum_{i=1}^{n_1} \|\hat{A}_i - \hat{B}_{C(i)}\|$$

We choose the ordering and signs of the eigenvectors for mesh M_2 which give the minimum $cost(C)$.

4. **Non-rigid alignment:** Once the eigenvectors for two shapes have consistent ordering and signs, we perform a non-rigid alignment using ICP modified by thin-plate splines (Refer to the Appendix). The pseudo-code for this alignment procedure is given below.

Given two spectral embeddings $\hat{A}$ and $\hat{B}$,

(a) Initialize parameters d, w, λ.

(b) Transform $\hat{B}$ into $\hat{\hat{B}}$ using the transformation parameters d and w.

(c) Update correspondence C using Equation (4.10) after replacing $\hat{B}_j$ with $\hat{\hat{B}}_j$.

(d) Given the correspondence C, update transformation parameters using Equation (A.2).

(e) Update the regularization parameter λ.

(f) Repeat from Step (b) until convergence.

We have found experimentally that 5 to 10 iterations of the iterative alignment are sufficient to align the embeddings. The value of the regularization parameter λ is set to be the mean distance between all embedded point pairs. As shown in [6], this scale-dependent assignment of λ is robust to scaling of the point sets.

5. **Proximity-aided matching:** For dense correspondence it is hard to distinguish between near-by points using an alignment and correspondence procedure based on optimizing a global energy, which is the case in our approach. As used for 2D shape correspondence, we use the proximity heuristic (described in Section 3.1.3) to improve correspondence locally. We briefly outline the heuristic here for convenience.

   First, the anchor point pairs are computed as follows. Consider the $(n_1 \times n_2)$ matrix Z of correspondence costs between all points of M_1 and M_2. That is,

$$Z_{ij} = \|\hat{A}_i - \hat{B}_j\|.$$

   The first anchor point pair $(a_1^{(1)}, a_2^{(1)})$, where $a_1^{(1)}$ is a vertex of M_1 and $a_2^{(1)}$ is a vertex of M_2, is selected as the pair with least correspondence cost. That is,

$$(a_1^{(1)}, a_2^{(1)}) = \text{argmin}_{(i,j)}(Z_{ij}).$$

   The second anchor point pair is calculated in the same way. However, we would need the anchor points to be far from each other over the mesh. Hence, before finding the second pair, we modify the matrix Z so that points close to the first anchor point are penalized. The new correspondence cost matrix is given by:

$$Z_{ij}^{(1)} = Z_{ij} - \frac{1}{2}[dist^{M_1}(i, a_1^{(1)}) + dist^{M_2}(j, a_2^{(1)})]$$

where $dist^M(i,j)$ is the geodesic distance between the i^{th} and j^{th} vertex of mesh M. Now, the second anchor point pair is given by:

$$(a_1^{(2)}, a_2^{(2)}) = \text{argmin}_{(i,j)}(Z_{ij}^{(1)}).$$

This process can be repeated to obtain more anchor point pairs. With the anchor points, we modify Equation (4.10) for finding the best correspondence C to incorporate the proximity cost:

$$C(i) = \text{argmin}_j \Big[\|\hat{A}_i - \hat{B}_j\| \sum_{l=1}^{h} \alpha_l \cdot \|dist^{M_1}(i, a_1^{(l)}) - dist^{M_2}(j, a_2^{(l)})\| \Big]$$

where h is the number of anchor point pairs chosen and the α_l's are free parameters set by the user.

The success of the proximity heuristic depends on two factors: quality of the anchor point pairs and geodesic distances. Since the meshes are already well aligned, choosing the anchor point pairs to be the most trusted matches are expected to be robust. However, the dependence on geodesic distances may cause sensitivity of the heuristic to stretching in the shapes. Hence, fixing a large number of anchor point pairs can render the matching non-robust. Thus we restrict to fixing only three anchor pairs and set $\alpha_1 = \alpha_2 = \alpha_3 = 1$ for all our experiments.

4.4 Shape Retrieval based on Spectral Embeddings

For 3D shape retrieval, we need to define a measure of similarity between any two given shapes. This similarity measure must be invariant to common geometric transformations and to bending (for articulated shapes). For this purpose we use the bending invariant spectral embeddings. The construction of such embeddings has already been explained in Section 4.3. In this Section we focus on describing other steps of our shape retrieval algorithm as well as heuristics designed to handle various degeneracies in shapes provided in existing shape databases. For any given 3D model, we first compute the spectral embedding as follows:

1. **Construction of structural graph:**

In order to achieve invariance to bending, we define the affinities based on geodesic distances. However, estimation of geodesic distances poses several problems. Conventional methods of geodesic distance estimation over a mesh depend largely on the connectivity of the mesh. This dependency limits the use of geodesic distances, as we have noticed that many shapes in all the well-known shape database have disconnected components (a small number of them are simply triangle soups), in which case, the geodesic distance estimation will completely fail. We thus turn to a heuristic to work around this problem, which we describe below.

We use shortest path graph distances over the mesh connectivity to approximate geodesic distances. This not only provides computational and implementational simplicity, but also removes the constraint that the shape has to be defined using a connected manifold mesh. However, this does not solve the problem with disconnected mesh components. Hence, we need to add extra edges to the connectivity graph while making sure that the structure of the shape remain largely unchanged. Given a 3D mesh, let $G_m = (V, E_m)$ be the connectivity graph of the mesh and $C_1, C_2, \ldots$ be its components. We construct a p-connected graph $G_p = (V, E_p)$ over the mesh vertices such that the graph faithfully represents the shape. This is done using Yang's efficient algorithm for constructing p-connected graph over point clouds in Euclidean space that locally minimizes edge lengths by computing and combining p Euclidean minimum spanning trees of the given point cloud [67]. As shown in [67] and verified by our experiments, the resultant graph G_p approximates the structure of the shape well. Given G_m and G_p, the final structural graph is defined as:

$$G = (V, E)$$

$$E = E_m \cup \{(i, j) \mid (i, j) \in E_p, i \in C_I, j \in C_J, I \neq J\}$$

Informally, the structural graph G includes all edges of G_m and only those edges of G_p that join two disconnected components. In our implementation, we restrict p to 1 or 2 since higher values of p may result in edges between far away (hence, unrelated) components which is undesirable.

Note that the definition of G makes sure that there is a single connected component. However, we include only the edges that are absolutely required to connect the components. The remaining edges come from the original mesh connectivity. This helps

in better preserving the structure of the mesh. Once the structural graph G has been constructed, the geodesic distance between two vertices can be approximated by the shortest path length in G computed using Dijkstra's shortest path algorithm.

2. **Spectral embedding**:

   Having robustly estimated geodesic distances between all pairs of points, we can compute the spectral embedding according to the procedures given in Section 4.3. We outline that procedure here to make the description of our shape retrieval approach self-contained.

   Define the affinity matrix A based on geodesic distances. The affinity matrix is given by a Gaussian of the distances:

   $$A_{ij} = e^{\frac{-d_{ij}^2}{2\sigma^2}},$$

   where d_{ij} is the approximate geodesic distance between the i^{th} and the j^{th} vertex computed from above, and $\sigma = \max_{(i,j)}\{d_{ij}\}$ is the Gaussian width. Now we eigendecompose A to obtain the spectral embedding:

   $$A = U\Lambda U^T$$
   $$\hat{A} = U\Lambda^{\frac{1}{2}}$$

   We truncate the last $n-k-1$ columns and the first column of $\hat{A}$ for reasons explained in Section 4.2 and 4.3 to obtain the k-dimensional embedding $\hat{A}_k$. For 3D shape retrieval and for all results reported in this paper, we represent every shape with a 3-D spectral embedding given by the 2^{nd}, 3^{rd} and 4^{th} eigenvectors (scaled) of the affinity matrix. The 3D embeddings of some articulated shapes from the McGill shape database [63] are shown in Figure 4.5.

3. **Other affinity measures**:

   Although the use of geodesic distances to define affinities provide invariance to bending, it might cause adverse effects in some cases. For example, consider two chair models. Suppose that the arm-rest of one model is connected directly to its back-rest, however, on the other model, it is not connected directly, but through the bottom seat. In the first case, the geodesic distance between a point on the arm-rest and a

Figure 4.5: Spectral embeddings (bottom row) of some articulated 3D shapes (top row) from the McGill shape database. Note that normalization has been carried out.

point on the back-rest is small, whereas in the second case it will be relatively large since it has to go through the bottom seat. Hence, the spectral embeddings of the two chairs could be radically different and the retrieval result will suffer. In general, due to the sensitivity of geodesic distances to the topological noise in the shapes, the spectral embeddings can be different for two similar shapes. Note that if we define affinities based on Euclidean distance between the points, it would resolve the problem described above. However, such definition of affinities can no longer be expected to be invariant (or even robust) to bending. Nevertheless, this discussion reveals the flexibility of our approach, with the use of affinity matrices, in that they can be easily tuned to render the retrieval process invariant to a particular class of transformations depending on the database in question.

In the following Sections, we show a comparison of retrieval results using different affinity measures. However, since our target database is that of articulated shapes, it is not surprising that the geodesic distance based affinities perform the best. Minor improvements over conventional shape descriptors can still be seen using other affinity measures, which strengthens our proposal of performing retrieval over embeddings instead of the original shapes. We use Gaussian affinity matrix in all experiments, while varying the term d_{ij} based on various measures. The affinity measures that are compared in subsequent Sections, other than geodesic distance based affinities, are:

(a) Euclidean distance: d_{ij} is the Euclidean distance between vertices i and j.

(b) Combined distance: d_{ij} is the uniform combination of the Euclidean and the (approximate) geodesic distance between vertices i and j.

4. **Global Shape Descriptors**:

We have transformed the given 3D shape into its 3D spectral embedding. We can now define global shape descriptors on this embedding in the same way as they are defined on the original shape. Conventional global shape descriptors can also be used on the embeddings by considering the embedding to be just another 3D shape. We present a comparative study of various conventional shape descriptors and new shape descriptors obtained from the spectral embeddings in Section 4.6

4.5 Nyström Approximation for Fast Spectral Embedding

For both correspondence and retrieval, we need to construct the spectral embedding of the input meshes. Note that the time complexity of constructing the full geodesic-based affinity matrix for a mesh with n vertices is $O(n^2 \log n)$, as the complexity for computing geodesic distance from one vertex to all other vertices using fast marching is $O(n \log n)$. Moreover, the eigen-decomposition of an $(n \times n)$ matrix takes $O(n^3)$ time or $O(kn^2)$ if only the first k eigenvectors are computed. This complexity does not affect the performance of shape retrieval drastically, since the spectral embeddings of all shapes in the database can be precomputed. However, the query model still needs to be processed. To speed things up, we use Nyström approximation [18] to efficiently approximate the eigenvectors of the affinity matrix.

Nyström approximation is a sub-sampling technique that reduces the time complexity of affinity matrix construction and eigen-decomposition to $O(ln \log n + l^3)$, where l is the number of samples selected with $l << n$. We adopt a furthest point sampling scheme, which at each step, chooses a sample which maximizes the minimum (approximated) geodesic distance from the new sample to the previously found samples; the first sample can be chosen randomly. We refer to this sampling strategy as "Max-Min" sampling. Our extensive experiments confirm that for the purpose of shape retrieval, only 10 to 20 samples, from meshes with thousands of vertices, are sufficient. Nyström approximation and Max-Min sampling are explained in details in Appendix B.

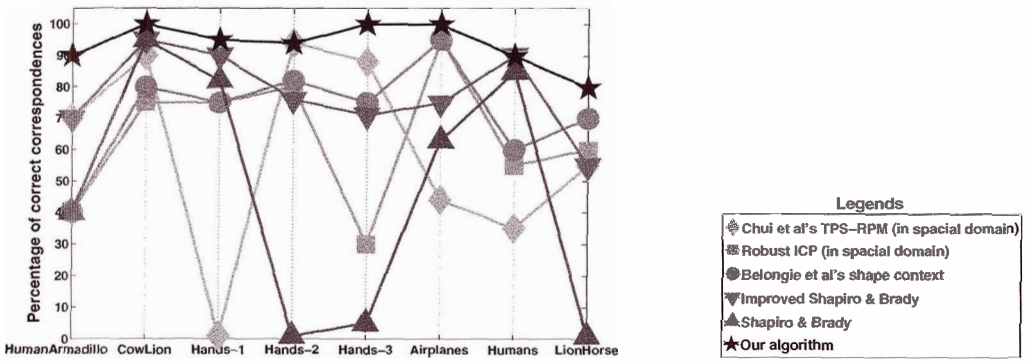


Figure 4.6: A comparison between several correspondence algorithms, including ours. The percentage of correct correspondences is plotted.

4.6 Experimental results

4.6.1 Shape Correspondence Results

To evaluate a mesh correspondence algorithm, we hand-pick a small number (17 to 20) of feature points on both of the meshes to be matched, where the number of vertices in these meshes varies between 100 to 300. The ground-truth correspondence between the features is determined by human. Now we compute a correspondence, only between feature points, using the algorithm and record the percentage of correct correspondences obtained. Figure 4.6 shows a comparison between our algorithm and other well-known schemes, on eight test cases. The shapes to be matched in the test cases are shown in Figure 4.7; each pair of shapes exhibit some degree of non-rigid deformations. We now describe briefly the other schemes we have experimented with and our experimental setup.

- **TPS-RPM [14] in spatial domain:** This is one of the most successful non-rigid ICP algorithms. It combines thin-plate splines, soft assign, and deterministic annealing to achieve robust correspondence. But as we have shown in Figure 2.1, it is susceptible to poor initial alignment, such as a difference in rotation.

  To improve its performance, we *manually* and rigidly align the two shapes to be matched, attempting to neutralize any rotation or translation between them; this is

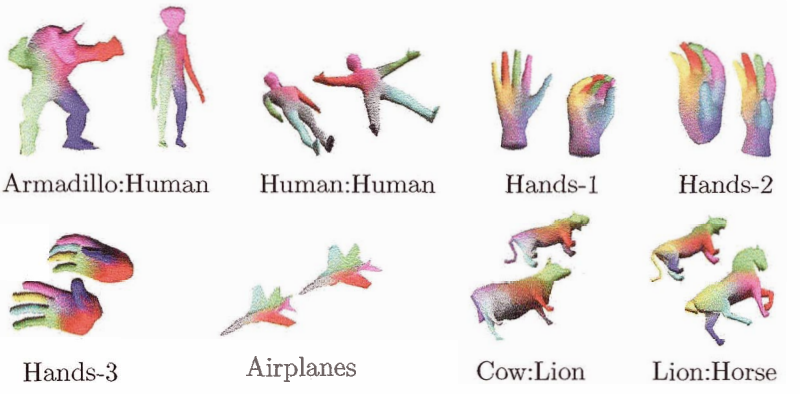


Figure 4.7: Correspondence results obtained from our algorithm, shown with color plots.

done for all the three spatial-domain schemes we have experimented with. However, bending in the shapes still cause the algorithm to perform poorly.

- **Robust ICP [73] in spatial domain:** This method is a recent variant of the original ICP [7] algorithm. It uses a hierarchical approach to achieve robust registration of 3D point sets. We use it as a representative of the rigid iterative alignment schemes.

- **Shape context [4, 5] in spatial domain:** We use a trivial 3D extension of the original 2D shape context of Belongie et al. [5] as a representative correspondence scheme based on local shape descriptors. Shape context is one of the most successful local descriptors for image analysis [44].

- **Shapiro and Brady [57]:** This is one of the early and best-known spectral point correspondence algorithms. It uses L_2 distance to compute a best matching, using their greedy sign correction but with no eigenvector reordering or eigenvalue scaling of eigenvectors.

- **Improved Shapiro and Brady:** Only eigenvalue scaling is incorporated into the original algorithm.

In each test case, $k = 6$ eigenvectors are used. Using more eigenvectors does not change the result due to eigenvalue scaling. Three out of the eight cases, Armadillo-Human, the Hands-1, and the Hands-3, have eigenvector switching occurring. In six of the eight cases, the greedy heuristic for eigenvector reordering and sign correction is successful; we shall

provide a remark on this issue in the next Section. The results shown are obtained by the expensive exhaustive search. Hence all results are limited to meshes with a few hundred vertices. In all test cases, no more than 10 iterations of our non-rigid ICP procedure are needed. In several cases, the procedure converges in less than 5 iterations. In terms of results, as can be seen from Figure 4.6, our algorithm clearly outperforms the state-of-the-art correspondence schemes mentioned above.

In Figure 4.7, we show some matching results obtained from our algorithm. The matching is shown by coloring the vertices of the meshes in an appropriate way. We first assign colors to the vertices of one of the two meshes, e.g., M_2. Then the color for the i^{th} vertex of mesh M_1 is set to be the color of the $C(i)^{th}$ vertex of M_2, where C is the correspondence found by our algorithm. This way, a good correspondence will induce a coloring that is consistent on both meshes. To show the meaningfulness of the correspondence obtained, we carefully assign different colors to different parts of the mesh M_2. Clearly, our algorithm matches bent shapes well, as well, it behaves robustly against moderate stretching in the shapes, e.g., the Armadillo vs. the human and the lion vs. the horse.

Note that in the Human:Human and Lion:Horse pairs, in Figure 4.7, which are of symmetric shapes, the correspondence is symmetrically switched. Namely, the right hand of one human is matched to the left hand of the other, etc. Similarly, the right leg of the lion is matched to the left leg of the horse, etc. This occurs since we define affinities based on geodesic distance, which is an intrinsic measure that cannot distinguish between symmetric points. As such, the left hand and the right hand of the human are equally good matches for the right hand of the other human. By chance, the Armadillo:Human, Cow:Lion, and Airplanes pairs return the right correspondences.

One solution to the above symmetry problem would be to carefully select the right sign of the eigenvectors. For shapes where there is one plane of symmetry, there will be two possible sign configurations of the eigenvectors that would give the minimum correspondence cost. These can be detected and the right configuration can be picked by inspecting it visually. As the number of symmetry planes increase, more sign configurations will give the minimum correspondence cost. A more analytical solution would be to define affinities in a symmetry-distinguishing way.

Figure 4.8 gives additional correspondence results obtained using our algorithm on numerous articulated shapes. In each of shape class, one per column, all the shapes are

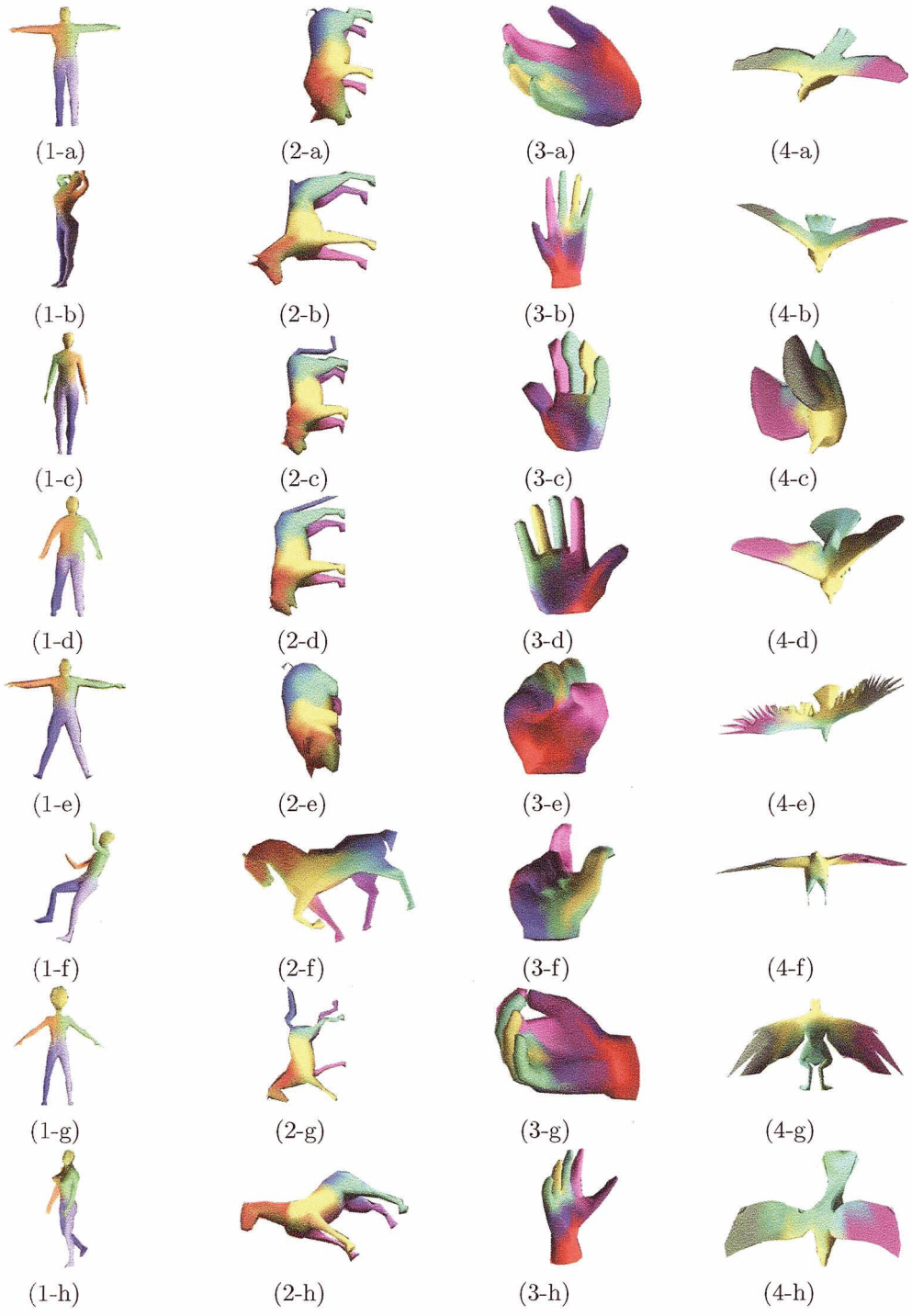


Figure 4.8: Column 1: correspondence results for human shapes. Column 2: for animal shapes. Column 3: for hand shapes. Column 4: for bird shapes.

matched to a single reference shape (the first shape in each column of 8) and correspondence obtained is color coded in accordance with the colors on the reference shape. Apart from showing the effectiveness of our method, e.g., see the second column of Figure 4.8, these examples also reveal some of its limitations which we discuss below:

1. **Effect of intrinsic shape symmetry:** As explained earlier, due to the intrinsic nature of the affinity matrix, our method is not guaranteed to match symmetric shapes correctly, as shown in Figure 4.8(1-c) and Figure 4.8(4-c) and (4-d). In all three cases, the sign configuration of the eigenvectors that gives the lowest correspondence cost leads to counterintuitive correspondence results. Our method succeeds in all the remaining cases in row 1, 2, and 4, although in each case, the next best eigenvector sign configuration, which has a correspondence cost extremely close to the lowest cost, would give a symmetrically flipped matching. This shows that the correspondence might very well have been symmetrically flipped in these cases too. Note that the correspondence may be symmetrically flipped, but it is nevertheless still consistent across the shape.

2. **Effect of topological changes:** Since our method largely depends on geodesic distances, topological changes can seriously harm the correspondence computation. This effect is visible in Figure 4.8(3-e) and (3-f), where the fingers of the hands are connected to the palm which would change the connectivity of the mesh, as well as the geodesic distances, drastically, resulting in unnatural correspondence results. Correct recovery of the correspondence between the fingers in this case appears to be a rather difficult problem, without some level of *prior* knowledge.

3. **Unreliable geodesic distances:** Figure 4.8(4-e) shows a bird shape that is very similar to the reference Figure for this group, Figure 4.8(4-a). However, the correspondence obtained is incorrect. We suspect that this is mainly due to the unreliability of geodesic distances on the wings of the bird that contains many "cuts". Hence, even though the shapes look similar in the spatial domain, their embeddings are rather different.

4. **Non-robustness of L_2 cost for exhaustive search:** Close inspection of Figure 4.8(1-b) reveals that the correspondence obtained is inconsistent: the left arm is colored orange which means that the left leg must be colored blue which is not

Figure 4.9: Incorrect eigenvector ordering is obtained even after exhaustively reordering the eigenvectors for shapes in Figure 4.8(1-a) and (1-b).

the case (note that this is different from the symmetry issue discussed above). This should not have been the case as the shapes are topologically sound and the geodesic distances are computed robustly. The problem becomes clear when we examine the result of the exhaustive reordering of eigenvectors. It turns out that for this shape, the exhaustive reordering does not give the right ordering of the eigenvectors, as shown in Figure 4.9. After further investigation we find that that the problem lies with the crude L_2 cost measure used in arriving at the reordering. A more robust cost measure should be sought.

Note that the examples shown in the figures do not reflect the performance of our method when applied to datasets containing outliers. In general, input meshes for graphics applications are well defined and free of outliers. Hence, this is not a primary concern. However, spectral methods are generally believed to be non-robust to outliers, hence, appropriate measures are required to apply such methods on contaminated datasets. We provide a brief analysis on this topic in Section 4.7.

4.6.2 Shape Retrieval Results

We now present a comparative study of two global shape descriptors, the spherical harmonics descriptor [34] (SHD) and the light field descriptor [13] (LFD), in the context of shape retrieval. Both of these descriptors have been shown to give excellent shape retrieval results in the Princeton shape benchmark [48]. In fact, the light field descriptor is the best among all the descriptors compared in [48]. We evaluate the performance of the descriptors when applied to the original meshes as compared to when they are applied to the spectral embeddings of the meshes. We use McGill's database of articulated 3D shape [63] for our experiments. We also present two very simple descriptors, easily obtained from the spectral embeddings that perform better than the other descriptors.

The McGill articulated shape database contains 255 models in 10 categories: Ants,

Crabs, Hands, Humans, Octopuses, Pliers, Snakes, Spectacles, Spiders and Teddy-bears. There are 20 to 30 models in each category. Some shapes from the database are shown in Figure 4.5. We now give a brief explanation of the descriptors we compare.

1. **Light Field Descriptor (LFD) [13]**: represents the model using histograms of 2D images of the model captured from a number of positions, uniformly placed on a sphere. The distance between two models is the distance between the two descriptors minimized over all rotations between the two models, hence attaining robustness to rotations. The main idea of this descriptor is to define shape similarity based on the visual similarity of the two shapes.

2. **Spherical Harmonics Descriptor (SHD) [34]**: is a geometry based representation of the shape which is invariant to rotations. It is obtained by recording the variation of the shape using spherical harmonic coefficients computed over concentric spherical shells.

3. **Spectral Descriptors**: The following are two descriptors that can be easily obtained from the spectral embeddings. As shown by the experiments, these absolutely outperform the LFD and SHD descriptors and are concise and easily obtainable from the spectral embeddings. As described below, the EVD descriptor consists of only twenty numbers and performs better than LFD and SHD. This shows the effectiveness of the affinity matrix and spectral embeddings in encoding shape information.

   (a) **Eigenvalue Descriptor (EVD)**: Note that while the eigenvectors of the affinity matrix form the spectral embedding which is a normalized representation of the shape, the eigenvalues specify the variation of the shape along the axes defined by the corresponding eigenvectors. Hence, as a simple descriptor, we use the square root of the first twenty eigenvalues of the affinity matrix to describe a shape. Note that more than twenty eigenvalues can also be used. Infact, our experiments indicate that increasing the number of eigenvalues improves the results in the beginning, however, no noticeable improvements are obtained if the number is increased beyond twenty. Also, the eigenvalues tend to decrease very quickly, hence, only the largest eigenvalues are the ones that encode significant shape information. Note that the eigenvalues are affected by the number of vertices in the

shape. Since, there are shapes with different number of vertices in the database, the eigenvalues cannot be used for shape comparison as is. However, this is not an issue here, since we use eigenvalues obtained from Nyström approximation, for which, the number of samples is the same for all models in the database. The distance between two meshes P and Q is given by the χ^2-distance between the square root of their first twenty eigenvalues:

$$Dist_{EVD}(P,Q) = \frac{1}{2}\sum_{i=1}^{2} 0\frac{[\,|\lambda_i^P|^{\frac{1}{2}} - |\lambda_i^Q|^{\frac{1}{2}}]^2}{|\lambda_i^P|^{\frac{1}{2}} + |\lambda_i^Q|^{\frac{1}{2}}}$$

(b) **Correspondence Cost Descriptor (CCD)**: The distance between two shapes in the scheme of CCD is derived from the correspondence between the vertices of the two shapes. Given the respective k-dimensional spectral embeddings of P and Q in the form of an $n_P \times k$ matrix V_P and an $n_Q \times k$ matrix V_Q, the CCD distance between the two shapes P and Q is given by:

$$Dist_{CCD}(P,Q) = \sum_{p \in P} \|V_P(p) - V_Q(match(p))\|$$

Here, $V_P(p)$ and $V_Q(q)$ are the p^{th} and the q^{th} rows of V_P and V_Q, respectively, p represents a vertex of P, and $match()$ is some computed mapping of the vertices of P to the vertices of Q. This matching can be obtained using any correspondence algorithm, e.g., [30, 57, 11]. We have chosen to compute correspondence using the spectral embeddings we have already obtained from the previous step. The correspondence algorithm used is a simple best matching based on Euclidean distance in the embedding space [57]. Specifically,

$$match(p) = \text{argmin}_{q \in Q} \|V_P(p) - V_Q(q)\|$$

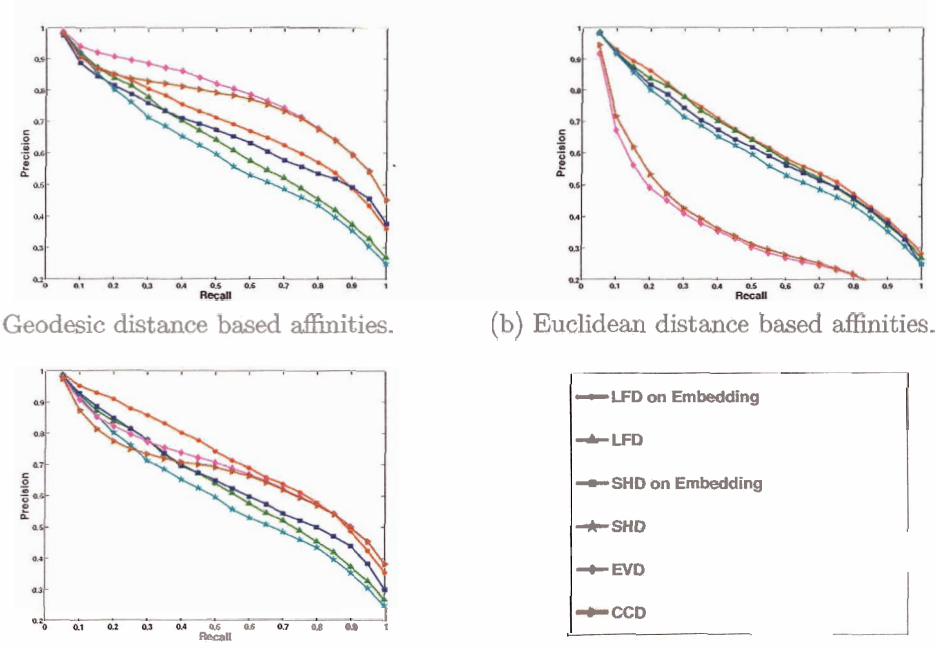
The intuition behind defining such a similarity cost is that if two shapes are similar (though they may differ by a bending transformation), their spectral embeddings would be similar, hence the Euclidean distance between a point and its match will be small, resulting in a smaller value of $Dist_{CCD}(P,Q)$. However, note that the time complexity of finding the distance between two shape in the CCD scheme is $O(n^2)$, where n is the number of vertices. This is extremely slow and is not feasible to apply for comparing the query model over and over again

with all the models in the database. Hence, we use CCD in conjunction with EVD. We first use EVD to filter out all poor matches by thresholding. Only the top few matches obtained from EVD are further refined using CCD.

We plot the precision-recall (PR) curves for the afore-mentioned descriptors when they are applied to the McGill database of articulated shapes, in Figure 4.10. In (a), the approximate geodesic distances are used to construct the affinities. Clearly, the descriptors show significant improvements when applied to bending invariant embeddings, compared to their spatial domain counterparts. In Figure 4.10(b), we show the performance of the same set of descriptors, however, we construct embeddings based on Euclidean distances. Note that the performance of spectral descriptors degrades considerably. This is mainly because spectral descriptors are naive descriptors that rely on the ability of the affinity matrix to normalize any transformations between the shapes. Since Euclidean distance based affinity matrix does not normalize the shapes against bending and the database in question is particularly that of articulated shapes, such poor performance is expected. Figure 4.10(c) shows the performance of the descriptors where the affinities are calculated using an average of geodesic and Euclidean distances. For LFD and SHD, both geodesic affinities and combined affinities give considerable improvements. Whereas, Euclidean affinities show only minor improvements since they fail to normalize the shapes against bending. EVD and CCD perform well only when the embeddings are bending normalized for the reason mentioned above.

The precision-recall curves plotted in Figure 4.10 show analytical results. We now show some visual results which emphasize the need for bending invariant spectral embeddings for more effective retrieval of articulated shapes. These results are shown in Figure 4.11. The spectral embeddings used in these results are all constructed using geodesic distance based affinities. Figure 4.11(a) shows the results of retrieving an ant shape from the database. Note the poor performance of SHD even when the amount of bending is moderate. Figure 4.11(b) and (c) show results for querying the database with a human and a plier shape, respectively, that have a relatively larger amount of bending. As we can see, LFD performs rather poorly on the original shapes.

It is quite evident from Figure 4.11 that shape descriptors applied to spectral embeddings show clear and consistent improvement over their spatial domain based counterparts. It is also interesting to note from Figure 4.11 that EVD, our simple shape descriptor based on eigenvalues, works the best. We have indeed observed that most, if not all, incorrect



(a) Geodesic distance based affinities.

(b) Euclidean distance based affinities.

(c) Combined distance based affinities.

Legends.

Figure 4.10: Precision-recall plots for various global descriptors, derived from different distance measures, when applied to the McGill database of articulated shapes [63].

retrieval results using EVD are caused by having parts of a shape incorrectly connected in our construction of the structural graph. Recovering the correct shape information from a soup of triangles or sparsely and nonuniformly sampled points (which occur often in the shape databases) is not an easy problem, but any improvements in this regard will improve the performance of the EVD even further. Our current heuristic is quite primitive and we would like to look into this problem in our future work.

In Figure 4.12, we show an image representations of the similarity matrix for all the shapes of the database. Here, a bright pixel represents greater similarity. The descriptors used are EVD and LFD on spectral embeddings. The diagonal structure of the image matrix shows that similar shapes have greater similarity value. Also, the matrix obtained from EVD has a more distinct diagonal structure than that obtained from LFD on embeddings.

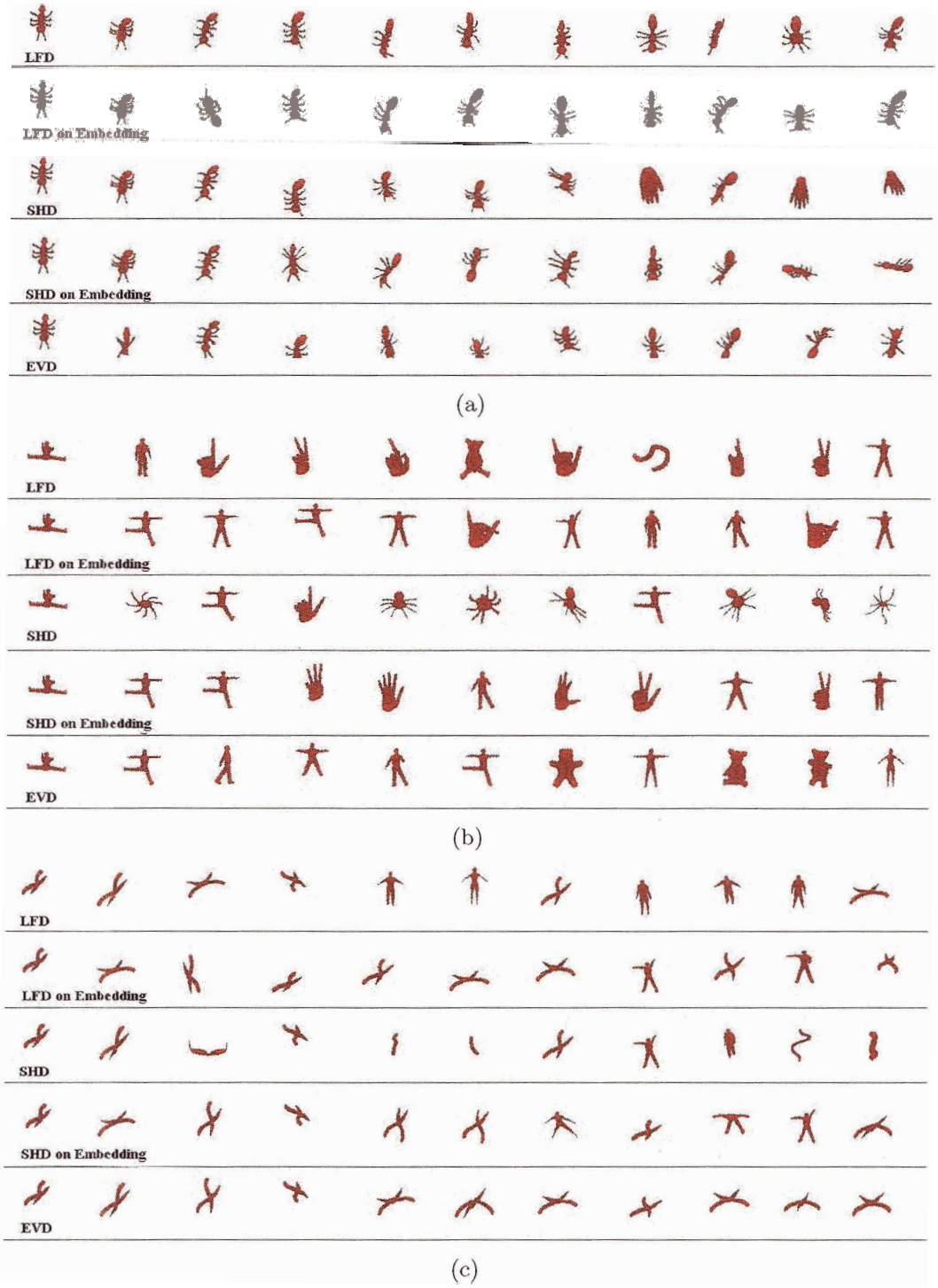


Figure 4.11: Retrieval results using the McGill database of articulated 3D shapes [63]: First column in each row is the query shape. This is followed by the top ten matches retrieved using the shape descriptor as indicated.

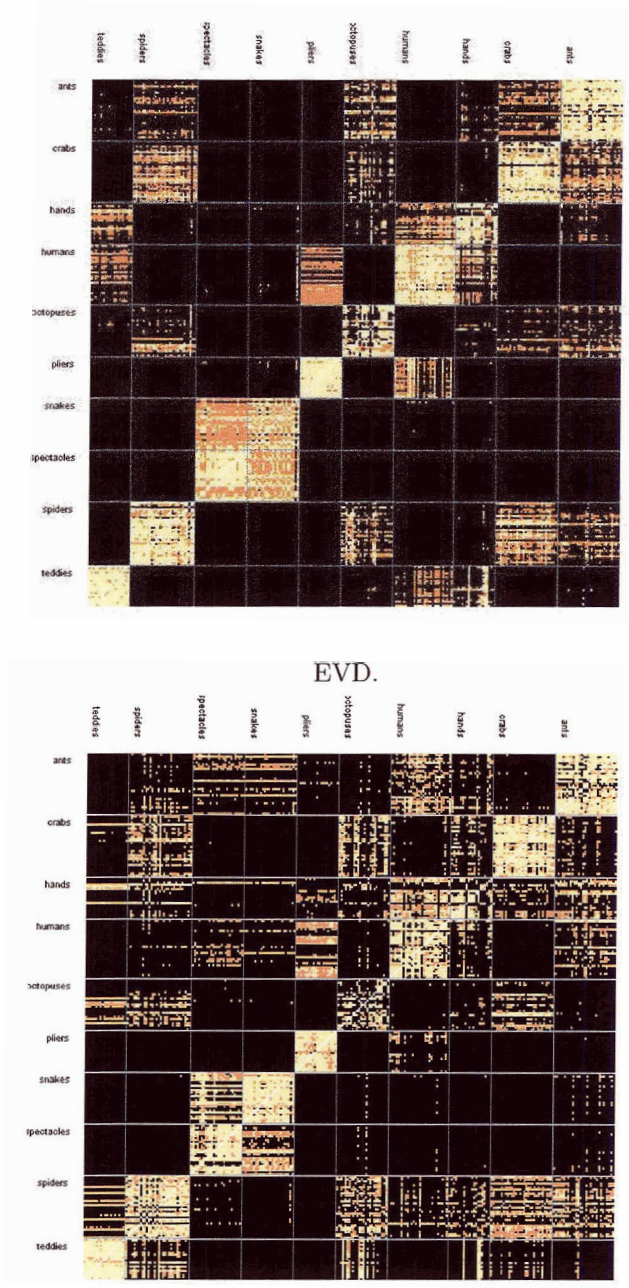


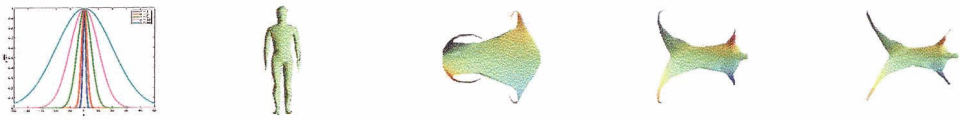
Figure 4.12: Similarity matrix for shapes from the McGill database of articulated 3D shapes [63], computed using EVD (top) and LFD on spectral embeddings (bottom).

4.7 Gaussian Width and Robustness to Outliers

One of the main drawbacks of spectral correspondence approaches reported in the related literature is their non-robustness against outliers in the data. The initial spectral correspondence method developed by Shapiro and Brady [57] and all subsequent modifications, for example, those of Hancock et al. [11, 10] and Caelli et al. [9], do not provide any analysis of their approaches with regards to the outlier problem. This is a prime concern when dealing with image data for computer vision problems as the result of edge or feature detection on images typically produce a large number of spurious data points. However, for graphics applications, this is not an issue of concern, as the triangular meshes are already well defined and they usually specify a surface, hence, outliers (if any) can be detected and removed easily. Nevertheless, in this Section, we provide a brief description of the outlier problem in spectral embeddings and explain its relation to the Gaussian width σ defined in equation 4.9.

Note that for a given affinity matrix A, the spectral embedding $\hat{A}_k$ defined in equation 4.2 is such that $\hat{A}_k \hat{A}_k^T$ is the best rank-k approximation of A. That is, the dot product of the i^{th} and the j^{th} row of $\hat{A}_k$ is an approximation to the ij^{th} entry of A. However, these rows are exactly the embedding coordinates of the i^{th} and j^{th} mesh vertices. Hence, it is clear that the affinities between the mesh vertices in the spatial domain are approximated by dot products of the embedded vertices in the spectral domain. Now, if an outlier is added to the mesh, the embedding will change so that the dot product between the original vertices and the newly added outlier approximate the corresponding affinities. Since, the dot product approximation is in a least-square sense, the embedding is fairly stable against a small number of outliers. However, as the number of outliers becomes large, the embedding can no longer be trusted. Nevertheless, this effect can be mitigated through appropriate choice of the Gaussian width.

Gaussian width specifies the "effective neighborhood" of a vertex. As shown in Figure 4.13(a), as the width is decreased, the Gaussian function tends to result in affinities closer to zero for large geodesic distances. This effect is seen in Figure 4.13(c), (d) and (e), where, as the Gaussian width is decreased, the embedding tends to *collapse*. Note that if the Gaussian width is decreased to almost zero, the affinity matrix becomes an identity matrix. Hence, the k-dimensional embedding in that case will be k points on the corners of a unit k-hypercube and the rest of the points will collapse to the origin. For example



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

Figure 4.13: Effect of changing Gaussian width: (a) The Gaussian function; (b) A human mesh; (c) Embedding with $\sigma = 0.5 \times Avg$; (d) Embedding with $\sigma = 1.0 \times Avg$; (e) Embedding with $\sigma = 1.5 \times Avg$. Here, Avg is the average geodesic distance between any two points on the mesh.

the two arms and legs come closer to the torso as the vertices on the arms and the legs are no longer affected by the points far from them. Hence, the extent, on the mesh, to which a vertex has influence, is reduced. Clearly, if the collapsing effect increases, the embedding will self intersect making it difficult for the matching process to distinguish between two points. This gives us a constraint that the Gaussian width must be sufficiently large.

Now we consider data with outliers. Note that the affinity matrix of a mesh with outliers can be written as:

$$A' = \begin{bmatrix} A & Z \\ Z^T & O \end{bmatrix},$$

where, A is the affinity matrix of the mesh without outliers, O contains the affinities between the outlier points and Z contains the affinities between outliers and mesh vertices. If A' is of the form of a block diagonal matrix (i.e. Z contains all zeros), then the eigenvectors of A' are given by the eigenvectors of A padded with zeros. Hence, we can extract the embedding of the original mesh easily from that of the contaminated mesh. This, however, would be the ideal situation which is generally not possible as setting the matrix Z to zeros would require full detection of outliers. This is where the Gaussian width can be used to make the matrix Z close to being zeros. Obviously a small Gaussian width would be favored to achieve this.

There is one more disadvantage of setting the Gaussian width small. With a small

Gaussian width, the affinity matrix becomes of the form:

$$A = \begin{bmatrix} A_1 & 0 & 0 & \cdots \\ 0 & A_2 & 0 & \cdots \\ 0 & 0 & \ddots & \\ \vdots & & & \end{bmatrix},$$

where, A_i are small matrices representing small effective neighborhoods of the mesh vertices. Note that, if the Gaussian width is small, the affinity matrix can always be transformed to the above form by permuting its rows and columns appropriately. This results in larger amount of eigenvector switching as the eigenvalues of A_i's would be approximately the same.

Experimental Results: It is clear that setting the Gaussian width to a value that provides a large effective neighborhood and is also robust to outliers would involve satisfying two conflicting goals, as large neighborhood can be achieved using large Gaussian width and robustness to outliers requires small Gaussian width. Moreover, we have shown that the right choice of the width mitigates the effect of outliers which is one of the main drawbacks of spectral correspondence. Indeed the conflicting nature of this problem is confirmed by our experiments. We proposed in earlier Sections to set the Gaussian width to the maximum geodesic distance on the mesh. For this experiment, however, we instead use the average of all geodesic distances as it is more robust to outliers. In all other experiments in this thesis, we have used maximum geodesic distance, as outliers are not an issue. In Figure 4.14, we plot the correspondence error against change in Gaussian width. For every experiment, we compute a matching between a mesh and a copy of itself with a number of outliers added at randomly selected locations. To add an outlier, a face of the mesh is selected randomly, and a vertex is added to the centroid of the face (connected to the three vertices of the face) and then pulled off the surface of the mesh resulting in a "spike" on the surface. The number of outliers is varied from 10 to 50. The meshes themselves contain 200 vertices. Since the original mesh vertices are not moved, the correct matching is trivial to compute. The error is then defined as the total geodesic distance between the matching point found by the algorithm and the correct matching point. The plots clearly indicate that choice of Gaussian width is important when dealing with data that contains outliers, and that in such a situation, lowering the Gaussian width can result in more robust matching. However, a definitive rule to choose Gaussian width requires further study.

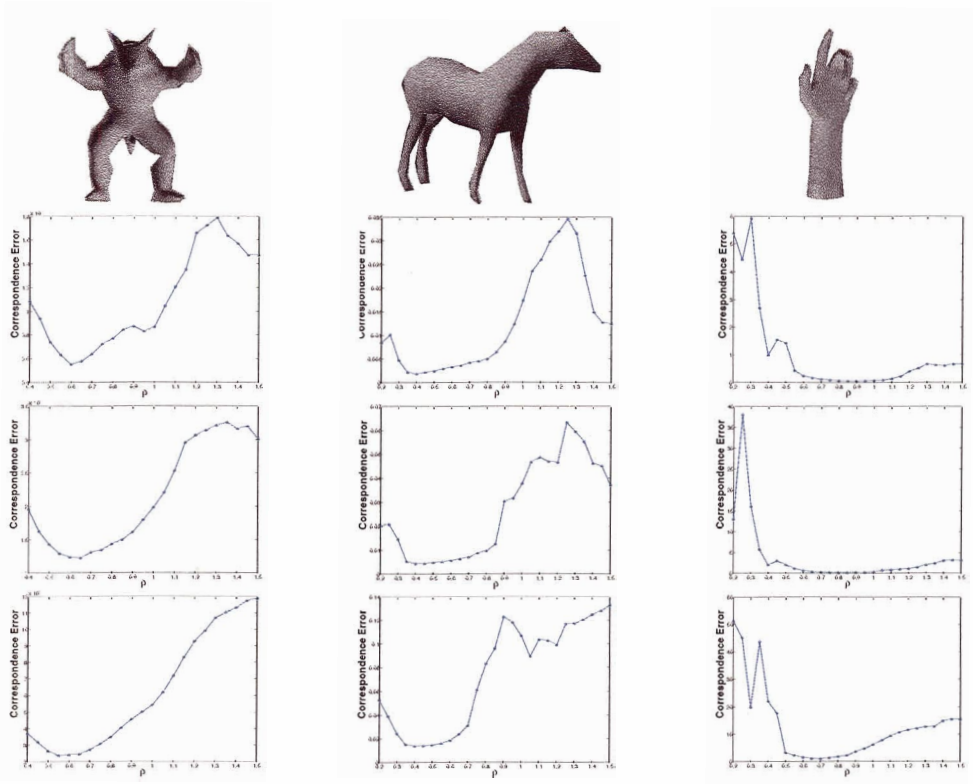


Figure 4.14: Correspondence error plots: in each plot, error is plotted against ρ, where, the Gaussian width is given by $\rho \times Avg$, where, Avg is the average geodesic distance. First row shows the mesh used, then from top to down are the plots showing correspondence error for 10, 20 and 50 outliers, respectively.

Chapter 5

Conclusions and Future Work

In this thesis, we have described methods for shape correspondence and retrieval. The main feature of our methods is the use of geodesic distances to construct intrinsic shape representation, which is subsequently used for correspondence and retrieval.

5.1 Geodesic Shape Context for 2D Shapes

For the case of 2D shapes, we have presented a new shape descriptor. Our method is simple and intuitive to understand and implement and is efficient since the final matching is quickly extracted by reading off the dissimilarity matrix. We experimentally show that our descriptor is robust against noise and non-rigid shape deformations such as bending and moderate amounts of stretching. Possible future work includes, generalizing the descriptor for 3D shape correspondence and building a framework for 3D shape searching and retrieval based on this descriptor. Since, the descriptor depends on the ordering of the neighborhood of a point, this generalization is not obvious. Moreover, estimation of curvature on a 3D meshes is typically more non-robust than curvature estimation on 2D contours, hence, extra measures are required to ensure the robustness of the descriptor.

Here, we also remark on the possible application of the spectral method to the correspondence of 2D contours. Note that the spectral method depends only on the definition of affinities between points. Clearly these affinities can be defined on 2D shapes as well. However, spectral embeddings based on geodesic affinities do not make much sense for 2D shapes. Note that any 2D contour can be obtained by performing arc-length preserving deformations on a circle. Since, the deformations are arc-length preserving, the geodesic

distance between two points on the contour will be preserved. Hence, the geodesic distance based spectral embedding of any contour will be same as that of a circle. Clearly such embedding cannot be used for correspondence or retrieval. Another option is to define affinities using Euclidean distances which is usually the case in most spectral correspondence algorithms. However, this is not robust for articulated shapes. Hence, we prefer the descriptor based approach for 2D shapes also because of the simplicity of the approach and the ease of computation of the descriptor.

5.2 Spectral Correspondence for 3D Shapes

For the case of 3D shapes, we have described a hybrid approach to finding a correspondence between the vertices of two 3D meshes. We first transform the meshes into the spectral domain, based on geodesic affinities, and then match the spectral embeddings after taking appropriate steps to ensure a consistent ordering and sign assignment of the eigenvectors. Eigenvalue scaling of the eigenvectors renders our algorithm robust against difference in mesh sizes and choice of the dimensionality of the embeddings. Our method does not need a pre-selected set of feature vertices and can be completely automated. It is invariant against rigid transformations, uniform scaling, and shape bending. Experimentally, we find it to be robust against moderate stretching in the shapes as well, relying on thin-plate splines for non-rigid alignment in the spectral domain, and it outperforms well-known existing shape correspondence schemes.

The time complexity for computing the spectral embeddings and the correspondence cost, provided that an ordering and signs of the eigenvectors have been determined, is $O(n^2 \log n)$, where n is the number of vertices in the larger mesh. Note that this is inherent to the spectral approach, since the first step, which computes the pair-wise affinities, already requires $O(n^2 \log n)$ time. However, we effectively reduce this complexity to $O(ln \log n)$, where l is the number of samples chosen, by using sub-sampling techniques and Nyström extrapolation [18] to obtain approximate spectral embeddings. To reduce the cost of extracting correspondences, where the naïve best matching would require $O(n^2)$ time, we can take advantage of the accurate alignments that have already been obtained and apply spatial partitioning to speed up the search for correspondence pairs.

One limitation of our current approach, in terms of computational cost, is its reliance on an exhaustive search to find a consistent eigenvector ordering and sign assignment. The

greedy reordering approach is fast but it does not always give the correct result. The pairwise flipping approach is also fast and works in almost all the cases in practice, however, it is more likely to give a locally minimum correspondence if the input meshes are not sufficiently similar. Moreover, the problem of sign flips is not solved if this heuristic is used. Analytically, the problem of finding a reordering and sign assignment which would lead to the best correspondence, e.g., according to the simple L_2 distance, is as hard as the graph isomorphism problem. It would be interesting to look into fast approximation algorithms for this problem and adopt it for our purpose.

Quality-wise, an important issue is related to the quality of the correspondence cost used in determining the eigenvector ordering and sign assignment. Currently, we are using L_2 and in some rare cases as shown earlier, even the exhaustive search would return a poor eigenvector ordering or sign assignment. This shows that a better cost function is still required. In addition, other limitations of our current method, including the handling of shape symmetry and non-manifold meshes, need to be addressed.

Finally, an investigation into possible definitions of the point affinities that are robust, if not invariant, to stretching within perceptually salient parts of a shape is needed. This would offer an alternative to using non-rigid alignment in the spectral domain and avoid having to find a consistent eigenvector ordering or sign assignment.

5.3 3D Shape Retrieval in Spectral Domain

We have also shown that spectral embeddings can be used for shape-based retrieval of 3D models from a database. Our focus here, is on articulated shapes. We present a method of making conventional shape descriptors invariant to shape articulation, with the use of spectral embeddings derived from an appropriately defined affinity matrix. The affinity matrix encodes pair-wise distances between the data points and invariance to a particular type of transformation can be achieved through a judicious choice of a distance measure. When conventional shape descriptors, e.g., LFD and SHD, are applied to spectral embeddings, derived from approximated geodesic distances, on the McGill database of articulated 3D shapes, absolute improvements are obtained for shape retrieval. Robustness of affinity matrices is also shown, as minor improvements for the LFD and SHD descriptors can be achieved, on the same database, even with Euclidean distance based affinities, which are not invariant to bending. We have also presented a new global shape descriptor, EVD, based on

the eigenvalues of the affinity matrix and empirically shown that its performance is better than that of LFD and SHD.

In general, spectral methods can be sensitive to the presence of outliers in the data. However, this issue is not of great concern for 3D model correspondence and retrieval, as the 3D models are mostly free of outliers. Moreover, since most models define a surface, outliers are easy to detect and remove. Nevertheless, robustness to outliers is necessary with respect to retrieval and recognition of more general form of data and we suspect that further study into appropriate definition of Gaussian width can help in achieving this robustness.

Appendix A

Thin Plate Splines

The thin plate spline, pioneered by Bookstein [8], is a generalization of cubic splines to higher dimensions and it contains affine transformation as a special case. With non-rigid transformations, there are infinitely many ways of transforming a point set into another. Thin plate splines are effective because of their smoothness constraints which discourage arbitrary mappings. In the limit of this smoothness constraint the thin plate spline model reduces to an affine transformation model. The thin plate spline transformation functions $f_p : x \in \mathbf{R}^k \rightarrow \mathbf{R}, 1 \leq p \leq k$ map a point set $X = \{x_1, x_2, \ldots, x_n\}$ in k (say $k = 2$) dimensional space to another point set $Y = \{y_1, y_2, \ldots, y_n\}$ by minimizing the following energy functionals:

$$\begin{aligned} E(f_p) = & \sum_{i=1}^{n} \|y_{ip} - f_p(x_i)\| + \\ & \lambda \int\int \left[(\frac{\partial^2 f_p}{\partial x^2})^2 + 2(\frac{\partial^2 f_p}{\partial x \partial y})^2 + (\frac{\partial^2 f_p}{\partial y^2})^2 \right] dxdy \end{aligned} \tag{A.1}$$

where λ is the regularization (smoothing) parameter. Note that the correspondence between X and Y is assumed to be given. Hence, point $y_i = (y_{i1}, y_{i2}, \ldots, y_{ik})$ is the matching point for x_i. The unique set of f_p's that minimize the above energy functionals can be written in mattrix form as:

$$f(x_i, d, w) = x_i \cdot d + \phi(x_i) \cdot w,$$

where x_i is now in $(k+1)$-dimensional homogeneous coordinates, d is a $(k+1) \times (k+1)$ affine transformation matrix, w is an $n \times (k+1)$ warping coefficients matrix and $\Phi(x_i)$ is a vector of length n such that $\phi_j(x_i) = -\|x_j - x_i\|$.

As shown in [6], the transformation (d, w) that minimizes the energy can be calculated by solving the following system:

$$\begin{bmatrix} K & X^T \\ X & 0 \end{bmatrix} \begin{bmatrix} w \\ d \end{bmatrix} = \begin{bmatrix} Y \\ 0 \end{bmatrix} \tag{A.2}$$

Here, K is the matrix $(\Phi - \lambda I)$ where, I is an identity matrix of appropriate size and Φ is an $(n \times n)$ matrix whose i^{th} row is $\phi(x_i)$, that is, $\Phi_{ij} = -\|x_j - x_i\|$.

Using these transformation parameters, we transform the point set X to point set Y and then recompute the correspondence. This process is iterated until convergence as described in Section 4.3.

Appendix B

Nyström Approximation

The computation of spectral embeddings described in this thesis starts by building a matrix which encodes certain relationship, called affinities, between each pair of vertices in a mesh. This matrix is then eigendecomposed and its eigenvectors and eigenvalues are used to obtain the spectral embeddings.

Computing spectral embeddings is time-consuming due to the quadratic complexity, in terms of the data size, of affinity computation and up to cubic-time complexity for eigenvalue decomposition. The affinity computation complexity increases to $O(n^2 \log n)$ for our case since the affinities are based on geodesic distances. Nyström method [18, 66] is therefore proposed to overcome this problem via sub-sampling and reconstruction.

Consider a set of n points $\mathcal{Z} = \mathcal{X} \bigcup \mathcal{Y}$, where $\mathcal{X}$ and $\mathcal{Y}$, $\mathcal{X} \bigcap \mathcal{Y} = \varnothing$, of sizes l and m, respectively, give a partition of $\mathcal{Z}$. Write the symmetric affinity matrix $W \in \mathbb{R}^{n \times n}$ in block form:

$$W = \begin{bmatrix} A & B \\ B^T & C \end{bmatrix},$$

where $A \in \mathbb{R}^{l \times l}$ and $C \in \mathbb{R}^{m \times m}$ are affinity matrices for points in $\mathcal{X}$ and $\mathcal{Y}$, respectively; $B \in \mathbb{R}^{l \times m}$ contains the *cross-affinities* between points in $\mathcal{X}$ and $\mathcal{Y}$. Without loss of generality, we designate the points in $\mathcal{X}$ as *sample points*. Let $A = U \Lambda U^T$ be the eigenvalue decomposition of A, then the eigenvectors of W can be approximated, using the Nyström method [18], as

$$\bar{U} = \begin{bmatrix} U \\ B^T U \Lambda^{-1} \end{bmatrix}. \tag{B.1}$$

This allows us to approximate the eigenvectors of W by only knowing the sampled sub-block

$[A\ B]$. The overall complexity is thus reduced from $O(n^3)$, without sub-sampling, down to $O(ln \log n) + O(l^3)$, where $l \ll n$, in practice.

The rows of $\bar{U}$ define the spectral embeddings of the original data points from $\mathcal{Z}$. From (B.1), we see that the i^{th} row of U, which is completely determined by A, gives the embedding $\bar{x}_i$ of point x_i in $\mathcal{X}$ and the j^{th} row of $B^T A^{-1} B$ is the embedding $\bar{y}_j$ of point y_j in $\mathcal{Y}$. If we let $\bar{y}_j^d$ denote the d^{th} component of $\bar{y}_j$, then equation (B.1) can be rewritten as

$$\bar{y}_j^d = \frac{1}{\lambda_d} \sum_{i=1}^{l} \bar{x}_i^d B(i,j) = \frac{1}{\lambda_d} \sum_{i=1}^{l} \bar{x}_i^d W(i, j+l), \qquad 1 \leq d \leq l. \tag{B.2}$$

Namely, the embedding $\bar{y}_j$ is extrapolated using the coordinates of the $\bar{x}_i$'s, weighted by the corresponding cross-affinities in B.

With $\bar{U}$, we obtain an approximation $\bar{W}$ of the original affinity matrix W,

$$\bar{W} = \bar{U} \Lambda \bar{U}^T = \begin{bmatrix} A & B \\ B^T & B^T A^{-1} B \end{bmatrix}.$$

Max-Min sampling scheme: The quality of Nyström approximation depends on the samples selected over the mesh as suggested by our experiments [43]. We have found the following simple sampling scheme to give excellent results: The sample points are chosen one at a time. At any time, the i^{th} sample point is the mesh vertex that maximizes the minimum geodesic distance to the already chosen $(i-1)$ sample points. The first sample point is chosen as follows: select a random mesh vertex (v_0), then find the vertex (v_1) farthest (geodesically) to v_0, then find v_2 farthest from v_1. After several such iterations point v_j is chosen to be the first sample. This process will most likely place the first sample close to an extremity of the mesh.

The motivation behind such sampling scheme is to select sample points mutually far away from each other. Note that $A^{-1}\mathbf{1}$ gives the coefficients of the expansion of $\mathbf{1}$ in the space whose basis are the columns of A. Moreover, the diagonals of A are 1 and its other entries lie in $(0, 1)$. It is easy to show, in 2D, that the sum of these coefficients, $\mathbf{1}^T(A^{-1}\mathbf{1})$, is no larger than l. The maximal l is obtained when A's columns are the canonical basis of the Euclidean space. This is generalizable to arbitrary dimensions. In order for A's columns to be close to the canonical basis, the off-diagonal entries should be close to zero. Thus samples should be taken mutually far away from each other. More details and a comparison of various sampling schemes can be found in [43].

Bibliography

[1] M. Alexa. Recent advances in mesh morphing. *Computer Graphics Forum*, 21(3):173–196, 2002.

[2] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, H. Pang, and J. Davis. The correlated correspondence algorithm for unsupervised registration of nonrigid surfaces. In *Proc. NIPS*, pages 33–40, 2004.

[3] M. Ankerst, G. Kastenmüller, H.-P. Kriegel, and T. Seidl. Nearest neighbor classification in 3d protein databases. In *Proc. ISMB*, pages 34–43, 1999.

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

[5] S. Belongie, J. Malik, and J. Puzicha. Matching shapes. In *Proc. ICCV*, pages 454–463, 2001.

[6] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. on PAMI*, 24:509–523, 2002.

[7] P. J. Besl and N. D. Mckay. A method for registration of 3d shapes. *IEEE Trans. on PAMI*, 14:239–256, 1994.

[8] F. L. Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE Trans. on PAMI*, 11:567–585, 1989.

[9] T. Caelli and S. Kosinov. An eigenspace projection clustering method for inexact graph matching. *IEEE Trans. on PAMI*, 26(4):515–519, 2004.

[10] M. Carcassoni and E. R. Hancock. Correspondence matching with modal clusters. *IEEE Transactions on PAMI*, 25(12):1609–1615, 2003.

[11] M. Carcassoni and E. R. Hancock. Spectral correspondence for point pattern matching. *Pattern Recognition*, 36:193–204, 2003.

[12] J. Chan, H. Tek, , and B. Kimia. Symmetry-based indexing of image databases. *J. Visual Communication and Image Representation*, 9:268–278, 1998.

[13] D.-Y. Chen, X.-P. Tian, Y.-T. Shen, and M. Ouhyoung. On visual similarity based 3d model retrieval. In *Computer Graphics Forum*, pages 223–232, 2003.

[14] H. Chui and A. Rangarajan. A new point matching algorithm for non-rigid registration. In *Proc. CVIU*, volume 89, pages 114–141, 2003.

[15] C. M. Cyr and B. B. Kimia. 3d object recognition using shape similarity-based aspect graph. In *Proc. Int. Conf. on Computer Vision*, pages 254–261, 2001.

[16] C. Eckart and G. Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1:211–218, 1936.

[17] A. Elad and R. Kimmel. On bending invariant signature of surfaces. *IEEE Trans. on PAMI*, 25(10):1285–1295, 2003.

[18] C. Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral grouping using the nyström method. *IEEE Trans. on PAMI*, 26(2):214–225, 2004.

[19] A. Frome, D. Huber, R. Kolluri, T. Bulow, and J. Malik. Recognizing objects in range data using regional point descriptors. In *European Conference on Computer Vision (ECCV)*, 2004.

[20] F. Funkhouser, P. Min, M. Kazhdan, J. Chen, A. Halderman, D. Dobkin, and D. Jacobs. A search engine for 3d models. *ACM Trans. Graph.*, 22(1):83–105, 2003.

[21] R. Gal and D. Cohen-Or. Salient geometric features for partial shape matching and similarity. Under revision for ACM Trans. on Graphics, 2005.

[22] M. Garland. Qslim simplification software — qslim 2.1.

[23] T. Gatzke, C. Grimm, M. Garland, and Zelinka S. Curvature maps for local shape comparison. In *Proc. SMI*, pages 246–255, 2005.

[24] N. Gelfand, N. Mitra, L. Guibas, and H. Pottmann. Robust global registration. In *Proc. SGP*, pages 197–206, 2005.

[25] G. H. Golub and C. F. Van Loan. Matrix computations. In *John Hopkins University Press*, 1996.

[26] C. Gotsman, X. Gu, and A. Sheffer. Fundamentals of spherical parameterization for 3d meshes. *ACM Transaction on Graphics*, 22:358–363, 2003.

[27] H. Hilaga, Shinagawa Y., Kohmura T., and T. K. Kunii. Topology matching for fully automatic similarity estimation of 3d shapes. In *Proc. of SIGGRAPH*, pages 203–212, 2001.

[28] D. D. Hoffman and W. A. Richards. Parts of recognition. *Cognition*, 18:65–96, 1984.

[29] A. Jain. Fundamentals of digital image processing. In *Prentice Hall*, 1989.

[30] V. Jain and H. Zhang. Robust 3d shape correspondence in the spectral domain. In *Proc. of Shape Modeling International*, page to appear, 2006.

[31] A. Johnson and M. Hebert. Recognizing objects by matching oriented points. In *Proc. CVPR*, pages 684–689, 1997.

[32] A. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21:433–449, 1998.

[33] S. Kang and K. Ikeuchi. Determining 3d object pose using the complex extended gaussian image. In *Proc. of Computer Vision and Pattern Recognition*, pages 580–585, 1991.

[34] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz. Rotation invariant spherical harmonic representation of 3d shape descriptors. In *Symposium on Geometry Processing*, pages 167–175, 2003.

[35] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz. Shape matching and anisotropy. *ACM Trans. Graph.*, 23(3):623–629, 2004.

[36] R. Kolluri, J. R. Shewchuk, and J. F. O'Brien. Spectral surface reconstruction from noisy point clouds. In *Proc. of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 11–21, 2004.

[37] M. Körtgen, G.-J. Park, M. Novotni, and R. Klein. 3d shape matching with 3d shape contexts. In *Seventh Central European Seminar on Computer Graphics*, 2003.

[38] V. Kraevoy and A. Sheffer. Cross-parameterization and compatible remeshing of 3d models. *ACM Transactions on Graphics*, 23:861–869, 2004.

[39] V. Kraevoy, A. Sheffer, and C. Gotsman. Matchmaker: Constructing constrained texture maps. In *ACM SIGGRAPH*, pages 326–333, 2003.

[40] X. Li and I. Guskov. Multiscale features for approximate alignment of point-based surfaces. In *Proc. SGP*, pages 217–226, 2005.

[41] H. Ling and D. W. Jacobs. Using the inner-distance for classification of articulated shapes. In *Proc. CVPR*, pages 719–726, 2005.

[42] L. Liu, G. Wang, B. Zhang, B. Guo, and H.-Y. Shum. Perceptually based approach for planar shape morphing. In *Proc. Pacific Graphics*, pages 111–120, 2004.

[43] R. Liu, V. Jain, and H. Zhang. Subsampling for efficient spectral mesh processing. In *Proc. Computer Graphics International*, pages 172–184.

[44] K. Mikolajczyk and C. Schimid. A performance evaluation of local descriptors. *IEEE Transactions on PAMI*, 17:1615–1630, 2005.

[45] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *NIPS*, pages 857–864, 2002.

[46] R. Ohbuchi, T. Minamitani, and T. Takei. *Shape-similarity Search of 3D Models by Using Enhanced Shape Functions*, pages 97–104. 2003.

[47] R. Osada, T. Funkhouseri, B. Chazelle, and D. Dobkin. Matching 3d shapes with shape distribution. In *Proc. of Shape Modeling International*, pages 154–166, 2001.

[48] M. Kazhdan P. Shilane, P. Min and T. Funkhouser. The princeton shape benchmark. In *Shape Modelling International*, pages 167–178, 2004.

[49] E. Praun, W. Sweldens, and P. Schr. Consistent mesh parameterizations. In *ACM SIGGRAPH*, pages 179–184, 2001.

[50] S. Rusinkiewicz and M. Levoy. Efficient variants of the icp algorithm. In *Proc. The Third International Conference on 3-D Digital Imaging and Modeling*, pages 145–152, 2001.

[51] B. Scholkopf, A. Smola, and K.-R. Muller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

[52] J. Schreiner, A. Asirvatham, E. Praun, and H. Hoppe. Inter-surface mapping. *ACM Transaction on Graphics*, 23(3):870–877, 2004.

[53] C. Scott and R. Nowak. Robust contour matching via the order preserving assignment problem. In *Technical Report TREE 0406, Department of Electrical and Computer Engineering, Rice University*, 2004.

[54] G. Scott and H. Longuet-Higgins. An algorithm for associating the features of two patterns. *Royal Soc. London*, 1991.

[55] G. L. Scott and H. C. Longuet-Higgins. Feature grouping by relocalisation of eigenvectors of the proximity matrix. In *British Machine Vision Conference*, pages 103–108, 1990.

[56] T. W. Sederberg and E. Greenwood. A physically based approach to 2d shape blending. In *Proc. SIGGRAPH*, volume 26, pages 25–34, 1992.

[57] L. S. Shapiro and Brady J. M. Feature based correspondence: An eigenvector approach. *Image and Vision Computing*, 10(5):283–288, 1992.

[58] H. Shum. On 3d shape similarity. In *Proc. of Computer Vision and Pattern Recognition*, pages 526–531, 1996.

[59] R. Sumner and J. Popovic. Deformation transfer for triangle meshes. *ACM Transactions on Graphics*, 23:399–405, 2004.

[60] T. Tangelder and R. Veltkamp. Polyhedral model retrieval using weighted point sets. In *Proc. of Shape Modeling International*, pages 119–129, 2003.

[61] T. Tangelder and R. Veltkamp. A survey of content based 3d shape retrieval methods. In *Proc. of Shape Modeling International*, pages 145–156, 2004.

[62] S. Umeyama. An eigen decomposition approach to weighted graph matching problem. *IEEE Transaction on PAMI*, 10:695–703, 1988.

[63] McGill University. Mcgill 3d shape benchmark: *http://www.cim.mcgill.ca/ shape/benchmark/*.

[64] R. C. Veltkamp and M. Hagedoorn. State of the art in shape matching. In *Technical Report UU-CS-1999-27, Utrecht*, 1999.

[65] D. Vranic. An improvement of rotation invariant 3d shape descriptor based on functions on concentric spheres. In *IEEE Int. Conf. on Image Processing*, pages 757–760, 2003.

[66] C. Williams and M. Seeger. Using the nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems*, pages 682–688, 2001.

[67] L. Yang. k-edge connected neighborhood graph for geodesic distance estimation and nonlinear data projection. In *International Conference on Pattern Recognition (ICPR)*, volume 1, pages 196–199, 2004.

[68] T. Zaharia and F. Prêteux. 3d shape-based retrieval within the mpeg-7 framework. In *Proc. SPIE Conference 4304*, pages 133–145, 2001.

[69] R. Zayer, C. Rössl, Z. Karni, and H.-P. Seidel. Harmonic guidance for surface deformation. In *Computer Graphics Forum (Proceedings of Eurographics)*, volume 24, pages 601–609, 2005.

[70] H. Zhang and R. Liu. Mesh segmentation via recursive and visually salient spectral cuts. In *Vision, Modeling, and Visualization*, pages 429–436, 2005.

[71] J. Zhang, K. Siddiqi, D. Macrini, A. Shokoufandeh, and A. Dickinson. Retrieving articulated 3d models using medial surfaces and their graph spectra. In *Int. Workshop On Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 285–300, 2005.

[72] Y. Zhang. A fuzzy approach to digital image warping. *IEEE Computer Graphics and Applications*, 16:33–41, 1999.

[73] T. Zinber, J. Schmidt, and H. Niemann. A refined icp algorithm for robust 3d correspondence estimation. In *Proc. of International Conference on Image Processing*, pages 695–698, 2003.