Shape Correspondence using Spectral Methods and Deep Learning

Alon Shtern
Shape Correspondence using Spectral Methods and Deep Learning

Research Thesis

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy

Alon Shtern

Submitted to the Senate of the Technion — Israel Institute of Technology

Av 5777 Haifa August 2017
This research was carried out under the supervision of Prof. Ron Kimmel, in the Faculty of Computer Science.

Some results in this thesis have been published as articles by the author and research collaborators in conferences and journals during the course of the author’s doctoral research period, the most up-to-date versions of which being:


The generous financial help of the Technion is gratefully acknowledged. Research leading to these results was supported by ERC - NORDIA advanced grant no. 267414.
# Contents

List of Figures

Abstract 1

1 Introduction 3
   1.1 Spectral Embedding 4
   1.2 Spectral Distance 5
   1.3 Nonrigid Shapes Deformation 6
   1.4 Volumetric Optical Flow 7

2 Matching the LBO Eigenspace of Non-Rigid Shapes via High Order Statistics 9
   2.1 Introduction 9
      2.1.1 Related Work 11
      2.1.2 Background 11
   2.2 Eigenfunction Matching 13
      2.2.1 Problem Formulation 13
      2.2.2 Matching Cost Function 14
      2.2.3 Solving the Minimization Problem 18
   2.3 Results 19
   2.4 Conclusions 23
   2.5 Discretization 25
      2.5.1 Laplace–Beltrami Eigendecomposition 25
      2.5.2 Gradient 25
   2.6 Application Specific Parameters 26

3 Spectral Gradient Fields Embedding for Nonrigid Shape Matching 29
   3.1 Introduction 29
      3.1.1 Contribution 30
      3.1.2 Metric Geometry 32
      3.1.3 The gradient and the Laplace-Beltrami operator 33
      3.1.4 Spectral embedding 33
   3.2 Spectral gradient fields embedding 34
List of Figures

2.1 The eigenfunction matching of two nearly isometric shapes. Hot and cold colors represent positive and negative values, respectively. (Top) The first pose of a dog; (center) the second pose of a dog; (bottom) the second pose of a dog after the matching algorithm. . . . . . . . . . . . . 20
2.2 Feature point correspondence of two nearly isometric shapes of a horse. 21
2.3 Eigenfunction matching of two nearly isometric shapes. Hot and cold colors represent positive and negative values, respectively. (Top) The first pose of a human; (center) the second pose of a human; (bottom) the second pose of a human after the matching algorithm. . . . . . . . . . . . . 21
2.4 Feature point correspondence of two nearly isometric shapes of a horse. 22
2.5 Eigenfunction matching of two nearly isometric shapes. Hot and cold colors represent positive and negative values, respectively. (Top) The first pose of a horse; (center) the second pose of a horse; (bottom) the second pose of a horse after the matching algorithm. . . . . . . . . . . . . 22
2.6 Feature point correspondence of two nearly isometric shapes of a horse. 23
2.7 Evaluation of the iterative spectral kernel maps’ correspondence framework applied to shapes from the TOSCA database, using the protocol of [KLF11] . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 24
2.8 Evaluation of the iterative closest spectral kernel maps’ correspondence framework applied to shapes from the SCAPE database, using the protocol of [KLF11]. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 24
3.1 Inner and external products of the gradient fields of a pair of eigenfunctions $\phi_1$, $\phi_3$. The inner product (third column, and bottom figure) exhibits finer details of the neck and hands of the human shape. Notice in the forth column, that the difference between front and back is captured by the external product. In the last two rows we see that these features are stable under a natural pose of the articulated object. In all our figures, red and blue colors represent high and low values of scalar functions on the shapes. . . . . . . . . . . . . . . . . . . . . . . . . 31
3.2 Spectral density of the spectral GFs feature vector computed on the human shape, compared to the spectrum of the GPS embedding. . . . . . 37
3.3 The spectral GFs feature vector induced the distance function $\tilde{d}_X(p,x)$ between a fixed point $p$ to all points $x \in X$ (left), and the distance function $\tilde{d}_Y(q,y)$ from the corresponding point $q = \varphi(p)$ to all points on shape $Y$ (right).

3.4 In the common space, the inter-shape distance $d_{X,Y}(p,y)$ is calculated between a fixed point $p$ on shape $X$ and all points in shape $Y$. From left to right: spectral GFs with the embeddings $J^\Phi$, $J^\Phi_\omega$ (inner-products), $J^\Phi_\nu$ (external products) and the GPS embedding.

3.5 The empirical probability that there is an eigenfunction $\phi_j^Y$ that matches $\phi_i^X$ up to a sign. The matching threshold is set to 0.1, i.e., we check if $E(i,j) < 0.1$ for some $j$ (see Eq. (3.4)). Notice, that this probability drops sharply beyond the seventh eigenfunction.

3.6 Evaluation of the correspondence system applied to shapes from the SCAPE database, using the protocol of [KLF11], (with allowed symmetries).

3.7 Evaluation of the correspondence system applied to shapes from the TOSCA database, using the protocol of [KLF11].

3.8 Bilateral symmetry. The two mirror halves of several shapes from the TOSCA database are found by embedding the shape using spectral gradient fields.

4.1 Evaluation of the iterative spectral kernel maps algorithm applied to shapes from the TOSCA database, using the protocol of [KLF11].

4.2 Evaluation of the iterative closest spectral kernel maps algorithm applied to shapes from the SCAPE database, using the protocol of [KLF11] with allowed symmetries.

4.3 Evaluation of the iterative closest spectral kernel maps algorithm initialized by landmark points.

4.4 Evaluation of the ICSKM algorithm applied to non-isometric shapes from the TOSCA database. For the primates and animals categories, the algorithm is provided with pairs of 18 and 11 landmark points, respectively. The distortion curves are evaluated by calculating the geodesic error of 18 corresponding points for the primates category, and 10 corresponding points for the animals category.

4.5 Texture mapping of non-isometric shapes. The textures of wolf and dog shapes were transferred to cat and horse shapes, respectively. The ICSKM algorithm was initialized by 7 landmark points selected using the farthest point strategy [HS85].

4.6 Symmetry axis of several shapes from the TOSCA database.
5.1 Deformation using the example-based LBO dictionary. The left portion of
the figure shows the cat and centaur ground-truth target shapes (colored in
gray). On the right we show the near perfect representation of these target
shapes by a linear combination of the dictionary’s atoms. In this case, the
weighting functions are the eigenfunctions of the Laplace-Beltrami operator that
correspond to the lowest 15 eigenvalues. The exemplar shapes that are used
to extract the example-based LBO dictionaries for representing the shapes are
shown inside the box.

5.2 Automatic feature point correspondence and shape interpolation. The four
examples of a horse (top middle) and the two sets of vertex positions (top
left, top right) were used to generate a sequence of frames. Correspondence of
the points on the four legs (circled in blue) was detected by minimizing the
example-based deformation energy for all permissible correspondences. The
example-based deformations (bottom left and right) were then interpolated at
four times the original frame rate to produce the movie sequence (bottom).

5.3 The four dog shapes are used as examples for our method (left). The deformed
shape (right) is found from the vertex positions (middle). In this case the
deformed shape is 50% larger than the reference ones.

5.4 Example-based deformed shapes from few vertex positions of a hand shape.

5.5 Nonrigid ICP. Left to right: The three exemplar shapes, the uncorrupted and
complete target shape, the acquired partial shape with eight known feature
points (marked as red dots), the initial deformation using the feature points
and the final deformation after applying the nonrigid ICP.

5.6 Evaluation of the shape completion and registration procedure applied to shapes
from the TOSCA database.

5.7 Deformed shapes constructed by omitting some of the steps in the proposed
example-based framework.

6.1 Multi-scale architecture.

6.2 Generative network $G_s$.

6.3 Convolution and multiplication neural network.

6.4 Turbulence prediction. In each of the 3 examples, the first row are the 6 ground
truth pressure slices, and the second row is their respective predictions.

6.5 Cardiac cycle prediction.

6.6 Cardiac cycle prediction.

6.7 Comparison of different methods to predict the next frame from UCF101. Left
to right: ground truth, V-Flow, V-Flow-Sharp, adversarial, optical flow.

6.8 Comparison of different methods on UCF101.

6.9 Comparison of different methods on UCF101.
Abstract

The interest in acquiring and analyzing three dimensional representations of the world is ever increasing, fueling a wide range of computer vision algorithms in the field of geometry processing. Spectral analysis has become key component in many applications involving non-rigid shapes modeled as two-dimensional surfaces, and recently, convolutional neural networks have shown remarkable success in a variety of computer vision tasks. In this dissertation, we designed a set of methods and tools that use these paradigms for applications such as shape correspondence, nonrigid deformations, and volumetric optical flow.

Finding the correspondences between pairs of shapes is a fundamental operation in the field of geometry processing. Measures of dissimilarity between surfaces have been found highly useful for this task. A powerful approach for measuring distance between two nonrigid shapes is to embed their two-dimensional surfaces into some common Euclidean space, defining the comparison task as a problem of rigid matching in that space. We introduce a novel spectral embedding, named the Spectral Gradient Fields Embedding, which exploits the local interaction between the eigenfunctions of the Laplace-Beltrami operator and the extrinsic geometry of the surface. The common embedding relies on the assumption that the Laplace-Beltrami eigenfunctions computed on each shape independently are compatible with each other. However, this assumption is often unrealistic. We address this issue by matching a small number of eigenfunctions, that are relatively stable, using a high order statistics (HOS) approach.

We also analyze the applicability of the spectral kernel distance as a measure of dissimilarity between surfaces, for solving the shape matching problem. To align the spectral kernels, we developed the Iterative Closest Spectral Kernel Maps (ICSKM) algorithm. ICSKM extends the Iterative Closest Point (ICP) algorithm to the class of deformable shapes. Instead of aligning the shapes in the three dimensional Euclidean domain, this method estimates the transformation that best fits the embeddings of the shapes into the spectral domain.

In case the data consists of an incomplete, occluded and disconnected parts of a shape the approach we took is to use a small set of representative natural poses. Using these few exemplar shapes, the method expresses an unseen appearance by a low-dimensional linear subspace, specified by a redundant dictionary of weighted vertex positions. The algorithm, called Fast Blended Transformations (FBT), finds the deformation that best
fits the partial data, minimizing a nonlinear functional that regulates the example manifold in a smooth way, and detects the pointwise mapping between the partial shape and the reference shapes.

Volumetric optical flow is a different way to describe the matching in a three-dimensional dynamic scene. We designed a multi-scale optical flow based architecture for predicting the next frame of a sequence of volumetric images. The fully differentiable model consists of specific crafted modules that are trained on small patches in an unsupervised manner. The approach, called V-Flow, is useful for analyzing the temporal dynamics of three-dimensional images in applications that involve, for example, motion of viscous fluid substances or real magnetic resonance imaging (MRI).
Chapter 1

Introduction

Establishing a meaningful shape correspondence is a challenging task in geometry processing, and is an important component of various three-dimensional shape processing applications, such as morphing, attribute transfer, shape deformation, shape interpolation and scan alignment. A fundamental problem of shape correspondence is to find point-to-point dense correspondences between isometric shapes. Two shapes are perfectly isometric, if there exists a distance-preserving mapping between the two shapes such that the geodesic distance between two points on one shape exactly equals the geodesic of the corresponding points on the other shape. A few natural expansions to this fundamental problem can farther be explored.

- Correspondence refinement. In this case we aim to interpolate an initial map that may be sparse into a dense point to point map.

- Nonisometry shape correspondence, where the isometry assumption is violated.

- Correspondence of a shape to itself, that would naturally find the symmetry axis of the shape.

- Partial matching, where at least one of the shapes consists of parts that may be incomplete, occluded or disconnected. A variation of this challenge is to complete the three dimensional detailed representation of the occluded and noisy shape.

- Optical flow that represents the apparent motion of objects, surfaces, and edges in a visual scene.

In this research we developed a set of methods that can be used to overcome these challenges. The Laplace-Beltrami eigenfunctions and their respective eigenvalues, are extensively used by these methods, as they are invariant to isometric deformations, and create a smooth basis for representing functions on the shape.
1.1 Spectral Embedding

The embedding of manifolds into some Euclidean space is often used for simplifying matching and comparison procedures [BBG94, EK03, CLL+05] between shapes. A useful property of such a target metric space is that corresponding points of different isometric shapes are mapped to nearby points in the target space. In that case, the embedding of multiple isometric shapes into this common target space naturally leads to distance between shapes that is easy to compute. Recently, attention has been given to spectral type of embedding that use the eigenvalues and eigenfunctions of the Laplace-Beltrami operator of the shape as a target space [EK03, Rus07, SOG09, ASC11]. The fact that the Laplace-Beltrami operator is invariant to isometric deformations, makes spectral embedding well suited for comparing the same object in different poses and expressions.

Bérard et al. [BBG94] exhibited the spectral embedding of Riemannian manifolds by their heat kernel. They embedded manifolds into a compatible common target space (infinite Hilbert space) and used the Hausdorff distance in that space to define a metric between isometry equivalent classes of Riemannian manifolds. It means, in particular, that two manifolds are at zero distance if and only if they are isometric.

Rustamov [Rus07] introduced the global point signature (GPS) of a point on a shape. It encodes both the eigenvalues and the eigenfunctions of the Laplace-Beltrami operator evaluated at that point. The GPS kernel is, in essence, the integration over all scales of the heat kernel on the surface [BB11]. The GPS kernel coincides with the Green’s function, and in some sense measures the extent to which two points are geometrically connected. He showed that the GPS embedding of a surface without self-intersections has no self-intersections as well.

A different approach was introduced by Sun et al. [SOG09]. Their signature, called the heat kernel signature (HKS), is defined for every point of the shape, by observing the heat kernel in that point over time. The set of all HKS on a shape characterizes a given surface up to an isometry under the condition that the eigenvalues of the Laplace-Beltrami operator are non-repeating. The invariance of the heat kernel signature to isometric deformations ensures that this signature can be used to find correspondence between different poses of the same shape.

In Chapter 3, we explore a novel spectral embedding, using gradient fields (GFs) of the Laplace-Beltrami operator eigenfunctions, for correspondence detection. We call the gradient fields of the eigenfunctions, spectral gradient fields (or spectral GFs), and refer to the new embedding as the spectral gradient fields embedding. As Laplace-Beltrami eigenfunctions computed independently for different shapes are often incompatible with each other, the aim of this construction is to embed the shapes using the eigenfunctions that correspond to the lowest eigenvalues. While existing methods [BBG94, Rus07] use the eigenfunctions themselves to define the target space, we embed the shapes using pairs of eigenfunctions. Thereby, more information is being extracted from the
interaction between the relatively stable first few eigenfunctions.

Although the eigenfunctions computed independently on isometric shapes have a nearly compatible eigenbasis, several inconsistencies arise. Eigenfunctions are defined up to a sign, the order of the eigenfunctions especially those representing higher frequencies is not repeatable across shapes, the eigenvalues of the Laplace–Beltrami operator may have a multiplicity greater than one with several eigenfunctions corresponding to each such eigenvalue. It is generally impossible to expect that an eigenfunction with a large eigenvalue of one shape will correspond to any eigenfunction of another shape. In Chapter 2, we present a novel method for matching eigenfunctions that were independently calculated for two nearly isometric shapes. We rely on the fact that for low order eigenfunctions, inconsistencies are usually governed by a small number of discrete parameters characterized by the sign sequence and permutation vector. We estimate these parameters by matching statistical properties over the spectral domain. The matching of the corresponding eigenfunctions enables the use of the eigenfunctions for consistent spectral embedding of multiple isometric shapes.

1.2 Spectral Distance

A common approach for shape matching is to define a measure of dissimilarity between shapes modeled as 2-manifolds. The well-established Gromov-Hausdorff distance measures the maximum geodesic discrepancy between pairs of corresponding points of the two given shapes [MS05]. The point-wise map can be inferred to as a byproduct of the evaluation of the Gromov-Hausdorff distance. This approach was embraced by the Generalized Multi-Dimensional Scaling (GMDS) framework [BBK06]. Within the Gromov-Hausdorff framework, Bronstein et al. [BBM+10] suggested replacing the geodesic distance by the diffusion distance [CLL+05], exploiting the apparent stability of diffusion distances to local changes in the topology of the shape. Despite its generality and theoretical beauty, it has been a challenge to apply the Gromov-Hausdorff framework in a straightforward manner to shape matching, mainly due to its intrinsically combinatorial nature.

Kasue and Kumura [KK94] extended the Gromov-Hausdorff distance framework to the family of spectral methods. The spectral kernel distance was constructed by replacing the metric defined on the manifolds with the heat kernel. The heat kernel provides a natural notion of scale, which is useful for multi-scale shape comparison. Recently, Mémoli [Mém09] introduced the spectral Gromov-Wasserstein distance, applying the theory of mass transportation. The spectral Gromov Wasserstein distance via the comparison of heat kernels satisfies all properties of a metric on the class of isometric manifolds.

The evaluation of the spectral kernel distance between two nearly isometric surfaces should be capable of discovering the mapping between them. Alas, the task is not straightforward, due to model impairments and the combinatorial nature of the problem.
Therefore, to achieve highly accurate and dense correspondence, we need to make adaptations to this distance measure and design an efficient and robust optimization algorithm.

In Chapter 4, we show how the alignment of the spectral kernels and the evaluation of the spectral kernel distance between two shapes can be achieved by extending the well established Iterative Closest Point (ICP) algorithm [YM92, BM92] to the class of nonrigid shapes. The classical ICP algorithm refines the correspondence between rigid shapes embedded in the three dimensional Euclidean space. The key idea is simple. Given an initial map between the shapes, find the best rotation and translation that aligns the shapes, apply it and calculate new correspondence by the nearest neighbor algorithm.

As for nonrigid shapes, a similar idea was presented by the iterative post-process refinement algorithm [OBCS+12]. Instead of aligning the shapes in the three dimensional Euclidean domain, this method estimates the transformation that best fits the shapes in the spectral domain. Given an initial map from shape \( X \) to shape \( Y \), one linear constraint is generated for each point \( x \in X \), and the least squares method is used to infer the transformation matrix.

The proposed iterative closest spectral kernel maps (ICSKM) algorithm extends this idea by finding the transformation that best matches the respective spectral kernels \( K(x, x') \) and \( \tilde{K}(y, y') \) of the shapes \( X \) and \( Y \). Now, each pair of points \( x, x' \in X \) generates a linear constraint by including its normalized kernel relation \( K(x, x')/K(x, x) \). The two dimensional information, effectively improves the refinement procedure. The optimization problem is solved by the least squares method with Tikhonov regularization [Tik63, HK70]. The algorithm is shown to be robust, flexible and easy to implement. It can be used efficiently as a refinement procedure of rough or sparse correspondence detection methods. The main advantage of the ICSKM algorithm over existing methods is in the combination of the iterative post-process refinement algorithm with the two dimensional constraints of the spectral kernel, resulting in highly accurate correspondence maps.

1.3 Nonrigid Shapes Deformation

The construction of an efficient automatic procedure that deforms one shape into another in a natural manner is a fundamental and well-studied challenge in computer graphics. Professional animators design deformable models for manually editing facial expressions, controlling postures and muscles of shapes, and creating sequences of gestures and motions of animated objects. Such models also play a key role in the field of shape analysis. For example, elastic surface registration techniques try to iteratively warp given shapes so as to establish an optimal alignment between them.

Recently, Jacobson et al. [JBK+12] suggested to deform a single shape by looking for transformations that minimize the nonlinear As-Rigid-As-Possible (ARAP) energy.
This energy penalizes deviations from rigidity of the underlying structural skeleton. The optimization process alternates between finding the minimal affine transformations and projecting them onto the group of rigid ones. The algorithm converges after a few iterations and provides realistic deformations with a low computational effort. The method was designed for modifying a single reference shape. As such, it does not effectively incorporate the nature of plausible non-rigid deformations that can be well captured by a few examples.

In Chapter 5, we present an efficient deformation method for the case where multiple exemplar shapes are available. We were motivated by the nonrigid 3D partial registration problem. This problem is considered a key challenge in the field of shape analysis. One of the most efficient approaches to solve this challenge is using deformation-driven correspondences [ZSCO+08]. A good deformation method for this purpose should efficiently produce plausible deformations that fits some known constraints. The deformation algorithm we designed uses the reference shapes to infer an expressive yet low dimensional model, which is computationally efficient and produces natural looking poses. The deformation algorithm constructs a dictionary that contains prototype signal-atoms of weighted vertex coordinates, that effectively span the space of deformations represented by the exemplar shapes. The deformation parameters are found by minimizing the As-Rigid-As-Possible energy functional that is reformulated to support multiple reference shapes.

**1.4 Volumetric Optical Flow**

Motion lies at the core of dynamical systems. One way to understand the motion of forms and structures in images is through optical flow [HS81, LK+81], which is an approximation of the motion of objects in an image, and its computation was traditionally based on local spatial and temporal derivatives in a given sequence of images. That is, in two dimensions it tries to specify how much the semantic content of each image pixel moves between adjacent images, while in three dimensions it specifies how much the content of each volume element (voxel) moves between adjacent volumes.

While several solutions to deep optical flow and video prediction are well established [RSB+14, SMS15, LKC16, PHC15, KOS+16, MCL15, WRHS13, FDI+15, RB16], volumetric temporal evolution learning remained unexplored. Part of the difficulty in devising a robust and efficient 3D optical flow is due to the large number of possibilities by which each voxel can move. Moreover, unlike the two dimensional case, where there exist some benchmarks with ground-truth optical flow, volumetric datasets lack such supervised information.

The approach introduced in Chapter 6, is motivated by several papers that predict the next frame of movie sequences, knowing the past and the present frames, in an unsupervised manner. The end-to-end differentiable architecture is based on multi-scale optical flow prediction. Each pyramid level consists of a deep generative network, which
is designed as a series of convolution layers with element-wise multiplication modules, followed by rectified linear units. The generative network recursively refines the future optical flow estimation and simultaneously adjusts the last frame that the warping module operates on. We refer to the proposed architecture as “V-Flow.”

The model is trained without any supervision effort, by minimizing the reconstruction error between the predicted volumetric frame and the ground truth next frame. Given previous volumetric frames, the system predicts 3D future optical flow. The neural network is specifically designed for the task of optical flow prediction and consists of multiple convolution and multiplication layers. Inherent property of the proposed approach is the ability to control the trade-off between image-sharpness and quantitative performance. This is done by introducing a latent frame which is a variation of the last frame. Each pixel of the latent frame is moved in keeping with the optical flow to produce the next frame prediction. As the latent frame is closer to the last frame, the predicted frame becomes sharper.
Chapter 2

Matching the LBO Eigenspace of Non-Rigid Shapes via High Order Statistics

A fundamental tool in shape analysis is the virtual embedding of the Riemannian manifold describing the geometry of a shape into Euclidean space. Several methods have been proposed to embed isometric shapes into flat domains, while preserving the distances measured on the manifold. Recently, attention has been given to embedding shapes into the eigenspace of the Laplace–Beltrami operator. The Laplace–Beltrami eigenspace preserves the diffusion distance and is invariant under isometric transformations. However, Laplace–Beltrami eigenfunctions computed independently for different shapes are often incompatible with each other. Applications involving multiple shapes, such as pointwise correspondence, would greatly benefit if their respective eigenfunctions were somehow matched. Here, we introduce a statistical approach for matching eigenfunctions. We consider the values of the eigenfunctions over the manifold as the sampling of random variables and try to match their multivariate distributions. Comparing distributions is done indirectly, using high order statistics. We show that the permutation and sign ambiguities of low order eigenfunctions can be inferred by minimizing the difference of their third order moments. The sign ambiguities of antisymmetric eigenfunctions can be resolved by exploiting isometric invariant relations between the gradients of the eigenfunctions and the surface normal. We present experiments demonstrating the success of the proposed method applied to feature point correspondence.

2.1 Introduction

The embedding of nonrigid shapes into a Euclidean space is well established and widely used by shape analysis applications. Usually, the mapping from the manifold to the Euclidean space preserves distances, that is the distance measured between two points on the manifold is approximated by the respective distance calculated in the Euclidean
The embedding of multiple isometric shapes into the same common Euclidean space seems to be ideal for applications like pointwise correspondence and shape editing. A useful property of this common embedding would be if any corresponding points of different isometric shapes were mapped to nearby target points in the Euclidean space. If this property is fulfilled, then the simultaneous processing of shapes in the target domain can be done in a straightforward manner.

Elad et al. [EK03] used classical multi-dimensional scaling (MDS) embedding into the geodesic kernel eigenspace. The MDS dissimilarity measure was based on the geodesic distances computed by the fast marching procedure [KS98]. Bérard et al. [BBG94] used the heat operator spectral decomposition to define a metric between two manifolds $M$ and $M'$. They embedded the two manifolds into their respective eigenspaces, and measured the Hausdorff distance [Hau14] between the two shapes in the spectral domain. The Hausdorff distance $d_H(M, M')$, being the greatest of all the distances from a point in one set to the closest point in the other set, can easily be calculated in this common Euclidean space. They showed that $d_H(M, M') = 0$ if and only if the Riemannian manifolds $M$ and $M'$ are isometric. Lafon et al. [CLL+05] defined the diffusion maps and showed that the embedding into the heat kernel eigenspace is isometry invariant and preserves the diffusion metric. Rustamov [Rus07] introduced the global point signature (GPS) embedding for deformation-invariant shape representation.

Although the diffusion maps computed independently on isometric shapes have a nearly compatible eigenbasis, several inconsistencies arise:

- Eigenfunctions are defined up to a sign.
- The order of the eigenfunctions, especially those representing higher frequencies, is not repeatable across shapes.
- The eigenvalues of the Laplace–Beltrami operator may have a multiplicity greater than one, with several eigenfunctions corresponding to each such eigenvalue.
- It is generally impossible to expect that an eigenfunction with a large eigenvalue of one shape will correspond to any eigenfunction of another shape.
- Intrinsic symmetries introduce self-ambiguity, adding complexity to the sign estimation challenge.

These drawbacks limit the use of diffusion maps in simultaneous shape analysis and processing; they do not allow using high frequencies and usually require some intervention to order the eigenfunctions or solve sign ambiguities.

In this paper, we present a novel method for matching eigenfunctions that were independently calculated for two nearly isometric shapes. We rely on the fact that for low order eigenfunctions, inconsistencies are usually governed by a small number of discrete parameters characterized by the sign sequence and permutation vector. We estimate these parameters by matching statistical properties over the spectral domain. The matching of the corresponding eigenfunctions enables the use of diffusion maps for consistent embedding of multiple isometric shapes into a common Euclidean space.
2.1.1 Related Work

The problems of eigenfunctions permutation and sign ambiguity were previously addressed in the context of simultaneous shape processing. Several authors, among them Shapiro and Brady [SMB92] and Jain et al. [JZvK07], proposed using either exhaustive search or a greedy approach for the eigenvalue ordering and sign detection. Umeyama [Ume88] proposed using a combination of the absolute values of the eigenfunctions and an exhaustive search. Mateus et al. [MCHB07] expressed the connection between the eigenfunctions of two shapes by an orthogonal matrix. They formulated the matching as a global optimization problem, optimizing over the space of orthogonal matrices, and solved it using the expectation minimization approach. Later, Mateus et al. [MHK+08] and Knossow et al. [KSMH09] suggested using histograms of eigenfunctions values to detect their ordering and signs. Dubrovina et al. [DK11] suggested using a coarse matching based on absolute values of eigenfunctions together with geodesic distances measured on the two shapes.

Most of these methods do not reliably resolve eigenfunction permutation [JZvK07, KSMH09, MCHB07, SMB92]. Some of the above algorithms are limited by high complexity and do not allow the matching of more than a few eigenfunctions [Ume88, DK11]. None of these methods reliably estimate the sign sequence of antisymmetric eigenfunctions.

At the other end, Kovnatsky et al. [KBB+13] proposed avoiding the matching problem by constructing a common approximate eigenbases for multiple shapes using approximate joint diagonalization algorithms. Yet, it relies on prior knowledge of a set of corresponding feature points.

Finally, the algorithm proposed by Pokrass et al. [PBB+13] mostly resembles our approach. They used sparse modeling to match the Laplace–Beltrami operator (LBO) eigenfunctions that span the wave kernel signature (WKS). Yet, that approach does not reliably infer the signs of the antisymmetric eigenfunctions.

2.1.2 Background

Laplace–Beltrami Eigendecomposition

Let us be given a shape modeled as a compact two-dimensional manifold \( M \). The divergence of the gradient of a function \( f \) over the manifold:

\[
\Delta_G f = \text{div} \, \text{grad} f
\]  

(2.1)

is called the Laplace–Beltrami operator (LBO) of \( f \) and can be considered as a generalization of the standard notion of the Laplace operator to manifolds [Tau95, LZ10]. The
Laplace–Beltrami operator is completely derived from the metric tensor $G$.

$$\Delta_G f = \text{div} \text{grad} f = \frac{1}{\sqrt{|G|}} \sum_i \partial_i \sqrt{|G|} \sum_j g^{ij} \partial_j f$$

(2.2)

where $g^{ij} = (G^{-1})_{ij}$ are the components of the inverse metric tensor.

Since the operator $-\Delta_G$ is a positive self-adjoint operator, it admits an eigendecomposition with non-negative eigenvalues $\lambda_i$ and corresponding orthonormal eigenfunctions $\phi_i$,

$$-\Delta_G \phi_i = \lambda_i \phi_i$$

(2.3)

where orthonormality is understood in the sense of the local inner product induced by the Riemannian metric on the manifold. Furthermore, due to the assumption that our manifold is compact, the spectrum is discrete. We can order the eigenvalues as follows $0 = \lambda_1 < \lambda_2 < \cdots < \lambda_i < \cdots$. The set of corresponding eigenfunctions given by \{\phi_1, \phi_2, \cdots, \phi_i, \cdots\} forms an orthonormal basis of functions defined on $M$.

**Diffusion Maps**

The heat equation describes the distribution of heat in time. On a manifold $M$, the heat equation is governed by the Laplace–Beltrami operator $\Delta_G$:

$$\frac{\partial u}{\partial t} = \Delta_G u$$

(2.4)

The heat kernel $K_t(x, y)$ is the diffusion kernel of the heat operator $e^{t\Delta_G} (t > 0)$. It is a fundamental solution of the heat equation with the point heat source at $x$ (the heat value at point $y$ after time $t$). The heat kernel can be represented in the Laplace–Beltrami eigenbasis as:

$$K_t(x, y) = \sum_i (\tilde{\lambda}_i)^t \phi_i(x)\phi_i(y) = \sum_i e^{-\lambda_i t} \phi_i(x)\phi_i(y)$$

(2.5)

where $\tilde{\lambda}_i$ are the eigenvalues of the heat operator, $\lambda_i$ are the eigenvalues of the LBO and $\tilde{\lambda}_i = e^{-\lambda_i}$.

Using the heat kernel, we can define the diffusion distance [CLL+05]:

$$d_{M,t}^2(x, y) = \|K_t(x, \cdot) - K_t(y, \cdot)\|_{L^2}^2$$

$$= \int_M (K_t(x, z) - K_t(y, z))^2 da(z)$$

(2.6)

where $da$ is the area element of $M$.

The diffusion distance $d_{M,t}(x, y)$ can be computed by embedding the manifold into
the infinite Euclidean space spanned by the LBO eigenbasis:

\[ d_{M,t}(x, y) = \left( \sum_i e^{-2\lambda_i t}(\phi_i(x) - \phi_i(y))^2 \right)^{\frac{1}{2}} \]  

(2.7)

The diffusion map \( \{\Phi_t\} \) embeds the data into the finite \( N \)-dimension Euclidean space:

\[
\Phi_t(x) = \begin{bmatrix}
 e^{-\lambda_1 t}\phi_1(x) \\
 e^{-\lambda_2 t}\phi_2(x) \\
 \vdots \\
 e^{-\lambda_N t}\phi_N(x)
\end{bmatrix}
\]  

(2.8)

so that in this space, the Euclidean distance is equal to the diffusion distance up to a relative truncation error:

\[ d_{M,t}(x, y) \approx ||\Phi_t(x) - \Phi_t(y)|| \]  

(2.9)

**Multivariate Distribution Comparison**

The distribution of \( N \) continuous random variables \( \phi_1, \phi_2, ..., \phi_N \) is directly represented by the probability density function \( f_{\phi_1,\phi_2,\ldots,\phi_N}(\phi_1, \phi_2, ..., \phi_N) \). The direct estimation of the multivariate probability density function from data samples is hard to accomplish. Therefore, instead of using a direct comparison of distribution functions [KL51, Lin91], an indirect representation is often being utilized. The probability distribution can be indirectly specified (under mild conditions) in a number of different ways, the simplest of which is by its raw moments:

\[ \mu_{i_1,i_2,...,i_N} \equiv E[\phi_1^{i_1}\phi_2^{i_2}\ldots\phi_N^{i_N}], \quad \{i_1, i_2, ..., i_N\} \in \mathbb{Z}_{\geq 0} \]  

(2.10)

In order to compare the multivariate distributions of two sets of \( N \) random variables \( \phi_1^X, \phi_2^X, ..., \phi_N^X \) and \( \phi_1^Y, \phi_2^Y, ..., \phi_N^Y \), we can use this indirect representation and compare the raw moments of the random variables. In practice, only a small set of the moments \( \mathcal{I} \) can be used for measuring the difference between the distributions:

\[ C_{X,Y} = \sum_{\{i_1,i_2,...,i_N\} \in \mathcal{I}} \rho_{i_1,i_2,...,i_N}(\mu_{i_1,i_2,...,i_N}^X - \mu_{i_1,i_2,...,i_N}^Y)^2 \]  

(2.11)

where \( \rho_{i_1,i_2,...,i_N} \) are the weights associated with each raw moment.

**2.2 Eigenfunction Matching**

**2.2.1 Problem Formulation**

Let us denote by \( X \) and \( Y \) the two shapes that we would like to match. We represent the correspondence between \( X \) and \( Y \) by a bijective mapping \( \varphi : X \mapsto Y \), such that for each point \( x \in X \), its corresponding point is \( \varphi(x) \in Y \). The diffusion map embeds...
each point $x \in X$ into the $N$ dimension Euclidean space $\mathbb{R}^N$ according to $\Phi_t^{X,N}(x)$. Correspondingly, each point $y \in Y$ is embedded by the mapping $\Phi_t^{Y,N}(y)$ into $\mathbb{R}^N$. We denote the diffusion map at $t = 0$ by $\Phi^X(x) = \Phi_{t=0}^{X,N}(x)$ and $\Phi^Y(y) = \Phi_{t=0}^{Y,N}(y)$, respectively.

We wish to find embeddings of shape $X$ and shape $Y$ to the finite dimensional Euclidean space, such that the corresponding points $x \in X$ and $\varphi(x) \in Y$ will be mapped to nearby points in the embedded space. Because of the inconsistencies described in the Introduction, the diffusion maps of shapes $X$ and $Y$ do not necessarily fulfill this property. Our task is to modify the diffusion map $\Phi^Y(y)$ by a small number of parameters $\theta$, such that the new embedding $\tilde{\Phi}^Y_\theta(y)$ will match $\Phi^X$, i.e., $\Phi^X(x) \approx \tilde{\Phi}^Y_\theta(\varphi(x))$.

For the $N$ low eigenvalues, the matching is characterized by the following parameters:

- The respective signs of the eigenfunctions $s_i \in \{+1, -1\}$.
- The permutation vector $\pi$ of the eigenfunctions: $\pi : \{1, 2, ..., N\} \mapsto \{1, 2, ..., N\}$.

We would like to find the parameters $\hat{\theta} = \{\hat{s}; \hat{\pi}\}$, which create the matched embedding $\tilde{\Phi}^Y_\theta(y)$ with elements $\tilde{\phi}^Y_i = \hat{s}_i \phi^Y_{\hat{\pi}(i)}$, $i \in 1, 2, ..., N$.

### 2.2.2 Matching Cost Function

The entire algorithm can be expressed as the minimization of the following cost function:

$$\{\hat{s}; \hat{\pi}\} = \arg\min_{s, \pi} (C(s, \pi) + C^S(s, \pi) + \alpha(C^P_\nabla(s, \pi) + C^{P,S}_\nabla(s, \pi))) \quad (2.12)$$

**Overview**

The objective function is comprised of four terms $C, C^S, C^P_\nabla, C^{P,S}_\nabla$. The first and second terms $(C, C^S)$ compare the mixed moments of compatible functions. Minimizing the terms $C, C^S$ is usually sufficient to correctly reorder the eigenfunctions and find the right sign sequence. Alas, in the presence of intrinsic symmetry, these mixed moments are ambiguous and cannot be compared effectively. The third and fourth terms $(C^P_\nabla, C^{P,S}_\nabla)$ work out this difficulty by using a pair of gradients of compatible functions. The gradients are inserted into the functional, by incorporating their cross-product in the direction of the outward pointing normal to the surface.

To enhance the discriminative properties of the algorithm and its robustness, we apply two additional techniques.

- Pointwise signatures as side information: We mix in stable compatible signatures, like the heat kernel signature (HKS), employed in $C^S, C^P_\nabla, C^{P,S}_\nabla$.
- Raw moments over segments: We blindly (i.e., without correspondence) segment the shapes into parts in a compatible way and integrate over these segments separately; employed in $C^P_\nabla, C^{P,S}_\nabla$. 

The terms of the cost function can be expressed by:

\[ C(s, \pi) = \sum_{i,j,k} \left( \mu_{i,j,k} - s_i s_j s_k \mu_{\pi(i),\pi(j),\pi(k)} \right)^2 \]

\[ \mu_{i,j,k} = E[\phi_i \phi_j \phi_k], \quad i, j, k \in \{1, 2, ..., N\} \]

\[ C^p(s, \pi) = \sum_{i,j,k,p} \left( \xi_{i,j,k,p} - s_i s_j s_k \xi_{\pi(i),\pi(j),\pi(k),p} \right)^2 \]

\[ \xi_{i,j,k,p} = E[\nu_{i,j,k,p} \phi_0], \quad i, j, k \in \{1, 2, ..., N\}, \quad p \in \{1, 2, ..., P\} \]

\[ C^s(s, \pi) = \sum_{i,q} \left( \mu_{i,q} - s_i \mu_{\pi(i),q} \right)^2 \]

\[ \mu_{i,q} = E[\phi_i \psi_q], \quad i \in \{1, 2, ..., N\}, \quad q \in \{1, 2, ..., Q\} \]

\[ C^{ps}(s, \pi) = \sum_{i,q,k,p} \left( \xi_{i,q,k,p} - s_i s_k \xi_{\pi(i),q,\pi(k),p} \right)^2 \]

\[ \xi_{i,q,k,p} = E[\nu_{i,q,k,p} \phi_0], \quad i, k \in \{1, 2, ..., N\}, \quad p \in \{1, 2, ..., P\}, \quad q \in \{1, 2, ..., Q\} \]

where:

- \( \phi_i \) are the eigenfunctions of the Laplace–Beltrami operator \(-\Delta G \phi_i = \lambda_i \phi_i\).
- \( w_p : \mathbb{R}_{\geq 0} \rightarrow [0, 1] \) are nonlinear weighting functions.
- \( \psi_q : M \rightarrow \mathbb{R} \) are the components of an external point signature.
- \( \nabla G \) is the gradient induced by the metric tensor \( G \).
- \( E[z] = \int_M z da_M \), where \( da_M \) is the area element of the manifold \( M \).
- \( n \) is the normal to the surface.
- \( \times \) is the cross-product in \( \mathbb{R}^3 \), and \( \cdot \) is the inner product in \( \mathbb{R}^3 \).
- The weighting parameter \( \alpha \) determines the relative weight of the gradient cost functions.

In Appendix 2.5, we give full details of the discretization that we have used to implement the matching algorithm.

The application specific parameters include:

- \( N \): the number of eigenfunctions to be matched.
• \{w_p\}_{p=1}^{P}$: the $P$ nonlinear weighting functions.

• \{ψ_q\}_{q=1}^{Q}$: the external point signature of size $Q$.

• $α$: the relative weight of the gradient cost functions.

In Appendix 2.6, we give the details of the application specific parameters that were used in our experiments.

Next, we review the different terms of the cost function.

**Resolving Sign Ambiguities and Permutations**

For now, let us limit our discussion to resolving the sign ambiguity $s$. If we had known the correspondence between the two shapes, the sign of the $i-$th eigenfunction $s_i$ could be inferred by pointwise comparison:

$$\hat{s}_i = \arg\min_{s_i} E[(\phi_i^X(x) - s_i\phi_i^Y(\varphi(x))^2]$$  \hspace{1cm} (2.13)

and the expectation is taken over the manifold:

$$E(f(x)) = \int_X f(x)da_X$$  \hspace{1cm} (2.14)

where $da_X$ is the area element of the shape $X$. Unfortunately, the correspondence is unknown. Hence, pointwise comparison cannot be used in a straightforward manner.

We now make the analogy between the values of the eigenfunctions over the manifold and $N$ random variables. We consider the vector of values of the diffusion map $\Phi^X(x)$ at point $x$ as a sample out of a multivariate distribution $f_{\Phi}^X(\phi_1(x), \phi_2(x), ..., \phi_N(x))$. We wish to match the multivariate distributions $f_{\Phi^X}$ and $f_{\Phi^Y}$. As explained in Section 2.1.2, an indirect representation of the distribution is suitable for comparing multivariate distributions. Specifically, we shall use the raw moments.

By way of construction, the non-trivial eigenfunctions have zero mean and are orthonormal. Hence, the first and second moments carry no information. Accordingly, we must use higher order moments to match the distributions. We propose to use the third order moments over the manifold $M$

$$\mu_{i,j,k} = E[\phi_i \phi_j \phi_k] = \int_M \phi_i \phi_j \phi_k da_M$$  \hspace{1cm} (2.15)

where $i, j, k \in \{1, 2, ..., N\}$.

**Resolving Antisymmetric Eigenfunctions**

For shapes with intrinsic symmetries (see [RBBK07]), some of the eigenfunctions have antisymmetric distributions. The distribution of the antisymmetric eigenfunctions is agnostic to sign change. Hence, the signs of the antisymmetric eigenfunctions cannot be resolved by the simple scheme described in Section 2.2.2.
The gradient of the eigenfunctions $\nabla \phi_k$ could be exploited to resolve the sign ambiguity:

- The gradient $\nabla f$ of an antisymmetric eigenfunction $f$ is not antisymmetric.
- The gradient is a linear operator. Consequently $\nabla (-f) = -\nabla f$, $\forall f$.

Therefore, we can farther expand the set of variables that are used in the calculation of the raw moments, by incorporating the gradient. The gradient vector is contained in the tangent plane. Thus, the cross-product of the gradients of two eigenfunctions points either outward or inward from an orientable surface. Changing the sign of one eigenfunction will flip the direction of the cross-product. We can use this property to define new functions $\nu_{i,j}$ over the manifold:

$$\nu_{i,j} = (\nabla \phi_i \times \nabla \phi_j) \cdot \mathbf{n}$$

(2.16)

where $\mathbf{n}$ is the outward pointing normal to the tangent plane. We shall use the joint moments of the eigenfunctions and their gradients:

$$\xi_{i,j,k} = E[\nu_{i,j} \phi_k], \quad i,j,k \in \{1,2,\ldots,N\}$$

(2.17)

We note that Equation (2.16) can be further simplified by:

$$\nu_{i,j} = (\nabla \phi_i \times \nabla \phi_j) \cdot \mathbf{n} = \nabla \phi_i \cdot (\nabla \phi_j \times \mathbf{n})$$

(2.18)

$(\nabla \phi_j \times \mathbf{n})$ can be computed only once for each $\phi_j$.

**Raw Moments over Segments**

Taking the expectation over the whole shape may be too crude, especially for detecting antisymmetric sign ambiguities. We can refine the minimization criterion by taking the expectation over different segments of the shapes. Remember that the correspondence between the shapes is yet unknown; therefore, directly dividing the shape into corresponding segments is impossible. Indirectly dividing the shape into different segments is possible by using the eigenfunctions themselves. The eigenfunctions $\phi_k, k \in \{1\ldots N\}$ have respective low eigenvalues, which means that they have a slow rate of change. Therefore, it is possible to define indicator functions $w_p(|\phi_k(x)|) : X \rightarrow \{0,1\}, p \in \{1..P\}$ in a way that will output one or zero values at different segments of the shape. For example, we can define $h(|\phi_k(x)|) = 1$ if $|\phi_k(x)| > TH$ and zero otherwise, where $TH$ is a scalar threshold. The output of these functions automatically divides the two shapes into a similar manner, without the use of pointwise correspondence. Moreover, because the function $w_p(|\phi_k|)$ is symmetric, its output does not depend on the sign of the eigenfunction $\phi_k$. We conclude that we can use $w_p(|\phi_k|)$ to make a weighted average of
the raw moments according to different segments:

\[ \xi_{i,j,k,p} = E[\nu_{i,j,\phi_k w_p}(|\phi_k|)] \]
\[ i, j, k \in \{1, 2, ..., N\}, \quad p \in \{1..P\} \] (2.19)

**Pointwise Signatures As Side Information**

We can easily use other signatures \((\psi_1, \psi_2, ..., \psi_Q)\) as side information to refine the minimization criterion. Specifically, we can use signatures that carry no inconsistencies among different shapes. In our experiments, we used the heat kernel signature (HKS) as an additional signature [SOG09]. We can use the joint moments of the diffusion maps and the additional signatures \(\psi_q\):

\[ \mu_{i,q}^S = E[\phi_i \psi_q], \quad i \in \{1, 2, ..., N\}, \quad q \in \{1, 2, ...Q\} \] (2.20)

and compute the cross of the eigenfunctions gradient \(\nabla \phi_i\) and the signature functions gradients \(\nabla \psi_q\):

\[ \nu_{i,q}^S = (\nabla \phi_i \times \nabla \psi_q) \cdot n \] (2.21)

\[ \xi_{i,q,k,p}^S = E[\nu_{i,q}^S \phi_k w_p(|\phi_k|)] \]
\[ i, k \in \{1, 2, ..., N\}, \quad p \in \{1..P\}, \quad q \in \{1..Q\} \] (2.22)

**2.2.3 Solving the Minimization Problem**

The minimization of Equation (2.12) is a non-convex optimization problem. Yet, it only involves a small number of discrete parameters. Therefore, an exhaustive search is possible. In practice, we implemented the search in four steps:

- **Step 1**: An initialization of \(s^0\) is determined by \(s_i = \text{sign}(\mu_{i,i,i}^X \mu_{i,i,i}^Y)\) and \(\pi^0 = [0, 1, ..., N]\).

- **Step 2**: The permutation vector \(\hat{\pi}\) is found by minimizing \(C(s, \pi) + C^S(s, \pi)\). We make an educated guess for the possible permutations, limiting the search for two permutation profiles:
  - two consecutive eigenfunction switching (with possible sign change), i.e., \([\pi_i, \pi_j, \pi_k, \pi_l] = [j, i, l, k], j = i + 1, l = k + 1\)
  - triplet permutation (with possible sign change), i.e., \([\pi_i, \pi_j, \pi_k] = [j, k, i]\) or \([k, i, j], j = k + 1, i = j + 1\);  

- **Step 3**: The sign sequence is resolved again by minimizing \(C(s, \pi) + C^S(s, \pi)\). In this step, all possible quadruple sign changes are checked, setting the permutation vector found in Step 2. If the cost function was decreased in Step 2 or Step 3,
then return to Step 2. While finding the optimal sign sequence and permutation vector, we keep a list of all possible good sign sequences for the next step.

- Step 4: The optimal sign sequence $\hat{s}$ is found by comparing the entire cost function $C(s, \pi) + C^S(s, \pi) + \alpha(C_P(s, \pi) + C^{PS}(s, \pi))$ for each sign sequence in the list created in Step 3.

2.3 Results

We tested the proposed method on pairs of shapes represented by triangulated meshes from the TOSCA database [BBK08]. Figures 2.1, 2.3 and 2.5 show how the proposed method succeeds in matching the eigenfunctions of several isometric shapes. In each figure, at the top, are the first four eigenfunctions of the first pose of the object. In the middle are the eigenfunctions of the second pose of the object. At the bottom are the first four eigenfunctions of the second pose with the correct sign sequence and permutations.

For example, we can see in Figure 2.1 that the eigenfunction matching algorithm swapped $\phi^Y_3$ and $\phi^Y_4$. It also correctly flipped the signs of $\phi^Y_1$, $\phi^Y_2$ and $\phi^Y_4$. We also notice that the matching algorithm was able to detect the correct signs of the antisymmetric eigenfunctions. For example, in Figure 2.3, the sign of the antisymmetric eigenfunction $\phi^Y_2$ was correctly flipped, while keeping the sign of $\phi^Y_3$.

We used the matched eigenfunctions for detecting feature point correspondence between the two shapes. A selected number of feature points from the first shape were matched to the second one using a combination of two signatures:

- The matched low order eigenfunctions that represent the global structure of the shapes.
- The heat kernel signature (HKS) derivative (Equation (2.33)) that, being a bandpass filter, expresses more local features.

Figures 2.2, 2.4 and 2.6 show that the correspondences between feature points were found correctly. Notice that this approach was able to resolve the symmetries of the given shapes.

The algorithm was implemented in MATLAB. All of the experiments were executed on a 3.00 GHz Intel Core i7 machine with 32 GB of RAM. The runtime for matching the eigenfunctions for a typical pair of shapes from the TOSCA database is 1.5 s, excluding the calculation of the eigenfunctions.

We utilized these feature points to propagate correspondence to the whole shape. The signatures we used for this application were:

- The global point signature (GPS) kernel (Equation (2.34)), propagating the correspondence of each feature point.
The heat kernel signature (HKS) derivative (Equation (2.33)), as before.

The entire cost function given in Equation (2.12) can be computed from the raw moments and the parameters, without the use of the eigenfunctions themselves.

We note that the computation of the moments can be done prior to the minimization algorithm. This calculation of the raw moments of shape $X$ is independent of shape $Y$ and can be performed for each shape before the matching procedure. The correspondence method was applied to pairs of shapes represented by triangulated meshes from both the TOSCA and SCAPE databases [BBK08, ASP+05]. The TOSCA dataset contains densely sampled synthetic human and animal surfaces, divided into several classes with given ground-truth point-to-point correspondences between the shapes within each class. The SCAPE dataset contains scans of real human bodies in different poses. We compare our work to several correspondence detection methods.

- Matched eigenfunctions: the method proposed in this paper.
- Blended: the method proposed by Kim et al. [KLF11] that uses a weighted combination of isometric maps.
- Best conformal: the least-distortive conformal map roughly describes what is the best performance achieved by a single conformal map without blending.
- Heat kernel matching (HKM) with one correspondence: This method is based on matching features in a space of a heat kernel for a given source point, as described...
Figure 2.2: Feature point correspondence of two nearly isometric shapes of a horse.

Figure 2.3: Eigenfunction matching of two nearly isometric shapes. Hot and cold colors represent positive and negative values, respectively. (Top) The first pose of a human; (center) the second pose of a human; (bottom) the second pose of a human after the matching algorithm.
Figure 2.4: Feature point correspondence of two nearly isometric shapes of a horse.

Figure 2.5: Eigenfunction matching of two nearly isometric shapes. Hot and cold colors represent positive and negative values, respectively. (Top) The first pose of a horse; (center) the second pose of a horse; (bottom) the second pose of a horse after the matching algorithm.
Figure 2.6: Feature point correspondence of two nearly isometric shapes of a horse.

in [OMMG10]. A full map is constructed from a single correspondence, which is obtained by searching a correspondence that gives the most similar heat kernel maps. We use the results shown in [KLF11].

• HKM with two correspondences: in a non-isometric case, the previous method might obtain better results by using a second correspondence. The matching is then performed in the augmented feature space of two heat kernel maps [OMMG10]. We use the results shown in [KLF11].

Note that all of the above methods, including the one we propose, can be followed by the post-processing iterative refinement algorithm [OBCS+12]. The refinement procedure is based on the iterative closest point (ICP) method [YM92, BM92], applied in the spectral domain, and is known to improve the quality of other isometric shape matching methods.

Figures 2.7 and 2.8 compare our correspondence framework with existing methods on the TOSCA benchmark and SCAPE, using the evaluation protocol proposed in [KLF11]. The distortion curves describe the percentage of surface points falling within a relative geodesic distance from what is assumed to be their true locations. For each shape, the geodesic distance is normalized by the square root of the shape’s area. We see that out of the methods we compared, the proposed method is second best to the blended method, which is currently state-of-the-art.

2.4 Conclusions

The Laplace–Beltrami operator (LBO) provides us with a flat eigenspace in which surfaces could be represented as canonical forms in an isometric invariant manner. However, the order and directions (signs) of the axes in this Hilbert space do not
Figure 2.7: Evaluation of the iterative spectral kernel maps’ correspondence framework applied to shapes from the TOSCA database, using the protocol of [KLF11].

Figure 2.8: Evaluation of the iterative closest spectral kernel maps’ correspondence framework applied to shapes from the SCAPE database, using the protocol of [KLF11].
have to correspond when two isometric surfaces are considered. In order to resolve such potential ambiguities, we resorted to high order statistics of the eigenfunctions of the LBO and their interaction with the surface normal. It appears that these cross moments allow for ordered directional matching of the components of corresponding eigenspaces. We demonstrated that resolving the sign and order correspondence allows for shape matching in various scenarios. In the future, we plan on extending the proposed framework to enable it to deal with more generic transformations, like the scale-invariant metric introduced by Aflalo et al. [ARK11].

2.5 Discretization

2.5.1 Laplace–Beltrami Eigendecomposition

We used the cotangent weight scheme for the Laplace–Beltrami operator discretization, proposed by Pinkall et al. [UPSDJKP93] and later refined by Meyer et al. [MDSB02]. In order to calculate the eigendecomposition of the Laplace–Beltrami operator, we solved the generalized eigendecomposition problem, as suggested by Rustamov [Rus07].

\[ W\phi = \lambda A\phi \]  \hspace{1cm} (2.23)

where

\[ w_{ij} = \begin{cases} \cot \alpha_{ij} + \cot \beta_{ij} & i \neq j \\ \sum_{k \neq i} w_{ik} & i = j \end{cases} \]  \hspace{1cm} (2.24)

and \( A \) is a diagonal matrix. \( A_{ii} \) equals the Voronoi area around vertex \( i \).

2.5.2 Gradient

We assume that the function \( f \) is linear over the triangle with vertices \( V_i, V_j, V_k \) with values \( f_i, f_j, f_k \) at the vertices. We define the local coordinates \((u, v)\) with coordinates \((0, 0), (0, 1), (1, 0)\) at the vertices \( V_i, V_j, V_k \). Because \( f \) is assumed to be linear:

\[ \frac{\partial f}{\partial u} = f_j - f_i \]  \hspace{1cm} (2.25)

and:

\[ \frac{\partial f}{\partial v} = f_k - f_i \]  \hspace{1cm} (2.26)

which can be written as:

\[ \frac{\partial f}{\partial (u, v)} = \begin{bmatrix} \frac{\partial f}{\partial u} \\ \frac{\partial f}{\partial v} \end{bmatrix}^T = (DF)^T \]  \hspace{1cm} (2.27)
where \( D = \begin{bmatrix} -1 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \) and \( F = \begin{bmatrix} f_i \\ f_j \\ f_k \end{bmatrix} \).

The Jacobian:

\[
J = \frac{\partial(x, y, z)}{\partial(u, v)} = \begin{bmatrix}
\frac{\partial x}{\partial u} & \frac{\partial x}{\partial v} \\
\frac{\partial y}{\partial u} & \frac{\partial y}{\partial v} \\
\frac{\partial z}{\partial u} & \frac{\partial z}{\partial v}
\end{bmatrix}^T = [V_j - V_i, V_k - V_i]^T
\]

(2.28)

By the chain rule \( \frac{\partial f}{\partial (u,v)} = \frac{\partial f}{\partial (x,y,z)} \frac{\partial (x,y,z)}{\partial (u,v)} \) and in matrix form:

\[
(DF)^T = (\nabla f)^T J^T
\]

(2.29)

or equivalently

\[
J^T \nabla f = DF
\]

(2.30)

By taking the pseudoinverse, we get the discrete gradient operator over a triangle:

\[
\nabla = J^T (JJ^T)^{-1} D
\]

(2.31)

### 2.6 Application Specific Parameters

In our experiments, we used the following specific parameters for the matching algorithm:

- We matched the first \( N = 8 \) eigenfunctions from one shape to the first 10 eigenfunctions of the other shape.

- Soft thresholding was used to define \( P = 2 \) nonlinear weighting functions \( w_p \):

\[
w_0(z) = \begin{cases} 
0, & \text{if } |z| < TH \\
1, & \text{if } |z| > 2TH \\
(|z| - TH)/TH & \text{otherwise}
\end{cases}
\]

\[
w_1(z) = 1 - w_0(z)
\]

where \( TH = 0.1 \frac{1}{\sqrt{\int_M da}} \).

- For generating the external pointwise signature \( \psi_q \), the heat kernel signature (HKS) was used [SOG09]. In the approximation of the heat kernel signature \( \text{HKS}_t(x) = \sum_{i=1}^{h} e^{-\lambda_it} \phi_i^2(x) \), we used \( h = 120 \) eigenfunctions. We used a bandpass filter form of the HKS by taking the derivative of the heat kernel signature. The HKS derivative was logarithmically sampled \( Q = 6 \) times at \( t = t_q, \quad q = 1, 2, \ldots, Q \).
with $t_1 = \frac{1}{50\lambda_1}$ and $t_Q = \frac{1}{\lambda_1}$. $\psi_q$ were normalized according to the inner product over the manifold.

$$\psi_q(x) = \frac{\tilde{\psi}_q(x)}{\sqrt{\int_M \tilde{\psi}_q^2(\tilde{x}) d\tilde{a}(\tilde{x})}}$$

$$\tilde{\psi}_q(x) = \frac{\partial}{\partial t}\text{HKS}_t(x) \text{ sampled at } t = t_q$$

(2.33)

$$\frac{\partial}{\partial t}\text{HKS}_t(x) = \sum_{i=1}^{h} -\lambda_i e^{-\lambda_i t}\phi_i^2(x)$$

- For propagating the correspondence of each feature point $p$, the global pointwise signature (GPS) kernel was used [Rus07].

$$\text{GPS}(x,p) = \sum_{i=1}^{h} \frac{1}{\lambda_i} \phi_i(x) \phi_i(p)$$

(2.34)

- The relative weight parameter $\alpha$ was set by balancing the influence of the terms of the cost function.

$$\alpha = \frac{\sum_{i,j,k} (\mu_{i,j,k}^X)^2 + \sum_{i,q} (\mu_{i,q}^{X,S})^2}{\sum_{i,j,k,p} (\xi_{i,j,k,p}^X)^2 + \sum_{i,q,k,p} (\xi_{i,q,k,p}^{X,S})^2}$$

(2.35)
Chapter 3

Spectral Gradient Fields Embedding for Nonrigid Shape Matching

A popular approach for finding the correspondence between two nonrigid shapes is to embed their two-dimensional surfaces into some common Euclidean space, defining the comparison task as a problem of rigid matching in that space. We propose to extend this line of thought and introduce a novel spectral embedding, which exploits gradient fields for point to point matching. With this new embedding, a fully automatic system for finding the correspondence between shapes is introduced. The method is demonstrated to accurately recover the natural maps between nearly isometric surfaces and shown to achieve state-of-the-art results on known shape matching benchmarks.

3.1 Introduction

The embedding of manifolds into some Euclidean space is often used for simplifying matching and comparison procedures [BBG94, EK03, CLL+05]. A useful property of such a target metric space is that corresponding points of different isometric shapes are mapped to nearby points in the target space. In that case, the embedding of multiple isometric shapes into this common target space naturally leads to distance between shapes that is easy to compute. Recently, attention has been given to spectral type of embedding that use the eigenvalues and eigenfunctions of the Laplace-Beltrami operator of the shape as a target space [EK03, Rus07, SOG09, ASC11]. The fact that the Laplace-Beltrami operator is invariant to isometric deformations, makes spectral embedding well suited for comparing the same object in different poses and expressions.

Bérard et al. [BBG94] exhibited the spectral embedding of Riemannian manifolds by their heat kernel. They embedded manifolds into a compatible common target space (infinite Hilbert space) and used the Hausdorff distance in that space to define a metric between isometry equivalent classes of Riemannian manifolds. It means, in particular, that two manifolds are at zero distance if and only if they are isometric.

Rustamov [Rus07] introduced the global point signature (GPS) of a point on a shape.
It encodes both the eigenvalues and the eigenfunctions of the Laplace-Beltrami operator evaluated at that point. The GPS kernel is, in essence, the integration over all scales of the heat kernel on the surface [BB11]. The GPS kernel coincides with the Green’s function, and in some sense measures the extent to which two points are geometrically connected. He showed that the GPS embedding of a surface without self-intersections has no self-intersections as well.

A different approach was introduced by Sun et al. [SOG09] Their signature, called the heat kernel signature (HKS), is defined for every point of the shape, by observing the heat kernel in that point over time. The set of all HKS on a shape characterizes a given surface up to an isometry under the condition that the eigenvalues of the Laplace-Beltrami operator are non-repeating. The invariance of the heat kernel signature to isometric deformations ensures that this signature can be used to find correspondence between different poses of the same shape.

We propose a novel spectral embedding, using gradient fields (GFs) of the Laplace-Beltrami operator eigenfunctions, for correspondence detection. We call the gradient fields of the eigenfunctions, spectral gradient fields (or spectral GFs), and refer to the proposed embedding as the spectral gradient fields embedding. As Laplace-Beltrami eigenfunctions computed independently for different shapes are often incompatible with each other, the aim of our construction is to embed the shapes using the eigenfunctions that correspond to the lowest eigenvalues. While existing methods [BBG94, Rus07] use the eigenfunctions themselves to define the target space, we embed the shapes using pairs of eigenfunctions. Thereby, more information is being extracted from the interaction between the relatively stable first few eigenfunctions.

| spectral gradient fields | \((\sqrt{\lambda_i \lambda_j})^{-1}(\nabla \phi_i(x), \nabla \phi_j(x))\), \((\sqrt{\lambda_i \lambda_j})^{-1}((\nabla \phi_i(x) \times \nabla \phi_j(x)) \cdot n(x)); i, j \geq 1\) |
|spectral embedding [BBG94] | \(e^{-\lambda_i t} \phi_i(x); i \geq 1, t > 0\) |
| global point signature [Rus07] | \((\sqrt{\lambda_i})^{-1} \phi_i(x); i \geq 1\) |
| heat kernel signature [SOG09] | \(\sum_k \exp(-\lambda_k t) \phi_k^2(x); t > 0\) |

Table 3.1: The spectral gradient fields embedding and other known spectral embeddings and signatures.

### 3.1.1 Contribution

We propose to find correspondence by embedding the shapes using a point-wise feature vector, which is based on the inner and external products between the GFs of pairs of eigenfunctions. For each pair of eigenfunctions \(\phi_i, \phi_j\), that correspond to the eigenvalues \(\lambda_i, \lambda_j\), we calculate their respective gradient fields \(\nabla \phi_i, \nabla \phi_j\), and compute the inner
Figure 3.1: Inner and external products of the gradient fields of a pair of eigenfunctions $\phi_1$, $\phi_3$. The inner product (third column, and bottom figure) exhibits finer details of the neck and hands of the human shape. Notice in the forth column, that the difference between front and back is captured by the external product. In the last two rows we see that these features are stable under a natural pose of the articulated object. In all our figures, red and blue colors represent high and low values of scalar functions on the shapes.
product $\langle \nabla \phi_i, \nabla \phi_j \rangle$, and the external product taken as the cross product in the normal direction $(\nabla \phi_i \times \nabla \phi_j) \cdot \mathbf{n}$. Figure 3.1 presents two eigenfunctions and their corresponding spectral GFs inner and external products. Because the inner and external products depend on the eigenfunctions and the gradient operator, they are relatively stable under isometric deformations, different poses and articulations. These features enable us to extract fine geometric information from a pair of eigenfunctions, as well as obtaining the global structure of an object. For the human shape shown as an example in Figure 3.1, the inner product (third column and bottom) distinguishes between the neck’s left and right sides, while the external product (fourth column) partitions the body’s front and back.

The proposed feature vector defines an embedding of shapes into Euclidean space. To the best of our knowledge, this is the first time that spectral gradient fields are used for such an embedding (see Table 3.1 for a summary of existing embedding methods). We prove that the proposed spectral gradient fields embedding is injective. Therefore, it naturally induces a distance function between any two points on the surface. In this metric space a distance between shapes is relatively simple to compute. We define such a distance measure, which we refer to as the spectral gradient fields distance, in a way similar to the definition of the spectral embedding distance defined by Bérard et al. [BBG94]. In the classic spectral embedding, the target space is described by the eigenfunctions of the Laplace-Beltrami operator. In the proposed target space, we embed the shapes into a much richer space, by using the conformal-based inner product between GFs of the eigenfunctions [ACG+13]. We describe the properties of the proposed distance measure, and prove that it is a metric.

3.1.2 Metric Geometry

A metric space $(X,d_X)$ is a pair, where $X$ is a set and $d_X : X \times X \to \mathbb{R}_{\geq 0}$ is a well defined distance function, satisfying the symmetry property $d_X(x_1,x_2) = d_X(x_2,x_1)$, the identity property $d_X(x_1,x_2) = 0$ iff $x_1 = x_2$, and the triangle inequality $d_X(x_1,x_3) \leq d_X(x_1,x_2) + d_X(x_2,x_3)$ for any $x_1, x_2, x_3 \in X$.

Suppose $A, B \in X$ are two closed subsets of a common bigger metric space $(X,d_X)$, and we want to compare $A$ to $B$ in order to decide whether they represent the same object or not, one might compute the Hausdorff distance $d_H(A,B)$ between $A$ and $B$

$$d_H(A,B) \equiv \max(\sup_{a \in A} \inf_{b \in B} d_X(a,b), \sup_{b \in B} \inf_{a \in A} d_X(a,b)),$$

and designate two objects $A, B \in X$ as being identical, if $d_H(A,B) = 0$.

A different approach to compare objects, is to treat them as metric spaces, and check if these metric spaces are isometric or not. We say the metric spaces $(X,d_X)$ and $(Y,d_Y)$ are isometric when there exists a bijective mapping $\varphi : X \to Y$ such that $d_X(x_1,x_2) = d_Y(\varphi(x_1),\varphi(x_2))$ for all $x_1,x_2 \in X$. Such a $\varphi$ is an isometry between
(X, d_X) and (Y, d_Y). In other words, an isometry between two shapes is a map between their two dimensional surfaces that preserves the distances between any two points.

For example, when considering shapes as two-dimensional Riemannian manifolds embedded in the three dimensional Euclidean space, and if one is interested in invariance to deformations of a surface that preserve the geodesic metric, then the surfaces can be treated as metric spaces equipped with the geodesic distances of the Riemannian manifold [MS05, BBK06].

3.1.3 The gradient and the Laplace-Beltrami operator

Let X be a Riemannian manifold. For any smooth function f : X → ℝ the gradient of f is the vector field ∇f defined through the Riemannian manifold’s inner product, such that for any vector field U,

\[ \langle \nabla f(x), U(x) \rangle = \partial_U f(x), \]

where \( \partial_U f(x) \) is the directional derivative of f at \( x \in X \), in the direction \( U(x) \).

The Laplace-Beltrami operator, denoted by \( \Delta \), is the divergence of the gradient

\[ \Delta f \equiv \text{div} \text{grad} f, \]

and can be considered as a generalization of the standard notion of the Laplace operator to compact Riemannian manifolds [Taa95, Lév06, Ros97]. The Laplace-Beltrami operator is invariant to geodesic-preserving deformations because it is defined in terms of the surface metric tensor which is isometry invariant.

The eigendecomposition of \(-\Delta\) consists of non-negative eigenvalues \( 0 = \lambda_0 < \lambda_1 \leq \cdots \leq \lambda_i \leq \cdots \), satisfying

\[ -\Delta \phi_i \equiv \lambda_i \phi_i. \]

The set of corresponding eigenfunctions given by

\[ \Phi \equiv \{ \phi_0, \phi_1, \cdots, \phi_i, \cdots \}, \]

forms an orthonormal eigenbasis, such that \( \int_X \phi_i(x)\phi_j(x)dV = \delta_{ij} \), where \( dV \) is the volume element on the manifold X.

3.1.4 Spectral embedding

Bérard et al. [BBG94] used the spectral properties of the heat operator \( e^{t\Delta} \) to define a metric between two Riemannian manifolds \( X, Y \in \mathcal{M} \). Here, \( \mathcal{M} \) is the set of all closed (i.e., compact without boundary) Riemannian manifolds of dimension n. Given a Riemannian manifold \( X \in \mathcal{M} \) with volume Vol(X) and some \( t > 0 \), they based their
metric on the eigendecomposition of the heat kernel and defined the spectral embedding $I_t^\Phi : X \to \ell^2$ from the Riemannian manifold $X$ into the Hilbert space $\ell^2$ of real valued square-summable sequences

$$I_t^\Phi (x) \equiv \{ \sqrt{\text{Vol}(X)} e^{-\lambda_i t/2} \phi_i(x) \}_{i \geq 1},$$

utilizing the eigenfunctions $\phi_i$ and eigenvalues $\lambda_i$ of the Laplace-Beltrami operator $\Delta$.

Given a pair of eigenbases $\Phi^X, \Phi^Y$, they embedded the two Riemannian manifolds $X$ and $Y$ into $I_t^\Phi^X$ and $I_t^\Phi^Y$, respectively. Then, they measured the Hausdorff distance $d_H$ between the manifolds in the common $\ell^2$ space. Bérard et al. defined the distance $d_{EMB}^t : \mathcal{M} \times \mathcal{M} \to \mathbb{R}_{\geq 0}$ between the manifolds $X, Y$, as the upper-bound of the Hausdorff distance between any eigenbasis $\Phi^X$ and its closest counterpart $\Phi^Y$.

$$d_{EMB}^t(X,Y) \equiv \max(d_{I_t^\Phi^X}(X,Y), d_{I_t^\Phi^Y}(Y,X)),
\quad d_{I_t^\Phi}(X,Y) \equiv \sup_{\{\Phi^X\}} \inf_{\{\Phi^Y\}} d_H(I_{t^\Phi^X}, I_{t^\Phi^Y}).$$

We call $d_{EMB}^t(X,Y)$ the spectral embedding distance (for a detailed description of the spectral embedding distance, we refer the reader to [BBG94]). They showed that for any fixed $t > 0$, the spectral embedding distance $d_{EMB}^t$ is a metric between isometry classes of Riemannian manifolds. In particular, $d_{EMB}^t(X,Y) = 0$ if and only if the Riemannian manifolds $X$ and $Y$ are isometric.

### 3.2 Spectral gradient fields embedding

**Inner and external products.** A spectral gradient field is a tangent vector field defined as the gradient of an eigenfunction $\nabla \phi_i$. The feature vector we define is based on the inner product between two spectral gradient fields, $\omega_{i,j} : X \to \mathbb{R}$ for all $i,j \geq 1$, where

$$\omega_{i,j}(x) \equiv \text{Vol}(X) (\sqrt{\lambda_i \lambda_j})^{-1} (\nabla \phi_i(x), \nabla \phi_j(x)). \quad (3.1)$$

For the special case that the Riemannian manifold is an oriented two-dimensional manifold embedded in $\mathbb{R}^3$, we enrich the feature vector by using the cross product between two spectral GFs in the normal direction, $\nu_{i,j} : X \to \mathbb{R}$ for all $i,j \geq 1$, where

$$\nu_{i,j}(x) \equiv \text{Vol}(X) (\sqrt{\lambda_i \lambda_j})^{-1} (\nabla \phi_i(x) \times \nabla \phi_j(x)) \cdot \mathbf{n}(x). \quad (3.2)$$

**Embedding.** Let us study the embedding and the distance between shapes that are induced by this feature vector. We limit our analysis to the inner product between the gradients of the eigenfunctions. Thus, we define the spectral GFs embedding $J_t^\Phi : X \to \ell^2$...
as
\[ J_t^\Phi(x) \equiv \{ e^{-(\lambda_i + \lambda_j)t/2} \omega_{ij}(x) \}_{i,j \geq 1}, \quad t > 0. \]

**Proposition 3.2.1.** Let \( x_1, x_2 \in X \) be two distinct points on the Riemannian manifold \( X \in \mathcal{M} \). Then, there exists a smooth function \( f \), such that \( \nabla f(x_1) \neq 0 \), and \( \nabla f(x_2) = 0 \).

The proof of Proposition 3.2.1 is given in 3.5.

**Theorem 3.1.** For a Riemannian manifold \( X \in \mathcal{M} \), the embedding \( J_\Phi t : X \to \ell^2 \) is injective, i.e., \( x_1 \neq x_2 \iff J_\Phi(t)(x_1) \neq J_\Phi(t)(x_2) \).

**Proof.** The proof is motivated by Rustamov’s analysis of his GPS embedding [Rus07]. Suppose that for two different points \( x_1, x_2 \in X \), we have \( J_\Phi{t}(x_1) = J_\Phi{t}(x_2) \). This means that for all \( i,j \geq 1 \), we have \( \langle \nabla \phi_i(x_1), \nabla \phi_j(x_1) \rangle = \langle \nabla \phi_i(x_2), \nabla \phi_j(x_2) \rangle \). Now, any smooth scalar function \( f : X \to \mathbb{R} \) can be represented as a linear combination of the eigenbasis \( f = \sum_i a_i \phi_i \). We can thereby write the norm of the gradient as
\[ \|\nabla f(x_1)\|^2 = \sum_{i,j} a_i a_j \langle \nabla \phi_i(x_1), \nabla \phi_j(x_1) \rangle = \sum_{i,j} a_i a_j \langle \nabla \phi_i(x_2), \nabla \phi_j(x_2) \rangle = \|\nabla f(x_2)\|^2. \]

On the other hand, using Proposition 3.2.1, there always exists a smooth function \( f \), such that \( \|\nabla f(x_1)\| \neq \|\nabla f(x_2)\| \), a contradiction. Therefore, \( J_\Phi(t)(x_1) \neq J_\Phi(t)(x_2) \).

**Theorem 3.2.** The family of maps \( \{ J_\Phi \}_{t>0} \) is invariant to global scaling of the metric.

See 3.6 for a proof of this Theorem.

**Point to point distance.** Given a Riemannian manifold \( X \in \mathcal{M} \), the embedding \( J_\Phi : X \to \ell^2 \) induces a metric \( \tilde{d}_X : X \times X \to \mathbb{R}_{\geq 0} \) on the manifold, in a way that the distance between any two points \( x_1, x_2 \in X \) coincides with the distance between the images of these points in the \( \ell^2 \) space
\[ \tilde{d}_X(x_1, x_2) \equiv \| J_\Phi(t)(x_1) - J_\Phi(t)(x_2) \|_{\ell^2}. \]

The metric \( \tilde{d}_X(x_1, x_2) \) is a well defined distance function which is invariant to the choice of the orthonormal eigenbasis \( \Phi \).

**Distance between Riemannian manifolds.** Let \( X, Y \in \mathcal{M} \) be the two closed Riemannian manifolds we would like to compare, and let us be given some \( t > 0 \). Given a pair of eigenbases \( \Phi^X, \Phi^Y \), we embed the two Riemannian manifolds \( X \) and \( Y \) into \( J_\Phi^X \) and \( J_\Phi^Y \), respectively, and measure the Hausdorff distance \( d_H \) between
the manifolds in the common $\ell^2$ space. We define the spectral gradient fields distance, denoted by $d_{t}^{GF} : \mathcal{M} \times \mathcal{M} \rightarrow \mathbb{R}_{\geq 0}$, as the supremum of the Hausdorff distance between any eigenbasis $\Phi^X$ and its closest counterpart $\Phi^Y$.

\[
d_{t}^{GF} (X,Y) \equiv \max(d_{J_t}(X,Y), d_{J_t}(Y,X)),
\]

\[
d_{J_t}(X,Y) \equiv \sup_{\{\Phi^X\}, \{\Phi^Y\}} \inf d_H(J_{t}^{\Phi^X}, J_{t}^{\Phi^Y}).
\]

We say that $X, Y$ are spectral gradient fields equivalent, if the metric spaces $(X, \tilde{d}_X)$, $(Y, \tilde{d}_Y)$ are isometric and if $d_{t}^{GF}(X,Y) = 0$.

**Theorem 3.3.** The spectral GFs distance $d_{t}^{GF}$ is a metric between spectral GFs equivalent classes of Riemannian manifolds.

The proof of Theorem 3.3 is given in 3.7.

**Remark.** The use of the spectral GFs external product $((\nabla \phi_i \times \nabla \phi_j) \cdot n(x))$ is obviously restricted to oriented two-dimensional surfaces embedded in the three-dimensional Euclidean space. Although the cross product of two tangent vectors is extrinsic in nature, its projection in the normal direction is invariant to isometric deformations. Consequently, for a two-dimensional surface embedded in $\mathbb{R}^3$, adding the elements $e^{-(\lambda_i + \lambda_j)t/2} \nu_{i,j}$ for all $i, j \geq 1$ to the embedding $J_{t}^{\Phi}$, leaves the above analysis unchanged. Hence, the statements of Theorems 3.1-3.3 hold for the extended feature vector as well.

### 3.3 Experiments and Results

**Truncated feature vector.** Let us study the spectral GFs feature vector when we are restricted to using the $N_0$ eigenfunctions that correspond to the lowest eigenvalues. Remember that we are driven by the task of finding correspondence between two nearly isometric shapes. Because empirical evidence suggests that there are only a few eigenfunctions that are stable to approximately isometric deformations, for the shape matching application we prefer to avoid eigenfunctions that correspond to high eigenvalues.

In our analysis (and implementation) we use the plain vanilla feature vector $J^\Phi \equiv J_\omega^\Phi \cup J_\nu^\Phi$, where $J_\omega^\Phi \equiv \{\omega_{i,j}\}_{1 \leq i \leq j \leq N_0}$ and $J_\nu^\Phi \equiv \{\nu_{i,j}\}_{1 \leq i < j \leq N_0}$ are the inner and external parts of the feature vector, respectively. $J^\Phi$ excludes the dependency on the parameter $t$. In that case, the number of nontrivial and unique features in $J^\Phi$ is $N_0^2$. This feature vector is invariant to global scaling as shown in the proof of Theorem 3.2.

**Spectrum.** As a case study, we analyze the spectral GFs embedding of a human shape. For this shape, assume we are given $N_0 = 6$ eigenfunctions. Accordingly, the size of the spectral GFs feature vector is $6^2 = 36$. Since $J^\Phi$ is nonlinear in $\phi$, it is interesting to visualize how the energy of the feature vector is distributed as a function of the
Figure 3.2: Spectral density of the spectral GFs feature vector computed on the human shape, compared to the spectrum of the GPS embedding.

eigenvalues. In Figure 3.2, we plot the energy of the spectral GFs features vector as a function of the eigenvalues, and compare it to the spectrum of the global point signature (GPS) feature vector. We see, that the spectrum of the inner and external products is more widely distributed, which means that they could reflect finer details of the surface structure.

Intra-shape point to point distances. The feature vector \( J^\Phi \) induces a distance \( \tilde{d}_X : X \times X \rightarrow \mathbb{R}_{\geq 0} \) between points on the shape

\[
\tilde{d}_X(x_1, x_2) \equiv \| J^\Phi(x_1) - J^\Phi(x_2) \| = \sum_{i,j=1}^{N_0} (\omega_{i,j}(x_1) - \omega_{i,j}(x_2))^2 + (\nu_{i,j}(x_1) - \nu_{i,j}(x))^2.
\]

Figure 3.3 depicts the distance function \( \tilde{d}_X(p, x) \), between a fixed point \( p \in X \) on the front of the human shape to any other point \( x \in X \). We see that \( \tilde{d}_X(p, x) \) is correlated to the geodesic distance between point \( p \) to point \( x \). We also observe that \( \tilde{d}_X(p, x) \) is invariant to the isometric deformation, as expected.

Inter-shape point to point distances. For two shapes \( X, Y \), let us assume we are
Figure 3.3: The spectral GFs feature vector induced the distance function $\tilde{d}_X(p, x)$ between a fixed point $p$ to all points $x \in X$ (left), and the distance function $\tilde{d}_Y(q, y)$ from the corresponding point $q = \varphi(p)$ to all points on shape $Y$ (right).

Figure 3.4: In the common space, the inter-shape distance $d_{X,Y}(p, y)$, is calculated between a fixed point $p$ on shape $X$ and all points in shape $Y$. From left to right: spectral GFs with the embeddings $J_\Phi, J_\omega^\Phi$ (inner-products), $J_\nu^\Phi$ (external products) and the GPS embedding.
given compatible eigenbases of size \(N_0\), meaning that \(\phi_i^X(x) \approx \phi_i^Y(\varphi(x))\), \(\forall i \in \{1 \ldots N_0\}\), for all points \(x \in X\). In the common embedding space, the distance \(d_{X,Y}(x, y) : X \times Y \to \mathbb{R}\) between a point on one shape to a point on the other shape is simply

\[
d_{X,Y}(x, y) = \left\| J^{\phi_X}(x) - J^{\phi_Y}(y) \right\|
\]

\[
= \sum_{i,j=1}^{N_0} (\omega_{i,j}^X(x) - \omega_{i,j}^Y(y))^2 + (\nu_{i,j}^X(x) - \nu_{i,j}^Y(y))^2.
\]  

(3.3)

In Figure 3.4 we visualize the distance \(d_{X,Y}(p, y)\) from the fixed point \(p \in X\) to all points \(y \in Y\). We see (column 1), that points on surface \(Y\) that are close to the image of \(p\), also have low values of \(\tilde{d}_{X,Y}(p, y)\). Hence, for a point to point correspondence application, we should assign the corresponding point of \(p\) as the one with the smallest distance from \(p\), in the common Euclidean space. Notice especially, that while the GPS embedding (column 4) makes points on the back mistakenly close to \(p\), by using the spectral GFs feature vector this effect is diminished. This can be explained by the use of the \(\nu_{i,j}\) features (column 3), that include the cross product operation which is an extrinsic operation. Pure intrinsic embeddings do not use this important attribute of the surface, so the embedding \(J^\phi\) presents additional global features. Combined with the inner product based embedding \(J^\omega\) which is more locally accurate (column 2), the spectral gradient fields embedding \(J^\phi\) is well suited for finding point to point correspondences.

**Compatible eigenfunctions.** For two nearly isometric shapes \(X, Y\), one might say that the eigenfunction \(\phi_i^X\) is matched to the eigenfunction \(\phi_j^Y\) up to a sign, if the difference between \(\phi_i^X\) and \(\phi_j^Y \circ \varphi\)

\[
E(i, j) \equiv \min_{S \in \{+1, -1\}} \int_X |\phi_i^X(x) - S\phi_j^Y(\varphi(x))|^2 da(x),
\]  

(3.4)

is lower than a certain threshold, where \(da\) is the area element of \(X\). In Figure 3.5, we see that for the databases we analyze [ASP+05, BBK08], the probability of finding a match for the \(i\)th eigenfunction drops sharply as \(i\) increases beyond a certain point (the seventh eigenfunction for the databases we tested). This means that \(N_0\), the number of eigenfunctions that is used for defining the common embedding, should be kept small.

**Implementation.** For the purpose of testing our ideas, we developed a holistic correspondence system. At the heart of the system we use the spectral GFs embedding, defined in Eq. (3.1) and Eq. (3.2), and compute the inter-shape point to point distances \(d_{X,Y}(x, y)\) of Eq. (3.3), where we use \(N_0 = 6\) compatible eigenfunctions that are found automatically (based on [SK14]). Then, we apply a nearest neighbor search to find the initial correspondence

\[
\hat{\varphi}_0(x) = \arg\min_{y \in Y} d_{X,Y}(x, y).
\]  

(3.5)
Figure 3.5: The empirical probability that there is an eigenfunction $\phi^Y_j$ that matches $\phi^X_i$ up to a sign. The matching threshold is set to 0.1, i.e., we check if $E(i,j) < 0.1$ for some $j$ (see Eq. (3.4)). Notice, that this probability drops sharply beyond the seventh eigenfunction.

As a preprocess step, pairs of points that have low possibility of being matched are filtered out by comparing a mix of global and local features, using the compatible eigenfunctions themselves and the wave kernel signature [ASC11]. We filter out pairs of points if their feature vectors are too far in the $L^2$ sense. After applying the spectral GFs embedding method using Eq. (3.5), we refine the correspondence with the iterative closest spectral kernel maps (ICSKM) algorithm [SK]. The alignment of the spectral kernels of the two shapes by the ICSKM algorithm, produces the final dense map $\hat{\phi} : X \rightarrow Y$.

Performance. We tested the proposed method on pairs of shapes represented by triangulated meshes from both the SCAPE database [ASP+05] and the TOSCA database [BBK08]. The SCAPE dataset contains 71 registered meshes of a particular person in different poses. The TOSCA dataset contains 80 densely sampled synthetic human and animal surfaces, divided into several classes with given ground-truth point-to-point correspondences between the shapes within each class. We compare the results of our framework to several correspondence detection methods.

- Spectral GFs Embedding - the method proposed in this paper.
- Spectral Maps - the correspondence system used in this paper, without the use of the spectral GFs embedding. At the preprocess stage we filter out all
SCAPE Correspondence

Figure 3.6: Evaluation of the correspondence system applied to shapes from the SCAPE database, using the protocol of [KLF11], (with allowed symmetries).

correspondences except one.

• Blended - the method proposed by Kim et al. that uses a weighted combination of isometric maps [KLF11].

Figure 3.6 and Figure 3.7 compare our correspondence method with existing methods on the SCAPE and TOSCA benchmarks, using the evaluation protocol proposed in [KLF11]. For each method we plot the distortion curves with the ICSKM refinement procedure (solid), and without it (dotted). The distortion curves describe the percentage of surface points falling within a relative geodesic distance from what is assumed to be their true locations. For each shape, the geodesic distance is normalized by the square root of the shape’s area. It is evident from the benchmarks that the proposed method is well suited for finding maps between approximately isometric shapes, and together with the ICSKM algorithm achieves state-of-the-art results.

Reflective symmetry. Intrinsic symmetry detection can be viewed as finding a map from a shape to itself [RBBK10]. When the shape has a reflective symmetry, its self-map flips the orientation of the surface. In that case, we can apply the proposed shape matching method, by adjusting the normal to the surface to point inward, and
TOSCA Correspondence

Figure 3.7: Evaluation of the correspondence system applied to shapes from the TOSCA database, using the protocol of [KLF11].

consequently change the external product features of the reflective shape to be $-\nu_{i,j}$.

Then, we can use the estimated reflective self-map $\hat{\varphi}_R : X \to X$ to partition the shape into its two mirror halves. First, we find a point $p$ that is well inside one of the two halves of the shape, for example by selecting a point that is as far as possible from its reflective image $\hat{\varphi}(p)$, i.e., we find the point $p = \arg\max_{x \in X} g(x, \hat{\varphi}_R(x))$, where $g : X \times X \to \mathbb{R}_{\geq 0}$ is the geodesic distance function between two points on the surface of the shape. Then, we check if each point $x \in X$ belongs to the half of the shape that includes $p$. We assign $x$ to this half if $g(x, p) + g(\hat{\varphi}_R(x), \hat{\varphi}_R(p))$ is less than $g(\hat{\varphi}_R(x), p) + g(x, \hat{\varphi}_R(p))$.

In Figure 3.8, we demonstrate the effectiveness of the proposed approach, by visualizing the bilateral symmetry for several shapes from the TOSCA database.

3.4 Conclusions

A new feature vector for embedding nonrigid shapes was introduced and integrated into a holistic shape matching system. The embedding, which is proved to be injective, induces a distance between points on the surface, and can be defined to measure a distance between shapes. We have demonstrated the effectiveness of the proposed approach by achieving state-of-the-art results on shape matching benchmarks.
Figure 3.8: Bilateral symmetry. The two mirror halves of several shapes from the TOSCA database are found by embedding the shape using spectral gradient fields.

In the future, we plan to examine the properties of the GFs embedding in conjunction with compatible functions on the two shapes, that are different from the eigenfunctions. For example, one may use the heat kernel signature (HKS), wave kernel signature (WKS) [ASC11] or the heat kernel maps (HKM) [OMMG10] set with landmark correspondences. These options seem particularly useful when the shapes are noisy, or the deformations are large, making the eigenfunctions incompatible.

3.5 Proof of Proposition 3.2.1

We need to prove that if \( x_1, x_2 \) are two distinct points on the Riemannian manifold, then, there exists a smooth function \( f \), such that \( \nabla f(x_1) \neq 0 \), and \( \nabla f(x_2) = 0 \). We use two technical Lemmas.

**Lemma 3.5.1.** Let \( U_1, U_2 \) be two disjoint closed sets. Then, there exists a smooth function \( u_1 : X \to \mathbb{R}_{\geq 0} \), such that

\[
\begin{align*}
  i & \quad u_1(x) = 1, \quad \forall x \in U_1, \\
  ii & \quad u_1(x) = 0, \quad \forall x \in U_2.
\end{align*}
\]

**Proof.** \( U_1, U_2 \) are disjoint, thereby \( \{X \setminus U_2, X \setminus U_1\} \) is an open cover for \( X \). Hence, there exists a partition of unity of nonnegative functions \( u_1, u_2 : X \to \mathbb{R}_{\geq 0} \), such that
the support of \( u_1 \) is contained in \( X \setminus U_2 \) and the support of \( u_2 \) is contained in \( X \setminus U_1 \). Moreover, \( u_1 + u_2 = 1 \) everywhere.

i The support of \( u_2 \) is contained in \( X \setminus U_1 \), hence, \( u_2(x) = 0, \forall x \in U_1 \). Therefore, 
\[ u_1(x) = 1 - u_2(x) = 1, \forall x \in U_1. \]

ii Because the support of \( u_1 \) is contained in \( X \setminus U_2 \), it follows that \( u_1(x) = 0, \forall x \in U_2 \).

\[ \square \]

**Lemma 3.5.2.** Let \( x_1, x_2 \) be two distinct points on the Riemannian manifold. Then, there exists a smooth function \( u_1 : X \to \mathbb{R}_{\geq 0} \), such that \( u_1(x_1) = 1, u_1(x_2) = 0, \nabla u_1(x_1) = 0 \) and \( \nabla u_1(x_2) = 0 \).

**Proof.** Construct two disjoint closed sets \( U_1, U_2 \), such that \( x_1 \) is in the interior of \( U_1 \), and \( x_2 \) is in the interior of \( U_2 \). Then, use Lemma 3.5.2 to make such \( u_1 \). \[ \square \]

Now, take a smooth function \( g : X \to \mathbb{R} \), with a non-zero gradient at point \( x_1 \), i.e., \( g(x_1) \neq 0 \). Using Lemma 3.5.2, construct the function \( f = u_1g \). By the chain rule

\[
\nabla f(x_1) = u_1(x_1)\nabla g(x_1) + g(x_1)\nabla u_1(x_1) = 1 \cdot \nabla g(x_1) + g(x_1) \cdot 0, \\
\nabla f(x_2) = u_1(x_2)\nabla g(x_2) + g(x_2)\nabla u_1(x_2) = 0 \cdot \nabla g(x_2) + g(x_2) \cdot 0.
\]

We conclude that \( \nabla f(x_1) \neq 0 \) and that \( \nabla f(x_2) = 0 \).

\[ \blacksquare \]

### 3.6 Proof of Theorem 3.2

**Proof.** If the Riemannian manifold \( \bar{X} \) is obtained by uniformly scaling the metric of the Riemannian manifold \( X \) by a factor \( \alpha > 0 \), then, the eigenvalues of the Laplace-Beltrami operator are scaled according to \( \lambda_i^X = \alpha^{-2}\lambda_i^\bar{X} \), the corresponding \( L^2 \)-normalized eigenfunctions become \( \phi_i^X = \alpha^{-1}\phi_i^\bar{X} \), and the gradient operator reads \( \nabla^X = \alpha^{-1}\nabla^\bar{X} \).

We have

\[
\omega_{i,j}^X = \frac{\text{Vol}(\bar{X}) \langle \nabla^X \phi_i^X, \nabla^X \phi_j^X \rangle}{\sqrt{\lambda_i^X \lambda_j^X}} \\
= \frac{\alpha^2\text{Vol}(X) \langle \alpha^{-1}\nabla^X \alpha^{-1}\phi_i^X, \alpha^{-1}\nabla^X \alpha^{-1}\phi_j^X \rangle}{\sqrt{\alpha^{-2}\lambda_i^X \alpha^{-2}\lambda_j^X}} \\
= \frac{\text{Vol}(X) \langle \nabla^X \phi_i^\bar{X}, \nabla^X \phi_j^\bar{X} \rangle}{\sqrt{\lambda_i^\bar{X} \lambda_j^\bar{X}}} = \omega_{i,j}^\bar{X},
\]

which means that \( \omega_{i,j}^X \) is global scale invariant. Now, if we set \( \bar{t} = \alpha^2t \)

\[
(J_{\bar{t}}^\bar{X})_{i,j} = e^{-(\lambda_i^\bar{X} + \lambda_j^\bar{X})\bar{t}/2} \omega_{i,j}^\bar{X}
\]

44
3.7 Proof of Theorem 3.3

We would like to show that the spectral gradient fields distance \(d_{t}^{GF} : \mathcal{M} \times \mathcal{M} \rightarrow \mathbb{R}_{\geq 0}\) satisfies the following properties of a metric space.

P1: Symmetry - \(d_{t}^{GF}(X, Y) = d_{t}^{GF}(Y, X)\) for every \(X, Y \in \mathcal{M}\).

P2: Triangle-inequality - For every three Riemannian manifolds \(X, Y, Z \in \mathcal{M}\),

\[d_{t}^{GF}(X, Z) \leq d_{t}^{GF}(X, Y) + d_{t}^{GF}(Y, Z).\]

P3: Identity -

(a) If \(X, Y \in \mathcal{M}\) are spectral GFs equivalent, then \(d_{t}^{GF}(X, Y) = 0\).

(b) If \(d_{t}^{GF}(X, Y) = 0\), then \(X, Y\) are spectral GFs equivalent.

Proposition 3.7.1. For every three Riemannian manifolds \(X, Y, Z \in \mathcal{M}\),

\[d_{t}^{GF}(X, Z) \leq d_{t}^{GF}(X, Y) + d_{t}^{GF}(Y, Z).\]

Proof. We denote

\[d_{X,Y}(x,y) \equiv \left\| J_{t}^{\Phi_{X}}(x) - J_{t}^{\Phi_{Y}}(y) \right\|,\]

\[\text{dis}(\varphi) \equiv \sup_{x \in X} d_{X,Y}(x, \varphi(x)),\]

\[\text{dis}(\psi) \equiv \sup_{y \in Y} d_{X,Y}(\psi(y), y).\]

The Hausdorff distance between \(J_{t}^{\Phi_{X}}\) and \(J_{t}^{\Phi_{Y}}\) is

\[d_{H}(J_{t}^{\Phi_{X}}, J_{t}^{\Phi_{Y}}) = \max(\sup_{x \in X} \inf_{y \in Y} d_{X,Y}(x, y), \sup_{y \in Y} \inf_{x \in X} d_{X,Y}(x, y)).\]

Lemma 3.7.2. if \(d_{H} = \epsilon\), then there exist \(\varphi : X \rightarrow Y\) and \(\psi : Y \rightarrow X\) such that \(d_{X,Y}(x, \varphi(x)) \leq \epsilon, \forall x \in X\) and \(d_{X,Y}(\psi(y), y) \leq \epsilon, \forall y \in Y\).

Proof. take \(\varphi(x) = \inf_{y \in Y} d_{X,Y}(x, y)\) and \(\psi(y) = \inf_{x \in X} d_{X,Y}(x, y).\)

Now, if \(d_{t}^{GF}(X, Y) \leq \epsilon\), then, for any \(\Phi_{X}\), one can find \(\Phi_{Y}, \varphi, \psi\) (using Lemma 3.7.2) such that \(\text{dis}(\varphi) \leq \epsilon, \text{dis}(\psi) \leq \epsilon, i.e.,\)

\[d_{X,Y}(x, \varphi(x)) \leq \epsilon, \forall x,\]

\[d_{X,Y}(\psi(y), y) \leq \epsilon, \forall y.\]
Let $d_{t}^{GF}(X, Y) = \epsilon_1$ and $d_{t}^{GF}(Y, Z) = \epsilon_2$. Hence, for any $\Phi^X$, there exist two eigenbases $(\Phi^Y, \Phi^Z)$, and two pairs of corresponding mappings $(\varphi_1 : X \to Y, \psi_1 : Y \to X)$ and $(\varphi_2 : Y \to Z, \psi_2 : Z \to Y)$ satisfying $\text{dis}(\varphi_1) \leq \epsilon_1$, $\text{dis}(\psi_1) \leq \epsilon_1$, and $\text{dis}(\varphi_2) \leq \epsilon_2$, $\text{dis}(\psi_2) \leq \epsilon_2$. Denote by $\varphi = \varphi_2 \circ \varphi_1 : X \to Z$, $\psi = \psi_1 \circ \psi_2 : Z \to X$. Invoking the triangle inequality for $\ell^2$ spaces, one has

$$d_{X,Z}(x, \varphi(x)) \leq d_{X,Y}(x, \varphi_1(x)) + d_{Y,Z}(\varphi_1(x), \varphi(x)) \leq \text{dis}(\varphi_1) + \text{dis}(\varphi_2) \leq \epsilon_1 + \epsilon_2, \forall x \in X,$$

$$d_{X,Z}(\psi(z), z) \leq d_{X,Y}(\psi(z), \psi_1(z)) + d_{Y,Z}(\psi_2(z), z) \leq \text{dis}(\psi_1) \leq \epsilon_1 + \epsilon_2, \forall z \in Z.$$

This means that for any $\Phi^X$, we can find $\Phi^Z, \varphi, \psi$, such that $d_{X,Z}(x, \varphi(x))$ and $d_{X,Z}(\psi(z), z)$ are bounded by $\epsilon_1 + \epsilon_2$. Consequently,

$$\inf_{\{\Phi^X\}} d_H(J_t^{\Phi^X}, J_t^{\Phi^Z}) \leq \epsilon_1 + \epsilon_2, \forall \Phi^X.$$

Clearly,

$$d_{H}(X, Z) = \sup_{\{\Phi^X\}} \inf_{\{\Phi^Z\}} d_H(J_t^{\Phi^X}, J_t^{\Phi^Z}) \leq \epsilon_1 + \epsilon_2.$$

In the same way $d_{H}(Z, X) \leq \epsilon_1 + \epsilon_2$, implying $d_{t}^{GF}(X, Z) \leq \epsilon_1 + \epsilon_2$. \hfill \Box

**Proposition 3.7.3.** If $d_{t}^{GF}(X, Y) = 0$, then $X$, $Y$ are spectral GFs equivalent.

**Proof.** We need to prove there exists a bijective and distance preserving map between $X$ and $Y$. Because $d_{t}^{GF}(X, Y) = 0$, we have eigenbases $\Phi^X$, $\Phi^Y$ and a map $\varphi : X \to Y$ between them, such that $J_t^{\Phi^X}(x) = J_t^{\Phi^Y}(\varphi(x))$, $\forall x \in X$. Therefore, for all $x_1, x_2 \in X$

$$\left\| J_t^{\Phi^X}(x_1) - J_t^{\Phi^X}(x_2) \right\|_{\ell^2} = \left\| J_t^{\Phi^Y}(\varphi(x_1)) - J_t^{\Phi^Y}(\varphi(x_2)) \right\|_{\ell^2}.$$

We also know that $X$, $Y$ are equipped with the metrics $\tilde{d}_X$, $\tilde{d}_Y$ derived from the distances in the common space

$$\tilde{d}_X(x_1, x_2) = \left\| J_t^{\Phi^X}(x_1) - J_t^{\Phi^X}(x_2) \right\|_{\ell^2}, \forall x_1, x_2 \in X,$$

$$\tilde{d}_Y(y_1, y_2) = \left\| J_t^{\Phi^Y}(y_1) - J_t^{\Phi^Y}(y_2) \right\|_{\ell^2}, \forall y_1, y_2 \in Y.$$

Hence, $\tilde{d}_X$ is a distance preserving map

$$\tilde{d}_X(x_1, x_2) = \tilde{d}_Y(\varphi(x_1), \varphi(x_2)).$$
In the same way
\[ \tilde{d}_Y(\psi(x_1), \psi(x_2)) = \tilde{d}_Y(x_1, x_2). \]

Moreover, \( x = \psi(\varphi(x)) \) for all \( x \in X \) and \( y = \varphi(\psi(y)) \) for all \( y \in Y \), or else \( J^\Phi_x \) or \( J^\Phi_y \) would not be injective. Therefore, \( \varphi, \psi \) are bijections.

We conclude that the distance \( d_{GF}^t(X, Y) : \mathcal{M} \times \mathcal{M} \rightarrow \mathbb{R}_{\geq 0} \) satisfies the properties of Theorem 3.3.

P1: Symmetry - by definition, the function \( d_{GF}^t(X, Y) \) is invariant to permutation of \( X \) and \( Y \).

P2: Triangle-inequality - by Proposition 3.7.1.

P3: Identity -

(a) - \( X, Y \) cannot be spectral GFs equivalent if \( d_{GF}^t(X, Y) \neq 0 \).

(b) - by Proposition 3.7.3.

\[ \square \]
Chapter 4

Iterative Closest Spectral Kernel Maps

An important operation in geometry processing is finding the correspondences between pairs of shapes. Measures of dissimilarity between surfaces, has been found to be highly useful for nonrigid shape comparison. Here, we analyze the applicability of the spectral kernel distance, for solving the shape matching problem. To align the spectral kernels, we introduce the iterative closest spectral kernel maps (ICSKM) algorithm. The ICSKM algorithm farther extends the iterative closest point algorithm to the class of deformable shapes. The proposed method achieves state-of-the-art results on the Princeton isometric shape matching protocol applied, as usual, to the TOSCA and SCAPE benchmarks.

4.1 Introduction

Correspondence detection between pairs of shapes lies at the heart of many operations in the field of geometry processing. The problem of acquiring correspondence between rigid shapes has been widely addressed in the literature. As for non-rigid shapes, this problem remains difficult even when the space of deformations is narrowed to nearly isometric surfaces, which approximately preserve the geodesic distances between corresponding points on each shape.

A common approach for shape matching is to define a measure of dissimilarity between shapes modeled as 2-manifolds. The well-established Gromov-Hausdorff distance measures the maximum geodesic discrepancy between pairs of corresponding points of the two given shapes [MS05]. The point-wise map can be inferred to as a byproduct of the evaluation of the Gromov-Hausdorff distance. This approach was embraced by the Generalized Multi-Dimensional Scaling (GMDS) framework [BBK06]. Within the Gromov-Hausdorff framework, Bronstein et al. [BBM+10] suggested replacing the geodesic distance by the diffusion distance [CLL+05], exploiting the apparent stability of diffusion distances to local changes in the topology of the shape. Despite its
generality and theoretical beauty, it has been a challenge to apply the Gromov-Hausdorff framework in a straightforward manner to shape matching, mainly due to its intrinsically combinatorial nature.

Kasue and Kumura [KK94] extended the Gromov-Hausdorff distance framework to the family of spectral methods. The spectral kernel distance was constructed by replacing the metric defined on the manifolds with the heat kernel. The heat kernel provides a natural notion of scale, which is useful for multi-scale shape comparison. Recently, Mémoli [Mém09] introduced the spectral Gromov-Wasserstein distance, applying the theory of mass transportation. The spectral Gromov Wasserstein distance via the comparison of heat kernels satisfies all properties of a metric on the class of isometric manifolds.

The evaluation of the spectral kernel distance between two nearly isometric surfaces should be capable of discovering the mapping between them. Alas, the task is not straightforward, due to model impairments and the combinatorial nature of the problem. Therefore, to achieve highly accurate and dense correspondence, we need to make adaptations to this distance measure and design an efficient and robust optimization algorithm.

4.1.1 Contribution

Our main observation is that the alignment of the spectral kernels and the evaluation of the spectral kernel distance between two shapes can be achieved by extending the well established Iterative Closest Point (ICP) algorithm [YM92, BM92] to the class of nonrigid shapes. The classical ICP algorithm refines the correspondence between rigid shapes embedded in the three dimensional Euclidean space. The key idea is simple. Given an initial map between the shapes, find the best rotation and translation that aligns the shapes, apply it and calculate new correspondence by the nearest neighbor algorithm.

As for nonrigid shapes, a similar idea was presented by the iterative post-process refinement algorithm [OBCS+12]. Instead of aligning the shapes in the three dimensional Euclidean domain, this method estimates the transformation that best fits the shapes in the spectral domain. Given an initial map from shape \( X \) to shape \( Y \), one linear constraint is generated for each point \( x \in X \), and the least squares method is used to infer the transformation matrix.

The proposed iterative closest spectral kernel maps (ICSKM) algorithm extends this idea by finding the transformation that best matches the respective spectral kernels \( K(x, x') \) and \( \tilde{K}(y, y') \) of the shapes \( X \) and \( Y \). Now, each pair of points \( x, x' \in X \) generates a linear constraint by including its normalized kernel relation \( K(x, x') / K(x, x) \). The two dimensional information, effectively improves the refinement procedure. The optimization problem is solved by the least squares method with Tikhonov regularization [Tik63, HK70]. The algorithm is shown to be robust, flexible and easy to implement.
It can be used efficiently as a refinement procedure of rough or sparse correspondence detection methods. The main advantage of the ICSKM algorithm over existing methods is in the combination of the iterative post-process refinement algorithm with the two dimensional constraints of the spectral kernel, resulting in highly accurate correspondence maps.

4.2 Related work

4.2.1 spectral kernel distance

The heat kernel \( K_t(x, x') \) is defined as the solution of the heat equation \( \frac{\partial u}{\partial t} = \Delta u \), with a point heat source at \( x \in X \), measured at point \( x' \in X \) after time \( t > 0 \), where \( \Delta \) denotes the Laplace-Beltrami (LB) operator.

Kasue and Kumura [KK94] defined the metric \( d(X, Y) \) between the Riemannian manifolds \( X \) and \( Y \) by comparing their respective heat kernels

\[
d(X, Y) \equiv \inf_{\varphi : X \to Y, \psi : Y \to X} \max\{\text{dis}(\varphi), \text{dis}(\psi)\},
\]

(4.1)
taking the supremum of kernel distortion for all \( t > 0 \)

\[
\text{dis}(\varphi) \equiv \sup_{x, x' \in X, t > 0} u(t)d_t(x, x', \varphi(x), \varphi(x')),
\]

\[
\text{dis}(\psi) \equiv \sup_{y, y' \in Y, t > 0} u(t)d_t(\psi(y), \psi(y'), y, y'),
\]

where \( d_t(x, x', y, y') \) measures the absolute discrepancy between the heat kernels \( K_t(x, x') \) and \( \tilde{K}_t(y, y') \)

\[
|\text{Vol}(X)K_t(x, x') - \text{Vol}(Y)\tilde{K}_t(y, y')|.
\]

\( \text{Vol}(X) \) and \( \text{Vol}(Y) \) are the volumes of \( X \) and \( Y \), respectively. The function \( u(t) \equiv e^{-(t+1)/t} \) is used to normalize the kernels for different values of \( t \), and make sure that it will not blow up as \( t \to 0 \). We denote \( d(X, Y) \) as the spectral kernel distance. The spectral kernel distance is a metric between isometry classes of Riemannian manifolds, which means, in particular, that two manifolds are at zero distance if and only if they are isometric.

In practice, a more tractable \( L_2 \) version of Eq. (4.1) can be optimized by finding the map \( \varphi : X \to Y \) that best aligns the spectral kernels of two shapes for a fixed time.
In the discrete setting, the spectral kernel distortion can be formalized as

$$\min_{\varphi: X \to Y} \sum_{x, x' \in X} |K_t(x, x') - \tilde{K}_t(\varphi(x), \varphi(x'))|^2.$$ (4.2)

### 4.2.2 Post-process iterative refinement algorithm

The post-process iterative refinement algorithm [OBCS+12] takes as input an initial map, iteratively finds the transformation matrix between the spectral bases of the two compared shapes, and outputs a dense correspondence between the shapes. Here, we use the first \(n\) Laplace-Beltrami eigenfunctions as the spectral basis [Bel64, Tau95, SKM98, Lév06]. The eigendecomposition of the LB operator consists of non-negative eigenvalues \(0 = \lambda_0 < \lambda_1 < \cdots < \lambda_i < \cdots\), with corresponding eigenfunctions \(\Phi = \{\phi_0, \phi_1, \cdots, \phi_i, \cdots\}\) that forms an orthonormal basis, which is well suited for representing near isometric shapes [OBCS+12, AK13]. In this case, the post-process iterative refinement algorithm is similar to the well known Iterative Closest Point (ICP) [YM92, BM92] in \(n\) dimensions, except that it is performed in the natural spectral domain, rather than the standard Euclidean space.

Let \(\varphi: X \mapsto Y\) be a bijective mapping between shapes \(X\) and \(Y\). If we are given a scalar function \(f: X \mapsto \mathbb{R}\), then, we can obtain a corresponding function \(g: Y \mapsto \mathbb{R}\) by the composition \(g = f \circ \varphi^{-1}\). Given the bases \(\Phi\) and \(\tilde{\Phi}\) on the shapes \(X\) and \(Y\), respectively, we can represent \(f\) as a row vector \(\mathbf{a}\) with coefficients \(a_i\), and equivalently, \(g\) as a row vector \(\mathbf{b}\) with coefficients \(b_i\). It is easy to show that we can write a linear transformation \(\mathbf{a} = \mathbf{b}C\), where the transformation matrix \(C\) is independent of \(f\) and is completely determined by the bases \(\Phi, \tilde{\Phi}\) and the map \(\varphi\).

Now, suppose we have point-to-point correspondences, such that each point \(x \in X\) corresponds to some point \(y \in Y\) by the mapping \(y = \varphi(x)\). In this case, the delta function \(\delta_x\) at point \(x \in X\) corresponds to the delta function \(\check{\delta}_y\) at point \(y = \varphi(x)\). We can represent the delta function \(\delta_x\) in the basis \(\Phi\) by

$$\mathbf{a}_x = \Phi(x) = (\phi_1(x), \phi_2(x), \ldots, \phi_i(x), \ldots).$$

Equivalently, the function \(\check{\delta}_y\) can be represented in the basis \(\tilde{\Phi}\) as

$$\mathbf{b}_y = \tilde{\Phi}(y) = (\tilde{\phi}_1(y), \tilde{\phi}_2(y), \ldots, \tilde{\phi}_i(y), \ldots).$$

Then, we can construct the function preservation constraints \(A = BC\), where the corresponding matrices \(A\) and \(B\) are built by stacking the row vectors \(\mathbf{a}_x\) and \(\mathbf{b}_y\), respectively. Therefore, at every iteration of the refinement procedure, we can infer the transformation matrix \(C\) from previous correspondences by solving \(A = BC\) with the least square method. Then, a new map can be found by searching for the point \(y \in Y\), such that the row vector \(\tilde{\Phi}(y)C\) is the closest to \(\Phi(x)\).
4.3 Iterative closest spectral kernel maps

Motivated by the definition of the spectral kernel distortion of Eq. (4.2), we wish to find the map $\varphi : X \mapsto Y$ that aligns the compatible spectral kernels, $K(x, x')$ and $\tilde{K}(y, y')$. We adopt a similar approach to the post-process iterative refinement algorithm, by constructing corresponding functions over the two shapes. The trivial functions that represent point-to-point correspondence are the delta functions. The key idea is to impose the spectral kernel constraints on these delta functions. Accordingly, if the point $x \in X$ maps to $y = \varphi(x)$ and the point $x' \in X$ maps to $y' = \varphi(x')$, then, the function

$$f_{x,x'} = \left( \frac{K(x, x')}{|K(x, x)|} \right) \delta_x,$$

should correspond to

$$g_{y,y'} = \left( \frac{\tilde{K}(y, y')}{|\tilde{K}(y, y)|} \right) \tilde{\delta}_y.$$

We point out that $x, x'$ are constant parameters that define the function $f_{x,x'}$. As seen in Section 4.2.2, the LB basis representation of the delta function $\delta_x$ at a point $x \in X$ is simply $\Phi(x)$. Therefore, the function $f_{x,x'}$ in the basis $\Phi$, and equivalently the function $g_{y,y'}$ in the basis $\tilde{\Phi}$, can be represented by

$$a_{x,x'} = \left( \frac{K(x, x')}{|K(x, x)|} \right) \Phi(x),$$

$$b_{y,y'} = \left( \frac{\tilde{K}(y, y')}{|\tilde{K}(y, y)|} \right) \tilde{\Phi}(y).$$

In this case, we can construct the corresponding matrices $A$, $B$ by stacking the row vectors $a_{x,x'}$ and $b_{y,y'}$, respectively. Notice that we normalize the kernels, so that $K(x, x') / |K(x, x)| = 1$, $\forall x = x'$.

By recalling that for nearly isometric shapes, the correspondence we are looking for should be represented by a nearly-diagonal $C$ [KBB+13], we can submit an element-wise off-diagonal penalty $W$ and formulate the following problem

$$\arg\min_C \|A - BC\|_F^2 + \beta \|W \odot C\|_F^2,$$

(4.3)

where $\beta$ is a tuning parameter. The symbol $\odot$ represents the matrix element-wise multiplication operation. The matrix $W$ is chosen, such that as $(i, j)$ is located farther from the diagonal of the matrix $W$, the element-wise penalty $W_{i,j}$ increases. The minimization of Eq. (4.3) can be obtained separately for each column of $C$ by the least squares method, with Tikhonov regularization [Tik63, HK70].

The iterative closest spectral kernel maps algorithm is summarized in Algorithm 4.1. For a given initial correspondence $\hat{\varphi}_0(x)$, $\forall x \in X$, the algorithm provides the transformation matrix $C$ computed in Step 3, and the point-wise map $\hat{\varphi}(x)$ found in Step 4, which can be used to approximate the spectral kernel distortion of Eq. (4.2).
Algorithm 4.1: Iterative Closest Spectral Kernel Maps

for $\ell = 1$ to $L$ do

1. Calculate the spectral coefficients

   \[
   a_{x,x'} = \frac{K(x,x')}{|K(x,x)|} \Phi(x),
   \]

   \[
   b_{y,y'} = \frac{\tilde{K}(y,y')}{|\tilde{K}(y,y)|} \tilde{\Phi}(y),
   \]

   for $x, x' \in X, y = \hat{\varphi}_{\ell-1}(x), y' = \hat{\varphi}_{\ell-1}(y')$,

   using the correspondence $\hat{\varphi}_{\ell-1}$ provided by the previous iteration.

2. Compose the constraint matrices $A_\ell, B_\ell$ by stacking the row vectors $a_{x,x'}, b_{y,y'}$ respectively.

3. Find the optimal transformation matrix $C_\ell$ that minimizes

   \[
   \|A_\ell - B_\ell C_\ell\|_F^2 + \beta \|W \odot C_\ell\|_F^2.
   \]

4. For each point $x \in X$, find new map $\hat{\varphi}_\ell(x)$ by searching for the point $y \in Y$, that minimizes the Euclidean distance between the row vectors $\tilde{\Phi}(y)C_\ell$ and $\Phi(x)$, applying

   $$
   \hat{\varphi}_\ell(x) = \arg\min_{y \in Y} \left\| \Phi(x) - \tilde{\Phi}(y)C_\ell \right\|_2.
   $$

end for

As an additional option, one can discard correspondences before estimating the transformation matrix in Step 3. For example, corresponding triangles with flipped orientation are expected to be outliers. In that case, it is advisable to filter out correspondences that belong to such inversely oriented coupled triangles.

The ICSKM algorithm can be viewed as a generalization of the post-process iterative refinement algorithm. This is noticed by setting the kernel $K(x,x')$ to be the heat kernel $K_t(x,x')$. In that case, as $t \to 0$ the normalized kernel $K_t(x,x') / |K_t(x,x)| \to 0$ for $x \neq x'$, and the only constraints that remain are $a_{x,x} \to \Phi(x)$ and $b_{y,y} \to \tilde{\Phi}(y)$.

Implementation

In all our experiments we used the same choice of parameters. In general, we chose our parameters for achieving the most accurate results in a reasonable time. To that end, we used $n = 200$ eigenfunctions of the LB operator.

Our empirical evidence suggests that the GPS kernel [Rus07, QH07], that is,

\[
K(x,x') = \sum_i \frac{1}{\lambda_i} \phi_i(x)\phi_i(x'),
\]

provides superior qualities for correspondence detection, compared to other kernels we tested. The number of iteration has been set to $L = 45$. At each iteration, a subset
Table 4.1: Run-times (in seconds) of the proposed method, evaluated on shapes from the TOSCA dataset.

<table>
<thead>
<tr>
<th># of vertices</th>
<th>4344</th>
<th>19248</th>
<th>25290</th>
<th>45659</th>
<th>52565</th>
</tr>
</thead>
<tbody>
<tr>
<td>n = 50 eigenfunctions</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>runtime</td>
<td>32</td>
<td>39</td>
<td>41</td>
<td>51</td>
<td>56</td>
</tr>
<tr>
<td>n = 100 eigenfunctions</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>runtime</td>
<td>50</td>
<td>59</td>
<td>62</td>
<td>74</td>
<td>80</td>
</tr>
<tr>
<td>n = 200 eigenfunctions</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>runtime</td>
<td>115</td>
<td>135</td>
<td>140</td>
<td>153</td>
<td>165</td>
</tr>
</tbody>
</table>

of 2000 points are matched. The off-diagonal penalty \( W^2_{i,j} = \frac{|\tilde{\lambda}_i - \lambda_j|}{\lambda_j} U_i \) was set to be proportional to the difference of the eigenvalues \( \tilde{\lambda}_i \) and \( \lambda_i \) that correspond to \( \tilde{\phi}_i \) and \( \phi_i \), and scaled by the \( i_{th} \) entry of \( U = \text{diag}(B^T B) \). The tuning parameter \( \beta \) was set to 0.1.

The system was implemented in MATLAB. All the experiments were executed on a 3.00 GHz Intel Core i7 machine with 32GB RAM. Run-times for pairs of shapes of various sizes from the TOSCA dataset are shown in Table 4.1.

### 4.4 Results

We tested the proposed method on pairs of shapes represented by triangulated meshes from both the TOSCA database [BBK08] and the SCAPE database [ASP+05]. The TOSCA dataset contains densely sampled synthetic human and animal surfaces, divided into several classes with given ground-truth point-to-point correspondences between the shapes within each class. The SCAPE dataset contains scans of real human bodies in different poses. We compare our results to several correspondence detection methods.

- **Iterative Closest Spectral Kernel Maps** - the method proposed in this paper. Initial coarse map is found by comparing the first few matched eigenfunctions of the LB operator [SK14]. Another option is to use a small number of landmark points.

- **Functional Maps + Blended (TOSCA only)** - the functional maps based post-process iterative refinement algorithm. We use the results shown in [OBCS+12]. There, the post-process procedure refines the correspondence provided by the Blended method [KLF11].

- **Blended** - the method proposed by Kim et al. that uses a weighted combination of isometric maps [KLF11].

- **Möbius Voting** - the method proposed by Lipman et al. counts votes on the conformal Möbius transformations [LF09].
Permuted Sparse Coding + MSER (SCAPE only) - the approach proposed by Pokrass et al. finds correspondence by using methods from the field of sparse modeling [PBB+13]. We note that this method depends on the ability to detect repeatable regions between shapes. There, maximally stable extremal regions (MSER) are used as a preprocessing step [LBB11].

Fig. 4.1 compares the ICSKM algorithm with existing methods on the TOSCA benchmark, using the evaluation protocol proposed in [KLF11]. The distortion curves describe the percentage of surface points falling within a relative geodesic distance from what is assumed to be their true locations. For each shape, the geodesic distance is normalized by the square root of the shape’s area. It is evident from the benchmark that the proposed method significantly outperforms existing ones.

**TOSCA correspondence**

![Graph showing comparison of ICSKM algorithm with existing methods on TOSCA benchmark](image)

Figure 4.1: Evaluation of the iterative spectral kernel maps algorithm applied to shapes from the TOSCA database, using the protocol of [KLF11].

Fig. 4.2 compares the proposed correspondence algorithm with existing methods on the SCAPE database, again using the evaluation protocol proposed in [KLF11], allowing symmetric flip for a selected number of feature points. Remark: In the evaluation, the correct symmetry is automatically chosen for the shape as a whole.

Table 4.2 displays the percentage of correspondences that fall within different values of relative geodesic distances. It is interesting to focus on large geodesic errors. Unlike other methods, in the proposed approach only one of 200 points has a geodesic error larger than 0.1.

We continue investigating the refinement capabilities of the ICSKM algorithm. For
Figure 4.2: Evaluation of the iterative closest spectral kernel maps algorithm applied
to shapes from the SCAPE database, using the protocol of [KLF11] with allowed
symmetries.

**ICSKM landmark points initialization**

(TOSCA+SCAPE)

Figure 4.3: Evaluation of the iterative closest spectral kernel maps algorithm initialized
by landmark points.
Table 4.2: Percentage of surface points falling within a relative geodesic error for different methods (TOSCA).

<table>
<thead>
<tr>
<th>Method</th>
<th>0.025</th>
<th>0.050</th>
<th>0.100</th>
<th>0.150</th>
</tr>
</thead>
<tbody>
<tr>
<td>Geodesic circle</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ICSKM</td>
<td>82.5</td>
<td>95.9</td>
<td>99.5</td>
<td>99.9</td>
</tr>
<tr>
<td>F. Maps + Blended</td>
<td>69.5</td>
<td>88.7</td>
<td>96.4</td>
<td>98.5</td>
</tr>
<tr>
<td>Blended</td>
<td>55.9</td>
<td>84.7</td>
<td>96.6</td>
<td>98.0</td>
</tr>
<tr>
<td>Möbius Voting</td>
<td>39.3</td>
<td>60.9</td>
<td>79.6</td>
<td>86.2</td>
</tr>
</tbody>
</table>

that goal, we provide the algorithm with 3 to 50 landmark points, that were randomly selected from the ground-truth mapping. Fig. 4.3 compares the dense maps produced by the algorithm with these initial constraints. Observe that with just five landmark points, the algorithm outperforms previous state-of-the-art methods. We have also applied the ICSKM algorithm to non-isometric shapes taken from the TOSCA database. Fig. 4.4, displays the distortion curves for different pair of classes. For each class, we used the manually selected landmark points specified in [KLF11]. Half of these points were used to provide the algorithm with initial correspondence. The rest of the points were used to evaluate the geodesic error. Fig. 4.5 demonstrates how the mapping produced by the ICSKM algorithm initialized with 7 landmark points, transfers the texture from a wolf to a cat and from a dog to a horse.

Finally, we illustrate how the proposed method is able to find the intrinsic reflective symmetry axis of nonrigid shapes. Intrinsic symmetry detection can be viewed as finding correspondence from a shape to itself [RBBK10]. Following this approach, we search for a self-map with flipped orientation. In Fig. 4.6 we visualize the distance between a point and its image for several shapes from the TOSCA database.

4.5 Conclusions

A new method for correspondence detection between nonrigid shapes was introduced. The method is based on the evaluation of the spectral kernel distance, optimized by an ICP based approach in the spectral domain. We have demonstrated the effectiveness of the ICSKM algorithm by achieving state-of-the-art results on shape matching benchmarks. In the future, we intend to apply the ICSKM algorithm for other purposes, such as registration of rigid shapes, matching stereo images, and comparing deformable shapes with texture, and to study the potential and the limitations of the proposed approach for refining correspondences between shapes with topological noise or partially missing data.
Non-isometric shapes correspondence

Figure 4.4: Evaluation of the ICSKM algorithm applied to non-isometric shapes from the TOSCA database. For the primates and animals categories, the algorithm is provided with pairs of 18 and 11 landmark points, respectively. The distortion curves are evaluated by calculating the geodesic error of 18 corresponding points for the primates category, and 10 corresponding points for the animals category.

Figure 4.5: Texture mapping of non-isometric shapes. The textures of wolf and dog shapes were transferred to cat and horse shapes, respectively. The ICSKM algorithm was initialized by 7 landmark points selected using the farthest point strategy [HS85].
Figure 4.6: Symmetry axis of several shapes from the TOSCA database.
Chapter 5

Fast Blended Transformations for Partial Shape Registration

Automatic estimation of skinning transformations is a popular way to deform a single reference shape into a new pose by providing a small number of control parameters. We generalize this approach by efficiently enabling the use of multiple exemplar shapes. Using a small set of representative natural poses, we propose to express an unseen appearance by a low-dimensional linear subspace, specified by a redundant dictionary of weighted vertex positions. Minimizing a nonlinear functional that regulates the example manifold, the suggested approach supports local-rigid deformations of articulated objects, as well as nearly isometric embeddings of smooth shapes. A real-time non-rigid deformation system is demonstrated, and a shape completion and partial registration framework is introduced. These applications can recover a target pose and implicit inverse kinematics from a small number of examples and just a few vertex positions. The result reconstruction is more accurate compared to alternative reduced deformable models.

5.1 Introduction

The construction of an efficient automatic procedure that deforms one shape into another in a natural manner is a fundamental and well-studied challenge in computer graphics. Professional animators design deformable models for manually editing facial expressions, controlling postures and muscles of shapes, and creating sequences of gestures and motions of animated objects. Such models also play a key role in the field of shape analysis. For example, elastic surface registration techniques try to iteratively warp given shapes so as to establish an optimal alignment between them.

A major challenge in automatic shape deformation is preserving the expressiveness of the model while reducing its complexity. This can be accomplished by exploiting the potential redundancy in natural motions. For instance, in non-rigid articulated objects as hands, the bending of a single finger mainly influences the movement of nearby skin.
The stiffness of the limbs restricts them to move freely and therefore the deformation of a shape as a whole can often be well approximated as a blend of a small number of affine transformations. One such skeletal deformation technique, the Linear Blend Skinning (LBS) [MTLT88], has been widely adopted by the gaming and the film industries due to its simplicity and efficiency.

More recently, Jacobson et al. [JBK+12] suggested to deform a single shape by looking for transformations that minimize the nonlinear As-Rigid-As-Possible (ARAP) energy [ACOL00, SA07]. This energy penalizes deviations from rigidity of the underlying structural skeleton. The optimization process alternates between finding the minimal affine transformations and projecting them onto the group of rigid ones. The algorithm converges after a few iterations and provides realistic deformations with a low computational effort. The method was designed for modifying a single reference shape. As such, it does not effectively incorporate the nature of plausible non-rigid deformations that can be well captured by a few examples. Therefore, this method requires a manually tailored pre-computation of biharmonically smooth blending functions, and relies on an initial pose of the shape that is usually selected as the previous frame in the motion sequence.

In many situations, while analyzing or synthesizing shapes, neither manual input nor the temporal state of the shape at the previous frame is available. In these circumstances, obtaining a natural initial pose for the nonlinear optimization procedure becomes a challenge. Nevertheless, in many of these events, static poses of the same shape might be available, such as in [BRLB14] where several human bodies in various postures were captured and reconstructed using range scanners. In this paper, we present an efficient generalization of the LBS model for the case where multiple exemplar shapes are available. To that end, the proposed framework uses the reference shapes to infer an expressive yet low dimensional model, which is computationally efficient and produces natural looking poses. The proposed method constructs a dictionary that contains prototype signal-atoms of weighted vertex coordinates, that effectively span the space of deformations represented by the exemplar shapes. We refer to [Ela10], for applications of overcomplete dictionaries for sparse and redundant data representations in other domains.

The proposed algorithm is mainly motivated by the nonrigid 3D partial registration problem. This problem is considered a key challenge in the field of shape analysis. One of the most efficient approaches to solve this challenge is using deformation-driven correspondences [ZSCO+08]. A good deformation method for this purpose should efficiently produce plausible deformations that fits some known constraints. In our setting we use several example shapes and a few known vertex positions. Although some example-based methods produce excellent deformations, in this context of partial registration, they usually carry three major drawbacks. First, most of these methods have high complexity. Second, they depend strongly on a good initial shape alignment. Third, they require many examples for constructing a model which plausibly captures
various poses. The proposed method tries to overcome these difficulties by using a redundant dictionary that spans a linear deformation subspace. The advantages of using a linear subspace are evident. Acceleration in this case is well established using the ARAP energy functional. Additionally, well known regularization techniques, such as $L_1$ and $L_2$ penalty terms, can easily be deployed in conjunction with the linear model to find a robust sparse representation for the initial shape alignment. Moreover, the simplicity and flexibility of using the linear representation enables the proposed algorithm to refine this initial shape deformation by gradually expanding the deformation space while simultaneously introducing more accurate model constraints.

The key contributions of the proposed approach include the following features.

- Given a few reference shapes, we construct a redundant, yet compact, dictionary of weighted positional-atoms that spans a rich space of deformations. A new deformation is represented as a linear combination of these atom-signals.

- Stable transformations are established by using sparse modeling over a limited subspace of deformations. The suggested framework ensures the use of only a few dictionary atoms relating a few given poses to a target one.

- The As-Rigid-As-Possible energy is reformulated to support multiple reference shapes and automatic global scale detection.

- Smooth deformations are realized by an additional biharmonic energy term that is computationally efficient to minimize when the skinning weights are set to be the eigenfunctions of the Laplace-Beltrami operator.

To demonstrate the fast blended transformations approach, animation sequences were generated given just a few reference shapes and a handful of point constraints that define each target frame. Quantitative evaluation indicates that the advantages of the proposed approach are fully realized when plugged into a shape completion and registration application that achieves low correspondence errors and deformation distortions.

5.2 Related efforts

Example-based deformation techniques attempt to establish a compact representation of shape deformations while trying to satisfy desirable properties. Forming these representations generally requires the processing of sets of poses, expressions, or identities of the same class of shapes. To fulfill this task, various methods have been proposed. Roughly speaking, they all share the following taxonomy.

**Displacement field interpolation.** This technique computes the pointwise difference between each example shape and a reference one at a resting pose, see for example [LCF00, SRIC01, KJP02]. More recent methods include statistical [FKY08] and rotational regressions [WPP07].

63
**Deformation gradient.** These methods interpolate the example poses using the gradient fields of the coordinate functions, and construct the deformed surface by solving a Poisson equation. In [XZWB06, SZGP05] the deformation is estimated for each triangle of the given mesh. Example based deformation gradients and its variants, like the Green strain tensor, are also used for static or dynamic simulation of elastic materials [MTGG11, KTUI12, BML+14, STC+12, ZZT15]. For lowering the computational cost, Der et al. [DSP06] proposed to cluster triangles that are subject to a similar rigid rotation with respect to a single reference shape. It allowed reformulating the problem in terms of transformations of a representative proxy point for each group of vertices.

**Edge lengths and dihedral angles interpolation.** Inspired by discrete shells [GHDS03], local properties were used for mesh interpolation [WDAH10], that naturally fits with the discrete shell energy for combined physics-based and example-driven mesh deformations [FB11].

**Transformation blending.** This approach describes the deformation by a set of affine transformations that are blended together to represent the deformed shape. In this case, the example shapes are used to find the skinning weights as well as the transformations by using non-linear optimization algorithms [JT05, KSO10, LD14, LG15].

**Linear subspace.** Similar in its spirit to the proposed approach is Tycowicz et al. [VTSSH15]. Their method computes an example-based reduced linear model for representing the high dimensional shape space using deformation energy derivatives and Krylov sequences. However, their framework and reduced linear subspace are specifically designed and restricted to the nonlinear shape interpolation problem.

The fast blended transformations method is affiliated with the class of transformation blending inspired by [JBK+12, WJBK15]. The deformation is performed by minimizing a nonlinear energy functional over the linear subspace of skinning transformations. Unlike previous efforts, we suggest to simultaneously blend affine transformations of several given poses of the same subject. The proposed framework allows us to learn the example manifold without estimating the explicit connections between the reference shapes. With these reference shapes, we construct an overcomplete dictionary that spans the space of allowed deformations up to a small tolerance. The nonlinear energy functional guides the transformations to achieve a physically-plausible deformation. Projecting a small set of constraints to the examples manifold, which is assumed to be of low dimensions, we obtain an efficient and accurate blending procedure for real time animation and for the partial shape registration task.
5.3 Notations and problem formulation

5.3.1 Linear blend skinning

Here, we follow the blend skinning model as described by Jacobson et al. in [JBK+12]. Let \( v_1, \ldots, v_n \in \mathbb{R}^d \) \((d = 3)\) be the vertex positions of the input reference mesh \( M \) with \( f \) triangles and \( n \) vertices. Denote the deformed vertex positions of a new target mesh \( \tilde{M} \) by \( \tilde{v}_1, \ldots, \tilde{v}_n \in \mathbb{R}^d \). The target vertex positions relate to the given reference vertices through \( m \) affine transformation matrices \( M_j \in \mathbb{R}^{d \times (d+1)} \), \( j = \{1, \ldots, m\} \) and real-valued skinning weight functions \( w_j \), that measure the influence of each affine transformation on each point of the shape. For a discrete mesh, we denote \( w_j(v_i) \) by \( w_{j,i} \), and readily have

\[
\tilde{v}_i = \sum_{j=1}^m w_{j,i} M_j(v_i) \quad .
\tag{5.1}
\]

Equation (5.1) can be rewritten in a matrix form as

\[
\tilde{V} = D_{\text{LBS}} T_{\text{LBS}},
\]

where \( \tilde{V} \in \mathbb{R}^{n \times d} \) is the matrix whose rows are the positions of the target vertices, and the matrices \( T_{\text{LBS}} \in \mathbb{R}^{(d+1)m \times d} \) and \( D_{\text{LBS}} \in \mathbb{R}^{n \times (d+1)m} \) are created by stacking the skinning parameters in the following fashion

\[
D_{\text{LBS}} = \begin{pmatrix}
w_{1,1} \begin{pmatrix} v^T_1, 1 \end{pmatrix} & \cdots & w_{m,1} \begin{pmatrix} v^T_m, 1 \end{pmatrix} \\
\vdots & \ddots & \vdots \\
w_{1,n} \begin{pmatrix} v^T_n, 1 \end{pmatrix} & \cdots & w_{m,n} \begin{pmatrix} v^T_n, 1 \end{pmatrix}
\end{pmatrix},
\]

\[
T_{\text{LBS}} = \begin{pmatrix} M_1 & \cdots & M_m \end{pmatrix}^T.
\]

5.3.2 Fast automatic skinning transformations

The most general form of representing the position of a new target vertex by a linear transformation of some dictionary (such as the linear blend skinning formulation) can be expressed by

\[
\tilde{V} = D T,
\]

where \( D \in \mathbb{R}^{n \times b} \) is a dictionary of size \( b \) (in case of standard linear blend skinning \( b = (d+1)n \)), and \( T \in \mathbb{R}^{b \times d} \) is a matrix of unknown coefficients that represents the vertex positions in terms of the dictionary.
Jacobson et al. [JBK+12] introduced a method for automatically finding the skinning transformations $T$ by minimizing the ARAP energy [SA07, LZX+08, CPSS10] between the reference shape $M$ and the target one $\tilde{M}$. Let $R_1, R_2, \ldots, R_r \in SO(d)$ and $E_1, E_2, \ldots, E_r$ be $r$ local rotations and their corresponding edge sets, respectively. The ARAP energy, which measures local deviation from rigidity, can be expressed as

$$E(V, \tilde{V}) = \sum_{k=1}^{r} \sum_{(i,j) \in E_k} c_{ijk} \| (\tilde{v}_i - \tilde{v}_j) - R_k(v_i - v_j) \|^2,$$

where $c_{ijk} \in \mathbb{R}$ are the cotangent weighting coefficients [PP93]. As indicated in [JBK+12], it is unnecessary to estimate the local rotation for each edge separately since vertices undergoing similar deformations can be clustered together into a small number of rotation clusters.

The ARAP energy can be expressed in a simple matrix form. Denote $A_k \in \mathbb{R}^{n \times |E_k|}$ as the directed incidence matrix corresponding to edges $E_k$, and let $C_k \in \mathbb{R}^{|E_k| \times |E_k|}$ be a diagonal matrix with weights $c_{ijk}$. Then, the ARAP energy can be written in matrix form as

$$2E(V, \tilde{V}) = \text{tr}(\tilde{V}^T L \tilde{V}) - 2 \text{tr}(RK\tilde{V}) + \text{tr}(V^T LV),$$

where $R = (R_1, \ldots, R_r)$, $K \in \mathbb{R}^{dr \times n}$ stacks the matrices $V^T A_k C_k A_k^T$, and $L \in \mathbb{R}^{n \times n}$ is the cotangent-weights Laplacian up to a constant scale factor. Plugging in the linear blend skinning formula $\tilde{V} = DT$ we obtain

$$2E(V, \tilde{V}) = \text{tr}(T^T \tilde{L} T) - 2 \text{tr}(R\tilde{K} T) + \text{tr}(V^T LV), \quad (5.2)$$

where $\tilde{L} = D^T LD$ and $\tilde{K} = KD$. For more details about the above derivation, we refer the reader to [JBK+12].

### 5.4 Example-based blended transformations

**Overview.** We now extend the framework described in the previous section for the case where multiple poses of the same shape are available. We begin by expressing the deformed shape as a combination of atoms from a dictionary that is constructed from the linear blend skinning matrices of the given examples. Then, we provide the details of various energy terms to be minimized with respect to the unknown transformations $T$ using the proposed model. Next, we describe the nonlinear optimization process and its initialization, and conclude by discussing optional extensions that can be incorporated into the algorithm.
5.4.1 Dictionary Construction

Suppose we are given \( q \) reference meshes \( M_1, M_2, \ldots, M_q \). Let \( v_1^\ell, \ldots, v_n^\ell \in \mathbb{R}^d \) be the positions of vertices belonging to the reference mesh \( M_\ell, \ell = 1, \ldots, q \), and let \( V_1, V_2, \ldots, V_q \in \mathbb{R}^{n \times d} \) be the matrices whose rows denote the positions of the corresponding vertices. We are also given some \( h \) linear constraints represented by the matrix \( H \in \mathbb{R}^{h \times n} \), such that \( H\tilde{V} \approx Y \), where \( Y \in \mathbb{R}^{h \times d} \) is the value of these constraints for the target shape. We can define the linear constraints to be simply the coordinates of points on the mesh or use more refined measures such as the Laplacian coordinates \([\text{Ale03, LSCO}]+\), or a weighted average of some vertex positions, to constrain our non-rigid blended shape deformation. Using this setup, we are interested in finding the positions of the target vertices as a result of a smooth transformation of the input meshes such that it approximately preserves local rigidity and satisfies the linear constraints up to a small error.

![Figure 5.1: Deformation using the example-based LBO dictionary. The left portion of the figure shows the cat and centaur ground-truth target shapes (colored in gray). On the right we show the near perfect representation of these target shapes by a linear combination of the dictionary’s atoms. In this case, the weighting functions are the eigenfunctions of the Laplace-Beltrami operator that correspond to the lowest 15 eigenvalues. The exemplar shapes that are used to extract the example-based LBO dictionaries for representing the shapes are shown inside the box.](image)

**Example-based dictionary.** Given \( m \) real-valued weight functions \( w_j, j = 1, \ldots, m \), we propose the example-based representation of the positions of the target vertices to be a combination of the linear blend skinning deformations of each given reference mesh

\[
\tilde{v}_i = \sum_{\ell=1}^{q} \tilde{v}_i^\ell,
\]

where

\[
\tilde{v}_i^\ell = \sum_{j=1}^{m} w_{j,i} M_j^\ell \begin{pmatrix} v_i^\ell \\ 1 \end{pmatrix}. \tag{5.3}
\]
We can explicitly write the new vertex positions as

\[
\tilde{v}_i = \sum_{\ell=1}^q \sum_{j=1}^m w_{j,i} \tilde{M}_\ell^j \left( v_{\ell,i}^1 \right) = \sum_{\ell=1}^q \sum_{j=1}^m w_{j,i} \tilde{M}_\ell^j v_{\ell,i} + \sum_{j=1}^m \sum_{\ell=1}^q w_{j,i} \tilde{M}_\ell^j ,
\]

where \(\tilde{M}_\ell^j \in \mathbb{R}^{d \times d}\) and \(\tilde{M}_\ell^j \in \mathbb{R}^{d \times 1}\) are sub-matrices of \(M_j^\ell\), such that \(M_j^\ell = \left( \tilde{M}_\ell^j, \bar{M}_\ell^j \right)\).

This formula can be equivalently expressed in the standard matrix form by

\[
\tilde{V} = DT,
\]

where \(D \in \mathbb{R}^{n \times (1+qd)m}\) is the proposed dictionary of size \(b = (1 + qd)m\), that multiplies the examples’ vertex positions \(v_{\ell,i}^1\) with the vertex weights \(w_{j,i}(v_i)\), and \(T \in \mathbb{R}^{(1+qd)m \times d}\) stacks the matrices \(\tilde{M}_j^\ell\) and \(\bar{M}_j^\ell\) in the following way

\[
D = \begin{pmatrix} \tilde{D} & \tilde{D}_1 & \ldots & \tilde{D}_q \end{pmatrix},
\]

\[
T = \begin{pmatrix} \tilde{T}^T & \tilde{T}_1^T & \ldots & \tilde{T}_q^T \end{pmatrix}^T,
\]

where

\[
\tilde{D}_\ell = \begin{pmatrix} w_{1,1} v_{1}^T & \ldots & w_{m,1} v_{1}^T \\ \vdots & \ddots & \vdots \\ \vdots & \ddots & \vdots \\ w_{1,n} v_{n}^T & \ldots & w_{m,n} v_{n}^T \end{pmatrix},
\]

\[
\tilde{T}_\ell = \begin{pmatrix} \tilde{M}_1^\ell & \ldots & \tilde{M}_m^\ell \end{pmatrix}^T,
\]

and

\[
\tilde{D} = \begin{pmatrix} w_{1,1} & \ldots & w_{m,1} \\ \vdots & \ddots & \vdots \\ \vdots & \ddots & \vdots \\ w_{1,n} & \ldots & w_{m,n} \end{pmatrix},
\]

\[
T = \begin{pmatrix} \sum_{\ell=1}^q \tilde{M}_1^\ell & \ldots & \sum_{\ell=1}^q \tilde{M}_m^\ell \end{pmatrix}^T.
\]

**Weighting functions.** There are many ways to choose weighting functions. One is to consider the weights of bones like in the standard linear blend skinning model. In that case, we name the constructed dictionary as the *example-based skeleton dictionary*. When there is no significant underlying skeletal structure, we suggest to use the first \(m\) eigenfunctions of the Laplace-Beltrami operator (LBO) \([\text{DRW}12, \text{BWP}13]\). This choice of a dictionary is useful, for example, when handling facial expressions, for analyzing internal organs in volumetric medical imaging applications, or for deforming non-rigid
objects such as an octopus. The eigendecomposition of the LBO consists of non-negative eigenvalues \( 0 = \lambda_0 < \lambda_1 < \cdots < \lambda_i < \ldots \), with corresponding eigenfunctions \( \Phi \equiv \{ \phi_0, \phi_1, \ldots, \phi_i, \ldots \} \), that can be considered as an orthonormal basis. We refer to this dictionary as the *example-based LBO dictionary*.

### 5.4.2 Nonlinear Energy Terms

**Linear constraints.** The energy of the \( h \) linear constraints can be calculated by

\[
2E_{lc}(\tilde{V}) = \| H\tilde{V} - Y \|^2_2 = \| HDT - Y \|^2_2 = \| XT - Y \|^2_2
\]

\[
= \text{tr} (T^T X^T XT) - 2 \text{tr} (Y^T XT) + \text{tr} (Y^T Y),
\]

(5.5)

where \( X = HD \).

**Smoothness energy.** Let \( v_{i,k} \), \( k \in \{1, \ldots, d\} \) be the \( k \)th coordinate of the vertex position \( v_i \). Notice from Equation (5.4), that the amount of influence of \( v_{i,k} \) on \( \tilde{v}_{i,k} \) is some linear combination of \( w_j(v_i), j = 1, \ldots, m \). Following the same reasoning as in [JBPS11], we search for a smooth variation of this influence, for example, one that minimizes the Laplacian energy \( \frac{1}{2} \int_M \Delta(\cdot)^2 \mathrm{da} \) of this linear combination, where \( da \) is an area element on the surface \( M \) of our shape. For the special case where the weights are the LBO eigenfunctions, the sum of all smoothness energy terms can be expressed as

\[
E_{sm} = \frac{1}{2} \text{tr}(T^T \Lambda T),
\]

(5.6)

where \( \Lambda \) is a diagonal matrix. The values of the diagonal are the squares of the eigenvalues of the respective eigenfunctions. Thus, in this case, the smoothness energy amounts to a simple quadratic regularization term. Note, that when the weighting functions are chosen in a different way, the smoothness energy expression is a bit more involved.

**Scaling.** In some applications, there is a scale difference between the example shapes and the linear constraints. To compensate for such a discrepancy, we introduce a scaling factor \( \alpha \) into the ARAP energy. It reflects the ratio between the reference shape and the deformed one, in the following manner,

\[
E_{sc}(V, \tilde{V}) = \frac{1}{2} \sum_{k=1}^r \sum_{(i,j) \in E_k} c_{ijk} \| (\tilde{v}_i - \tilde{v}_j) - \alpha R_k(v_i - v_j) \|^2.
\]

(5.7)

Hence, the ARAP energy with the global scale factor reads

\[
2E_{sc}(V, \tilde{V}) = \text{tr}(T^T \tilde{L} T) - 2 \alpha \text{tr}(R \tilde{K} T) + \alpha^2 \text{tr}(V^T L V).
\]

(5.8)
**Average ARAP energy.** One way to define an example-based energy functional is by taking the average between all as-rigid-as-possible energies, namely,

$$E_{av} = \frac{1}{q} \sum_{\ell=1}^{q} E_{sc}(V_\ell, \tilde{V}),$$  \hspace{1cm} (5.9)

with the additional linear constraints and the smoothness energies,

$$E_{total}(\tilde{V}) = E_{av}(\tilde{V}) + \beta_{lc}E_{lc}(\tilde{V}) + \beta_{sm}E_{sm}(\tilde{V}),$$  \hspace{1cm} (5.10)

where $\beta_{lc}$, $\beta_{sm}$, are some tuning parameters that control the importance of the linear constraints and the smoothness term. We can simplify this expression, plugging in Equations (5.6), (5.5) and (5.8)

$$2E_{total}(\tilde{V}) = tr(T^T \tilde{L} T) - \frac{1}{q} \sum_{\ell=1}^{q} (2\alpha tr(R_\ell \tilde{K}_\ell T) + \alpha^2 tr(V_\ell^T LV_\ell)) + \beta_{lc} tr(T^T X^T XT) - 2\beta_{lc} tr(Y^T XT) + \beta_{lc} tr(Y^T Y) + \beta_{sm} tr(T^T \Delta T).$$  \hspace{1cm} (5.11)

**Minimal ARAP energy.** Another way to define an example-based energy functional is to find the minimal ARAP energy between the deformed mesh and each of the input meshes separately,

$$E_{mn}(\tilde{V}) = \min_{\ell} E_{\ell}(\tilde{V}, T_\ell),$$  \hspace{1cm} (5.12)

where

$$E_{\ell}(\tilde{V}, T_\ell) = E_{sc}(V_\ell, \tilde{V}) + \beta_{lc}E_{lc}(\tilde{V}) + \beta_{sm}E_{sm}(\tilde{V}).$$  \hspace{1cm} (5.13)

This can be expressed as

$$2E_{\ell}(\tilde{V}, T_\ell) = tr(T_\ell^T \tilde{L} T_\ell) - 2\alpha tr(R_\ell \tilde{K}_\ell T_\ell) + \alpha^2 tr(V_\ell^T LV_\ell) + \beta_{lc} tr(T_\ell^T X^T XT) - 2\beta_{lc} tr(Y^T XT) + \beta_{lc} tr(Y^T Y) + \beta_{sm} tr(T_\ell^T \Delta T_\ell).$$  \hspace{1cm} (5.14)

### 5.4.3 Optimization

To minimize the energy $E_{total}(\tilde{V})$ and find the local rotations $R_\ell$, $\ell = 1, \ldots, q$, the global scale factor $\alpha$ and the transformations $T$, we follow the local-global approach of
Figure 5.2: Automatic feature point correspondence and shape interpolation. The four examples of a horse (top middle) and the two sets of vertex positions (top left, top right) were used to generate a sequence of frames. Correspondence of the points on the four legs (circled in blue) was detected by minimizing the example-based deformation energy for all permissible correspondences. The example-based deformations (bottom left and right) were then interpolated at four times the original frame rate to produce the movie sequence (bottom).
[SA07] with an additional step to find the global scale $\alpha$. First we fix $T$ and $\alpha$ and solve for $R_\ell$ (local step). Then, we find $\alpha$ by fixing $T$, $R_\ell$ (scale step). Finally, we fix $R_\ell$ and $\alpha$, and solve for $T$ (global step).

**Local step.** For fixed $\alpha$ and $T$, maximizing $\text{tr}(R_\ell S_\ell)$, $\ell = 1,\ldots,q$, where $S_\ell = \tilde{K}_\ell T_\ell$ is constant, amounts to maximizing $\text{tr}(R_\ell S_\ell)$, $k = 1,\ldots,r$, which is obtained by taking $R_\ell = \Psi_\ell^T \Phi_\ell^T$, where

$$S_\ell = \Phi_\ell \Sigma_\ell \Psi_\ell^T$$

is given by the singular value decomposition of $S_\ell$.

**Scale step.** For fixed $T$ and $R_\ell$, $\ell = 1,\ldots,q$, we can differentiate by $\alpha$

$$\frac{\partial E_{\text{total}}}{\partial \alpha} = -\frac{1}{q} \sum_{\ell=1}^{q} (\text{tr}(R_\ell \tilde{K}_\ell T_\ell)) + \alpha \text{tr}(V_\ell^T L V_\ell).$$

Setting the derivative to zero, we get

$$\alpha = \frac{1}{q} \sum_{\ell=1}^{q} (\text{tr}(R_\ell \tilde{K}_\ell T_\ell)) / \text{tr}(V_\ell^T L V_\ell).$$

**Global step.** For fixed $\alpha$ and $R_\ell$, $\ell = 1,\ldots,q$, we differentiate $E_{\text{total}}$

$$\frac{\partial E_{\text{total}}}{\partial T} = \frac{1}{q} \sum_{\ell=1}^{q} (\tilde{L} T - \alpha \tilde{K}_\ell^T R_\ell^T)$$

$$+ \beta_{lc} (X^T X T - X^T Y) + \beta_{sm} \Lambda T$$

$$=(\tilde{L} + \beta_{lc} X^T X + \beta_{sm} \Lambda) T - \beta_{lc} X^T Y - \frac{\alpha}{q} \sum_{\ell=1}^{q} \tilde{K}_\ell^T R_\ell^T.$$ 

Setting these derivatives to zero, we obtain

$$(\tilde{L} + \beta_{lc} X^T X + \beta_{sm} \Lambda) T = \beta_{lc} X^T Y + \frac{\alpha}{q} \sum_{\ell=1}^{q} \tilde{K}_\ell^T R_\ell^T.$$  

Let us define $\Gamma = (\tilde{L} + \beta_{lc} X^T X + \beta_{sm} \Lambda)$. Then, we can solve for $T$ by precomputing the Cholesky factorization of $\Gamma$

$$T = \Gamma^{-1} (\beta_{lc} X^T Y + \frac{\alpha}{q} \sum_{\ell=1}^{q} \tilde{K}_\ell^T R_\ell^T).$$

As for optimizing the minimal ARAP energy $E_{\text{mn}}(\tilde{V})$, in the local step we find each set of rotations $R_\ell$ by maximizing $\text{tr}(R_\ell S_\ell)$, where $S_\ell = \tilde{K}_\ell T_\ell$. We then find the global
scale factor relative to each reference shape

\[ \alpha_\ell = \text{tr}(R_\ell \tilde{K}_\ell T_\ell) / \text{tr}(V_\ell^T L V_\ell). \]

In the global step we calculate the respective blended transformations \( T_\ell \), by

\[ T_\ell = \Gamma^{-1} (\beta_{lc} X^T Y + \alpha_\ell \tilde{K}_\ell^T R_\ell^T). \]

Then, we calculate the minimal energy \( E_\ell(\tilde{V}, T_\ell), \ell = 1, \ldots, q \) of Equation (5.13).

**Initial transformations.** In the first global step, there are no rotation matrices that can be used. Hence, the energy that we need to minimize is

\[ 2E_{\text{init}}(\tilde{V}) = \beta_{lc} \| X^T Y \|^2 + \beta_{sm} T^T \Lambda T \]
\[ = \beta_{lc} \text{tr}(T^T X^T X T) - 2\beta_{lc} \text{tr}(Y^T X T) \]
\[ + \beta_{lc} \text{tr}(Y^T Y) + \beta_{sm} \text{tr}(T^T \Lambda T). \]  

(5.20)

We readily have,

\[ \frac{\partial E_{\text{init}}}{\partial T} = \beta_{lc} (X^T X T - X^T Y) + \beta_{sm} \Lambda T \]
\[ = (\beta_{lc} X^T X + \beta_{sm} \Lambda) T - \beta_{lc} X^T Y. \]  

(5.21)

Setting these derivatives to zero, we obtain

\[ T = (\beta_{lc} X^T X + \beta_{sm} \Lambda)^{-1} (\beta_{lc} X^T Y). \]  

(5.22)

**Sparse initial transformations.** A more robust initial transformation can be achieved by adding an \( L_1 \) penalty to the energy given in Equation (5.20)

\[ E_{\text{sp}}(\tilde{V}) = E_{\text{init}}(\tilde{V}) + \beta_{sp} \| T \|_{L_1}. \]  

(5.23)

The effect of this additional penalty is that it makes the initial transformations sparse, which results in a deformation with less artifacts. The parameter \( \beta_{sp} \) controls the amount of sparsity in the initial solution of \( T \). Equation (5.23) can be solved efficiently using the elastic net regression method [ZH05].

**5.4.4 Extensions**

**Updating constraints.** It may happen that some of the \( h \) linear constraints are unavailable due to noise or occlusions. This can be easily solved by deleting the appropriate rows of \( X \) and \( Y \) and efficiently updating the Cholesky factorization.

**Dictionary reduction.** When the input meshes are similar to each other, the proposed
example-based dictionary becomes redundant. The dictionary can be reduced consider-
ably by clustering similar dictionary atoms. For this purpose, we use the k-medoids
clustering algorithm [KR87]. The advantage of k-medoids over k-means clustering is
that each cluster center of the k-medoids procedure is represented by one of the original
dictionary atoms. This makes the appearance of the deformed shape more plausible
compared to using k-means clustering for dimensionality reduction.

**Change of dictionaries.** It is sometimes useful to work with two different dictionaries.
In that case, the representations of the mesh in these two subspaces can be converted
from one to the other in a simple way. Suppose we are given the dictionaries $D_1$, $D_2$
and a good approximation of the transformation $T_1$. Then, the transformation $T_2$
can be set to

$$T_2 = (D_2^T D_2)^{-1} D_2^T T_1. \quad (5.24)$$

This is particularly useful when one wants to initialize the transformations using a
low dimensional dictionary by applying Equation (5.23), and then change to a richer
dictionary for obtaining more refined transformations.

**Figure 5.3:** The four dog shapes are used as examples for our method (left). The deformed
shape (right) is found from the vertex positions (middle). In this case the deformed shape is
50% larger than the reference ones.

5.5 **Experimental results**

**Implementation Considerations.** In our implementation we use $m \leq 15$ eigenfunc-
tions as the weighting functions for the example-based LBO dictionary. To support
natural articulated shapes deformation, we construct the example-based skeleton dictio-
Figure 5.4: Example-based deformed shapes from few vertex positions of a hand shape.

Table 5.1: Model parameters and performance (in milliseconds).

<table>
<thead>
<tr>
<th>class</th>
<th>Input mesh</th>
<th>Model</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>n f r b q 1-iter full</td>
</tr>
<tr>
<td>woman</td>
<td>45659</td>
<td>91208</td>
<td>30 178 5 1.3 16.7</td>
</tr>
<tr>
<td>centaur</td>
<td>15768</td>
<td>31532</td>
<td>31 166 5 1.2 16.3</td>
</tr>
<tr>
<td>wolf</td>
<td>4344</td>
<td>8684</td>
<td>14 82 2 0.7 10.3</td>
</tr>
<tr>
<td>dog</td>
<td>25290</td>
<td>50528</td>
<td>25 136 4 1.0 13.9</td>
</tr>
<tr>
<td>man</td>
<td>52565</td>
<td>105028</td>
<td>31 235 9 1.9 23.2</td>
</tr>
<tr>
<td>cat</td>
<td>27894</td>
<td>55712</td>
<td>22 148 6 1.2 15.3</td>
</tr>
<tr>
<td>hand</td>
<td>2224</td>
<td>4424</td>
<td>18 163 7 1.2 15.5</td>
</tr>
<tr>
<td>horse</td>
<td>16843</td>
<td>8431</td>
<td>26 139 4 1.1 14.7</td>
</tr>
</tbody>
</table>
Table 5.2: Example-based dictionary. Maximal relative distortion (in percent) of the deformed shapes.

<table>
<thead>
<tr>
<th>q</th>
<th>size</th>
<th>example-based LBO dict.</th>
<th>example-based skeleton dict.</th>
<th>one example</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>94</td>
<td>2.18</td>
<td>3.59</td>
<td>2.39</td>
</tr>
<tr>
<td>2</td>
<td>139</td>
<td>1.97</td>
<td>2.50</td>
<td>2.10</td>
</tr>
<tr>
<td>3</td>
<td>184</td>
<td>1.82</td>
<td>2.16</td>
<td>1.95</td>
</tr>
<tr>
<td>4</td>
<td>218</td>
<td>1.72</td>
<td>1.89</td>
<td>1.88</td>
</tr>
<tr>
<td>5</td>
<td>258</td>
<td>1.63</td>
<td>1.78</td>
<td>1.83</td>
</tr>
<tr>
<td>6</td>
<td>293</td>
<td>1.56</td>
<td>1.68</td>
<td>1.77</td>
</tr>
<tr>
<td>7</td>
<td>339</td>
<td>1.52</td>
<td>1.60</td>
<td>1.71</td>
</tr>
<tr>
<td>8</td>
<td>382</td>
<td>1.51</td>
<td>1.50</td>
<td>1.65</td>
</tr>
</tbody>
</table>

Figure 5.5: Nonrigid ICP. Left to right: The three exemplar shapes, the uncorrupted and complete target shape, the acquired partial shape with eight known feature points (marked as red dots), the initial deformation using the feature points and the final deformation after applying the nonrigid ICP.
nary. Its weights are generated using an automatic example-based skinning software package [LD14]. These skeleton weights are also used to define the rotation clusters. After constructing the dictionary from our mesh examples, we decrease the size of the dictionary using the k-medoids clustering algorithm. This step typically reduces the size of the dictionary in half.

The transformations are found in several steps. We begin by estimating the sparse initial transformations using Equation (5.23). Typically, we start with \( m \leq 4 \) eigenfunctions as the weighting functions. Then, we apply a two stage optimization procedure. In the first step we minimize the average ARAP energy of Equation (5.11). This energy, although robust, tends to smooth out some of the details of the shape. Therefore, in the second step we optimize the minimal ARAP energy of Equation (5.12), that effectively selects one example pose which seems to be closest to the target pose. After a few iterations, we apply Equation (5.24), and change to a richer dictionary that can reflect finer details of the shape. We construct this richer dictionary according to the properties of the subject we want to deform. For articulated shapes, we use the example-based skeleton dictionary. For non-articulated objects, we increase the number of eigenfunctions used to construct the example-based LBO dictionary.

The algorithm was implemented in MATLAB with some optimizations in C++. We use the SVD routines provided by McAdams et al. [MST+11]. All the experiments were executed on a 3.00 GHz Intel Core i7 machine with 32GB RAM. In Table 5.1 we give the settings for different mesh classes [BBK08, SP04] and typical performance of the algorithm. For these settings the algorithm takes between 10 and 25 milliseconds.

**Example-based dictionary.** The example-based dictionary spans natural deformations of a given shape with a small error. In Figure 5.1 we show some examples of deformations created using the example-based LBO dictionary with 15 eigenfunctions. The mesh parameters and number of example shapes used is as in Table 6.1. Observe, that there are no noticeable artifacts in these deformations.

We note that the experiments indicate that the accuracy of the proposed model increases with the number of example shapes. For each shape in the database [ASP+05], we found the closest deformed shape in the \( L_2 \) sense. We calculated the maximal Euclidean distortion between the deformed and the original shape and normalized it by the square root of the original shape's area. Then, we average this maximal distortion for all shapes. We notice, that the maximal distortion decreases as the number of example shapes grows. We also compare the example-based LBO and the skeleton dictionaries. Although quantitatively, the example-based LBO dictionary seems to perform better, our experience suggests that for shapes that have a well-defined skeleton, the example-based skeleton dictionary is more pleasing to the eye, as it captures the stiffness of the bones. Another conclusion is that using many examples improves the deformation accuracy. This can be seen by calculating the distortion of the deformed shape when the example-based LBO dictionary is constructed using one shape only,
while keeping the number of dictionary atoms the same and without applying the
dictionary reduction step. Table 5.2 summarizes the results.

**Example-based deformation from few vertex positions.** Perhaps, the most
powerful application of our example-based framework is finding a naturally deformed
shape from just a couple of vertex positions. In this scenario, we are given the positions
of just a few points of a single depth image of a target shape. Given prior example
shapes in different postures, we are able to faithfully and reliably reconstruct the target
shape. In Figures 5.3 and 5.4, we show reconstructed dog and hand, shapes from a
small number of feature points. In these examples, the feature points were sampled in a
scale different than that of the example-shapes by a factor of 1.5 (dog), and 0.7 (hand).

**Automatic feature point correspondence.** The example-based deformation energy
can be used to find correspondence between the example shapes and the given feature
points [ZSCO+08]. Because our method does not rely on good initialization nor on many
input points, it is ideal for such a purpose. For example, in Figure 5.2, we are given four
reference shapes and eight feature points. In this demonstration, the correspondences
of the four feature points that belong to each leg (circled in blue) are difficult to find.
We can resolve this ambiguity by running our optimization algorithm for all 24 options
of permissible correspondences, and calculate the example based energy of Equation
(5.12) for each. Then, the correspondence can be found by choosing the option that
gave the minimal deformation energy.

**Shape interpolation.** A nice application that can easily be performed, is to inter-
polate between two deformed shapes. In our setting, we are given two instances of
positional constraints. From these constraints we find two deformed shapes and their
rotations. Then, we are able to interpolate between these rotations. To produce the
new transformations, we apply one additional global-step. Figure 5.2 demonstrates an
interpolation between two deformed shapes of a galloping horse. Four example meshes
are used as an input. In the supplementary material we add a video of a galloping
horse reconstructed from few feature points. The video frames are interpolated by a
factor of eight. Based on the proposed ideas, we developed a computer program that
automatically finds a natural deformed shape from a user’s specified vertex locations
and interpolates between the start pose and the final deformation of the shape, creating
a smooth and intuitive motion of the shape. We provide a video that shows how this
software is used to make an animation sequence of a moving person.

**Nonrigid ICP.** The blended transformations can be plugged into a simple nonrigid
ICP framework [ACP03, ARV07, LSP08]. Nonrigid ICP registration alternates between
finding pointwise correspondences and deforming one shape to best fit the other. Hence,
we propose the following strategy. To find correspondences compare the vertex positions
of all points and their surface normal vectors. In each iteration, we set new linear
constraints according to the vertex positions of the obtained point-to-point correspon-
dences, and apply our blended transformations method to wrap the nonrigid shapes
Figure 5.6: Evaluation of the shape completion and registration procedure applied to shapes from the TOSCA database.
while keeping the deformed shape inside the example manifold. We note that because the representation space is defined by the blended transformations it is suffice to match only a subset of points on the two shapes.

**Shape completion and registration.** In many depth data acquisition scenarios, the acquired data consists of an incomplete, occluded and disconnected parts of a shape. Given some known feature points in those parts of the shape, we want to find the deformation that best fits the partial data and detect the pointwise mapping between acquired partial shape and the reference shapes. To this end, we propose a two step procedure. In the first step, the feature points are used to find an initial deformation. In the second step, the deformation is refined by applying a nonrigid ICP procedure. Since our deformation technique is able to find a good approximation from just a few vertex positions it is ideal to be plugged into this procedure. Figure 5.5 shows an example of partial data of a cat shape (middle) with some known feature points (marked in red). The initial deformation was found by applying the proposed Fast Blended Transformation algorithm using the known feature points (second from left). The final deformation was attained by applying the nonrigid ICP algorithm in conjunction with our blended transformations approach (left). Notice that the nonrigid ICP algorithm corrected the tilt of the cat’s head.

We tested the proposed shape completion and registration procedure on shapes represented by triangulated meshes from the TOSCA database [BBK08]. We performed 50 random experiments with different example and target shapes. For each experiment, we were given eight reference shapes and one target shape for which some of its vertices were removed. We assume that the remaining shape includes some predefined parts that amount to more than 50% of the shape’s area. We further assume that in these parts there are a number of identifiable feature points and that around each feature point, within a certain geodesic circle, no vertices were removed. In the test we performed the number of feature points was set to eight and the radius of the geodesic circle about each point was 15% of the square-root of the shape’s area.

We studied the performance of our approach with different number of example shapes. In our implementation, we set the initial linear constraints to be the weighted average of the vertex positions in 10 different geodesic circles around each feature point. The weights of each vertex were proportional to its voronoi area. Using these linear constraints, we found an initial guess of the deformation. Then, we employed the nonrigid ICP algorithm for the rest of the mesh. We also compared our results with the ones obtained by plugging in the deformation method proposed by Der et al. [DSP06] into our shape completion procedure, using the same skeleton structure. This method applies an example-based deformation gradient model on the problem, and is computationally comparable to the proposed fast blended transformations algorithm. To achieve better results, we used a modified version of the deformation gradient model that supports soft constraints. For leveling the playing field, the automatic skeleton
structure was found in the same way for all methods [LD14].

Figure 5.6 (left) compares the accuracy of the achieved deformations. The distortion curves describe the percentage of surface points falling within a relative distance from the target mesh. For each shape, the Euclidean distance is normalized by the square root of the shape’s area. As for the partial registration, the distortion curves shown in Figure 5.6 (right) describe the percentage of correspondences that fall within a relative Euclidean distance from what is assumed to be their true locations, similar to the protocol of [KLF11]. We see, that both the deformation quality and the correspondence accuracy increase with the number of reference shapes. This is expected, since as more example poses are introduced, the example-based dictionary better spans the space of natural deformations and we have more poses to compare against. We also notice that for these experiments, our deformation approach (even with one reference shape as in [JBK+12]) significantly outperforms the inverse kinematics method of Der et al. [DSP06]. This can be explained by the fact that the reduced deformable model of Der et al. is based on explicit interpolation between the reference poses using deformation gradients. Apparently, this model needs a large number of reference poses to cover all the allowed isometric transformations. In contrast, our model implicitly finds the example manifold by a linear combination of the dictionary atoms and the ARAP energy. Hence, it needs far fewer examples.

5.6 Discussion

We tested some deficient versions of our example-based deformation framework. In Figure 5.7 we show several examples of how these partial versions of the algorithm behave. For comparison to the complete method see Figures 5.3 and 5.4. We notice that the most important part of the proposed framework is the construction of the dictionary from multiple examples. If only one example is used, as in [JBK+12], the deformation algorithm fails when the shape has many degrees of freedom.

Although the method is robust and usually performs very well, some limitations and failures in particular cases do exist. Despite the usually pleasing to the eye deformations of the proposed example-based approach, sometimes undesirable artifacts might occur. This is the result of the collinearity between different dictionary atoms. As for the performance of the algorithm, the deformation can be produced in real time but the algorithm cannot accommodate for video applications with many objects that need to be simultaneously deformed. This problem can be solved by using the proposed algorithm only for objects for which a previous pose cannot be used for the initialization of the current one. Another drawback is that if the example shapes do not incorporate enough information for extracting the right rotation clusters, then, the algorithm will ultimately fail. Also, the current evaluation system does not prevent self-intersections.
Figure 5.7: Deformed shapes constructed by omitting some of the steps in the proposed example-based framework.

1. Restricting the example-based dictionary to use one example.
2. Comparing the deformed shape to only one reference shape.
3. Skipping the scale-step, by setting $\alpha = 1$. 
Table 5.3: List of Mathematical Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{M}$</td>
<td>reference and deformed meshes</td>
</tr>
<tr>
<td>$d$</td>
<td>dimension of the shape</td>
</tr>
<tr>
<td>$n$</td>
<td>number of vertices</td>
</tr>
<tr>
<td>$f$</td>
<td>number of faces</td>
</tr>
<tr>
<td>$m$</td>
<td>number of blending weight functions</td>
</tr>
<tr>
<td>$q$</td>
<td>number of example shapes</td>
</tr>
<tr>
<td>$r$</td>
<td>number of rotation clusters</td>
</tr>
<tr>
<td>$v^f_i$</td>
<td>$i$th vertex of the $\ell$th shape</td>
</tr>
<tr>
<td>$w_{j,i}$</td>
<td>$j$th blending weight function at the $i$th vertex</td>
</tr>
<tr>
<td>$\phi_j$</td>
<td>$j$th eigenfunction of the Laplace-Beltrami operator</td>
</tr>
<tr>
<td>$\lambda_j$</td>
<td>$j$th eigenvalue of the Laplace-Beltrami operator</td>
</tr>
<tr>
<td>$M^f_{\ell}$</td>
<td>$j$th transformation of the $\ell$th shape</td>
</tr>
<tr>
<td>$\mathbf{V}, \tilde{\mathbf{V}}$</td>
<td>set of reference and deformed vertices</td>
</tr>
<tr>
<td>$D$</td>
<td>example-based dictionary</td>
</tr>
<tr>
<td>$T$</td>
<td>transformation matrix</td>
</tr>
<tr>
<td>$R^f_k$</td>
<td>$k$th rotation of the $\ell$th shape</td>
</tr>
<tr>
<td>$\mathcal{E}_k$</td>
<td>set of vertices of the $k$th rotation cluster</td>
</tr>
<tr>
<td>$c_{ijk}$</td>
<td>cotangent weight of the edge $(i,j)$ in the $k$th rotation cluster</td>
</tr>
<tr>
<td>$\mathbf{H}$</td>
<td>constraint sampling matrix</td>
</tr>
<tr>
<td>$\mathbf{Y}$</td>
<td>constraint matrix</td>
</tr>
<tr>
<td>$\Lambda$</td>
<td>diagonal eigenvalue matrix</td>
</tr>
<tr>
<td>$E_{sm}$</td>
<td>smoothness term</td>
</tr>
<tr>
<td>$E_{lc}$</td>
<td>linear constraint term</td>
</tr>
<tr>
<td>$E_{sc}$</td>
<td>scaled as-rigid-as-possible term</td>
</tr>
<tr>
<td>$E_{av}$</td>
<td>averaged as-rigid-as-possible term</td>
</tr>
<tr>
<td>$\beta_{lc}$</td>
<td>linear constraint weight</td>
</tr>
<tr>
<td>$\beta_{sm}$</td>
<td>smoothness weight</td>
</tr>
<tr>
<td>$\alpha_{\ell}$</td>
<td>scaling parameter of the $\ell$th shape</td>
</tr>
</tbody>
</table>

5.7 Conclusions

We applied the concept of overcomplete dictionary representation to the problem of shape deformation. The proposed example-based deformation approach extends the subspace of physically-plausible deformations, while controlling the smoothness of the reconstructed mesh. The blended transformations enable us to find a new pose from a small number of known feature points without any additional information. It is well-suited for real-time applications as well as offline animation and analysis systems. In the future, we plan to apply the proposed framework to various problems from the
field of shape understanding, such as gesture recognition, registration of MRI images, and prior based object reconstruction from depth images.
Chapter 6

V-Flow: Deep Unsupervised Volumetric Next Frame Prediction

Predicting the temporal dynamics of three-dimensional images is an important means for analyzing volumetric data. We propose an unsupervised learning of a multi-scale optical flow based approach for predicting the next frame of a sequence of volumetric images. The fully differentiable model consists of specific crafted modules that are trained on small patches. To test the proposed approach, we ran unsupervised experiments on synthetic incompressible two-fluid Navier-Stokes simulation and real magnetic resonance imaging (MRI) of the cardiac cycle. Comparison of a spatial version of our architecture to recent methods in predicting the next frame of movie sequences shows significant quantitative and visual improvements.

6.1 Introduction

Motion lies at the core of dynamical systems. One way to understand the motion of forms and structures in images is through optical flow [HS81, LK+81], which is an approximation of the motion of objects in an image, and its computation was traditionally based on local spatial and temporal derivatives in a given sequence of images. That is, in two dimensions it tries to specify how much the semantic content of each image pixel moves between adjacent images, while in three dimensions it specifies how much the content of each volume element (voxel) moves between adjacent volumes. While several solutions to deep optical flow and video prediction are well established [RSB+14, SMS15, LKC16, PHC15, KOS+16, MCL15, WRHS13, FDI+15, RB16], volumetric temporal evolution learning remained unexplored. Part of the difficulty in devising a robust and efficient 3D optical flow is due to the large number of possibilities by which each voxel can move. Moreover, unlike the two dimensional case, where there exist some benchmarks with ground-truth optical flow, volumetric datasets lack such supervised information.

The approach introduced here, is motivated by several papers that predict the next
frame of movie sequences, knowing the past and the present frames, in an unsupervised manner. The end-to-end differentiable architecture is based on multi-scale optical flow prediction. Each pyramid level consists of a deep generative network, which is designed as a series of convolution layers with element-wise multiplication modules, followed by rectified linear units. The generative network recursively refines the future optical flow estimation and simultaneously adjusts the last frame that the warping module operates on. We refer to the proposed architecture as “V-Flow.”

The key contributions of V-Flow include the following features.

- The model is trained without any supervision effort, by minimizing the reconstruction error between the predicted volumetric frame and the ground truth next frame. The optimizer minimizes the absolute error criteria aggregated with a volumetric gradient difference loss function.

- Given previous volumetric frames, the system predicts 3D future optical flow. Multiple scales are combined linearly in a Laplacian pyramid like fashion.

- The neural network is specifically designed for the task of optical flow prediction and consists of multiple convolution and multiplication layers. A novel three dimensional warping module is introduced, comprised of 3D grid generator and a linear tri-sampler.

- Inherent property of the proposed approach is the ability to control the trade-off between image-sharpness and quantitative performance. This is done by introducing a latent frame which is a variation of the last frame. Each pixel of the latent frame is moved in keeping with the optical flow to produce the next frame prediction. As the latent frame is closer to the last frame, the predicted frame becomes sharper.

### 6.2 Related efforts

As learning of three dimensional volume prediction is yet unexplored, we draw inspiration from papers related to video prediction. Prediction of both two and three dimensional video dynamics is a challenging problem due to its complexity and the inherent intensity ambiguity in image sequences. Srivastava et al. [SMS15] introduced Long Short Term Memory (LSTM) networks [HS97] to learn representations of video sequences in an unsupervised manner. Lotter et al. [LKC16] designed a neural network architecture, inspired by the concept of predictive coding to continually predict the appearance of future video frames, using a deep, recurrent convolution network with both bottom-up and top-down connections. Patraucean et al. [PHC15] used a convolution LSTM network that integrates changes over time and an optical flow [HS81, LK81] prediction module that extends the Spatial Transformer [JSZ15] by using a per-pixel transformation for
each position instead of a single transformation for the entire image. This approach is somewhat related to supervised deep optical flow models like DeepFlow [WRHS13], FlowNet [FDI15], and SPyNet [RB16].

Ranzato et al. [RSB14] defined a recurrent network architecture inspired from language modeling, predicting the frames in a discrete space of patch clusters. Brabandere et al. introduced the Dynamic Filter Network, where filters are generated dynamically conditioned on an input and demonstrated the effectiveness of the dynamic filter network on the task of video prediction. The Video Pixel Networks proposed by Kalchbrenner et al. [KOS16] use a generative video model, that reflects the factorization of the joint distribution of the pixel values in a video. Oh et al. [OGL15] proposed an action conditional auto-encoder model, and predicted next sequences of Atari games from a single screen image. To deal with the inherently blurry predictions obtained from the standard Mean Squared Error (MSE) loss function, Mathieu et al. [MCL15] proposed a multi-scale architecture, and improved the quality of predicted images by using a Laplacian pyramid [DCF15] and an image gradient difference loss function.

6.3 Architecture

6.3.1 Optical flow process

Suppose we are given an input sequence of $k$ volumetric frames (patches) $V^1, V^2, \ldots, V^k \in \mathbb{R}^{c \times t \times h \times w}$ with $c$ color channels and grid size of $t \times h \times w$. We would like to predict the next frame $Y \equiv V^{k+1} \in \mathbb{R}^{c \times t \times h \times w}$. We assume the existence of an underlying optical flow process that deforms a latent frame $Z \in \mathbb{R}^{c \times t \times h \times w}$ into $Y$. The latent frame $Z$ is guided by an optical flow vector field $F \in \mathbb{R}^{d \times t \times h \times w}$ ($d = 3$), such that each pixel of $Y$
is found by moving the corresponding pixel of $Z$ in line with the displacements coded in $F$. This can be expressed by

$$Y(i, x, y, z) = Z(i, \tilde{x}, \tilde{y}, \tilde{z})$$

where $\tilde{x} = x - F(1, x, y, z)$, $\tilde{y} = y - F(2, x, y, z)$, $\tilde{z} = z - F(3, x, y, z)$, and $i \in \{1 \ldots c\}$, $x \in \{1 \ldots t\}$, $y \in \{1 \ldots h\}$, $z \in \{1 \ldots w\}$. We denote the operator that takes $Z$ and $F$ as inputs and calculates $Y$ as the \textit{FlowWarp} operator.

Our goal is to estimate the latent frame $Z$ and the optical flow $F$. Let $\hat{Z} \in \mathbb{R}^{c \times t \times h \times w}$, $\hat{F} \in \mathbb{R}^{d \times t \times h \times w}$ be the estimations of $Z$, $F$, respectively, and let $\hat{Y} \in \mathbb{R}^{c \times t \times h \times w}$ be the next frame prediction of $Y$ that is calculated by $\hat{Y} = \text{FlowWarp}(\hat{Z}, \hat{F})$. We wish $\hat{Y}$ to be as close as possible to the target $Y$. This can be formalized by introducing a loss function between the predicted next frame and the ground truth next frame. Let $\text{Loss}(\hat{Y}, Y)$ denote this loss function. Therefore, $\hat{Z}$, $\hat{F}$ are found by optimizing

$$\hat{Z}, \hat{F} = \arg\min_{Z, F} \text{Loss}(\text{FlowWarp}(Z, F), Y).$$

A basic approach would set $\hat{Z}$ to be exactly equal to $V^k$. Here, we allow some leeway. We introduce the auxiliary variable $A \in \mathbb{R}^{c \times t \times h \times w}$. $A$ is linearly combined with $V^k$ to get the latent frame $\hat{Z}$.

Theoretically, there are many solutions for $Z$, $F$ that are equivalent. But in practice, because we are setting $\hat{Z}$ as a linear combination of $V^k$ and $A$, we observe that the network converges to a specific solution, such that $\text{FlowWarp}(V^k, \hat{F})$ is a good
estimation of $Y$. Essentially, we see that the auxiliary variable $A$ absorbs most of the blurring effect inherent in video prediction. Hence, in the evaluation phase, it is possible to generate sharper predictions by omitting $A$ altogether. We denote this option as V-Flow-Sharp. Deriving the latent frame $Z$ and the optical flow $F$ is challenging due to the fact that the movement of each pixel between adjacent frames has many degrees of freedom. Therefore, to estimate $\hat{Z}$ and $\hat{F}$ we adopt a multi-scale architecture.

6.3.2 Multi-scale network

Overview. The structure of the multi-scale network is based on pyramid decomposition [BA83, DCF$^+$15, MCL15, RB16] and is made of a series of generative networks that follow one another as illustrated in Figure 6.1. The generative networks use down-sampled versions of the input sequence to make their predictions, ranging from the lowest to the highest resolutions. The predictions are recursively passed on as starting points for consequent pyramid levels, and are refined in such a way that the output of the last generative network reaches the desired original resolution.

More specifically, let $s \in \{1 \ldots N\}$ be the index of the pyramid level. Let $d_s(\cdot)$ be the downsampling function that decimates a frame by a factor of $2^{N-s}$, and let $V_s^1, V_s^2, \ldots, V_s^k \in \mathbb{R}^{c \times t_s \times h_s \times w_s}$ be a down-sampled version of the input volumetric frames in increasing resolution, such that $V_s^j = d_s(V^j), \forall j \in \{1 \ldots k\}$. The network consists of a sequence of $N$ generative networks denoted by $G_s$. Let $\hat{F}_s \in \mathbb{R}^{d \times t_s \times h_s \times w_s}$ be the optical flow estimation, and $\hat{Z}_s \in \mathbb{R}^{c \times t_s \times h_s \times w_s}$ be the latent frame of the $s$ level of the pyramid. The generative network $G_s$ receives the down-sampled input volumetric frames, and the up-sampled products of the preceding network $G_{s-1}$, computes $\hat{F}_s$ and $\hat{Z}_s$ by a multi-layered neural network, and outputs the next frame prediction $\hat{Y}_s \in \mathbb{R}^{c \times t_s \times h_s \times w_s}$ by warping the frame $\hat{Z}_s$ according to the optical flow $\hat{F}_s$. Then, $\hat{F}_s$, $\hat{Z}_s$ and $\hat{Y}_s$ are upsampled by a factor of two and passed to the succeeding generative network $G_{s+1}$. This is done recursively from the lowest resolution to the finest one.

Generative network. The generative network is shown in Figure 6.2. On the left of the Figure is the prediction module and on the right is the warping module. Two sources feed the generative network. The first is the given input sequence of volumetric frames which are down-sampled to the current resolution. The second information source is the set of up-sampled products of the preceding generative network $G_{s-1}$, computes $\hat{F}_s$ and $\hat{Z}_s$ by a multi-layered neural network, and outputs the next frame prediction $\hat{Y}_s \in \mathbb{R}^{c \times t_s \times h_s \times w_s}$ by warping the frame $\hat{Z}_s$ according to the optical flow $\hat{F}_s$. Then, $\hat{F}_s$, $\hat{Z}_s$ and $\hat{Y}_s$ are upsampled by a factor of two and passed to the succeeding generative network $G_{s+1}$. This is done recursively from the lowest resolution to the finest one.

Prediction module. The prediction module calculates the optical flow $\hat{F}_s$ and the latent frame $\hat{Z}_s$. It includes a neural network that is specifically designed for the task of optical flow prediction.

Convolution and multiplication neural network. The multi-layered neural network outputs the flow residual $R_s \in \mathbb{R}^{d \times t_s \times h_s \times w_s}$ and the auxiliary variable $A_s \in \mathbb{R}^{c \times t_s \times h_s \times w_s}$, that are used to calculate $\hat{F}_s$ and $\hat{Z}_s$. It is based on a classic structure of volumetric convolution layers followed by rectified linear units (ReLU). To better capture the temporal correlations between adjacent frames, we insert an element-wise
Figure 6.3: Convolution and multiplication neural network.

Table 6.1: Network configuration

<table>
<thead>
<tr>
<th>Volumetric next frame prediction</th>
<th># feature maps</th>
<th>kernel size</th>
</tr>
</thead>
<tbody>
<tr>
<td>G1</td>
<td>64,128,64</td>
<td>5,3,3,5</td>
</tr>
<tr>
<td>G2</td>
<td>64,128,256,128,64</td>
<td>5,3,3,3,3,3,5</td>
</tr>
<tr>
<td>G3</td>
<td>64,128,256,128,64</td>
<td>7,5,5,5,5,5,7</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Video next frame prediction</th>
<th># feature maps</th>
<th>kernel size</th>
</tr>
</thead>
<tbody>
<tr>
<td>G1</td>
<td>128,256,128</td>
<td>3,3,3,3</td>
</tr>
<tr>
<td>G2</td>
<td>128,256,128</td>
<td>5,3,3,5</td>
</tr>
<tr>
<td>G3</td>
<td>128,256,512,256,128</td>
<td>5,3,3,3,3,3,5</td>
</tr>
<tr>
<td>G4</td>
<td>128,256,512,256,128</td>
<td>7,5,5,5,5,7</td>
</tr>
</tbody>
</table>

Table 6.2: Mean Square Error (MSE) of pressure predictions (in micro-units).

<table>
<thead>
<tr>
<th>Method</th>
<th>1st Frame</th>
<th>2nd Frame</th>
<th>3rd Frame</th>
<th>4th Frame</th>
<th>5th Frame</th>
<th>6th Frame</th>
<th>7th Frame</th>
<th>8th Frame</th>
</tr>
</thead>
<tbody>
<tr>
<td>V-Flow (test)</td>
<td>3.7</td>
<td>21.0</td>
<td>67.7</td>
<td>160.6</td>
<td>342.3</td>
<td>639.7</td>
<td>1031.4</td>
<td>1509.1</td>
</tr>
<tr>
<td>V-Flow-Sharp (test)</td>
<td>23.3</td>
<td>109.2</td>
<td>280.4</td>
<td>500.9</td>
<td>780.6</td>
<td>1095.5</td>
<td>1434.1</td>
<td>1781.9</td>
</tr>
<tr>
<td>Last (test)</td>
<td>68.3</td>
<td>243.7</td>
<td>477.4</td>
<td>734.3</td>
<td>995.3</td>
<td>1251.8</td>
<td>1500.1</td>
<td>1739.5</td>
</tr>
<tr>
<td>V-Flow (train)</td>
<td>3.7</td>
<td>21.1</td>
<td>67.0</td>
<td>158.8</td>
<td>335.9</td>
<td>619.6</td>
<td>983.1</td>
<td>1416.4</td>
</tr>
<tr>
<td>V-Flow-Sharp (train)</td>
<td>23.0</td>
<td>107.3</td>
<td>276.8</td>
<td>494.2</td>
<td>771.3</td>
<td>1082.1</td>
<td>1415.6</td>
<td>1757.0</td>
</tr>
<tr>
<td>Last (train)</td>
<td>74.1</td>
<td>259.3</td>
<td>499.9</td>
<td>758.9</td>
<td>1018.8</td>
<td>1270.4</td>
<td>1510.1</td>
<td>1737.4</td>
</tr>
</tbody>
</table>
Figure 6.4: Turbulence prediction. In each of the 3 examples, the first row are the 6 ground truth pressure slices, and the second row is their respective predictions.

Table 6.3: Mean Square Error (MSE) of MRI predictions (in milli-units).

<table>
<thead>
<tr>
<th>Method</th>
<th>1st Frame</th>
<th>2nd Frame</th>
<th>3rd Frame</th>
</tr>
</thead>
<tbody>
<tr>
<td>V-Flow (test)</td>
<td>3.2</td>
<td>8.0</td>
<td>13.0</td>
</tr>
<tr>
<td>V-Flow-Sharp (test)</td>
<td>3.6</td>
<td>9.1</td>
<td>14.4</td>
</tr>
<tr>
<td>Last (test)</td>
<td>5.1</td>
<td>12.5</td>
<td>18.8</td>
</tr>
<tr>
<td>V-Flow (train)</td>
<td>2.9</td>
<td>7.3</td>
<td>11.9</td>
</tr>
<tr>
<td>V-Flow-Sharp (train)</td>
<td>3.3</td>
<td>8.4</td>
<td>13.4</td>
</tr>
<tr>
<td>Last (train)</td>
<td>4.7</td>
<td>11.5</td>
<td>17.4</td>
</tr>
</tbody>
</table>
Figure 6.5: Cardiac cycle prediction.
Figure 6.6: Cardiac cycle prediction.
Figure 6.7: Comparison of different methods to predict the next frame from UCF101. Left to right: ground truth, V-Flow, V-Flow-Sharp, adversarial, optical flow.

multiplication block, as shown in Figure 6.3. The configuration we used in our experiments are given in Table 6.1. In all layers, the size of the multiplication block is 1/4 of the overall number of feature maps.

Flow prediction. Let $u(\cdot)$ be a function that increases the resolution of the flow prediction of the preceding pyramid level by a factor of two. The optical flow estimation $\hat{F}_s$ is obtained by applying a pyramid style refinement

$$\hat{F}_s = \beta u(\hat{F}_{s-1}) + R_s.$$ 

The weight $\beta$ regulates the propagation of the optical flow. We typically initialize $\beta = 0.7$.

Latent frame. The latent frame $\hat{Z}_s$ is determined by linearly combining the last input volumetric frame $V^k_s$ and the auxiliary variable $A_s$ as follows

$$\hat{Z}_s = \alpha V^k_s + A_s.$$ 

$\alpha$ is a weight that controls the power of the last frame in $\hat{Z}$. We initialize $\alpha = 1.0$.

Warping module. The refined optical flow $\hat{F}_s$ drives a 3D optical flow warping module. This module is implemented similarly to the method of Patraucean et al. [PHC15] for video prediction. A three dimensional grid generator represents the optical flow as a dense transformation map that maps $\hat{Z}_s$ to $\hat{Y}_s$. A novel trisampler module follows. It uses the map to interpolate $\hat{Z}_s$ linearly, and effectively moves voxels of the estimated latent frame $\hat{Z}_s$ to obtain $\hat{Y}_s$. Hence, we have

$$\hat{Y}_s = FlowWarp(\hat{Z}_s, \hat{F}_s) =$$

$$= \text{Trisampler}(\hat{Z}_s, \text{Grid Generator}(\hat{F}_s)).$$
Figure 6.8: Comparison of different methods on UCF101.
Figure 6.9: Comparison of different methods on UCF101.
6.3.3 Training

The model is trained by minimizing the reconstruction error between the predicted next frame and the ground truth next frame. One way is to minimize the $L_p$ distance

$$L_p(\hat{Y}, Y) = \|\hat{Y}_s - Y_s\|_p^p.$$ 

In our multi-scale architecture we use $p = 1$.

Another option is to penalize the differences between the prediction gradients and the image gradients. We adapt the gradient difference loss (GDL) of [MCL15] to our volumetric setting. Thus, we define the volumetric gradient loss (VGDL)

$$L_{\text{VGDL}}(\hat{Y}, Y) = \sum_{x,y,z} \left( |Y_{x,y,z} - Y_{x-1,y,z}| - |\hat{Y}_{x,y,z} - \hat{Y}_{x-1,y,z}| + |Y_{x,y,z} - Y_{x,y-1,z}| - |\hat{Y}_{x,y,z} - \hat{Y}_{x,y-1,z}| + |Y_{x,y,z} - Y_{x,y,z-1}| - |\hat{Y}_{x,y,z} - \hat{Y}_{x,y,z-1}| \right).$$

The total loss combines the $L_1$ and the $L_{\text{VGDL}}$ loss functions with different weights, and is expressed by

$$\text{Loss}(\hat{Y}_s, Y_s) = \lambda_{L_1} \sum_{s=1}^S L_1(\hat{Y}_s, Y_s) + \lambda_{\text{VGDL}} \sum_{s=1}^S L_{\text{VGDL}}(\hat{Y}_s, Y_s).$$

In our experiments the parameters $\lambda_{L_1}$ and $\lambda_{\text{VGDL}}$ are set to 1.0 and 0.5, respectively.

Table 6.4: PSNR, SSIM and sharpness results on UCF101 dataset

<table>
<thead>
<tr>
<th>Method</th>
<th>1st Frame</th>
<th>2nd Frame</th>
<th>3rd Frame</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSNR</td>
<td>SSIM</td>
<td>Sharpness</td>
</tr>
<tr>
<td>V-Flow</td>
<td>32.10</td>
<td>0.93</td>
<td>25.65</td>
</tr>
<tr>
<td>V-Flow-Sharp</td>
<td>31.65</td>
<td>0.93</td>
<td>25.50</td>
</tr>
<tr>
<td>Mathieu et al. [MCL15]</td>
<td>31.47</td>
<td>0.91</td>
<td>25.38</td>
</tr>
<tr>
<td>Last</td>
<td>28.50</td>
<td>0.89</td>
<td>24.58</td>
</tr>
<tr>
<td>EpicFlow et al. [RWHS15]</td>
<td>31.97</td>
<td>0.93</td>
<td>25.57</td>
</tr>
</tbody>
</table>

6.4 Experiments

To assess qualitatively and quantitatively the behavior of the proposed architecture and its components, we ran unsupervised experiments on synthetic as well as real datasets. In order to predict more than one frame, we apply the model recursively by using the newly generated frame as an input. In all our experiments we used four input frames. Our implementation is based on Torch library [CKF11] and was trained on a TITAN-X nvidia GPU with 12GB memory. The optimization was done using adagrad [DHS11]. The spatial and volumetric configuration have 15,808,667 and 16,648,465 trainable parameters.
parameters, respectively.

### 6.4.1 Two-fluid Navier-Stokes simulation

Fluid dynamics has a wide range of applications, including calculating forces and moments on aircraft, determining the flow rate of water through pipelines, and predicting weather conditions and ocean currents. The motion of viscous fluid substances is governed by the Navier-Stokes equations, which are a non-linear set of differential equations that describes the flow of a fluid whose stress depends linearly on flow velocity gradients and pressure.

We test our architecture on the homogeneous buoyancy driven turbulence database [Pul15, LPW\textsuperscript{+}08] that simulates the turbulence of incompressible two fluids of different molar masses by solving the Navier-Stokes equations. We note that the numerical solution of the Navier-Stokes equations for turbulent flow is extremely difficult and the computational time in some situations becomes infeasible for calculation.

The database simulation grid is of size $1024^3$ at 1015 time frames. The input to our model is the pressure volume derived from the solution of the Navier-Stokes equations. The pressure does not fully define the temporal evolution of the buoyancy driven turbulence. Nevertheless, we try to predict the future pressure volumes. We randomly extract sequences of 20 time frames and of size $64^3$ from the simulation grid. The frames were normalized so that their values are mapped to the interval $[-1, 1]$. We used 144 sequences for training and 49 for testing.

Figure 6.4 shows examples of the predicted pressure at six consecutive future time frames using the proposed volumetric optical flow architecture. Table 6.2 shows the mean square error for predicting eight future frames. We compare V-Flow to the baseline of the last frame, and to V-Flow-Sharp on both the train and test sets. We see that the model generalizes well. We see that in this case the latent variable is very important, since it can predict the mixing of the two fluids. Hence there is a gap between the performance of V-Flow and of V-Flow-Sharp.

### 6.4.2 Volumetric MRI

Magnetic Resonance Imaging (MRI) is considered the gold standard test to accurately assess the heart’s squeezing ability. Analyzing the heart’s motion is important for estimating the amount of blood ejected from the left ventricle with each heartbeat. The 2015 Data Science Bowl (DSB) dataset [KH15] consists of cardiac MRI images in DICOM format across the cardiac cycle, with a minimum of 8 slices at each time frame. We have extracted the region of interest [KNH15] of size $128 \times 128$ from each slice. We trained the multi-scale architecture on $8 \times 16 \times 16$ patches on 455 training sequences of volumetric data to predict the next volume in the sequence. Figures 6.5, 6.6 show examples of the next volume predictions of the cardiac cycle. We tested the model on 174 test sequences. Table 6.3 shows the performance of the proposed approach for
the testing and training sequences. Again, we see that the algorithm generalizes well. Remark: careful inspection of the results shows that there is no significant flow across slices. This can be explained by the fact that the slice thickness is between 4 to 10 times larger than the pixel spacing within slice.

### 6.4.3 Video prediction

To quantitatively compare our approach to existing methods, we had to downscale our approach to two dimensions. For training, we used a subset of the Sports1m dataset [KTS+14]. All frames were down-sampled to a $240 \times 320$ pixel resolution and normalized. We train our network by randomly selecting temporal sequences of patches of size $64 \times 64$ pixels.

We evaluated the quality of our video predictions on 738 test videos from the UCF101 dataset [SZS12]. We compute the Peak Signal to Noise Ratio (PSNR), the Structural Similarity Index Measure (SSIM) [WBSS04] and sharpness of the images as in [MCL15]. As some of the images in the UCF101 dataset do not involve any motion, we use the approach presented in [MCL15] and compute the different quality measures only in the regions where the optical flow is higher than a fixed threshold. In some of the sequences, the last frame predicts the next frame almost perfectly. These sequences are discarded. We evaluate our architecture with and without modifying the last. We compare to the baseline last image and to the adversarial learning method of Mathieu et al. [MCL15, GPAM+14]. We also include the optical flow method that extrapolates the pixels of the next frame by using the optical flow from the last two frames [RWHS15, Dol]. The results are given in Table 6.4. We see that V-Flow presents a significant improvement of all measures. Figures 6.7, 6.8 and 6.9 show examples of the next frame prediction on test sequences from the UCF101 dataset. Although V-Flow present the best quantitative results, its output is blurred. By applying the optical flow on the last frame, we get much sharper images which visually appear realistic. The results of the adversarial net are blurry and there are some artifacts. The optical flow images are usually sharp and visually appealing, but in Figure 6.7 the skate-boarder’s head is squeezed and the trumpeters are filtered out.

### 6.5 Conclusions

A volumetric optical flow based next frame prediction method has been presented. We defined an underlying optical flow process that is flexible enough to model a range of problems. We used neural networks at each level of the volumetric pyramid to estimate the nature of the optical flow process, and trained the networks in an unsupervised manner.

We believe that the proposed approach can open up doors to new insights and applications. As the three dimensional world around us is dynamic, learning temporal
volumetric principles is key for our comprehension of our surroundings, and in the proposed method is an attempt to make a small step forward in this exciting new direction of understanding three dimensional system dynamics through deep learning.
Chapter 7

Discussion and Conclusions

In this thesis, we addressed the question of correspondence in three-dimensions in various forms. We discussed the correspondence problem of isometric and non-isometric shapes, considered refinement of dense and sparse correspondences, developed methods for occluded, disconnected and partial shapes, and addressed the issue of optical flow of whole scenes.

In Chapter 2 we discussed the problem of eigenfunction matching. Although the Laplace-Beltrami operator provides us with a flat eigenspace in which surfaces could be represented as canonical forms, the order and directions (signs) of the axes in this Hilbert space do not have to be compatible when two nearly-isometric surfaces are considered. In order to resolve such potential ambiguities, we resorted to high order statistics of the eigenfunctions of the LBO and their interaction with the surface normal.

In Chapter 3 we extracted a new feature vector for embedding nonrigid shapes, which is based on the inner and external products between the gradient fields of pairs of eigenfunctions, and integrated it into a holistic shape matching system. The embedding, which is proved to be injective, induces a distance between points on the surface, and can be defined to measure a distance between shapes.

In Chapter 4 we further developed our shape matching pipeline, introducing a new correspondence refinement method. The method is based on the evaluation of the spectral kernel distance, optimized by an ICP based approach in the spectral domain. The algorithm is shown to be robust, flexible and easy to implement. It can be used efficiently as a refinement procedure of rough or sparse correspondence detection methods. We have demonstrated the effectiveness of the algorithm by achieving state-of-the-art results on shape matching benchmarks.

A somewhat different approach is presented in Chapter 5. We applied the concept of overcomplete dictionary representation to the problem of shape deformation and partial similarity. The proposed example-based deformation approach extends the subspace of physically-plausible deformations, while controlling the smoothness of the reconstructed mesh. The blended transformations enable us to find a new pose from a small number of known feature points without any additional information. It is well-suited for real-time
applications as well as offline animation and analysis systems.

In Chapter 6, we discussed the challenge of volumetric next frame prediction, and implemented a fully differentiable model, based on optical flow. We defined an underlying optical flow process that is flexible enough to model a range of problems. We used neural networks at each level of a volumetric pyramid to estimate the nature of the optical flow process, and trained the networks in an unsupervised manner. Overall, we strongly believe that the combination of deep learning and geometry processing opens up doors to many insights and applications that involve correspondence detection in three dimensions, and that we are only at the beginning of exploring this path.
Bibliography


אריתקטורה מבוססת זרימה אופטית מורחבת סקלולה דיכ דלעת ואת התומנה הבאה בזרף של תומנה
נפוחות. המודל גורם ל vítima של כל מודולם של/Images המבקרת דל בזירה של מבקרת על בלוקים קטנים של מודים
ורשימים, שמכנים, V-Flow, שמכנים לentlich הדינמייה של תומנה של תומנה התלטטת הארץ שימוש
שלכלים, נשלוח תנועה של נולמות טמוניות ולזרף התומנה של שמסור בזר襞ת התויה מנכית

Technion - Computer Science Department - Ph.D. Thesis PHD-2017-13 - 2017
הנושא הבכירו יונית של מצじゃない תלמידי של הולランדר ינגב, מנותרה כמيين קמיע רהב
ללאגרטימור אירה פנומ쨌バンド אנטומור היישוב. האלצ'ה ספקטרליות הפכה ליהור פручיב
השבג אפקטיוון למצלמת רמכ ועפעה קומפוזיט קוריור ודסומריית. אלאחרות שרוחות
קונבוקלאציה הראת بحي ציון מתלוי בתחתית של ראיתו מוסמבה.ющה, היא קיטנה
שים כל עוד בערユーコ בביריינטיה ולא שימשו באלקטרה קרן התמצאה ביהור, פורמיציה של
אいると מונחון גומירה אפקטי התלמידי.

מתיאז התמצאה ה׳נקודה ב Springfield היא פועל בכיסית בחור והאנטומור היישוב. מודד
שהיות ביר ע Xperia שמיישם בויר להישמעו וי scraped ייהושם שקרי ביהור איה לשכן
את התרות ספקטרלית לדידיו ו yakın בצורת יבר אנטיידים ביווימרי ולגパーデיר על הביעה התמצאה
Spectral Gradient Fields Embedding מקט את המרחק בברבר, היא אמו Snowden ספקטרליים חק דגא
שמיצע קסם לקוליאבוןينا הצווארת הערמה ובאפרוניט קלפולה-בטולון והאנטומור הדיאגוניט
שהמשטח לע הערמה חשש התמצאה בין הפונקציות הגאומטריה של יחידי יזורו
антימגנטים. השכון מונסב על הערמה שה hakk התמצאה בין הפונקציות המנוגמתות
שהמשטח באמות כלכל תליו. הנה וב העים כורבדו אל פראיות. היא מתאימה יפעו מוכני
את ידי התמצאה לש מוספר קף לש פונקציית המתומת, שם ייצוב חסיט, על ידי שימש בספקטרליות
 PSGEM
מסדר נבדל

כפי כן, בחר את הא saturaton לתחמש חמר הביניעב הספקטרליות של יזורוכ מדרד
ברית יכל להתקין את עלייה התמצאה. על מזג טלוף בין埃尔ג потому שתמצאה ספקטרליות
Iterative Closest Spectral Kernel Maps, (ICSKM) "ה
ר" ימי קצף הם עיקרות מתאימה בין הפרספקטיבים שהאנטומור הדיאגוניט
בהברת התלמידים, ישו וחושית את הפרספקטיביות האופטימלית מתאימה בין יזורוכ בברבר
הספקטרליים

במקורה והתמצאות לכלול תלガイド רהזם ממסתתת. אל שמלmom ומוסמובה, יהיו הבנקים שחיות측מטים
בכפרים קף לש מהות אופטימי של יזורוכ. רצים שענים התמצאה ספקטריה הם גם הנחיה
ודרש על ידי תרימבכ reiterated של מוספר קף את המרחב מורדר באפרוניטי מפרק פתייה.
Fast Blended מילול המילה מזרבה משקל פומיים של קורודיקי הדנסטים. האלגוריתמים, שמקרא
Transformations, (FBT) מזא את התמצאה הפרמקנומות יאיר הפרש הركة על ידי פיצה פוגוס
לפוכציית עלאת אל ליאנות הש觭ון לע תלктив של מרחב התنقص האפיניות, זוכאז היא מעלה את
התמצאות יני הפרש הלקט חלבומזמות.

את הדרכיס ילע התמצאה בפשעה יונימן תלמידים זה על ידי ירוי אופטימי פיתוח. גאגר חכון

תקציר


התאמת צורות על ידי שיטות ספקטרליות
וכלידה עmozilla.

תIntroduction

לשם מילוי חלקו של הדירוגת במסלול התואר
דוקטור לפילוסופיה

אלון שטרן

רגשה לנטט הטכניון ... מרכז טכנולוגיה לישראל
אב הטכניון ריפה אוניברס 2017
התאמות צורות על ידי שיטות ספקטרליות
ולמידהعمוקה.

אלון슈טרן