FAKULTÄT FÜR INFORMATIK

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Informatics

Dense 2D-to-3D Shape Matching via Low-Rank Functional Correspondence

Aneta Stevanović

FAKULTÄT FÜR INFORMATIK

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Informatics

Dense 2D-to-3D Shape Matching via Low-Rank Functional Correspondence

Dichtes 2D-zu-3D Shape Matching mit niedrig dimensionalen, funktionalen Korrespondenzen

Author:	Aneta Stevanović
Supervisor:	Prof. Dr. Daniel Cremers
Advisor:	Dr. Emanuele Rodolà
Submission Date:	May 15, 2015

I confirm that this master's thesis is my own work and I have documented all sources and material used.

Garching, May 15, 2015 ....

Aneta Stevanović

Acknowledgments

I would like to offer my sincere gratitude to my advisor, Dr. Emanuele Rodolà, for sharing his knowledge with me, as well as always showing patience and encouragement throughout the course of this thesis. Furthermore, I wish to thank M.Sc. Luca Cosmo for his help with the implementation details and for including me in the discussions of his research. I also thank Prof. Dr. Daniel Cremers for the opportunity to write my thesis at his chair.

Abstract

This work presents a method for 2D-to-3D non-rigid shape matching via functional correspondence. A point-to-point mapping between the shapes is computed from a functional correspondence matrix that displays sparsity, distinct diagonal structure, and low rank. Three objective functions with different regularisation terms are proposed to obtain the correspondence matrix with such characteristic features. The objective functions are convex and the optimisation process produces globally optimal solutions. The effect the regularisation terms along with the use of different spectral shape descriptors have on the accuracy of the computed point-to-point mappings is shown experimentally.

Contents

Part I.

Introduction and Theory

1. Introduction

Shape matching is a fundamental problem in computer vision and graphics with as varied applications as shape reconstruction, scan registration, and deformation transfer amongst others.

Rigid shape matching is well understood and can be expressed by rotations and translations [ELF97]. However, matching shapes that are undergoing non-rigid deformations is a challenging task. Matchings in this case are often represented as a correspondence between points on two given shapes. The problem of 2D-to-3D shape matching being tackled in this work is no exception. The goal is to find dense correspondence between a 2D sketch and a 3D shape as illustrated in Figure 1.1. A large number of approaches have been proposed addressing the issue of how to obtain these point-to-point correspondences.

An established approach is to compute features on each shape that capture their geometry, and then match the features by minimising some distortion measure [Ovs+10], [Hua+08], [SY11]. Some of the feature descriptors that can be used are Heat- [SOG09] and Wave Kernel Signature [ASC11], and Global Point Signature [Rus07]. Once the features are matched, they may be employed as an initialisation for calculating the correspondence between all points of the shapes and, by doing so, obtain a dense point-to-point map. The approach is motivated by the assumption that commonly encountered non-rigid deformations, the ones caused by articulated motion, can be expressed as intrinsic near isometries. That is, the geodesic distances between pairs of points on the shape are more or less preserved after applying the deformation. The performance of such an approach depends on the quality of the feature descriptors and their invariance to isometric deformations and noisy data.

The introduction of functional maps approach [Ovs+12] has provided an easy way of using different feature descriptors alongside each other to find correspondence between 3D shapes. In this way, the strengths of various descriptors can be combined. The functional maps approach is also more flexible, since it is possible to choose appropriate bases in which to represent descriptor functions.

Besides 3D shape matching, functional maps have been used in formulating shape difference operator [Rus+13], in exploring shape collections [HWG14], in symmetry matching [Ovs+13], in image segmentation [Wan+14]. However, it is not known whether functional maps approach can be used successfully in the case of 2D-to-3D matching and which shape descriptors are capable of capturing adequate geometric information across 2D and 3D domains. This work attempts to answer these questions.

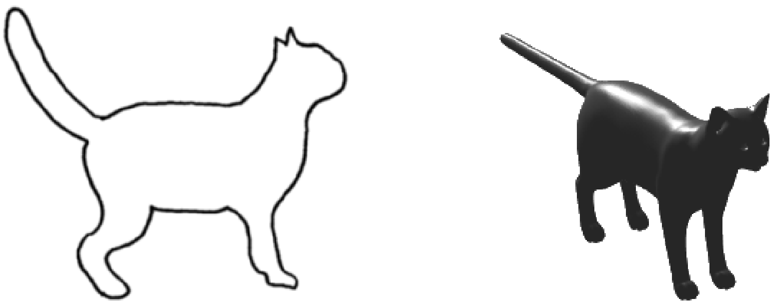


Figure 1.1.: Example of a 2D sketch on the left and a 3D shape on the right. The problem of 2D-to-3D shape matching aims at finding a dense correspondence between two shapes like these.

1.1. Related Work

Since the proposed approach for 2D-to-3D matching is inspired by the recent developments in the area of 3D shape matching, a brief overview of the related work in that area is presented here. Moreover, the work pertaining to the 2D-to-3D case is also presented which is in most cases a part of shape retrieval and sketch based modelling approaches.

Functional 3D shape matching. In 2012, the functional maps approach [Ovs+12] for matching 3D shapes was introduced in which a correspondence map between a pair of shapes is represented as a linear transformation between their function spaces. The map is expressed simply as a matrix and is computed from a set of corresponding scalar functions (e.g wave kernel signatures, region indicators) on both shapes. These functions are encoded in the shapes' harmonic bases and the approach relies on them being compatible. However, harmonic bases computed independently on different shapes do not always exhibit this behaviour. In [Kov+12] the authors addressed this issue by proposing to obtain common bases between shapes using joint diagonalization. The use of common bases allows to take advantage of the sparsity of the correspondence matrix and hence improve computation efficiency. However, a set of point or region correspondences has to be provided as input to this approach.

Pokrass et al. [Pok+12] proposed a method that builds on [Ovs+12] and is able to match shapes using very little known information, mainly the ability to detect repeatable regions. Their ordering is not known nor how many regions are in correspondence with one another. The functional map is computed using methods from the field of sparse modelling and weight matrix is introduced to better promote the diagonal structure of the correspondence matrix. The most recent work [Kov+15] treats finding a functional map between pairs of shapes as geometric matrix completion problem. It is shown that adding geometric structure is beneficial when scarce functional correspondences are provided and when matching highly non-isometric shapes.

2D-to-3D shape matching. Several methods have been proposed for sketch driven shape deformations. The main prerequisite in such a process is establishing correspondence between a sketch and a 3D shape. The methods of [T0L10] and [KSP09] use a contour of the sketch to align it to the 3D shape, and therefore only the correspondence of the points on the contour and the silhouette of the 3D shape are calculated. These methods require user interaction to determine a common orientation between the contour and the shape, and to specify several 2D-to-3D point correspondences to aid in the alignment process. Another method [LA11] represents the sketch and 3D shape as a combination of circle-triangle-square primitives and then puts these primitive shapes into correspondence. This method has high computational cost and is not able to produce dense point-to-point correspondence. The approach of Xu et al. [Xu+11] is designed to work on man-made models. Region correspondences are found between a photo object and the 3D shape for the purpose of model-driven modelling. User interaction is involved in several situations due to the semantic region segmentation algorithm used.

A different approach to 2D-to-3D matching comes from the area of sketch-based 3D shape retrieval. The matching process is cast to 2D domain by sampling views of 3D shape and finding feature correspondences between the sketch and the created views. The goal of the matching is to produce a similarity measure which does not necessitate point-to-point mapping. Therefore a method to compute such a map is not available. Various shape descriptors and feature views have been proposed for matching and more information on how they are used in the context of shape retrieval can be found in the following survey paper [Li+14].

1.2. Goal

The overall goal is to apply functional maps approach to 2D-to-3D non-rigid shape matching. The expectation is that this approach is able to produce quality point-to-point correspondence. To achieve this goal, three convex objective functions with different regularisation terms aimed at capturing the characteristic features of functional correspondence matrix such as sparsity, diagonal structure, and low rank are proposed. The convexity allows for the globally optimal solutions to be computed. The performance of the objective functions is shown by conducting experiments using well-known 3D shape data set augmented with 2D shapes extracted from images of those 3D shapes.

2. Background Knowledge

Laplace-Beltrami operator provides a base for a variety of geometry processing tasks. Its eigenfunctions and eigenvalues are the main components in the functional maps framework and in the shape descriptors used. The following sections present these topics in more detail and set the stage for the proposed 2D-to-3D shape matching method.

2.1. Laplace-Beltrami Operator

Given a real-valued, twice differentiable function f defined on a Riemannian manifold M (written in mathematical notation as $f \in L^2(M)$), the Laplace-Beltrami operator Δ is defined as:

$$\Delta f = -\text{div}\,(\nabla f) \tag{2.1}$$

where ∇f is the gradient of f, and div the divergence on the manifold. In Euclidean plane $\mathbb{R}^2$, the Laplace-Beltrami operator becomes the familiar Laplacian:

$$\Delta f = -\frac{\partial^2 f}{(\partial x)^2} - \frac{\partial^2 f}{(\partial y)^2} \tag{2.2}$$

The Laplace-Beltrami operator admits an eigendecomposition with eigenvalues λ and associated eigenfunctions ϕ:

$$\Delta \phi = -\lambda \phi \tag{2.3}$$

The eigenvalues of Equation 2.3, which is known as the Helmholtz equation, are non-negative and represent a discrete set ($0 \leq \lambda_1 \leq \lambda_2 \leq ... + \infty$). The eigenfunctions are real-valued functions defined on the whole manifold. Moreover, the eigenfunctions corresponding to distinct eigenvalues are orthogonal with respect to the inner product:

$$\langle \phi_i, \phi_j \rangle = \int_M \phi_i(x)\phi_j(x)dx = 0, \quad i \neq j \tag{2.4}$$

Eigenfunctions and eigenvalues have a physical world interpretation. One example where they play a key role is in describing what happens when a membrane of a drum is hit. The eigenfunctions can been seen as the natural vibration modes of the membrane, while the eigenvalues represent the corresponding resonant or vibration frequencies.

The *spectrum* is defined to be the family of eigenvalues of Equation 2.3. The multiplicity of an eigenvalue λ in the spectrum equals to the dimension of the vector space spanned by the eigenfunctions associated with the eigenvalue.

2.1.1. Properties

The Laplace-Beltrami operator and in turn the spectrum and the eigenfunctions have some nice properties [Reu+09b] which make them useful in the field of 3D shape analysis. These properties are:

(a) The Laplace-Beltrami operator is isometry invariant as it only depends on the gradient and the divergence which are intrinsic properties of the manifold. In other words, they are independent of the coordinate space in which the manifold is embedded. The isometry invariance is exhibited by the spectrum and the eigenfunctions as well.

(b) Scaling a manifold by the factor a has a known effect on the eigenvalues: they are scaled by the factor $\frac{1}{a^2}$. This makes comparing spectra possible, regardless of the difference in the sizes of manifolds.

(c) The spectrum and the eigenfunctions change continuously when the shape is being deformed.

(d) A fair amount of geometrical and topological information can be derived from the spectrum. On a Riemannian manifold with boundary it is shown that knowing the spectrum allows for deducing the area, perimeter and genus of the manifold [MS67].

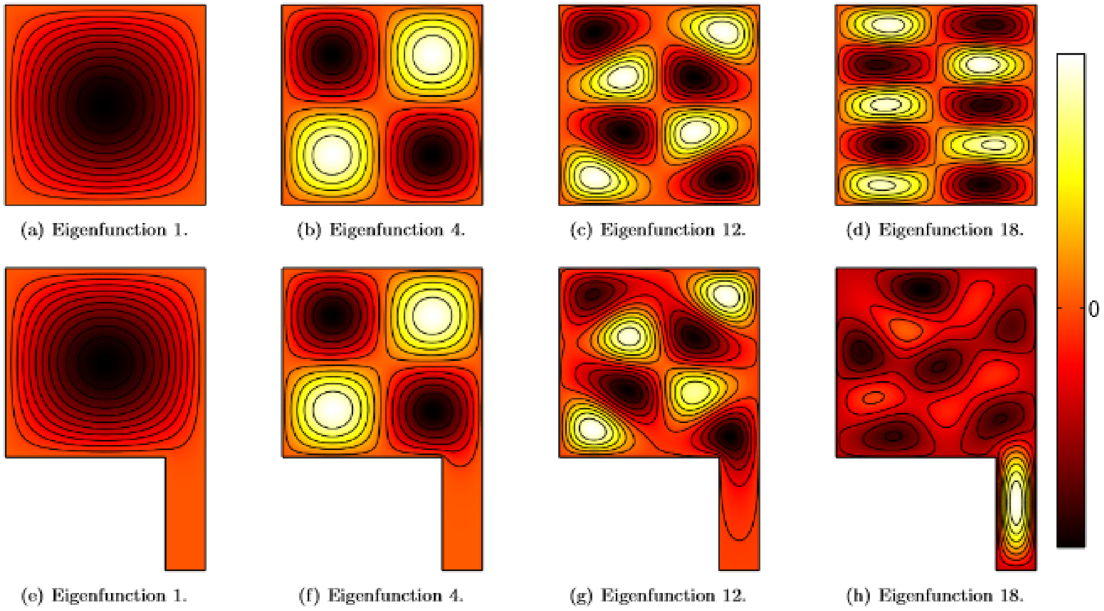
(e) The eigenfunctions with the smallest eigenvalues are more robust against shape change or noise, as they present the lower frequency modes.

2.1.2. Boundary conditions

Two types of boundary conditions can be used to solve Equation 2.3: the Dirichlet and the Neumann boundary condition. The former imposes that the value on the boundary of M is fixed ($\phi = 0$), whereas the latter imposes that the derivative in the normal direction of the boundary is fixed ($\frac{\partial \phi}{\partial n} = 0$). Of particular interest is the effect these boundary conditions have on eigenfunctions since they are the major building blocks in the functional shape matching approach.

Figures 2.1 and 2.2 display the different behaviour of the two boundary conditions on a 2D example of a square with a tail (S_T) and a unit square (S_1). In the Dirichlet case (Figure 2.1), the lower eigenfunctions are not able to pick up the presence of the attached tail of S_T. Only when higher eigenfunctions are considered (h) does the value of the tail area stop being constant or close to it. This is due to the restrictive condition of having the values on the boundary fixed to zero. Also worth pointing out is that as soon as the tail is detected, the corresponding eigenfunctions of S_T and S_1 do not look alike (d - h).

The Neumann boundary condition behaves differently (Figure 2.2). The addition of features like the tail influence the eigenfunctions very early, because the boundary is free to assume different values, not just zero. The first non-constant eigenfunction (d) is very different from the first non-constant eigenfuntions of the square (a). However, the



(a) Eigenfunction 1. (b) Eigenfunction 4. (c) Eigenfunction 12. (d) Eigenfunction 18.

(e) Eigenfunction 1. (f) Eigenfunction 4. (g) Eigenfunction 12. (h) Eigenfunction 18.

Figure 2.1.: Eigenfunctions for the Dirichlet boundary conditions for a square (top) and a square with tail (bottom). Low frequency (low eigenvalue) eigenfunctions do not probe the tail region due to the restrictive Dirichlet boundary conditions. Picture is taken from [Reu+09b].

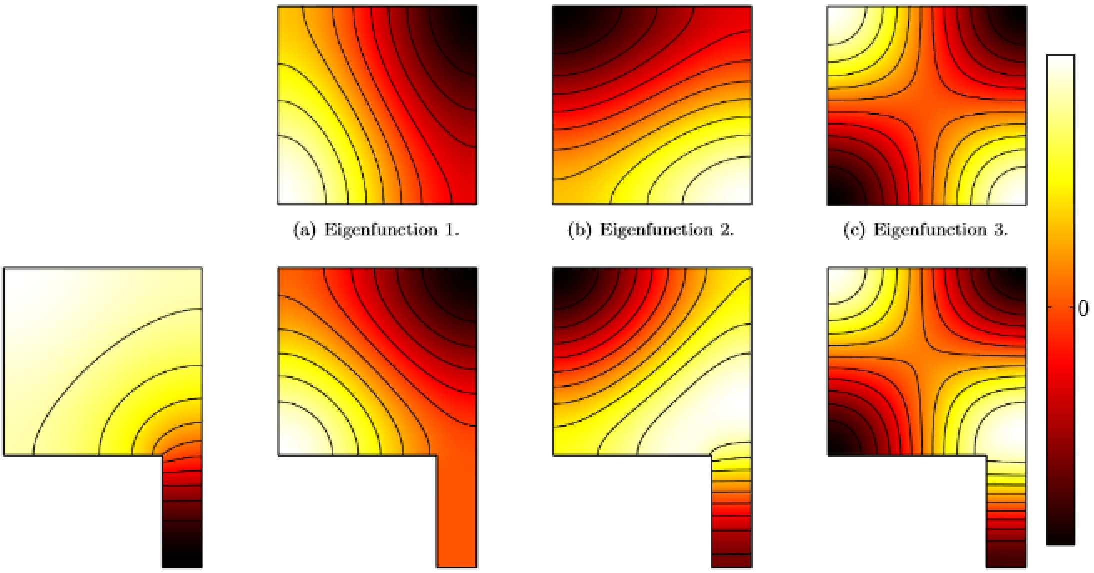
next eigenfunction (e) of S_T is almost identical to (a). Also the next few eigenfunctions (b - f and c - g) correspond with each other on the square region.

The work of Rodolà et al. [Rod+15] provides a mathematical justification of the behaviour of eigenfunctions in the presence of missing shape parts. They show that the variation of eigenfunctions in Figure 2.2 is caused only by their interactions along the boundary (or cut) between S_1 (square) and $S_T \setminus S_1$ (tail). The variation is not a function of the missing area. If the cut goes through low values of eigenfunctions, many of them are preserved across the shapes, whereas if it goes through high values, the eigenfunctions vary by a large amount.

For the purpose of shape matching, the Neumann boundary condition is advantageous as it picks up the significant changes in geometry much earlier. Also the addition of a new geometric feature does not drastically change the appearance of the eigenfunctions. They look similar to the corresponding eigenfunctions on the old shape.

2.1.3. Eigenfunctions as bases

Eigenfunction of the Laplace-Beltrami operator are analogue to Fourier bases on Euclidean domains. Hence, any square integrable real-valued function $f \in L^2(M)$ defined on a compact Riemannian manifold M can be expanded in terms of the eigenfunctions



(a) Eigenfunction 1. (b) Eigenfunction 2. (c) Eigenfunction 3.

(d) Eigenfunction 1. (e) Eigenfunction 2. (f) Eigenfunction 3. (g) Eigenfunction 4.

Figure 2.2.: Eigenfunctions for Neumann boundary conditions for a square (top) and a square with a tail (bottom). Picture is taken from [Reu+09b].

with the respective eigenvalues $0 \leq \lambda_1 \leq \lambda_2 \leq ...$ as:

$$f = \sum_{j=1}^{\infty} a_j \phi_j \tag{2.5}$$

for some coefficients $a_j \in \mathbb{R}$. The coefficients provide a canonical parametrisation of functions defined on the surface. Truncating the above infinite expansion results in a smoother function. Figure 2.3 illustrates this smoothness by showing how a chosen number of eigenfunctions affects the finer details of the Dragon model when its vertex coordinates are reconstructed using 100, 200, 300, 500, and 900 eigenfunctions, respectively.

2.1.4. Discretisation

A triangular mesh (V, E, F) is a commonly used discretisation of a 3D shape, where $V = \{\mathbf{v}_1, ..., \mathbf{v}_n\}$ is the vertex set, $E \subseteq V \times V$ is the set of edges between the vertices, and $F \subset V^3$ is the set of faces (or triangles in this case). A function f on the triangular mesh is represented as the vector $\mathbf{f} = (f(\mathbf{v}_1), ..., f(\mathbf{v}_n))$. The Helmholtz equation (Equation 2.3) can be written in the matrix form as:

$$\mathbf{L\Phi} = \mathbf{\Phi\Lambda} \tag{2.6}$$

where $\mathbf{\Lambda} = \text{diag}(\lambda_1, ..., \lambda_n)$ contains the eigenvalues, $\mathbf{\Phi} = (\phi_1, ..., \phi_n)$ contains the eigenfunctions arranged as columns, and the discrete Laplace-Beltrami operator is

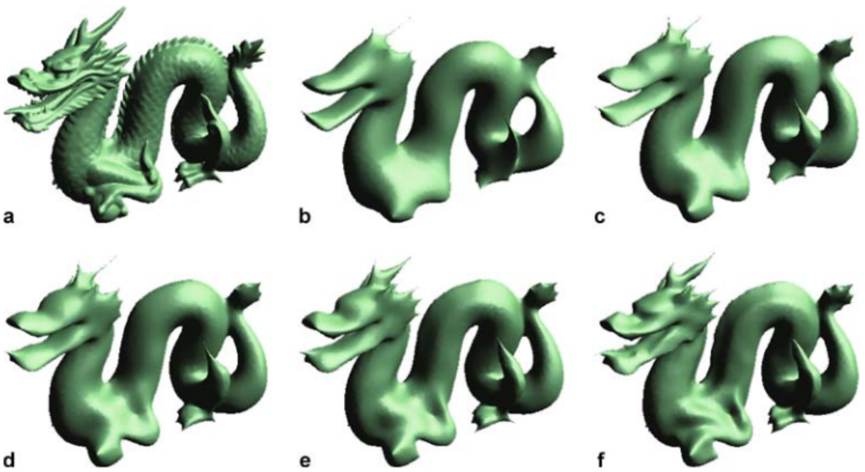


Figure 2.3.: The results of reconstructing the original Dragon model's (**a**) vertex coordinates using different number of eigenfunctions are shown in (**b - f**). The number of eigenfunctions used are 100, 200, 300, 500 and 900, respectively. This picture is taken from [RCG08].

represented by the $n \times n$ matrix **L** defined as [Reu+09a]:

$$\mathbf{L} = \mathbf{A}^{-1}\mathbf{S}. \tag{2.7}$$

The *mass matrix* **A** is diagonal ($\mathbf{A} = \text{diag}(a_1, ...a_n)$) and its entries represent the vertex weights, while the *stiffness matrix* **S** is symmetric with entries that express the edge weights. The matrix **S** is defined as $\mathbf{S} = \mathbf{D} - \mathbf{W}$ with $\mathbf{W} = (w_{i,j})$ being the weighted adjacency matrix, and $\mathbf{D} = \text{diag}(d_1, ...d_n)$ having the elements $d_i = \sum_{j \in N(i)} w_{i,j}$. Using the cotangent discretisation, the weight of the edge ij is computed in the following manner:

$$w_{i,j} = \begin{cases} (\cot \alpha_{i,j} + \cot \beta_{i,j})/2 & \text{if edge } ij \in E \\ 0 & \text{otherwise} \end{cases} \tag{2.8}$$

where $\alpha_{i,j}$, $\beta_{i,j}$ are the angles opposite the edge ij. The vertex weight a_i is defined as the local area element of $\mathbf{v}_i$ [Mey+02]:

$$a_i = \frac{1}{8} \sum_{j \in N(i)} (\cot \alpha_{i,j} + \cot \beta_{i,j}) \left\| \mathbf{v}_i - \mathbf{v}_j \right\| \tag{2.9}$$

Figure 2.4 shows the vertex $\mathbf{v}_i$ and its one-ring neighbourhood $N(i)$. The local area element a_i is displayed in grey and the edge ij in red.

The eigendecomposition problem can be posed as the minimisation:

$$\min_{\mathbf{\Phi}} \text{off}(\mathbf{\Phi}^T \mathbf{W} \mathbf{\Phi}) \quad \text{s.t. } \mathbf{\Phi}^T \mathbf{D} \mathbf{\Phi} = \mathbf{I} \tag{2.10}$$

of the sum of squared off-diagonal elements, $\text{off}(\mathbf{A}) = \sum_{i \neq j} a_{ij}^2$. This optimisation problem can be solved iteratively using for example the Jacobi eigenvalue algorithm.

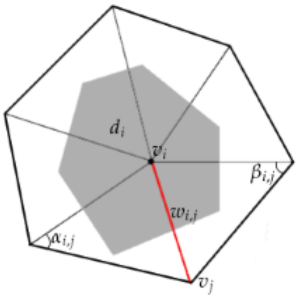


Figure 2.4.: The vertex $\mathbf{v}_i$ and its one-ring neighbourhood. The local area element of the vertex $\mathbf{v}_i$ is marked in grey. The edge ij is drawn in red and the $\alpha_{i,j}$ and $\beta_{i,j}$ angles used to compute its edge weight are also shown.

The discrete eigenfunctions are orthogonal with respect to the weighted inner product. Therefore, $\mathbf{\Phi}^T\mathbf{A}\mathbf{\Phi} = \mathbf{I}$, where $\mathbf{A}$ is previously defined mass matrix holding the vertex weights and $\mathbf{I}$ is the identity.

2.2. Functional Maps

The functional maps approach [Ovs+12] is a flexible framework for representing maps between shapes.

Let M and N denote two shapes discretised as triangular meshes with m and n vertices respectively. A map $T : M \to N$ between them induces a linear map $T_F : L^2(M) \to L^2(N)$ between their function spaces. The functional map T_F takes a scalar function $f \in L^2(M)$ and transfers it to $g \in L^2(N)$ by composition, as in $g = T_F(f)$. The original mapping T can be recovered from T_F since $T_F = f \circ T^{-1}$.

Let $m \times k$ matrix $\mathbf{\Phi} = (\phi_1, ..., \phi_k)$ and $n \times l$ matrix $\mathbf{\Psi} = (\psi_1, ..., \psi_l)$ be the truncated bases of M and N respectively. As previously mentioned in Section 2.1.3, the eigenfunctions ϕ_i of the Laplace-Beltrami operator can serve as bases, and consequently, any function $f : M \to \mathbb{R}$ can be expressed as $f = \sum_{i=1}^k a_i\phi_i$ using the first k eigenfunctions. Similarly, the basis functions ϕ_i once mapped onto the shape N are represented in terms of its basis $\mathbf{\Psi}$ as $T_F(\phi_i) = \sum_{j=1}^l c_{ij}\psi_j$. Then a linear functional mapping T_F is defined as:

$$\begin{aligned} T_F(f) = T_F\left(\sum_{i=1}^k a_i\phi_i\right) &= \sum_{i=1}^k a_i T_F\left(\phi_i\right) \\ = \sum_{i=1}^k a_i \sum_{j=1}^l c_{ij}\psi_j &= \sum_{j=1}^l \sum_{i=1}^k a_i c_{ij}\psi_j \end{aligned} \tag{2.11}$$

By projecting the discretized versions of functions f and $g = T_F(f)$ onto the respective shape's basis (i.e. $\mathbf{a} = (\mathbf{\Phi}^T\mathbf{A}_M)\mathbf{f}$ and $\mathbf{b} = (\mathbf{\Psi}^T\mathbf{A}_N)\mathbf{g}$), the functions can be represented as their expansion coefficients $\mathbf{a} = (a_1, ..., a_k)$ and $\mathbf{b} = (b_1, ..., b_l)$. Subsequently, the Equation 2.11 is equivalent to $b_j = \sum_{i=1}^k a_i c_{ij}$ or in matrix form $\mathbf{b} = \mathbf{C}\mathbf{a}$. The $l \times k$ matrix

$\mathbf{C} = (c_{ij})$ translates the expansion coefficients from the basis $\mathbf{\Phi}$ to the basis $\mathbf{\Psi}$ and it fully encodes the T_F mapping. Therefore, the original mapping T can be recovered if $\mathbf{C}$ is known.

The matrix $\mathbf{C}$ is a solution to the following system of equations:

$$\mathbf{B} = \mathbf{CA} \tag{2.12}$$

where matrices $\mathbf{A}$ and $\mathbf{B}$ hold functional correspondences (also known as constraints) projected onto the respective shape's basis. Namely, the i-th column of $\mathbf{A}$ represents a function on M that has a corresponding function on N represented by the i-th column of $\mathbf{B}$. The matrices $\mathbf{A}$ and $\mathbf{B}$ both have q columns, where q marks the number of constraints used. The functional correspondences usually come from shape descriptors, or points and regions indicator functions. The ones used in this work are presented in the following section. The Equation 2.12 is solved by optimising the least-squares problem:

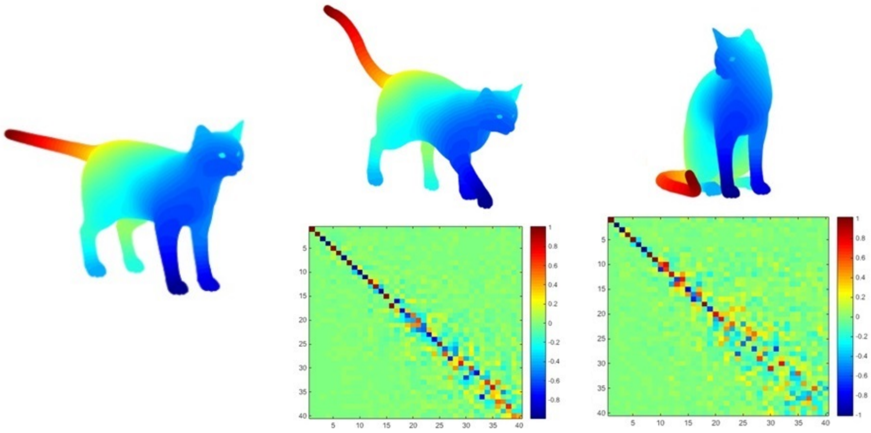
$$\min_{\mathbf{C}} \|\mathbf{CA} - \mathbf{B}\|_{\mathrm{F}}^2 \tag{2.13}$$

where the Frobenius norm of a matrix $\mathbf{X}$ is defined as $\|\mathbf{X}\|_{\mathrm{F}} = \sqrt{\sum_{ij} x_{ij}^2}$. Note that the size of matrix $\mathbf{C}$ depends only on the number of basis functions used, not on the size of the meshes.

If two shapes M and N are isometric and their eigenvalues are non-repeating, then the corresponding eigenfunctions of each shape are perfectly compatible. Consequently, $\mathbf{C}$ is a diagonal matrix with $c_{ii} = \pm 1$ and zeros everywhere else. A near-isometric map between two shapes is more commonly encountered in practice. The farther from isometry the shapes are, the more elements of $\mathbf{C}$ spread off the diagonal in a funnel-shaped manner. This is due to the eigenfunctions not being perfectly compatible between shapes which is more prominent in higher frequencies. However, most elements of $\mathbf{C}$ are still close to zero. Figure 2.5 illustrates this by showing the ground-truth maps between the source shape (a) and two target shapes (b, c) where the second target (c) is more deformed relative to the source. It can be seen that the elements of matrix $\mathbf{C}$ between the source and the more deformed target shape display more deviation from the diagonal.

2.2.1. Point-to-Point Conversion

The point-wise map between shapes can be recovered by the use of point indicator functions. Given a point $x \in M$ and its indicator function f (value 1 at x and 0 elsewhere), the corresponding point $y \in N$ is the maximum of the function g obtained by applying T_F mapping to f (i.e. $g = T_F(f)$). However, this method of recovering point-wise maps has $\mathcal{O}(mn)$ complexity and can be impractical when dealing with meshes with large number of vertices. A more efficient method is to consider for every point of $\mathbf{C\Phi}^T$ (represented as a column) its nearest neighbour in $\mathbf{\Psi}^T$, which requires $\mathcal{O}(m \log n + n \log m)$ using a fast search algorithm. *Post-processing iterative refinement* employs this more efficient technique to refine a given matrix $\mathbf{C}$ in order to make it



(a) cat0 (source) (b) cat10 (target 1) (c) cat6 (target 2)

Figure 2.5.: The ground-truth correspondence matrix **C** between the shapes cat0 and cat10 (b), and cat0 and cat6 (c).

produce a more optimal point-wise map. The steps of the refinement are as follows: first, for each column x of $\mathbf{C}\mathbf{\Phi}^T$ find the closest $\tilde{x}$ in $\mathbf{\Psi}^T$; second, find an orthonormal **C** minimising $\sum_i \|Cx_i - \tilde{x}\|$; third, repeat until convergence. Note that **C** is required to be square for this procedure, so the number of eigenfunctions that make up $\mathbf{\Phi}$ and $\mathbf{\Psi}$ basis are the same. Also, constraining **C** to be orthonormal results in the area-preserving maps being computed.

2.2.2. Regularisation of Correspondence Matrix

To help induce the diagonal structure of **C**, Pokrass et al. [Pok+12] use an ℓ_1-norm regularisation of **C** multiplied with the weight matrix **W** whose diagonal elements are zero. They also show that a quality shape correspondence can be obtained using only indicator functions of repeatable regions detected on the shapes without prior knowledge of the correct ordering of said regions. The regions are brought into correspondence by the permutation matrix $\mathbf{\Pi}$ during the optimisation phase:

$$\min_{\mathbf{C},\mathbf{\Pi}} \|\mathbf{CA} - \mathbf{B\Pi}\|_{\mathrm{F}}^2 + \lambda \|\mathbf{W} \odot \mathbf{C}\|_1 \tag{2.14}$$

where λ controls the importance of the ℓ_1-term and $\odot$ denotes element-wise matrix multiplication. It should be noted that post-processing iterative refinement provides a significant help in obtaining a good point-to-point shape correspondence from such little information as only region indicator functions.

Kovnatsky et al. [Kov+15] look at the problem of finding the functional map $\mathbf{T} = \mathbf{\Psi C \Phi}^T$ from a matrix completion point of view. They search for a matrix that explains the data in a "simple" way. That is, the matrix should result in a smooth correspondence, meaning that if two points are close on shape M their respective corresponding functions are similar. Likewise, for close points on shape N. Also, the matrix should be sparse, which is achieved by the nuclear and ℓ_1-norm regularisations. Using the decomposition $\mathbf{T} = \mathbf{U}\mathbf{V}^T$ and then approximating $\mathbf{U} \approx \mathbf{\Psi}_{k'}\mathbf{A}$ and $\mathbf{V} \approx \mathbf{\Phi}_{k'}\mathbf{B}$, where matrices $\mathbf{A}$ and $\mathbf{B}$ of the expansion coefficients are of size $k' \times k$, the following minimisation is proposed:

$$\begin{aligned}\min_{\mathbf{A},\mathbf{B}} \left\|\mathbf{A}\mathbf{B}^T\mathbf{\Phi}_{k'}^T\mathbf{F} - \mathbf{\Psi}_{k'}^T\mathbf{G}\right\|_{\mathrm{F}}^2 &+ \mu_1 \mathrm{tr}(\mathbf{A}\mathbf{B}^T\mathbf{L}_M\mathbf{B}\mathbf{A}^T) + \mu_2(\mathbf{B}\mathbf{A}^T\mathbf{L}_N\mathbf{A}\mathbf{B}^T) \\ &+\mu_3 \left\|\mathbf{\Psi}_{k'}\mathbf{A}\mathbf{B}^T\mathbf{\Phi}_{k'}^T\right\|_1 + \frac{\mu_4}{2}(\|\mathbf{\Psi}_{k'}\mathbf{A}\|_{\mathrm{F}}^2 + \|\mathbf{\Phi}_{k'}\mathbf{B}\|_{\mathrm{F}}^2)\end{aligned} \tag{2.15}$$

Matrices $\mathbf{F}$ and $\mathbf{G}$ of size $m \times q$ and $n \times q$ respectively hold q corresponding functions. The smoothness of the correspondence functions is achieved by the μ_1- and μ_2-term. The nuclear norm of a matrix (sum of its singular values) is often used to obtain a solution with low rank, and it can be written in the form presented by the μ_4-term.

2.3. Shape Descriptors

The shape descriptors used in this work are the ones based on the eigendecomposition of the Laplace-Beltrami operator associated with the shape.

2.3.1. Heat Kernel Signature

The Heat kernel signature (HKS) [SOG09] is a point descriptor based on the heat diffusion process on the shape which is governed by the Heat Equation:

$$\Delta u(x,t) = -\frac{\partial}{\partial t}u(x,t). \tag{2.16}$$

The heat kernel $k_t(x,x)$ represents the solution to the heat equation. It describes how much heat remains at point x after some time t, given that at $t = 0$ all the heat energy was concentrated at x. The eigendecomposition of heat kernel is expressed as:

$$k_t(x,x) = \sum_i^k e^{-\lambda_i t}\phi_i^2(x) \tag{2.17}$$

where λ_i and ϕ_i are the first k eigenvalues and the eigenfunction of the Laplace-Beltrami operator. The heat kernel function $k_t(x,x)$ is sampled at a certain set of times $t_1, ..., t_n$, so HKS at point x is defined as a vector of heat kernels:

$$HKS(x) = (k_{t_1}(x,x), ..., k_{t_n}(x,x)). \tag{2.18}$$

HKS is invariant under isometric deformation as it is computed using only intrinsic properties of the manifold. It is robust to small perturbations on the shape mesh. It also has a multi-scale property, meaning that for small t, HKS captures the local properties of the shape, whereas for large t, it captures the global structure.

2.3.2. Wave Kernel Signature

The Wave Kernel Signature (WKS) [ASC11] is also a point descriptor and like HKS inspired by physical phenomenon. Rather than observing the heat diffusion, what is considered here is the probability of a quantum particle to be located at a certain point on a shape. The movement of a quantum particle on the shape is governed by its wave function $\psi(x,t)$ which is a solution of the Schrödinger equation:

$$\frac{\partial}{\partial t}\psi(x,t) = i\Delta\psi(x,t). \tag{2.19}$$

Given an initial energy distribution f_E of a quantum particle, where energy is related to the eigenvalues, the wave function of the particle can be expressed in the spectral domain as:

$$\psi_E(x,t) = \sum_{k=1}^{\infty} e^{iE_K t}\phi_k(x) f_E(\lambda_k). \tag{2.20}$$

The probability to measure the particle at a point x is then $|\psi_E(x,t)|^2$. Subsequently, the average probability (over time) to measure a particle at the location x is defined as:

$$p(x) = \lim_{T\to\infty}\frac{1}{T}\int_0^T |\psi_E(x,t)|^2 = \sum_{k=0}^{\infty}\phi_k(x)^2 f_E(\lambda_k)^2. \tag{2.21}$$

Note that the time parameter has been replaced by the eigenvalues of the Laplace-Beltrami operator which capture an intrinsic notion of scale of the shape. Therefore, different choices of energy distributions f_E characterise the shape properties at different scales.

The following family of normal energy distributions centred at some mean $\log e$ with variance σ^2 are considered:

$$f_E(\lambda) \propto \exp(-\frac{(\log e - \log\lambda)^2}{2\sigma^2}). \tag{2.22}$$

Using logarithmically sampled energy E at $e_1, ..., e_n$, the WKS at a point x is defined as a vector:

$$WKS(x) = (p_{e_1}(x), ..., p_{e_n}(x)) \tag{2.23}$$

where each $p_e(x)$ is the probability of observing a quantum particle with initial energy distribution $f_e(\lambda)$ at point x.

The authors show both theoretically and in quantitative experiments that WKS is more discriminative than HKS and therefore performs better in shape matching.

2.3.3. Consensus Segmentation of Deformable Shapes

Rodolà et al. [RBC14] proposed a method that detects repeatable regions of 3D deformable shapes by extracting a consensus segmentation from an ensemble of candidate segmentations of the shape itself. The method is motivated by the observation that consensus segmentation tends to be more stable against non-isometric deformations.

The ensemble of candidate segmentations is created by running a clustering algorithm on the Global Point Signatures (GPS) of all the points on the shape. GPS of a point p is an infinite dimensional vector of scaled eigenfunctions:

$$g(p) = \left(\frac{1}{\sqrt{\lambda_1}} \phi_1(p), \frac{1}{\sqrt{\lambda_2}} \phi_2(p), \frac{1}{\sqrt{\lambda_3}} \phi_3(p), ... \right). \tag{2.24}$$

GPS characterises points intrinsically and is therefore a good choice for producing isometry-invariant segmentations. The goal of clustering is to assign each $g(p)$ to its nearest cluster centre. That is, given the partition function $C : g(p) \to i$, which assigns a cluster index $i \in 1, ..., k$ to every point on the shape, the goal is to find such a partition C that minimises the distances between all n points and their assigned cluster centres c_l. How well a specific partition C minimises said distances is measured in the following way:

$$w(C) = \frac{1}{n} \sum_{l=1}^{k} \sum_{C(g(p_i))=l} \|g(p_i) - c_l\|^2 \, \mathrm{d}a(p_i), \tag{2.25}$$

where $\mathrm{d}a(p_i)$ is the local area element of point p_i, c_l is the mean of the l-th cluster based on the current partition C, and $\|\cdot\|$ is the Euclidean distance in $\mathbb{R}^\infty$ induced by the inner product defined as:

$$\langle g(p), g(q) \rangle = \sum_{1}^{\infty} \frac{1}{\lambda_k} \phi_k(p) \phi_k(q). \tag{2.26}$$

The authors use the k-means clustering algorithm to minimise Equation 2.25. Starting from some initial seeds $\{c_l\}_{l=1}^{k}$, the algorithm assigns every $g(p_i)$ to its nearest centre c_l. Then, the position of centres is recomputed and the assignment is performed again. This is repeated until a certain stopping criteria is reach. The variety of the segmentations in an ensemble produced in this way come from choosing different initial seeds as well as the fact that the number of minimisers for Equation 2.25 could be infinite.

An ensemble of m candidate segmentations over a shape consisting of n triangles can be written as a matrix $\mathbf{E} \in \{1, .., n\}^{m \times n}$. Each entry E_{ui} holds the segment index of the u-th candidate segmentation that the triangle t_i belongs to. The rows represent individual segmentations and are denoted by $\mathbf{E}_{u*}$. Given a maximum number of segments k, a consensus segmentation is a segmentation $\mathbf{x}^* \in \{1, ..., k\}^n$ that is as close as possible to all segmentations in $\mathbf{E}$. This can be expressed formally as:

$$\mathbf{x}^* \in \arg\min_{\mathbf{x}} \sum_{u=1}^{m} d^2(\mathbf{E}_{u*}, \mathbf{x}) \tag{2.27}$$

where d measures the consistency of the segmentations:

$$d^2(\mathbf{x}, \mathbf{z}) = \sum_{i,j \in 1,...,n} w_{ij} (\mathbb{1}_{x_i = x_j} - \mathbb{1}_{z_i = z_j})^2. \tag{2.28}$$

The indicator function $\mathbb{1}_P$ returns 1 if property P is true, and 0 otherwise. Essentially, d is the weighted count of how many triangles do not belong to the same segment across the segmentations. The w_ij's are the product of the areas of the respective triangles.

The distance measure *d* in Equation 2.27 has quadratic complexity and depends on the number of triangles in the shape mesh. A detailed discussion on how these complexity issues are overcome can be found in the paper [RBC14].

Part II.

2D-to-3D Shape Matching via Low-Rank Functional Correspondence

3. Proposed Approach

The focus of this work is computing a point-to-point matching between organic 2D and 3D shapes, such as animals and humans, using functional correspondences. More precisely, the matchings being considered are between the 2D side and front views of a shape, and the semantically similar 3D shapes. Organic shapes usually have a bilateral (two-sided) symmetry. The symmetry helps in choosing the part of the 3D shape to be targeted in the matching process when the source of the 2D shape is a side view. In case of the front views, no symmetry cue is available to help determine the part of the 3D shape to be matched. Instead, the full shape is being considered as a target. Therefore, matching of the front views is restricted to the human shapes, where the "depth" of the shape is relatively small when observed from the front. The 2D shapes are represented as 3D triangular meshes where the z coordinate of all vertices is set to zero. Figure 3.1 shows a side view of a cat and a front view of a man and their respective triangular mesh.

The desired shape matching algorithm should be modelled as a convex problem so that a computed solution can be guaranteed to be a globally optimal one. Another advantage of convexity is that efficient and well-known methods are available for the optimisation process. The algorithm should also be robust to noise in order to work with the shapes and deformation models found in real world. The following sections first provide the motivation for the proposed shape matching algorithm by looking at the behaviour of eigenfunctions across 2D and 3D shapes. Then, the optimisation problem and how to solve it is presented.

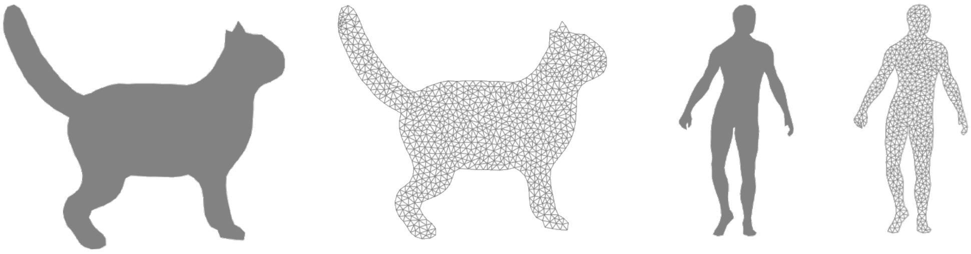


Figure 3.1.: A side view of a cat and a front view of a man along with their respective triangular mesh.

3.1. Eigenfunctions in 2D-to-3D Shape Matching

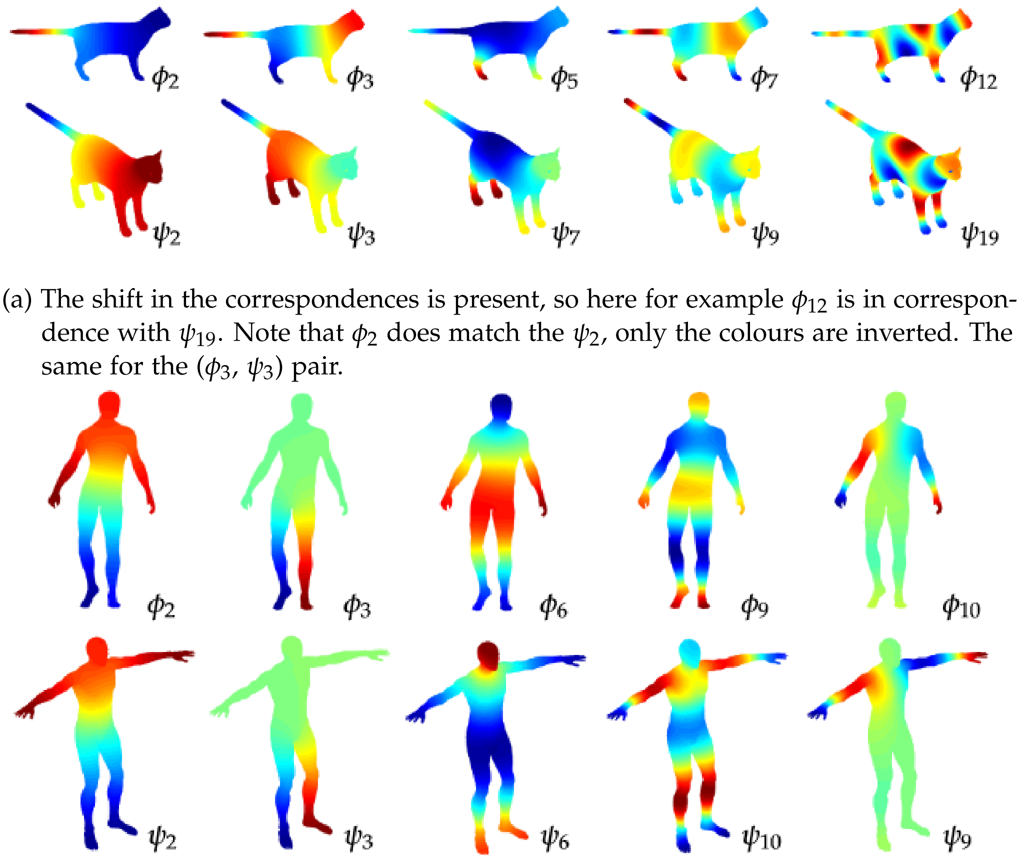
The success of shape matching via functional correspondence depends on the compatibility of shapes' eigenfunctions. Consistent behaviour of eigenfunctions across different shapes is only guaranteed if the shapes are perfectly isometric. In the case of 2D-to-3D matching, the shapes are far from isometric. They differ in their curvature (one is flat, the other is not), and in which features are present in the shape (only those seen from the side or front view, as opposed to the whole shape). Nevertheless, in practice the eigenfunctions with the smallest eigenvalues still behave in a similar way across 2D and 3D shapes if Neumann boundary condition is used to compute them.

This behaviour can be explained by the background information presented in Section 2.1. To summarise, eigenfunctions corresponding to lower frequencies are more robust to shape changes and noise. When Neumann boundary condition is used, the eigenfunctions detect the differences in shapes faster. More importantly, when a new part is added or removed from the base of the shape, the eigenfunctions on the base still correspond with each other. These global geometrical changes made to the shape do not have hazardous effect on the eigenfunctions. However, the correspondence of the eigenfunctions may be shifted, as the addition of new parts can bring about completely new eigenfunctions. Figure 3.2a shows the shift in the eigenfunction correspondence observed between the 2D view of a cat and a 3D cat. If the 2D view does not coincide with the 3D shape's intrinsic symmetry, like in the example in Figure 3.2b, the shift in correspondences is not present.

As previously discussed in Section 2.2, the correspondence matrix $\mathbf{C}$ is sparse and has non-zero elements only along its diagonal when the eigenfunctions of the two shapes being matched are perfectly compatible. This perfect compatibility does not occur in practice. Its absence is noticeable in 2D-to-3D shape matching case as illustrated by Figure 3.3, which shows the matrix $\mathbf{C}$ obtained by using the original functional maps approach (without the iterative refinement step), and the point-wise correspondence it produces. When the 2D shape is matched to the whole 3D shape (Figure 3.3a), the largest entries of $\mathbf{C}$ are on the main diagonal due to it being likely that the i-th eigenfunction on the 2D shape is in correspondence with the i-th eigenfunction on the 3D shape. By contrast, when the 2D shape is matched to one side of the 3D shape (Figure 3.3b), the largest entries of $\mathbf{C}$ form a slanted diagonal structure as a consequence of the shift in the eigenfunction correspondence. In other words, when looking at the pairs of corresponding eigenfunctions under partiality, the indices of $\mathbf{\Phi}$ increase one at a time, while the indices of $\mathbf{\Psi}$ increase faster because some are skipped (Figure 3.2a). These uneven corresponding index sequences result in the slanted diagonal structure of $\mathbf{C}$.

3.2. Problem Definition of 2D-to-3D Shape Matching

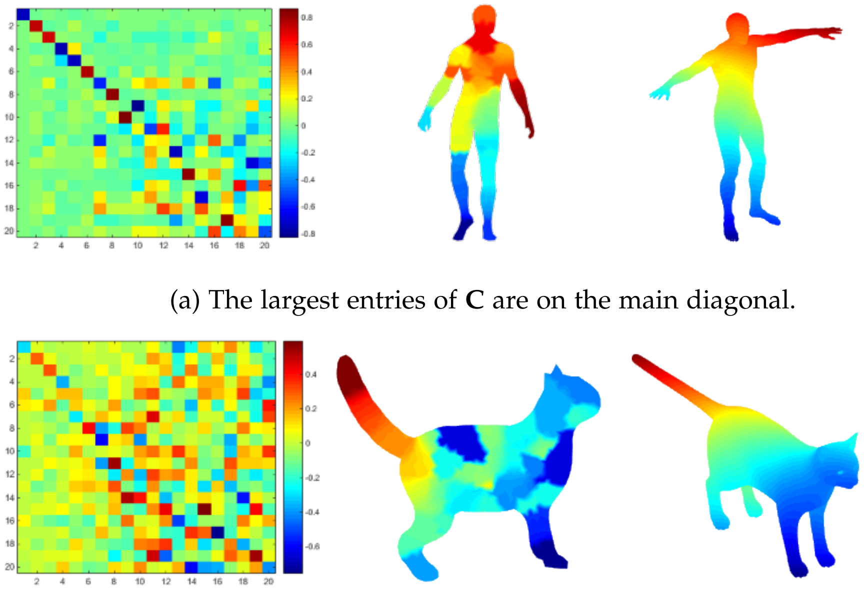
Optimising only the data term $\|\mathbf{CA} - \mathbf{B}\|_F^2$ is not able to cope with the noise introduces by the not perfectly compatible eigenfunctions. Therefore, the following regularised



(a) The shift in the correspondences is present, so here for example ϕ_{12} is in correspondence with ψ_{19}. Note that ϕ_2 does match the ψ_2, only the colours are inverted. The same for the (ϕ_3, ψ_3) pair.

(b) The shift in the correspondences is not present when the 2D view does not coincide with one half of the intrinsic symmetry of the 3D shape. Due to numerical instabilities, the correspondence of the 9^{th} and 10^{th} eigenfunctions are switched.

Figure 3.2.: Corresponding eigenfunctions of (a) the 2D side view of a cat and the 3D cat shape, and (b) the front view of a human and the 3D human shape.



(a) The largest entries of **C** are on the main diagonal.

(b) The largest entries of **C** form a slanted diagonal, due to the shift in the eigenfunction correspondence.

Figure 3.3.: Example of the point-wise correspondence between 2D and 3D human (a) and 2D and 3D cat shapes, and the matrix **C** from which the correspondence is derived. **C** is computed using only the data term $\|\mathbf{CA} - \mathbf{B}\|_F^2$.

optimisation problem is considered:

$$\min_{\mathbf{C}} \|\mathbf{CA} - \mathbf{B}\|_F^2 + \gamma\rho(\mathbf{C}) \tag{3.1}$$

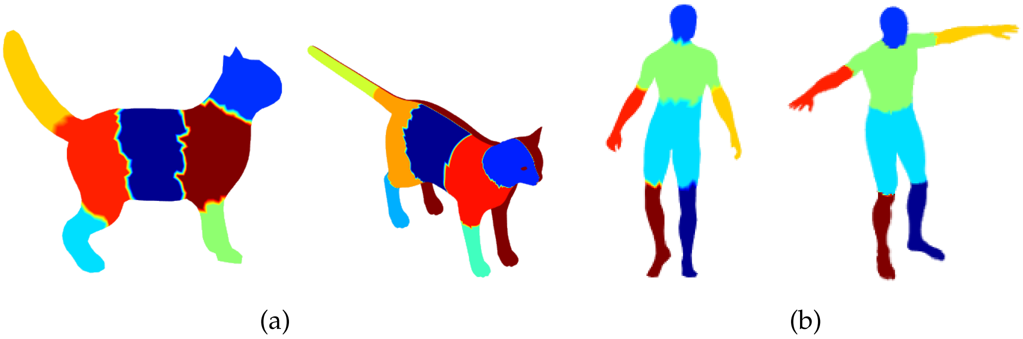
where γ controls the relative importance of the two terms, and $\rho(\mathbf{C})$ is a regularisation term for matrix **C** introduced to give preference to a solution that is sparse and diagonally dominant. Three different regularisations are proposed based on the weighted ℓ_1- and ℓ_2-norm, and the nuclear norm of **C**:

$$\rho(\mathbf{C}) = \|\mathbf{W} \odot \mathbf{C}\|_F^2\,; \quad \rho(\mathbf{C}) = \|\mathbf{W} \odot \mathbf{C}\|_1\,; \quad \rho(\mathbf{C}) = \|\mathbf{C}\|_*\,. \tag{3.2}$$

As before, $\odot$ denotes the element-wise matrix product. The weight matrix **W** helps to obtain the desired diagonal structure of the correspondence matrix **C** by having zero values where the diagonal should be. The effect these three regularisations have on **C** can be seen in Figure 4.4.

3.2.1. Data Term

The shape descriptors presented in Section 2.3 are used as the functional correspondences represented by **A** and **B** matrices. The surface area of a 2D and 3D shape being matched


(a)

(b)

Figure 3.4.: Example of the consistent segmentation across 2D and 3D shapes. The left side of the 3D cat in (a) is not represented by any functional correspondence.

are usually not the same. Therefore, the spectrum of the 3D shape is scaled by $\frac{1}{a^2}$, where $a = area(2D)/area(3D)$, and this normalized spectrum is used in the HKS and WKS computation. The consensus segmentation method (Section 2.3.3) is chosen for its ability to detect repeatable regions across 2D and 3D shapes. In practice, a different value for the parameter k (maximum number of regions) has to be specified for the method to produce the same number of regions on a given pair of 2D and 3D shapes. The obtained regions are manually put into correspondence and are represented as binary indicator functions. The values of HKS and WKS are segmented according to the computed regions. That is, every vector component of HKS and WKS, in Equation 2.18 and 2.23 respectively, is multiplied by every region indicator function. As a result, the HKS and WKS contained in region i on the shape M are in functional correspondence with the HKS and WKS contained in region i on the shape N. If there are x detected regions and y components in HKS and WKS each, then the total number of functional constraints used is $x + 2(xy)$. Figure 3.4 shows a consistent segmentation between a pair of cat (a) and human shapes (b).

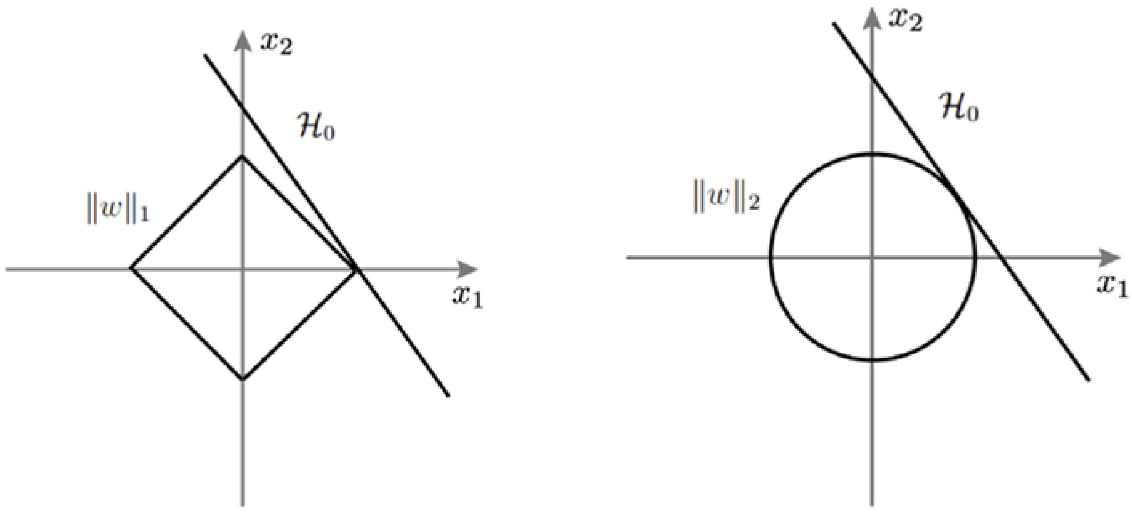
3.2.2. Weighted ℓ_2-Regularisation Term

$$\rho(\mathbf{C}) = \|\mathbf{W} \odot \mathbf{C}\|_F^2 \tag{3.3}$$

Probably the most common norm regularisation is the ℓ_2-norm, which given a vector $\mathbf{x}$ is defined as $\|\mathbf{x}\| = \sqrt{\sum_i x_i^2}$. When applied to a matrix, the ℓ_2-norm is the Frobenius norm. The ℓ_2-norm puts greater weight on the large entries of the matrix and is therefore more interested in decreasing or shrinking them than the small entries in the minimisation process. As a result, the ℓ_2-regularised solution tends to have relatively fewer large entries because they incur a much larger penalty.

3.2.3. Weighted ℓ_1-Regularisation Term

$$\rho(\mathbf{C}) = \|\mathbf{W} \odot \mathbf{C}\|_1 \tag{3.4}$$



(a) ℓ_1-regularisation

(b) ℓ_2-regularisation

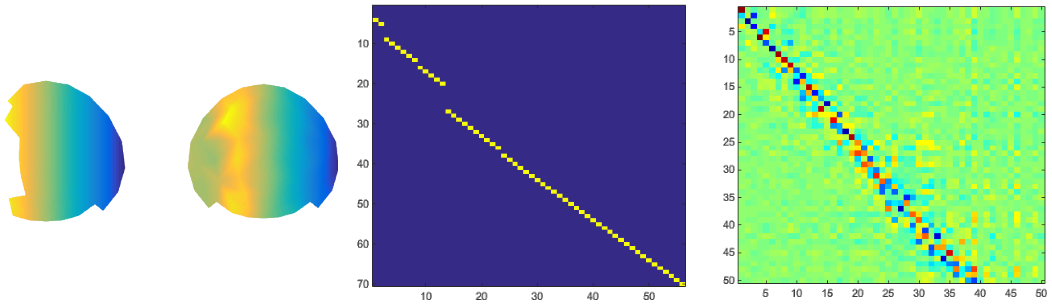
Figure 3.5.: The difference in behaviour of ℓ_1-regularisation and ℓ_2-regularisation when searching for the least costly path to the line H_0 in 2D space. In the case of ℓ_1-penalty, the best way is simply to step along the x_1 axis, and make zero steps along the x_2 axis. Using the ℓ_2-penalty, the path of the minimum cost uses both x_1 and x_2 directions.

The ℓ_1-norm of a vector $\mathbf{x}$ measures the sum of the absolute values of its entries, $\|\mathbf{x}\|_1 = \sum_i |x_i|$. The main motivation for using the ℓ_1-regularisation in the optimisation problems is that it induces sparsity, i.e. the produced solution has relatively few non-zero entries. By contrast, the ℓ_2-regularisation does not provide enough incentives to make the entries equal to zero, but rather shrinks them towards it in order to minimise the ℓ_2-norm. The explanation why the ℓ_1-regularisation induces sparsity can be explained by looking at the geometrical difference of the level set of the ℓ_1- and ℓ_2-norm. A level set is the set of points equidistant to the origin. Figure 3.5 shows that the level set of the ℓ_1-norm is diamond shaped (a), whereas the level set of the ℓ_2-norm is a circle (b). The minimum ℓ_1 penalty of walking to the line H_0 is achieved if the direction of x_1 axis is followed. No steps are made in the direction of the x_2 axis, so that component is set to zero. On the other hand, if the ℓ_2-penalty is applied, it is not necessary to strictly keep to the directions of the axes to minimise the cost.

3.2.4. Nuclear Norm Regularisation Term

$$\rho(\mathbf{C}) = \|\mathbf{C}\|_* \tag{3.5}$$

The nuclear (or trace) norm $\|\mathbf{X}\|_*$ of the matrix $\mathbf{X}$ with rank r is the ℓ_1-norm of the singular values of $\mathbf{X}$, $\|\mathbf{X}\|_* = \mathrm{tr}(\sqrt{\mathbf{X}^T\mathbf{X}}) = \sum_{i=1}^{r} \sigma_i$. The nuclear norm encourages sparsity in the vector of singular values, which has the effect of making the matrix be low rank.



(a) Two shapes matched. (b) Permutation matrix. (c) Matrix **C**.

Figure 3.6.: An example of partial shape matching and a low rank **C** it induces.

.

When looking at matching a 2D shape to one side of a 3D shape, the slanted diagonal structure is present in the matrix **C** (Figure 3.3b), which means that **C** should be low rank. In fact, low rank is a consequence of partiality. Figure 3.6a shows an example of partial matching between two small shapes. Some points on the right shape do not have a correspondence on the shape shown on the left. For this reason, a permutation matrix representing the shape correspondence in the standard basis has some empty rows (Figure 3.6b). When moving to the functional map representation, an orthogonal change of basis is performed from the standard to the eigenbasis, and this action preserves the zero rows of the permutation matrix resulting in a low rank **C** (Figure 3.6c).

In the case when the characteristic structure of **C** is on the main diagonal, a low rank solution can also be advantageous as it disregards the tail of the diagonal which is sometimes overly spread out to be able to hold quality information.

3.2.5. Weight Matrix

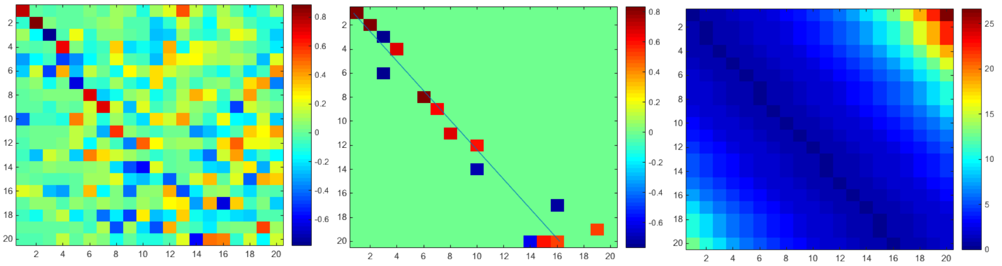


Figure 3.7.: Computation of weight matrix **C**. The diagonal structure (middle) is recovered from the correspondence matrix (left). A straight line is fitted through the recovered structure to locate where the zero elements of **C** should be (right).

The weight matrix **W** is computed based on the **C** that is obtained using the original

```
   Data: $\mathbf{\Phi}, \mathbf{y}, s$
   Result: $s$-sparse approximation $\mathbf{a}$ of $\mathbf{x}$
1  $\mathbf{r} = \mathbf{y}$
2  $\mathbf{x}^0 = \mathbf{0}$
3  repeat
4      $\mathbf{u} = \mathbf{\Phi}^T \mathbf{r}$
5      $\Omega = \operatorname{supp}(\mathbf{u}_{2s})$
6      $T = \Omega \cup \operatorname{supp}(\mathbf{a}^{k-1})$
7      $\mathbf{b} = \arg\min_{\mathbf{b}} \left\| \mathbf{\Phi}^T \mathbf{\Phi} \mathbf{b} - \mathbf{\Phi}^T \mathbf{y} \right\|_2^2, \quad \operatorname{supp}(\mathbf{b}) \subseteq T$
8      $\mathbf{a}^k = \mathbf{b}_s$
9      $\mathbf{r} = \mathbf{y} - \mathbf{\Phi} \mathbf{a}^k$
10 until a stopping criterion is met;
```

Algorithm 1: CoSaMP algorithm

functional maps approach. Matrix **C** can be seen as a compressible signal which can be approximated by an s-sparse one. **C** is compressible as it has a few large coefficients and many small ones. To recover the signal, the CoSaMP [TN08] algorithm is used.

CoSaMP is a greedy iterative method for reconstructing an s-signal from compressive samples $\mathbf{y} = \mathbf{\Phi x}$, where $\mathbf{x}$ is a target signal, and $\mathbf{\Phi}$ is a sampling matrix. In every iteration, the following steps are performed (Algorithm 1): first, a proxy of the residual is formed (line 4) and $2s$ largest entries are identified (line 5); these entries are united with the non-zero entries that appear in the current signal approximation (line 6); then, a least-squares problem is solved to approximate the target signal on the merged set of entries (line 7) and only s largest ones are retained to form a new approximation (line 8); finally, the residual is updated (line 9).

Figure 3.7 depicts the computation of **W**. In the middle is the recovered s-sparse matrix, where $s = 0.8k$ in this case. The input matrix **C** is shown on the left. The underlying diagonal structure of **C** can be seen easily on the recovered matrix. In this example, the number of eigenfunctions used to form the basis is $k = 20$, therefore the recovered structure has $s = 0.8k = 16$ non-zero entries. A straight line is fitted through these entries to determine the location of the diagonal and subsequently where zero elements in the weight matrix **W** are. The values of **W** elements increase by the power of 1.2 the farther away they are from the established diagonal.

A different approach to computing weight matrix is taken by Rodolà et al. [Rod+15]. They estimate the rank of **C** by comparing the Neumann spectra of the two shapes being matched. The two spectra align only partially, and the rank of **C** is the index of the largest eigenvalue computed on shape N smaller or equal to the largest eigenvalue computed on shape M. Knowing the rank allows for estimating where the non-zero values that promote the diagonal structure should be in **W**.

3.3. Numerical Optimisation

The ℓ_2-regularised optimisation is convex and differentiable, and can be minimised by standard gradient methods. On the other hand, ℓ_1- and nuclear norm regularised optimisation problems are convex but not differentiable. Here they are optimised using a proximal algorithm. Note that all three of the proposed optimisation problems are convex and solving them produces globally optimal solutions. The optimisations are initialised by $\mathbf{C}^0$ computed using the method in the original functional maps paper [Ovs+12].

3.4. ℓ_2-Regularised Optimisation

The optimisation problem

$$\min_{\mathbf{C}} h_{\ell_2}(\mathbf{C}) = \min_{\mathbf{C}} \|\mathbf{CA} - \mathbf{B}\|_{\mathrm{F}}^2 + \gamma \|\mathbf{W} \odot \mathbf{C}\|_{\mathrm{F}}^2 \tag{3.6}$$

is convex and differentiable and this work uses the conjugate gradient method [She94] as it is implemented in the Matlab toolbox *manopt* [Bou+14] to solve it.

3.4.1. Conjugate Gradient Method

Conjugate gradient is a method for solving quadratic functions $f : \mathbb{R}^n \to \mathbb{R}$,

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} + \mathbf{x}^T\mathbf{b} + \mathbf{c}, \tag{3.7}$$

where $\mathbf{x} \in \mathbb{R}^n$ is an unknown vector, $\mathbf{b}, \mathbf{c} \in \mathbb{R}^n$ are known vectors, and $\mathbf{A} \in \mathbb{R}^{n \times n}$ is a known symmetric, positive-definite matrix. The gradient of f is

$$g(\mathbf{x}) = \nabla f(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{b}. \tag{3.8}$$

The conjugate gradient (CG) method solves the quadratic functions in an iterative way by generating a sequence $\mathbf{x}^k$, $k > 0$, given an initial guess $\mathbf{x}^0$,

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha^k \mathbf{d}^k. \tag{3.9}$$

Here the step size α^k is obtained by a line search, and $\mathbf{d}^k$ are the search directions defined as

$$\mathbf{d}^{k+1} = -g(\mathbf{x}^{k+1}) + \beta^k \mathbf{d}^k, \quad \mathbf{d}^0 = -g(\mathbf{x}^0) \tag{3.10}$$

where β^k is the update parameter. The key feature of CG method is that all search directions are $\mathbf{A}$-orthogonal to one another. Two vectors $\mathbf{d}^i$ and $\mathbf{d}^j$ are $\mathbf{A}$-orthogonal, or *conjugate*, if

$$(\mathbf{d}^i)^T \mathbf{A} \mathbf{d}^j = 0. \tag{3.11}$$

The $\mathbf{A}$-orthogonal search directions ensure that the minimisation of $f(\mathbf{x}^k)$ along one direction does not undo the minimisation along another. In other words, when moving

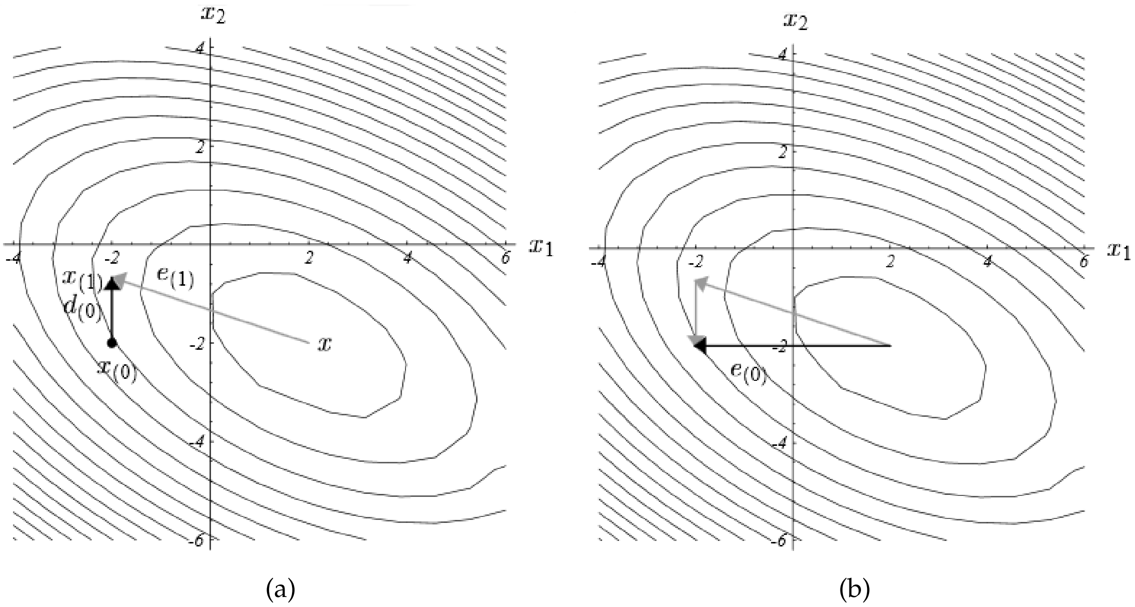


Figure 3.8.: Conjugate search directions in $\mathbb{R}^2$. Starting from the initial point $\mathbf{x}^0$, two steps are needed to reach the optimal point $\mathbf{x}$. Picture is taken from [She94].

the length of α^k in the direction $\mathbf{d}^k$ to reach the minimum point $\mathbf{x}^k$, and then moving the length of α^{k+1} in the direction $\mathbf{d}^{k+1}$, the obtained point $\mathbf{x}^{k+1}$ is still at the minimum in the direction $\mathbf{d}^k$. Subsequently, after n steps, $f(\mathbf{x}^n)$ will be minimised over all searched directions, which means that the CG method converges in n iterations.

Figure 3.8 shows the behaviour of the conjugate search directions in $\mathbb{R}^2$ space. Starting from the point $\mathbf{x}^0$ (3.8a), the first step in the direction $\mathbf{d}^0$ is just the right length to line up evenly with the optimal solution $\mathbf{x}$, i.e. $\mathbf{e}^1 = \mathbf{x}^1 - \mathbf{x}$ is $\mathbf{A}$-orthogonal to $\mathbf{d}^0$. The initial error $\mathbf{e}^0$ can be expressed as a sum of $\mathbf{A}$-orthonormal components (grey arrows) (3.8b). Each step in the conjugate search direction eliminates one of these components.

Different CG methods correspond to different ways of calculating the scalar β^k. Here the Hestenes-Stiefel formula [HZ06] is used

$$\beta^k = \frac{g(\mathbf{x}^{k+1})^T(g(\mathbf{x}^{k+1}) - g(\mathbf{x}^k))}{(\mathbf{d}^k)^T(g(\mathbf{x}^{k+1}) - g(\mathbf{x}^k))}. \tag{3.12}$$

The ideal step size α is the global minimiser of the function $f(\mathbf{x}^k + \alpha\mathbf{d}^k)$. However, it is too computationally expensive to identify this precise value. In practice, a line search algorithm is used. It is able to find a value α that adequately reduces f at a minimal cost. The backtracking line search is the default choice in the *manopt* toolbox. The algorithm sets a relatively large initial estimate $\tilde{\alpha}$ for the step size, and iteratively shrinks or "backtracks" the step size until the sufficient decrease condition is met [NW06,

Chapter 3]:

```
Choose $\tilde{\alpha}, \rho, c \in (0,1)$; set $\alpha \leftarrow \tilde{\alpha}$;
repeat until$f(\mathbf{x}^k + \alpha\mathbf{d}^k) \leq f(\mathbf{x}^k) + c\alpha\nabla f(\mathbf{x}^k)^T\mathbf{d}^k$
    $\alpha \leftarrow \rho\alpha$;                                        (3.13)
end
Terminate with $\alpha^k = \alpha$.
```

3.4.2. Optimisation of Proposed Function

In order to make it easier to see how the objective function in Equation 3.6 fits the quadratic form in Equation 3.7, the matrix $\mathbf{C}$, $\mathbf{B}$ and $\mathbf{W}$ are reshaped into vectors $\mathbf{c}$, $\mathbf{b}$ and $\mathbf{w}$, respectively, by stacking their columns one below the other. As for the matrix $\mathbf{A}$, its every column is repeated k times in a block diagonal fashion, and then these blocks are put one next to the other into one big $kk \times qk$ matrix $\tilde{\mathbf{A}}$.

It is now possible to express the objective function in terms of the reshaped matrices $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{W}} = \mathrm{diag}(w_1, ..., w_{kk})$, and vectors $\mathbf{c}$ and $\mathbf{b}$ in the following manner:

$$\begin{aligned} \left\|\tilde{\mathbf{A}}\mathbf{c} - \mathbf{b}\right\|^2 + \left\|\tilde{\mathbf{W}}\mathbf{c}\right\|^2 &= (\tilde{\mathbf{A}}\mathbf{c} - \mathbf{b})^T(\tilde{\mathbf{A}}\mathbf{c} - \mathbf{b}) + \mathbf{c}^T\tilde{\mathbf{W}}^T\tilde{\mathbf{W}}\mathbf{c} \\ &= \mathbf{c}^T\tilde{\mathbf{A}}^T\tilde{\mathbf{A}}\mathbf{c} - \mathbf{c}^T\tilde{\mathbf{A}}^T\mathbf{b} - \mathbf{b}^T\tilde{\mathbf{A}}\mathbf{c} + \mathbf{b}^T\mathbf{b} + \mathbf{c}^T\tilde{\mathbf{W}}^T\tilde{\mathbf{W}}\mathbf{c}. \end{aligned} \tag{3.14}$$

By introducing the notation $\mathbf{D} = \mathbf{A}^T\mathbf{A} + \tilde{\mathbf{W}}^T\tilde{\mathbf{W}}$ and $\mathbf{y} = \tilde{\mathbf{A}}^{\mathbf{T}}\mathbf{b}$, the Equation 3.14 can be written in a nice quadratic form as:

$$\mathbf{c}^T\mathbf{D}\mathbf{c} - 2\mathbf{c}^T\mathbf{y} + \mathbf{b}^T\mathbf{b} \tag{3.15}$$

which conforms to the type of optimisation problem the CG method can solve.

The CG method requires the computation of the gradient of the objective function. Using the fact that the squared Frobenius norm of a matrix $\mathbf{X}$ can be expressed as $\|\mathbf{X}\|_{\mathrm{F}}^2 = \mathrm{tr}(\mathbf{X}\mathbf{X}^T)$, where tr denotes the trace of a square matrix and is defined as the sum of the elements on the main diagonal, the objective function can be written as:

$$\begin{aligned} h(\mathbf{C}) &= \mathrm{tr}((\mathbf{C}\mathbf{A} - \mathbf{B})(\mathbf{C}\mathbf{A} - \mathbf{B})^T) + \gamma\mathrm{tr}((\mathbf{W} \odot \mathbf{C})(\mathbf{W} \odot \mathbf{C})^T) \\ &= \mathrm{tr}((\mathbf{C}\mathbf{A} - \mathbf{B})(\mathbf{A}^T\mathbf{C}^T - \mathbf{B}^T)) + \gamma\mathrm{tr}((\mathbf{W} \odot \mathbf{C})(\mathbf{W} \odot \mathbf{C}^T)) \\ &= \mathrm{tr}(\mathbf{C}\mathbf{A}\mathbf{A}^T\mathbf{C}^T) - \mathrm{tr}(\mathbf{C}\mathbf{A}\mathbf{B}^T) - \mathrm{tr}(\mathbf{B}\mathbf{A}^T\mathbf{C}^T) + \mathrm{tr}(\mathbf{B}\mathbf{B}^T)) \\ &\quad + \gamma\mathrm{tr}((\mathbf{W} \odot \mathbf{C})(\mathbf{W} \odot \mathbf{C}^T)) \end{aligned} \tag{3.16}$$

This form is more suitable for deriving the gradient because the following trace derivative identities are known:

$$\frac{\partial}{\partial \mathbf{X}} \mathrm{tr}(\mathbf{AX}) = \mathbf{A} \tag{3.17}$$

$$\frac{\partial}{\partial \mathbf{X}} \mathrm{tr}(\mathbf{AXB}) = \mathbf{A}^T\mathbf{B}^T \tag{3.18}$$

$$\frac{\partial}{\partial \mathbf{X}} \mathrm{tr}(\mathbf{AX}^T\mathbf{B}) = \mathbf{BA} \tag{3.19}$$

$$\frac{\partial}{\partial \mathbf{X}} \mathrm{tr}(\mathbf{XABX}^T) = (\mathbf{ABX}^T)^T + \mathbf{XAB}. \tag{3.20}$$

Applying these identities, the gradient of the objective function h_{ℓ_2} with respect to $\mathbf{C}$ is expressed as:

$$\nabla h_{\ell_2}(\mathbf{C}) = \frac{\partial}{\partial \mathbf{C}} h_{\ell_2}(\mathbf{C}) = 2\mathbf{CAA}^T - 2\mathbf{BA}^T + \gamma 2\mathbf{W}^2\mathbf{C} \tag{3.21}$$

3.5. ℓ_1- and Nuclear Norm Regularised Optimisation

The optimisation problems

$$\min_{\mathbf{C}} h_{\ell_1}(\mathbf{C}) = \min_{\mathbf{C}} \|\mathbf{CA} - \mathbf{B}\|_F^2 + \gamma \|\mathbf{W} \odot \mathbf{C}\|_1 \tag{3.22}$$

and

$$\min_{\mathbf{C}} h_{\ell_*}(\mathbf{C}) = \min_{\mathbf{C}} \|\mathbf{CA} - \mathbf{B}\|_F^2 + \gamma \|\mathbf{C}\|_* \tag{3.23}$$

are convex, and while the data term is differentiable, this is not the case for the ℓ_1 and nuclear norm terms. Generally speaking, these types of optimisation problems have the following form:

$$\min_{\mathbf{x}} f(\mathbf{x}) + g(\mathbf{x}) : \mathbf{x} \in \mathbb{R}^n, \tag{3.24}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ and $g : \mathbb{R}^n \to \mathbb{R}$ are both convex functions and only f is required to be smooth. More precisely, f is differentiable with Lipschitz continuous gradient:

$$\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \leq L \|\mathbf{x} - \mathbf{y}\| \quad \forall \mathbf{x}, \mathbf{y},$$

where $\|\cdot\|$ is the ℓ_2-norm and $L > 0$ is the Lipschitz constant.

A proximal algorithm called *fast gradient method* [Nes07] is used to solve the minimisation problem of the form presented in 3.24. Proximal algorithms received that name based on their use of the proximal operator to obtain the solution.

3.5.1. Proximal Operator

The operator $\mathbf{prox}_{\lambda f} : \mathbb{R}^n \to \mathbb{R}^n$ of a function f is defined as

$$\mathbf{prox}_{\lambda f}(\mathbf{v}) = \arg\min_{\mathbf{x}} (f(\mathbf{x}) + \frac{1}{2\lambda} \|\mathbf{x} - \mathbf{v}\|_2^2). \tag{3.25}$$

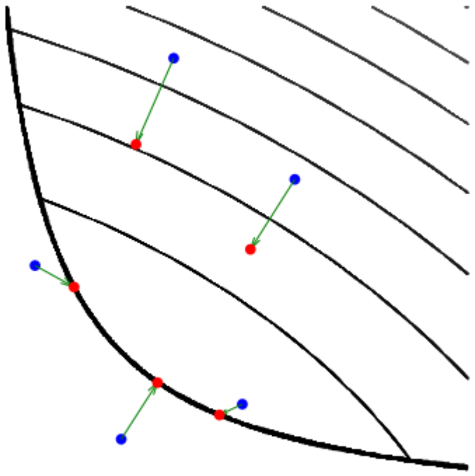


Figure 3.9.: The result of applying the proximal operator at various points in $\mathbb{R}^2$ space. Picture is taken from [PB14].

The resulting minimiser of $\mathbf{prox}_{\lambda f}(\mathbf{v})$ is a point that compromises between minimising f and being close to $\mathbf{v}$. The parameter λ controls the relative weight between these terms. The λ can also be seen as a step size. The larger the value of λ, the smaller the emphasis on the squared norm term, hence the bigger the movement of the point toward the minimum of f.

The visual interpretation of what the proximal operator does in $\mathbb{R}^2$ is given in Figure 3.9. The thick black line is the convex function f and the thin black lines are its contour curves. The proximal operator is evaluated at the blue points which results in them being moved towards the minimum of the function to their corresponding red points. The three points that are in the domain of f stay in the domain, while the other two move to the boundary of the domain.

3.5.2. Proximal Gradient Method

The proximal gradient method solves the convex optimisation problem in Equation 3.24 iteratively, starting from an initial guess $\mathbf{x}^0$:

$$\mathbf{x}^{k+1} = \mathbf{prox}_{\lambda^k g}(\mathbf{x}^k - \lambda^k \nabla f(\mathbf{x}^k)), \tag{3.26}$$

where $\lambda^k > 0$ is a step size. It can be shown that the method converges to a solution at a rate of $\mathcal{O}(1/k)$ if λ^k is fixed to a value from the $(0, 1/L]$ interval, where L is a Lipschitz constant of ∇f. The solution $\mathbf{x}^{k+1}$ is a trade-off between minimising g and following the standard gradient step $\mathbf{x}^k - \lambda \nabla f(\mathbf{x}^k)$.

The fast gradient method (FGM) is based on the previously defined proximal gradient method. The FGM uses an extrapolated point $\mathbf{y}^{k+1}$ to compute the solution of every iteration:

$$\begin{aligned} \omega^{k+1} &= \frac{1+\sqrt{1+4(\omega^k)^2}}{2} \\ \mathbf{y}^{k+1} &= \mathbf{x}^k + \frac{\omega^k - 1}{\omega^{k+1}}(\mathbf{x}^k - \mathbf{x}^{k-1}) \\ \mathbf{x}^{k+1} &= \mathbf{prox}_{\lambda^k g}(\mathbf{y}^{k+1} - \lambda^k \nabla f(\mathbf{y}^{k+1})), \end{aligned} \tag{3.27}$$

where $\omega^0 = 1$ and $\mathbf{y}^0 = \mathbf{x}^0$.

3.5.3. Optimisation of Proposed Functions

The concrete optimisation problems in Equation 3.22 and 3.23 can be split into the f and g parts:

$$f(\mathbf{C}) = \|\mathbf{CA} - \mathbf{B}\|_\mathrm{F}^2, \quad g(\mathbf{C}) = \gamma \|\mathbf{W} \odot \mathbf{C}\|_1 \tag{3.28}$$

and

$$f(\mathbf{C}) = \|\mathbf{CA} - \mathbf{B}\|_\mathrm{F}^2, \quad g(\mathbf{C}) = \gamma \|\mathbf{C}\|_* \tag{3.29}$$

respectively. The gradient of the both functions is the same:

$$\nabla f(\mathbf{C}) = 2\mathbf{CAA}^T - 2\mathbf{BA}^T, \tag{3.30}$$

and its derivation is similar to the one presented in Section 3.4.2. The proximal operator of the ℓ_1-term is defined as:

$$\mathbf{prox}_{\gamma g}(\mathbf{W} \odot \mathbf{C}) = S_\gamma(\mathbf{W} \odot \mathbf{C}). \tag{3.31}$$

It is evaluated via the soft-thresholding operator S_γ [PB14]:

$$(\mathbf{prox}_{\gamma g}(\mathbf{W} \odot \mathbf{C}))_i = S_\gamma(w_i c_i) = \begin{cases} c_i - \gamma\lambda w_i & c_i > \gamma\lambda w_i \\ c_i + \gamma\lambda w_i & c_i < -\gamma\lambda w_i \\ 0 & -\gamma\lambda w_i \le c_i \le \gamma\lambda w_i. \end{cases} \tag{3.32}$$

As before, λ is the step size, and γ the relative weight between f and g parts of the objective function. The S_γ is applied individually to each element of the $\mathbf{W} \odot \mathbf{C}$ matrix. Equation 3.32 can be written more compactly as:

$$\mathbf{prox}_{\gamma g}(\mathbf{W} \odot \mathbf{C}) = \max(|\mathbf{C}| - \gamma\lambda\mathbf{W}, 0) \odot \operatorname{sign}(\mathbf{C}) \tag{3.33}$$

where the absolute value and the sign function are applied element-wise.

As for the proximal operator of the nuclear norm, it is defined as:

$$\mathbf{prox}_{\gamma g}(\mathbf{C}) = \sum_{i=1}^{k} \max(\sigma_i - \gamma\lambda, 0) u_i v_i^T, \tag{3.34}$$

where $\mathbf{C} = \sum_{i=1}^{k} \sigma_i u_i v_i^T$ is the singular value decomposition of $\mathbf{C}$. This operation is called singular value thresholding since the singular values are being shrank towards zero by a constant amount [PB14].

As previously stated, the step size λ is bounded by the Lipschitz constant of ∇f, which in this case depends on the maximum eigenvalue of $\mathbf{A}^T\mathbf{A}$.

4. Experimental Results

4.1. Experiment Set-up

The 3D shapes used in the experiments are taken from TOSCA data set [BBK08]. The set contains triangular meshes (10K-50K vertices) of animals and humans in different poses. The 2D shapes are mostly obtained by taking pictures of the TOSCA shapes and then using the Delaunay triangulation to create the meshes (<1K vertices). The ground truth correspondence between pairs of 2D and 3D shapes is a set of one hundred point correspondences selected manually.

All the experiments were run using $k = 20$ eigenfunctions. The regularisation parameter γ in ℓ_1, ℓ_2, and nuclear norm regularised optimisation was set to 1, 10, and 20 respectively. The stopping criteria in the optimisation was the number of iterations, which was 200. The number of regions detected on shapes was 6-7, while the vectors of HKS and WKS had 100 components. Time parameter t in HKS was sampled from $[1, 20000]$ interval and the variance in WKS was 6. The point-to-point correspondences are obtained using the point indicators conversion method described in Subsection 2.2.1. No post-processing iterative refinement was used to compute the correspondences unless otherwise stated.

4.2. Evaluation Method

To measure the quality of the computed point correspondences, the evaluation method of [KLF11] is used. For every point p on 2D shape M that is in the ground truth correspondence set, the geodesic distance between its computed image and its true correspondence on 3D shape N is measured:

$$Err(f, f_{\text{true}}) = \sum_{p \in M} d_N(f(p), f_{\text{true}}(p)). \tag{4.1}$$

The error is normalised by $\sqrt{Area(N)}$ or by $\sqrt{Area(N)/2}$ depending on whether the 2D shape is being matched to the whole 3D shape or to one of its sides. The accuracy measure plots that follow in the next section show what percentage of correspondences has geodesic distance error less than some threshold value D, i.e. $d_N(f(p), f_{\text{true}}(p)) < D$.

4.3. Results

The results of matching one side of the 3D shape and the whole 3D shape are presented separately. An example of a point-wise correspondence, one for each matching case, is

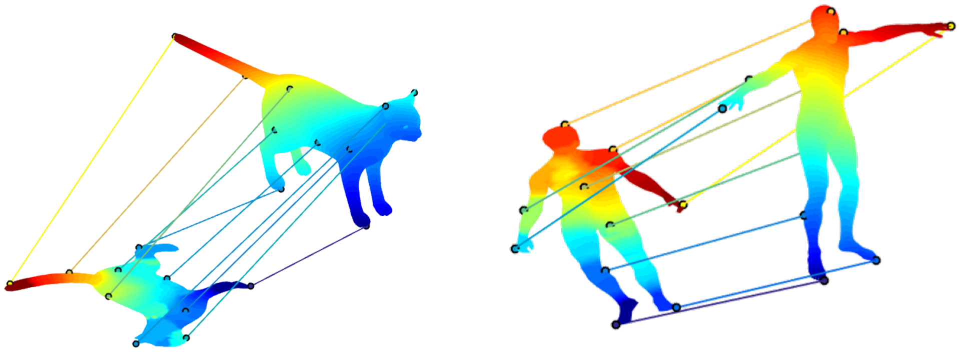


Figure 4.1.: Example of the obtained point-to-point correspondence.

shown in Figure 4.1. Points on the extremities tend to be matched with higher accuracy. WKS and HKS descriptors are not able to precisely localise the features on the torso. Also, when matching to the whole shape, some points are matched to the front and some to the back side of the body. More examples can be found in the appendix.

The performance of using different combinations of shape descriptors as functional constraints in all three proposed regularisations is shown in Figure 4.2. As expected, using only region indicator functions produces the least accurate point correspondences. When combining the regions with WKS and HKS, there is no clear advantage of using one over the other or both of them together. The only noticeable difference in the correspondence accuracy is displayed by the nuclear norm regularisation (Figure 4.2c) where it is better to use WKS.

When comparing the three proposed regularisations against each other, the ℓ_1 and ℓ_2 one achieve better results in the case when one side of the 3D shape is matched (Figure 4.3a). On the other hand, the nuclear norm regularisation is able to reach higher percentage of correct correspondences at a lower error measure than the other two methods when the whole 3D shape is being matched (Figure 4.3b).

How the correspondence matrix $\mathbf{C}$ looks when computed using the proposed method can be seen in Figure 4.4. The ℓ_1 and ℓ_2-regularisations output very similar looking matrices (left and middle columns). The application of the weight matrix $\mathbf{W}$ has the desired effect of inducing the diagonal structure, regardless of the norm used. When this observation is supported by the result of the previously discussed Figure 4.3, it seems that it does not matter whether ℓ_1 or ℓ_2 norm is chosen as the regularisation term. The right column of Figure 4.4 shows the low rank matrix produced by the nuclear norm regularisation. The entries corresponding to the higher frequency eigenfunctions are discarded.

As mentioned before, the point-to-point conversion is done by transferring point indicator functions from one shape to the other. The more efficient conversion method of looking for the nearest neighbour of every $\mathbf{C\Phi}^T$ column in $\mathbf{\Psi}^T$ produces mixed results. It performs very similar to the point indicators method when matching to the whole 3D

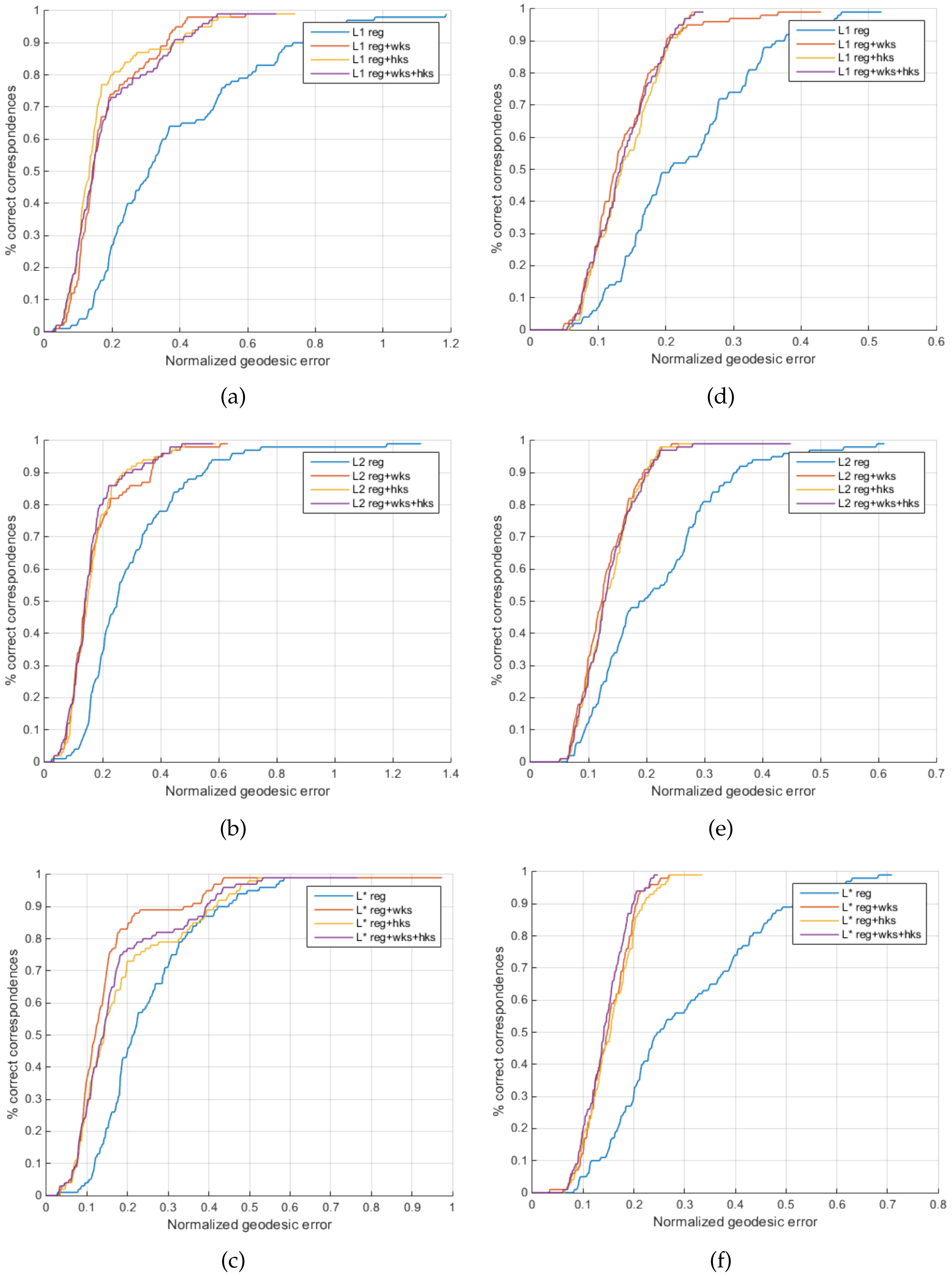
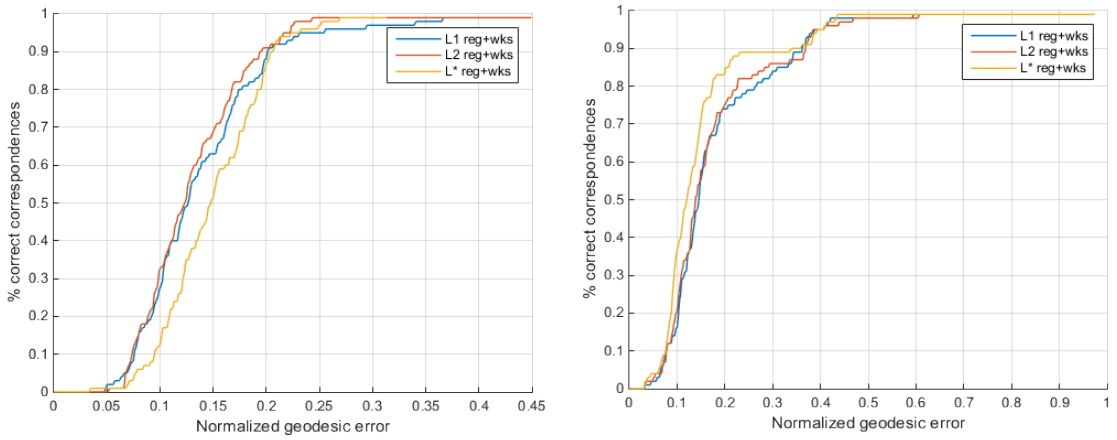


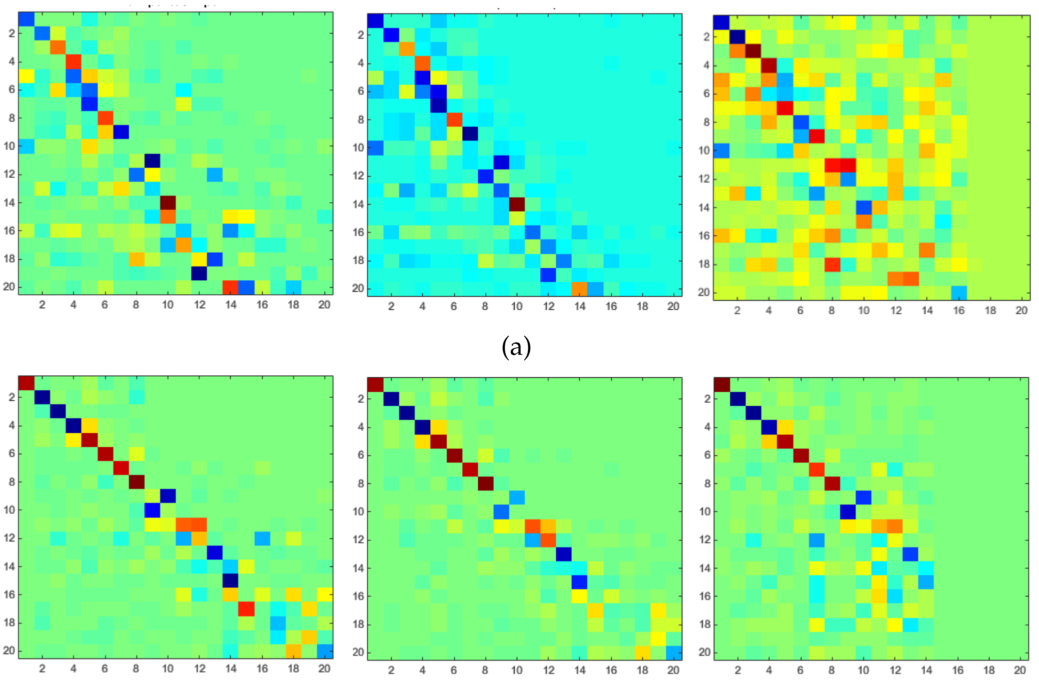
Figure 4.2.: Comparison of different regularisation terms and shape descriptors used. The plots (a-c) show the results of matching 2D shapes to one side of the 3D shape, while plots (d-f) show matching to the whole shape.



(a) One side of the 3D shape is matched.

(b) The whole 3D shape is matched.

Figure 4.3.: The three proposed regularisation terms plotted against each other for comparison. The functional constraints used are region indicator functions and WKS.

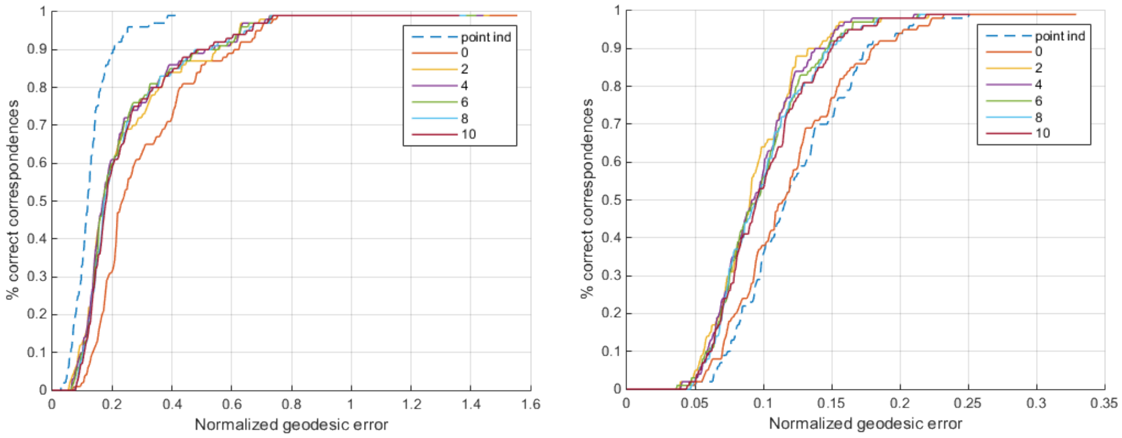


(a)

(b)

Figure 4.4.: The look of a typical correspondence matrix **C** computed when one side of the 3D shape (a) and the whole shape (b) is being matched. Left is ℓ_1, middle is ℓ_2, and right is using nuclear norm regularisation term.

shape (Figure 4.5b, see red full and blue dashed line). The situation is very different when one side of the 3D shape is matched. The point indicators method achieves significantly better accuracy, even when iterative refinement is used, which improves the result in both matching cases. The 2D meshes used in the experiments have less than a thousand vertices, so the conversion does not take more than a couple of seconds to be executed.



(a) One side of the 3D shape is matched.

(b) The whole 3D shape is matched.

Figure 4.5.: The effect the two methods for point-to-point conversion (Section 2.2.1) have on the accuracy. The dashed blue line represents the point-to-point correspondence obtained using point indicator functions. The full lines represent the more efficient method with 0-10 iterations of post-processing refinement.

5. Application

5.1. Deformation Transfer

Transferring deformations from one shape to another can accelerate mesh animation. A correspondence between two shapes has to be established before the deformation transfer can be performed. The shape matching approach in this work produces the required correspondence as illustrated by Figure 5.1. How a 2D shape can be deformed and how the obtained deformation can be transferred onto a 3D shape is explained below.

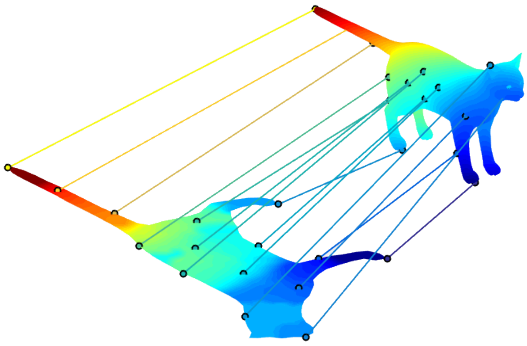


Figure 5.1.: Dense point correspondence obtained by the proposed shape matching approach.

5.1.1. 2D Shape Deformation

Bounded Biharmonic Weights (BBW) [Jac+11] approach is used to deform the 2D shapes. BBW is a linear blending scheme that produces smooth and intuitive deformations for the combination of handle types (points, skeleton bones, cages) in real-time. The user specifies an affine transformation T_j for each handle $H_j \subset \Omega, j = 1, ..., m$, and all points $\mathbf{p} \in \Omega$ are deformed by their weighted combinations:

$$\mathbf{p}' = \sum_{j=1}^{m} w_j(\mathbf{p}) T_j \mathbf{p} \tag{5.1}$$

where Ω is the domain obtained by the union of the shape and cage controls (if any), and $w_j : \Omega \to \mathbb{R}$ is a weight function associated with handle H_j.

The weights are computed by minimising the Laplacian energy subject to the certain constraints that specify desirable properties of weights such as smoothness and non-negativity:

$$\begin{aligned}
&\min_{w_j, j=1,\dots,m} \sum_{j=1}^{m} \frac{1}{2} \int_{\Omega} \left\| \Delta w_j \right\|^2 dV \\
&\text{subject to: } w_j|_{H_k} = \delta_{jk} \\
&\qquad w_j|_F \text{ is linear} \qquad\qquad \forall F \in F_C \\
&\qquad \sum_{j=1}^{m} w_j(\mathbf{p}) = 1 \qquad\qquad \forall \mathbf{p} \in \Omega \\
&\qquad 0 \leq w_j(\mathbf{p}) \leq 1, j = 1, \dots, m \quad \forall \mathbf{p} \in \Omega
\end{aligned} \tag{5.2}$$

where F_C is the set of all cage faces and δ_{jk} is a delta function.

5.1.2. 2D-to-3D Deformation Transfer

Deformation transfer for triangular meshes proposed by Sumner and Popović [SP04] copies the deformations of the triangles of the source mesh onto those of the target. The approach is general and does not require that the number of vertices and faces of the two meshes be the same.

A correspondence map M between the triangles of the source and target mesh

$$M = \{(s_1, t_1), (s_2, t_2), \dots, (s_{|M|}, t_{|M|})\} \tag{5.3}$$

is built from a small set of point correspondences. M is in general many-to-many mapping and it specifies that a triangle t_i on the target mesh should deform like a triangle s_i on the source mesh. The set of triangle transformations $\mathbf{S}_1, \dots, \mathbf{S}_{|S|}$ induced by the deformation of the source mesh is mapped through the correspondence M to the target mesh. However, the transform $\mathbf{S}_i$ cannot be applied directly to the corresponding target triangle since $\mathbf{S}_i$ encodes only the change in orientation and size, and not the relative position of the triangle to its neighbours. Therefore, a constraint has to be put in place which requires that shared vertices are transferred to the same place. That is, for the set of target affine transformations $\mathbf{T}_1 + \mathbf{d}_1 \dots \mathbf{T}_{|T|} + \mathbf{d}_{|T|}$ the requirement is:

$$\mathbf{T}_j \mathbf{v}_i + \mathbf{d}_j = \mathbf{T}_k \mathbf{v}_i + \mathbf{d}_k, \qquad \forall i, \forall j, k \in p(\mathbf{v}_i) \tag{5.4}$$

where $p(\mathbf{v}_i)$ is the set of all triangles that share vertex $\mathbf{v}_i$. The deformation transfer then solves the following constrained optimisation problem to consistently apply the source transformations to the target shape:

$$\begin{aligned}
&\min_{\mathbf{T}_1 + \mathbf{d}_1 \dots \mathbf{T}_{|T|} + \mathbf{d}_{|T|}} \sum_{j=1}^{|M|} \left\| \mathbf{S}_{s_j} - \mathbf{T}_{t_j} \right\|_F^2 \\
&\text{subject to: } \mathbf{T}_j \mathbf{v}_i + \mathbf{d}_j = \mathbf{T}_k \mathbf{v}_i + \mathbf{d}_k, \qquad \forall i, \forall j, k \in p(\mathbf{v}_i)
\end{aligned} \tag{5.5}$$

The results in the paper show deformation transfer only on 3D meshes. However, the approach can be applied to transferring deformations from 2D to 3D meshes as well. Figure 5.2 shows one such transfer. In this example, a set of 20 point correspondences obtained by the shape matching algorithm proposed in this work is used to establish the map *M* between the two reference cats. Four different deformations of the 2D reference cat are transferred onto the 3D one, generating four new poses which are shown in the bottom row labelled as "Output". The deformations are transferred to one side of the 3D cat first. Then, the side is copied and reflected through the bilateral symmetry plane to make up the full shape.

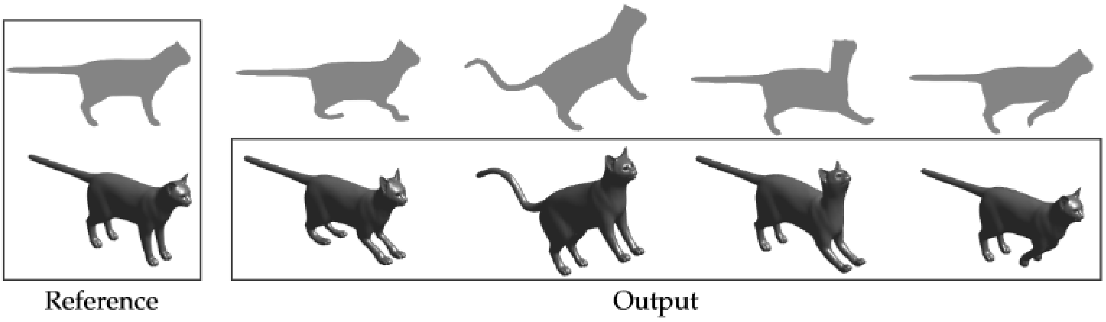


Figure 5.2.: Deformations of the 2D reference mesh are transferred to the 3D one, creating new 3D poses.

6. Conclusion

This work introduced a method for 2D-to-3D dense matching of non-rigid shapes. The method is based on the functional maps framework modified to address the 2D-to-3D case. Namely, a functional correspondence matrix is regularised in the optimisation process in order to promote its characteristic structure, which in turn makes the method be more robust to noise introduced by the spectral descriptors and eigenfunctions computed on the shapes. The proposed optimisation of correspondence matrix is convex allowing for computation of globally optimal solutions. The method is shown to be able to produce quality dense correspondence. In addition, the produced results can be used successfully in the application of deformation transfer.

The method is limited to matching side and front views of the shape as there is no procedure used to automatically determine which part of the 3D shape is being matched. Secondly, the accuracy of the computed shape matchings depends on the localisation abilities of shape descriptors. However, WKS and HKS descriptors do not have high localisation accuracy on more featureless parts of the shapes like human torso. The use of non-spectral shape descriptors could improve the quality of matching. Finally, post-processing iterative refinement improves the results of the optimisation process when partiality is not present. Using a similar iterative refinement for partial matching could further improve the results.

Appendix

A. Additional Shape Matching Results

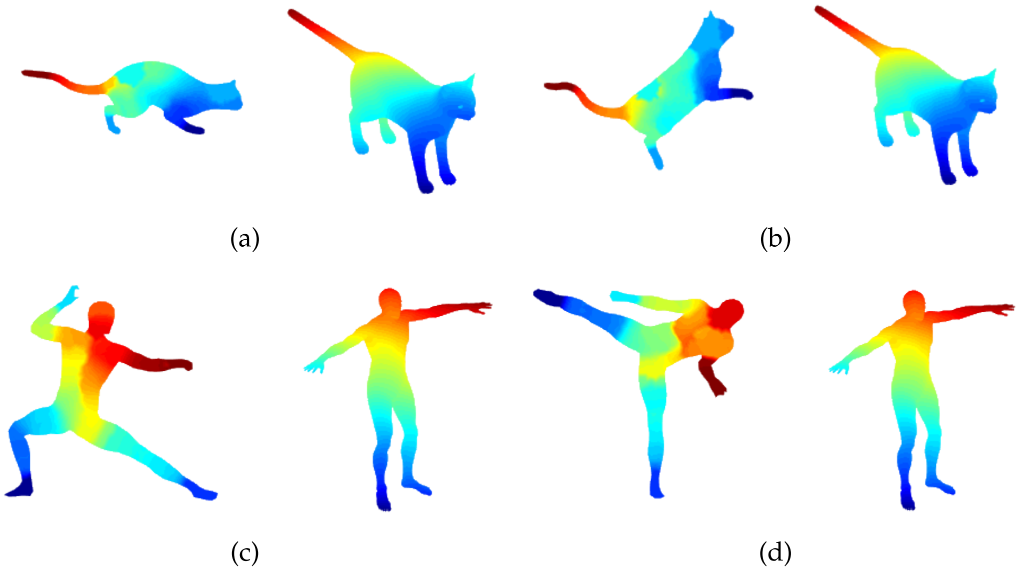


Figure A.1.: Dense point correspondence between pair of shapes (2D on the left, 3D on the right).

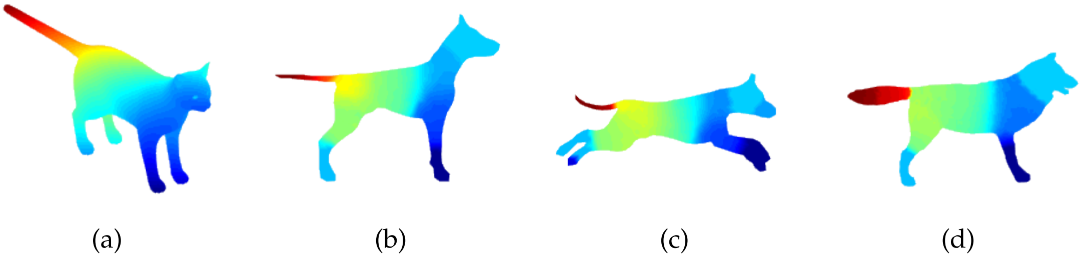
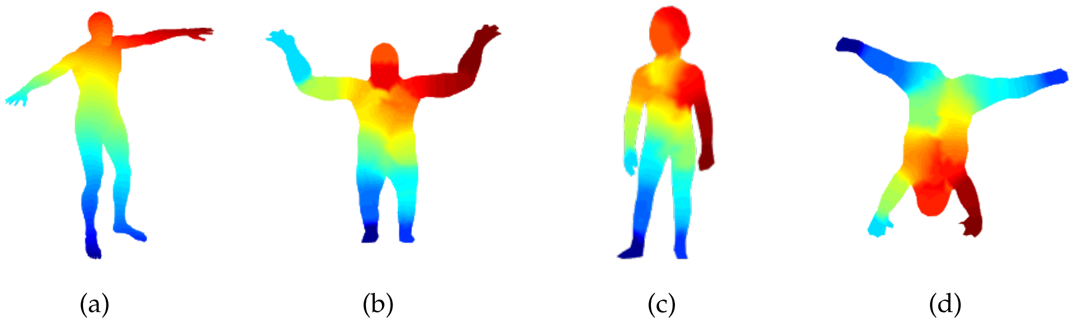


Figure A.2.: Dense point correspondence under partiality between non-isometric shapes. The three 2D shapes (b-d) are matched to the 3D cat shape (a).



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

Figure A.3.: Dense point correspondence between non-isometric shapes. The three 2D shapes (b-d) are matched to the 3D human shape (a). The 2D front views of (c,d) are a picture of the kid shapes taken from [Rod+14].

Bibliography

[ASC11] M. Aubry, U. Schlickewei, and D. Cremers. "The Wave Kernel Signature: A Quantum Mechanical Approach To Shape Analysis." In: *IEEE International Conference on Computer Vision (ICCV) - Workshop on Dynamic Shape Capture and Analysis (4DMOD)*. 2011.

[BBK08] A. Bronstein, M. Bronstein, and R. Kimmel. *Numerical Geometry of Non-Rigid Shapes*. 1st ed. Springer Publishing Company, Incorporated, 2008.

[Bou+14] N. Boumal, B. Mishra, P.-A. Absil, and R. Sepulchre. "Manopt, a Matlab Toolbox for Optimization on Manifolds." In: *Journal of Machine Learning Research* 15 (2014), pp. 1455–1459.

[ELF97] D. W. Eggert, A. Lorusso, and R. B. Fisher. "Estimating 3-D Rigid Body Transformations: A Comparison of Four Major Algorithms." In: *Mach. Vision Appl.* 9.5-6 (Mar. 1997), pp. 272–290.

[Hua+08] Q.-X. Huang, B. Adams, M. Wicke, and L. J. Guibas. "Non-rigid Registration Under Isometric Deformations." In: *Proceedings of the Symposium on Geometry Processing*. SGP '08. Copenhagen, Denmark: Eurographics Association, 2008, pp. 1449–1457.

[HWG14] Q. Huang, F. Wang, and L. Guibas. "Functional Map Networks for Analyzing and Exploring Large Shape Collections." In: *ACM Trans. Graph.* 33.4 (July 2014), 36:1–36:11.

[HZ06] W. W. Hager and H. Zhang. "A survey of nonlinear conjugate gradient methods." In: *Pacific journal of Optimization* 2.1 (2006), pp. 35–58.

[Jac+11] A. Jacobson, I. Baran, J. Popović, and O. Sorkine. "Bounded Biharmonic Weights for Real-Time Deformation." In: *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH)* 30.4 (2011), 78:1–78:8.

[KLF11] V. G. Kim, Y. Lipman, and T. Funkhouser. "Blended Intrinsic Maps." In: *ACM SIGGRAPH 2011 Papers*. SIGGRAPH '11. Vancouver, British Columbia, Canada: ACM, 2011, 79:1–79:12.

[Kov+12] A. Kovnatsky, M. M. Bronstein, A. M. Bronstein, K. Glashoff, and R. Kimmel. "Coupled quasi-harmonic bases." In: *CoRR* abs/1210.0026 (2012).

[Kov+15] A. Kovnatsky, M. M. Bronstein, X. Bresson, and P. Vandergheynst. "Functional Correspondence by Matrix Completion." In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2015.

[KSP09] V. Kraevoy, A. Sheffer, and M. van de Panne. “Modeling from Contour Drawings.” In: *Proceedings of the 6th Eurographics Symposium on Sketch-Based Interfaces and Modeling*. SBIM ’09. New Orleans, Louisiana: ACM, 2009, pp. 37–44.

[LA11] M. Li and G. Ashraf. “Transactions on Edutainment V.” In: ed. by Z. Pan, A. D. Cheok, W. Müller, and X. Yang. Berlin, Heidelberg: Springer-Verlag, 2011. Chap. Sketch Based 3D Character Deformation, pp. 177–188.

[Li+14] B. Li, Y. Lu, A. Godil, T. Schreck, B. Bustos, A. Ferreira, T. Furuya, M. J. Fonseca, H. Johan, T. Matsuda, R. Ohbuchi, P. B. Pascoal, and J. M. Saavedra. “A comparison of methods for sketch-based 3D shape retrieval.” In: *Computer Vision and Image Understanding* 119 (2014), pp. 57–80.

[Mey+02] M. Meyer, M. Desbrun, P. Schröder, and A. H. Barr. “Discrete Differential-Geometry Operators for Triangulated 2-Manifolds.” In: *Proc. VisMath*. 2002, pp. 35–57.

[MS67] H. P. McKean Jr. and I. M. Singer. “Curvature and the eigenvalues of the Laplacian.” In: *J. Differential Geom.* 1.1-2 (1967), pp. 43–69.

[Nes07] Y. Nesterov. *Gradient methods for minimizing composite objective function*. CORE Discussion Papers 2007076. Université catholique de Louvain, Center for Operations Research and Econometrics (CORE), 2007.

[NW06] J. Nocedal and S. J. Wright. *Numerical Optimization, second edition*. World Scientific, 2006.

[Ovs+10] M. Ovsjanikov, Q. Mérigot, F. Mémoli, and L. J. Guibas. “One Point Isometric Matching with the Heat Kernel.” In: *Comput. Graph. Forum* 29.5 (July 2010), pp. 1555–1564.

[Ovs+12] M. Ovsjanikov, M. Ben-Chen, J. Solomon, A. Butscher, and L. Guibas. “Functional Maps: A Flexible Representation of Maps Between Shapes.” In: *ACM Trans. Graph.* 31.4 (July 2012), 30:1–30:11.

[Ovs+13] M. Ovsjanikov, Q. Mérigot, V. Patraucean, and L. J. Guibas. “Shape Matching via Quotient Spaces.” In: *Comput. Graph. Forum* 32.5 (2013), pp. 1–11.

[PB14] N. Parikh and S. Boyd. “Proximal Algorithms.” In: *Found. Trends Optim.* 1.3 (Jan. 2014), pp. 127–239.

[Pok+12] J. Pokrass, A. M. Bronstein, M. M. Bronstein, P. Sprechmann, and G. Sapiro. “Sparse Modeling of Intrinsic Correspondences.” In: *CoRR* abs/1209.6560 (2012).

[RBC14] E. Rodolà, S. R. Bulò, and D. Cremers. “Robust Region Detection via Consensus Segmentation of Deformable Shapes.” In: *Comput. Graph. Forum* 33.5 (2014), pp. 97–106.

[RCG08] G. Rong, Y. Cao, and X. Guo. “Spectral mesh deformation.” English. In: *The Visual Computer* 24.7-9 (2008), pp. 787–796.

[Reu+09a] M. Reuter, S. Biasotti, D. Giorgi, G. Patanè, and M. Spagnuolo. "Discrete Laplace-Beltrami operators for shape analysis and segmentation." In: *Computers & Graphics* 33.3 (2009), pp. 381–390.

[Reu+09b] M. Reuter, F.-E. Wolter, M. Shenton, and M. Niethammer. "Laplace-Beltrami Eigenvalues and Topological Features of Eigenfunctions for Statistical Shape Analysis." In: *Comput. Aided Des.* 41.10 (Oct. 2009), pp. 739–755.

[Rod+14] E. Rodolà, S. R. Bulò, T. Windheuser, M. Vestner, and D. Cremers. "Dense Non-rigid Shape Correspondence Using Random Forests." In: *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*. 2014, pp. 4177–4184.

[Rod+15] E. Rodolà, L. Cosmo, M. M. Bronstein, A. Torsello, and D. Cremers. "Partial Functional Correspondence." In: *Comput. Graph. Forum* 34.5 (2015).

[Rus+13] R. M. Rustamov, M. Ovsjanikov, O. Azencot, M. Ben-Chen, F. Chazal, and L. Guibas. "Map-based Exploration of Intrinsic Shape Differences and Variability." In: *ACM Trans. Graph.* 32.4 (July 2013), 72:1–72:12.

[Rus07] R. M. Rustamov. "Laplace-Beltrami Eigenfunctions for Deformation Invariant Shape Representation." In: *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*. SGP '07. Barcelona, Spain: Eurographics Association, 2007, pp. 225–233.

[She94] J. R. Shewchuk. *An Introduction to the Conjugate Gradient Method Without the Agonizing Pain*. Tech. rep. Pittsburgh, PA, USA, 1994.

[SOG09] J. Sun, M. Ovsjanikov, and L. Guibas. "A Concise and Provably Informative Multi-scale Signature Based on Heat Diffusion." In: *Proceedings of the Symposium on Geometry Processing*. SGP '09. Berlin, Germany: Eurographics Association, 2009, pp. 1383–1392.

[SP04] R. W. Sumner and J. Popović. "Deformation Transfer for Triangle Meshes." In: *ACM Trans. Graph.* 23.3 (Aug. 2004), pp. 399–405.

[SY11] Y. Sahillioglu and Y. Yemez. "Coarse-to-Fine Combinatorial Matching for Dense Isometric Shape Correspondence." In: *Comput. Graph. Forum* 30.5 (2011), pp. 1461–1470.

[T0L10] G. hua Tan, W. C. 0001, and L. gang Liu. "Image driven shape deformation using styles." In: *Journal of Zhejiang University - Science C* 11.1 (2010), pp. 27–35.

[TN08] J. A. Tropp and D. Needell. "CoSaMP: Iterative signal recovery from incomplete and inaccurate samples." In: *CoRR* abs/0803.2392 (2008).

[Wan+14] F. Wang, Q. Huang, M. Ovsjanikov, and L. Guibas. "Unsupervised Multi-Class Joint Image Segmentation." In: *EEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014.

[Xu+11] K. Xu, H. Zheng, H. Zhang, D. Cohen-Or, L. Liu, and Y. Xiong. “Photo-inspired Model-driven 3D Object Modeling.” In: *ACM SIGGRAPH 2011 Papers*. SIGGRAPH ’11. Vancouver, British Columbia, Canada: ACM, 2011, 80:1–80:10.