הטכניון
מכון טכנולוגי לישראל
TECHNION
Israel Institute of Technology

ספריות הטכניון
The Technion Libraries

בית הספר ללימודי מוסמכים ע"ש ארווין וג'ואן ג'ייקובס
Irwin and Joan Jacobs Graduate School

Non Isometric Shape Correspondence

Danielle Ezuz

Non Isometric Shape Correspondence

Research Thesis

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

Danielle Ezuz

Submitted to the Senate
of the Technion — Israel Institute of Technology Iyar
5779 Haifa May 2019

This research was carried out under the supervision of Prof. Mirela Ben-Chen, in the Computer Science Department.

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:

Danielle Ezuz and Mirela Ben-Chen. Deblurring and denoising of maps between shapes. *Computer Graphics Forum*, volume 36, pages 165-174, 2017.

Danielle Ezuz, Justin Solomon, Vladimir G. Kim and Mirela Ben-Chen. GWCNN: A metric alignment layer for deep shape analysis. *Computer Graphics Forum*, volume 36, pages 49-57, 2017.

Danielle Ezuz, Justin Solomon and Mirela Ben-Chen. Reversible harmonic maps between discrete surfaces. *ACM Transactions on Graphics (TOG)*, volume 38, pages 15:1-15:12, 2019.

Danielle Ezuz, Behrend Heeren, Omri Azencot, Martin Rumpf and Mirela Ben-Chen. Elastic Correspondence between Triangle Meshes. *Computer Graphics Forum*, volume 38, 2019.

The generous financial help of the Technion, the Computer Science Department, the Irwin and Joan Jacobs Fellowship, and the Prof. Rahamimoff Travel Grant for Young Scientists of the US-Israel Binational Science Foundation (BSF) is gratefully acknowledged.

Contents

List of Figures

Abstract

Shape correspondence is a fundamental task in shape analysis, and has a variety of applications in computer graphics and computer vision. Generally, given two shapes, the goal is to compute for each point on the source shape a corresponding point on the target shape. Example applications include morphing (gradually deforming one shape to another), deformation or texture transfer, and statistical shape analysis.

Shape correspondence can be classified into different categories, based on the properties of the input shapes, and the desired properties of the result. A common category is *isometric* shape correspondence, that usually characterizes matching between the same object in different poses. In other cases, where correspondence is computed between *different* objects, the task is more challenging and not well defined mathematically.

This research studies non isometric shape correspondence. The main challenge is to explicitly define the mathematical properties of the desired results, and to design algorithms that generate results with the desired properties. As the desired result mostly depend on the downstream application, it is instrumental to consider the application when designing the method.

We propose semi-automatic methods for computation of shape correspondence between highly non isometric shapes, as well as a method that optimizes an existing correspondence method for shape classification and retrieval using deep learning. Our shape correspondence methods are designed to generate locally smooth results, that are advantageous for applications in computer graphics such as texture transfer and shape interpolation. This document contains a detailed description of the problem and the motivation, the proposed algorithms, and demonstrates the results and a variety of applications.

Chapter 1

Introduction

Shape correspondence, or shape matching is a fundamental task in shape analysis, and has many applications in computer graphics and computer vision. Example applications include statistical shape analysis [MDW08], texture, segmentation and deformation transfer [SP04], and shape interpolation [HRWW12]. Generally, the objective is to compute for each point on the source shape a corresponding point on the target shape (visualized in Figure 1.1).

Shape correspondence can be classified into different categories, based on the properties of the input shapes, and the desired properties of the result. A common classification distinguishes between *isometric* shape correspondence, that usually characterizes matching between the same object in different poses, and *non isometric* shape correspondence where different objects are matched. Another possible classification distinguishes between *full* and *partial* correspondence; full correspondence means that every point on the source shape has a matching point on the target and vice versa, while partial correspondence assigns a matching point only to a *subset* of points.

Isometric correspondence can be mathematically characterized by pairwise distance preservation, while there is no consensus regarding the mathematical characteristics of correspondence between different objects. In fact, in some cases different applications

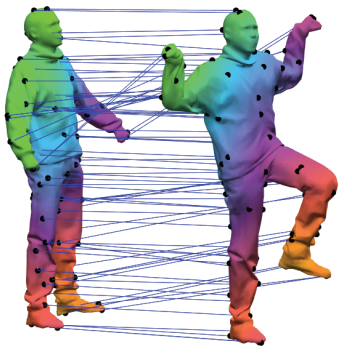


Figure 1.1: A map between shapes visualized using color transfer and lines that connect a sparse subset of correspondences.

require different correspondence between the same objects. Consider for example the shapes in Figure 1.2, which are not isometric. Seemingly, these shapes can be classified into the *partial correspondence* category, since the shape on the left has a part of the arm that the shape on the right does not have. For applications such as texture transfer, only points on the hand itself should have a corresponding point on the target. However, to successfully *interpolate* between these shapes, a bijective correspondence is required, where the arm segment is smoothly mapped to the bottom part of the target hand. Such a bijective map is visualized in the figure using texture transfer, and the interpolation results are demonstrated as well.

The main challenge this thesis addresses is mathematically defining the desired properties of correspondence between non isometric shapes. Since we handle shapes of different objects, we do not assume that semantic correspondence can be determined solely by the shape geometry. We therefore assume that additional information is given, such as a sparse set of corresponding landmarks, or an initial map that we refine.

Map refinement is instrumental to leverage existing correspondence methods that generate reasonable results with local inaccuracies. For example, many existing methods use *functional maps* [OBCS$^+$12], which are linear operators that match between *smooth functions* rather than points. Functional maps have many advantages; the discrete representation is compact, and its computation is efficient since many properties can be formulated as linear constraints. However, eventually the functional map needs to be converted to a pointwise map, and extracting a good pointwise map is not straightforward. We therefore devised a method that efficiently and accurately converts functional maps to pointwise maps [EBC17]. It is useful for a variety of methods that rely on the functional map representation. The method can also be used to refine a pointwise map, by converting it to a functional map (which removes high frequencies) and back. The details and the results are described in section 2.

When a sparse set of corresponding landmarks is given, it is natural to extend it to the entire shape in a smooth fashion. The Dirichlet energy is a known function that measures smoothness of maps between shapes, and a discretization for triangle

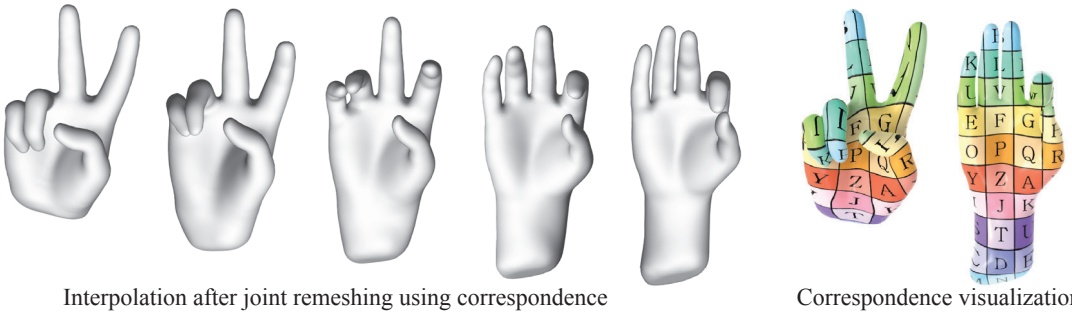


Figure 1.2: Non trivial correspondence example. While the shapes may seem to partially correspond, interpolation requires bijective correspondence.

meshes has been studied as well [IN05]. While optimizing the Dirichlet energy of a map improves local smoothness, it also *shrinks* the map in the sense that entire regions of the target shape may remain unmatched. We therefore recently suggested an energy that combines a discrete approximation of the Dirichlet energy and a term that promotes bijectivity [ESBC19]. Our discretization allows efficient and robust optimization, as well as locally accurate results. The details are provided in section 3.

While optimizing the Dirichlet energy is robust and leads to smooth results, it does not align *extrinsic features* such as creases and corners. In many cases, aligning extrinsic features is crucial for semantic correspondence. This can be done by replacing the Dirichlet energy with a non-linear elastic energy [EHA$^+$19], that is widely used for shape deformation. The use of a deformation energy for correspondence computation is natural, since shape correspondence is actually equivalent to a constrained deformation of the source to the target shape. To allow robust optimization of the elastic energy, some modifications of the energy have been made, and the modified energy is applicable to highly degenerate initializations. Since the energy is not linear in this case, its optimization is slower than the optimization of the Dirichlet energy, but the results are more accurate. The method is presented in section 4.

Finally, in section 5 a method that utilizes shape correspondence for deep learning is described [ESKBC17]. While deep learning methods are widely used for various tasks that involve *images*, applying deep learning techniques on 3D data is more challenging. The reason for the difference is that while images can be easily represented by a uniform structure (array of pixels), 3D data is usually not uniformly structured. If a uniform representation of 3D data (such as a voxel grid) is used, deep learning techniques can be applied, but such a representation usually does not make use of *intrinsic* properties of shapes. For example, a volumetric grid would encode completely differently shapes of the same object in a different pose. To overcome this problem, one can use shape correspondence to map *geometric descriptors* from a shape to a 2D grid, forming an image that represents the shape, that can be used as input to standard neural networks that operate on images. If the correspondence preserves pairwise distances, isometric shapes would have a similar image representation. We proposed a method that computed such a representation, where the correspondence is *optimized for the application*, which was classification and retrieval of 3D shapes in our case. Similarly, this method can be extended to other downstream applications that are not in the scope of this research.

To conclude, this thesis describes a few methods for non-isometric shape correspondence, that differ in the input type and the intended application. Studying correspondence in a context of specific applications is crucial, as different applications may require significantly different correspondence between shapes. We devised methods that can be used for various applications, such as texture transfer, morphing, quad transfer and classification.

1.1 Related Work

Shape correspondence, or shape matching, is a fundamental topic in geometry processing, with a considerable body of existing work (see [VKZHCO11, TCL+13] for a detailed survey of this field). We will focus our literature review on the shape correspondence methods that are most relevant to this thesis.

1.1.1 Fully Automatic Shape Correspondence

Kim et al. [KLF11] suggested one of the first fully automatic methods for non-isometric shape matching, that consistently generates high-quality results on a benchmark of shapes. This approach, denoted by BIM, generates a blending of conformal maps, with blending weights optimized to minimize isometric distortion. While providing excellent results in many cases, BIM is limited to genus-zero surfaces and can introduce large distortions for some shapes. Recently, Zheng et al. [ZWL+17] suggested to map between high-genus surfaces with the same genus, by decomposing the surfaces using a pants decomposition, and then computing harmonic maps between a set of intermediate cylindrical domains. This leads to a piecewise harmonic map between the input surfaces, which is further relaxed using geodesic heat flow. While this approach can be used without user intervention, if a globally semantic map is needed then accurate input landmarks are required, and it is limited to shapes of the same genus. Lahner et al. [LVB+17] suggested a method that computes *vertex-to-vertex* maps based on a set of matching descriptors and pairwise distances. While the method is robust to topological changes, it may leave areas unmapped, and is therefore less appropriate for applications such as texture transfer. Recently, a genetic algorithm was used for automatic computation of isometric shape correspondence [Sah18].

Some fully automatic methods compute a *sparse* correspondence between the shapes. For example, Kezurer et al. [KKBL15] formulated the shape correspondence problem as a Quadratic Assignment Matching problem, and suggested a convex semi-definite programming (SDP) relaxation to solve it efficiently. Dym et al. [DML17] suggested to combine doubly stochastic and spectral relaxations to optimize the Quadratic Assignment Matching problem, which is not as tight as the SDP relaxation, but much more efficient.

Functional maps. The functional map approach, introduced by Ovsjanikov et al. [OBCS+12], was originally designed for nearly-isometric shapes but has since been extended to non-isometric matching [KBB+13, PBB+13, SK14, KBBV15, ERGB16, BDK17, RPWO18]. This method generates a *generalized* map which puts in correspondence the function spaces on the mapped shapes (rather than points). Nogneng et al. [NO17] recently suggested a method that computes functional maps using fewer input descriptors by formulating commutativity constraints, and Huang et al. [HO17] suggested to use *adjoint* functional maps to improve and analyze correspondences. Using a hybrid approach, Maron et al. [MDK+16] optimize jointly for

a functional map and sparse correspondences, which are then extended to a dense vertex-to-vertex map. Recently, this framework was extended to computing partial correspondence [RCB+16, LRB+16, LRBB17], and to computing correspondences in shape collections [SBC14, HWG14, KGB16]. In addition, functional maps have been used for analysis and visualization of maps [OBCCG13, ROA+13], and image segmentation [WHG13]. See the recent survey [OCB+16] for a thorough overview of the functional map framework and its applications.

Other generalized representations. Solomon et al. suggested to use another generalized representation: *fuzzy* or a *soft* maps [SNB+12], that can be interpreted as a probability distribution over pairs of points, that determines the likelihood that each pair of points is in correspondence. Kim et al. [KLM+12] used fuzzy maps for exploring shape collections, and Solomon et al. [SPKS16] used the Gromov-Wasserstein objective to compute fuzzy maps between general domains, such as triangle meshes, point clouds or graphs. While generalized maps are beneficial for challenging mapping problems, such as mapping between shapes of different genus, extracting a precise map (*deblurring*) is a necessary post-processing step if the output map is to be used for transferring high-frequency data such as textures, normals, or deformations. A simple and efficient deblurring method has been proposed in [OBCS+12], and has been used in many subsequent papers. This approach is based on the assumption that indicator functions projected onto the reduced basis should correspond under a rotation in the spectral domain. However, while this assumption is suitable for isometric matching, in general cases it no longer holds, leading to inaccurate results. Other deblurring methods are described in the semi-automatic method section.

When the shapes are geometrically different and the optimal map is not expected to be isometric, the shape correspondence problem is ill-posed without additional *semantic* information. Such information can be given in the form of landmark constraints or an initial generalized map, from which a refined dense map can be computed.

1.1.2 Semi-automatic Shape Correspondence

Input: landmarks. Parameterization-based approaches compute bijective smooth maps to a common intermediate domain, and define precise maps between arbitrary shapes as the composition of the maps to the common domain. A variety of intermediate domains have been used in the literature, e.g. the plane [APL14, APL15], the sphere [GWC+04], the hyperbolic disk [SZS+17] and orbifolds [TFV+13, AL15, AL16, AKL17], to mention a few. These methods optimize the distortion of the map from the shapes to the target domain, but the composed map is not guaranteed in general to have low distortion. Furthermore, mapping through an intermediate domain places a topological restriction on the type of mapped shapes, as they should be topologically equivalent. Alternatively, Panozzo et al. [PBDSH13] compute a direct map between two

triangle meshes without requiring an intermediate domain. Their method computes on-surface barycentric coordinates with respect to the source landmarks, and then uses them with respect to the target landmarks to compute the corresponding point on the target shape. They use a high-dimensional Euclidean embedding to speed up the computation. Despite excellent results, in some cases a large number of landmarks is required to compute a correct correspondence. More recently, Mandad et al. [MCSK$^+$17] use landmarks or extrinsic alignment for initialization and then optimize simultaneously for a generalized map and a precise map.

Gehre et al. [GBKS18] suggested an interactive method, that gets corresponding *curves* as input from the user. Similarly to their method, that relied on functional maps [OBCS$^+$12], most methods that compute functional maps can easily incorporate soft landmark constraints.

Input: generalized maps. Rodolà et al. [RMC15] have suggested a deblurring approach that is suitable for non-isometric maps, by matching projected indicator functions using a *non-rigid* deformation, yet their approach is only applicable to shapes with the same number of vertices. This method was later extended to handle partial matching [RMC17]. Similar to our denoising approach, namely starting from an input pointwise map, Shtern et al. [SK14] use functional maps to refine the input map iteratively, by aligning the spectral kernels of the shapes. While their approach improves the ground truth error, it introduces significant conformal distortion in the map.

Vestner et al [VLB$^+$16, VLR$^+$17] recover a bijective vertex-to-vertex map by solving a linear assignment problem. While vertex-to-vertex bijections are beneficial for shapes with a similar triangulation, they highly depend on the tessellation of the input shapes. Furthermore, while remeshing the shapes to have the same number of vertices is possible, the sampled vertices are not likely to match bijectively, especially if the shapes are not isometric.

Finally, Corman et al. [COC15] as well as Azencot et al. [AVBC16] suggested a recovery method which reformulates the problem in terms of an unknown *vector field*. The output map is guaranteed to be continuous, as it is computed as the optimal *flow* which transports an arbitrary continuous initial map to the given functional map. Their setup requires as input a smooth pointwise map *in addition* to the input functional map.

1.1.3 Machine Learning Methods for Shape Correspondence

To deal with aspects of shape matching that are difficult to capture using mathematical models alone, some methods compute correspondences using concepts from machine learning, e.g. support vector machines [SSB05], PCA [SBS06], random forests [RRBW$^+$14], subspace analysis [COC14]. Litman et al. [LB14] suggested a method that learns shape descriptors, inspired by Wiener filter. More recently, deep learning approaches were used for shape correspondence. In [WHC$^+$16] they constructed an artificial neural network

that maps between depth maps: they rendered the 3D shapes from a certain angle, to generate an embedding of the geometry as an image. This approach depends on the view direction that was used for the rendering. Other methods use local isotropic or anisotropic filters in intrinsic convolution layers [MBBV15, BMRB16, MBM+17]. Haibin et al. [HKC+17] used the multi-view approach, where shapes are rendered from different angles and the multiple images represent the shape. Litany et al. [LRR+17] suggested *deep functional maps*, a network architecture that is based on functional maps.

While the mentioned deep learning methods depend on available labeled training data, Halimi et al. [HLR+18] as well as Roufosse et al. [RO18] suggested *self supervised* methods that do not necessarily require labeled data. Both methods relied on the deep functional map architecture [LRR+17], but suggested different quality criteria; Halimi et al. [HLR+18] relied on pairwise distance preservation, while Roufosse et al. [RO18] relied on the functional map energy terms that were used by Ren et al. [RPWO18].

Chapter 2

Deblurring and Denoising of Maps between Shapes

Since computing a high-quality map between two general surfaces is challenging, recent approaches suggest to relax the concept of a pointwise map and use *generalized* map representations. Such approaches put in correspondence, for example, functions [OBCS+12] or probability distributions [SPKS16] instead of points. While such generalized maps can successfully tackle challenging scenarios, e.g. matching between surfaces with different topologies [SPKS16] and partial matching [RCB+16], some applications do require a high quality pointwise map for transferring information between the shapes. This issue is exacerbated when the data to be transfered changes rapidly on the surface, and thus has *high frequencies*, such as in texture transfer or map-aware quadrangulation [PLPZ12]. Unfortunately, generalized maps often hold information only about the correspondence of smooth, *low frequency* functions, due to the use of the truncated eigenfunctions of the Laplace-Beltrami operator as a basis for representing the map [OBCS+12], or due to the entropy incorporated in the map to improve efficiency [SPKS16]. Hence, successful *map deblurring*, namely extracting a high-quality pointwise map from a semantic low frequency map, is paramount to the usability of generalized maps in applications.

A good map deblurring technique should fulfill a few required properties. First, it should be applicable in a general setting, without requiring the input shapes to be close to isometric, or the output map to be bijective. Otherwise, we may lose in this step the benefits we have gained by using generalized map representations. Second, the deblurred map should have a low conformal distortion, to avoid distorting textures during transfer. This requirement implies that the map should be a vertex-to-point map, also denoted as a *precise* map, where each vertex on the source shape is mapped to a point anywhere on the target surface and not necessarily to a vertex. Finally, the deblurred map should be as robust to the triangulation of the target surface as the generalized map. While map deblurring techniques exist, none fulfill all the required properties.

We propose a new method for map deblurring, by introducing a smoothness prior on the reconstructed map. Surprisingly, this straightforward approach results in better

maps than using existing approaches, especially when the shapes are not isometric. While we do not have a theoretical guarantee on the distortion bounds implied by our prior, our method generates in practice precise maps with lower conformal distortion than existing methods for deblurring and denoising. In addition, our technique can be easily incorporated into existing map computation pipelines, significantly improving the results. Furthermore, using the same framework we can perform *map denoising*, by projecting a given noisy map to a blurred map and reconstructing. Finally, we show that our approach outperforms the state of the art for a benchmark of non-isometric shapes, as well as show applications to map extraction from computed functional maps, and high quality intrinsic symmetry computation for challenging surfaces.

2.1 Method

Notation. We represent a triangle mesh M by its vertex set and face set $(\mathcal{V}, \mathcal{F})$, where we denote $n = |\mathcal{V}|, m = |\mathcal{F}|$, and its embedding by $V \in \mathbb{R}^{n\times 3}$. $\mathcal{H}(M)$ denotes the space of continuous piecewise linear functions on M, and $\mathcal{S}(M)$ denotes a space of functions given in a reduced basis of size k. The basis transformation between $\mathcal{S}$ and $\mathcal{H}$ is given by a matrix $\Psi \in \mathbb{R}^{n\times k}$, whose columns are the basis elements. The projection from the full space to the reduced space is given by the pseudo-inverse of the basis matrix, $\Psi^\dagger$. We denote scalar functions $f : M \to \mathbb{R}$ by their vector of coefficients in a basis, with either $f \in \mathbb{R}^n$ or $f \in \mathbb{R}^k$, for the full and reduced basis, respectively. The squared norm of a function on the surface is given by $\|f\|_M^2 = f^\top A f$, where $A \in \mathbb{R}^{n\times n}$ is the diagonal (lumped) mass matrix of the vertices. Similarly, for matrices we use the matrix trace: $\|F\|_M^2 = \mathrm{Tr}(F^\top A F)$. We denote the i-th row and j-th column of a matrix F by F_{i*} and F_{*j}, correspondingly.

When two meshes are involved we use a subscript, for example $\mathcal{H}_i = \mathcal{H}(M_i)$ is the space of piecewise linear functions on M_i. A pointwise map between two triangle meshes is denoted by $T_{12} : \mathcal{V}_1 \to M_2$, and it can be applied to any vertex v on M_1 to give any point $p \in \mathbb{R}^3$ on M_2 (not restricted to the vertices). The matrix $T_{12}(\mathcal{V}_1) \in \mathbb{R}^{n_1\times 3}$ thus represents the 3D coordinates of the mapped vertices of M_1. Maps between the functional

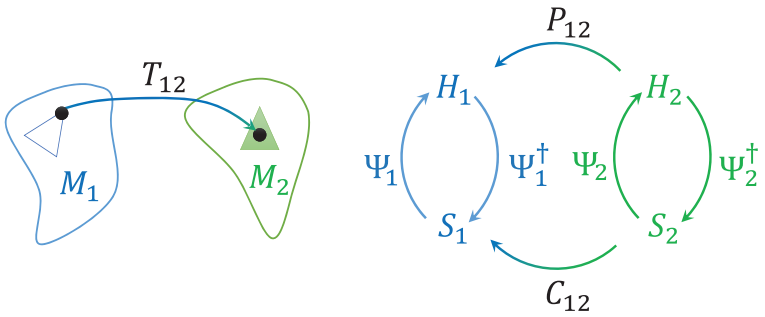


Figure 2.1: Our notation, see the text for details.

spaces are denoted by $C_{12} : \mathcal{S}_2 \to \mathcal{S}_1$ and $P_{12} : \mathcal{H}_2 \to \mathcal{H}_1$, and are represented by matrices $C_{12} \in \mathbb{R}^{k_1 \times k_2}$ and $P_{12} \in \mathbb{R}^{n_1 \times n_2}$, respectively. These spaces and transformations are visualized in Figure 2.1.

Background. The term *functional map* [OBCS+12], denotes a map between scalar functions on the two shapes. Given a pointwise map $T_{12} : \mathcal{V}_1 \to M_2$, its functional representation in the hat basis $P_{12} : \mathcal{H}_2 \to \mathcal{H}_1$ fulfills [OBCS+12]

$$(P_{12}f)(v) = f(T_{12}(v)), \quad \forall v \in \mathcal{V}_1, f \in \mathcal{H}_2.$$

The embedding of M_2 plays a special role in the relation between the functional and pointwise maps, as by definition we have:

$$T_{12}(\mathcal{V}_1) = P_{12}V_2.$$

We define the feasible set $\mathcal{P}_{12}$ such that $P_{12} \in \mathcal{P}_{12}$ if and only if there exists a map T_{12} such that $P_{12} = P(T_{12})$, where P is the operator that converts a vertex-to-point map to a matrix, which we will describe later.

To represent the functional map in a reduced basis $C_{12} : \mathcal{S}_2 \to \mathcal{S}_1$, we apply the basis transformations on both sides and get:

$$C_{12} = \Psi_1^\dagger P_{12} \Psi_2.$$

A common choice for the basis functions Ψ is the first k eigenfunctions of the Laplace-Beltrami (LB) operator, such that smooth functions can be well approximated using a small number of coefficients. We use the standard area weighted cotangent LB operator [BKP+10], thus we have $\Psi^\top A \Psi = Id$ and $\Psi^\dagger = \Psi^\top A$.

2.1.1 Map Deblurring

Given a map in a reduced basis, C_{12}, our goal is to find the "best" corresponding pointwise map T_{12}. We formalize this using the following optimization problem:

$$\begin{aligned} \underset{P_{12}}{\text{minimize}} \quad & R(P_{12}) + \|C_{12} - \Psi_1^\dagger P_{12} \Psi_2\|_F^2 \\ \text{subject to} \quad & P_{12} \in \mathcal{P}_{12} \end{aligned}, \tag{2.1}$$

where R is some regularizer that favors "good" maps. Given P_{12} we extract the map $T_{12}(\mathcal{V}_1) = P_{12}V_2$.

We suggest to incorporate a *smoothness* assumption, namely:

$$P_{12}\Psi_2 \in \text{span}(\Psi_1). \tag{2.2}$$

Intuitively, out assumption implies that functions on M_2 that are well represented

with Ψ_2 will be well represented with Ψ_1 after applying the map. When Ψ_i are the eigenfunctions of the LB operator this assumption implies that the map P_{12} does not introduce spurious high frequencies. To incorporate the smoothness prior into the optimization problem, we use the regularizer

$$R(P_{12}) = \|(Id - \Psi_1\Psi_1^\dagger)P_{12}\Psi_2\|_{M_1}^2, \tag{2.3}$$

which penalizes the component of $P_{12}\Psi_2$ that is orthogonal to Ψ_1.

Figure 2.2 illustrates that indeed, in practice, for large enough values of k_1 our assumption 2.2 holds. We show the value of the regularizer $R(P_{12})$, where P_{12} is the ground truth map, as a function of k_1 for a fixed $k_2 = 50$. We use two pairs of shapes from FAUST [BRLB14]: the same target M_2, and two source shapes M_1^I, M_1^{NI}, representing shapes isometric and non-isometric to M_2 respectively: M_1^I is the same person in a different pose, and M_1^{NI} is a different person in the same pose. Note that while for M_1^I the error reduces greatly when k_1 reaches 50, for M_1^{NI} we need a larger k_1, but after it is reached, the error drops.

Incorporating the regularizer (2.3) into the optimization problem (2.1) leads to the optimization problem:

$$\begin{aligned} \underset{P_{12}}{\text{minimize}} \quad & \|\Psi_1 C_{12} - P_{12}\Psi_2\|_{M_1}^2 \\ \text{subject to} \quad & P_{12} \in \mathcal{P}_{12} \end{aligned} . \tag{2.4}$$

We formally prove the equivalence between these optimization problems in Appendix A.1. Intuitively, $P_{12}\Psi_2$ are functions in $\mathcal{H}_1$, which can be represented using their projection on the basis Ψ_1 and the projection on its orthogonal complement $\Psi_1^\perp$. The term $\|C_{12} - \Psi_1^\dagger P_{12}\Psi_2\|_F^2$ only constrains the projection of $P_{12}\Psi_2$ on Ψ_1 to be close to the data C_{12}, and says nothing about the projection on the orthogonal complement. We add as a regularizer the requirement that the projection on the orthogonal complement is as small as possible, thus fully specifying constraints on $P_{12}\Psi_2$, leading to the second optimization problem.

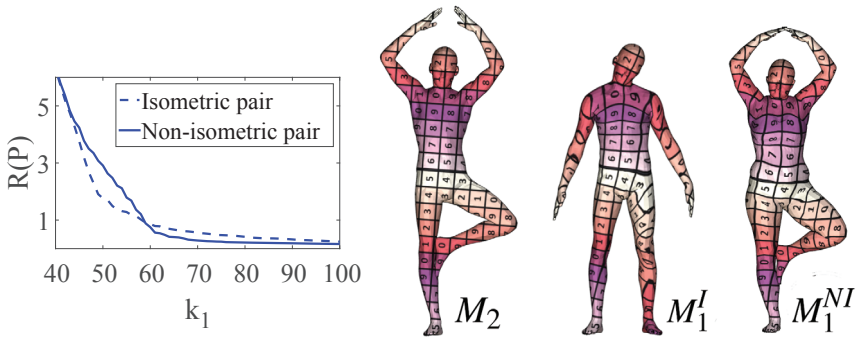


Figure 2.2: The value of the regularizer $R(P)$ for a pair of isometric (M_1^I, M_2) and non-isometric (M_1^{NI}, M_2) shapes from FAUST [BRLB14]. See the text for details.

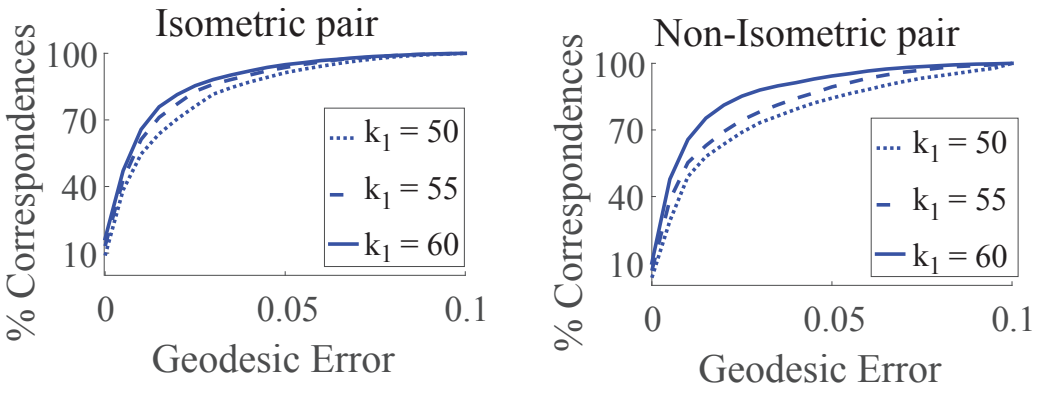


Figure 2.3: Deblurring with a varying number of basis functions k_1 for an isometric (left) and non-isometric (right) pair. A larger k_1 is required to achieve the same error for a non-isometric pair.

An important advantage of our formulation is that the objective and the constraint are *row separable* in P_{12}. Therefore, to fulfill the difficult constraint that P_{12} is in the feasible set $\mathcal{P}_{12}$, we can solve separately for each row of P_{12}, finding a global minimizer of the optimization problem. We elaborate on the numerical approach for solving the optimization problem in the next Section, and first demonstrate some illustrative results.

We explore the parameter choice for our deblurring method using the same two pairs from Figure 2.2. We keep $k_2 = 50$ fixed, and vary $k_1 \in [50, 55, 60]$. We compute the blurred map from the ground truth map as $C_{12} = \Psi_1^\dagger P_{12} \Psi_2$, deblur it by solving the optimization problem (2.4), and measure the error with respect to the ground truth. To avoid bias in the results by using the same triangulation for both meshes, we have remeshed the FAUST dataset, and propagated the ground truth map to the new meshes. Figure 2.3 shows the resulting error graphs, where we use the same protocol as in [KLF11]. As expected, taking larger values for k_1 leads to a smaller error, where in general to achieve the same error, larger values are required for non-isometries than for isometries. Note that, in contrast to original deblurring approach [OBCS+12], increasing k_1 does not affect the complexity of our approach, as we measure distances in $\mathbb{R}^{k_2}$.

2.1.2 Map Denoising

Given a noisy pointwise map $\tilde{P}_{12}$ we would like to improve it. Using our smoothness prior, we optimize for a map such that the projection of $P_{12}\Psi_2$ on Ψ_1 is close to the input map's projection, and the projection on $\Psi_1^\perp$ is minimal. This leads to:

$$\begin{aligned} \underset{P_{12}}{\text{minimize}} \quad & R(P_{12}) + \|\Psi_1^\dagger \tilde{P}_{12} \Psi_2 \ - \Psi_1^\dagger P_{12} \Psi_2\|_{M_1}^2 \\ \text{subject to} \quad & P_{12} \in \mathcal{P}_{12} \end{aligned} , \tag{2.5}$$

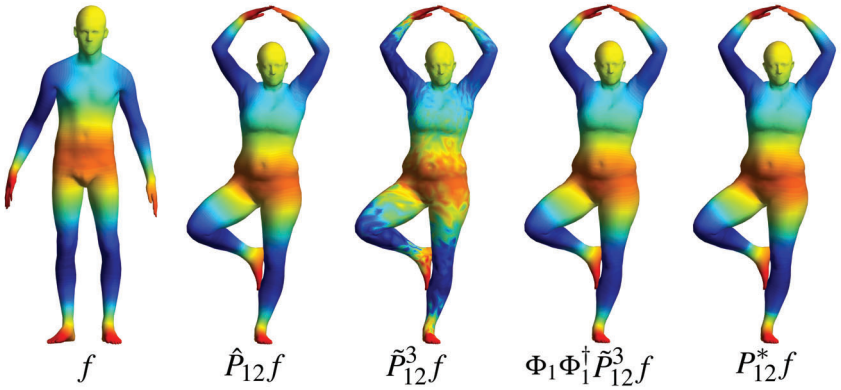


Figure 2.4: r-ring noise and low pass effect on $P_{12}\Psi_2$. From left to right: eigenfunction on the target f, f pulled back to the source using the ground truth map, the noisy map, and the smoothed noisy map. The last image on the right is f pulled back to the source using our denoised map. See the text for details.

which is equivalent to the optimization problem from Equation (2.4), when taking $C_{12} = \Psi_1^\dagger \tilde{P}_{12}\Psi_2$. Intuitively, we are removing the high frequencies in $\tilde{P}_{12}$ by blurring it, and then reconstructing the best pointwise approximation using our deblurring approach.

Figure 2.4 demonstrates the effect of projecting $P_{12}\Psi_2$ to the span of Ψ_1. Starting from a ground truth map $\hat{P}_{12}$ between the non-isometric pair from the previous experiment, we introduce noise by randomly mapping each vertex to one of its r-ring neighborhood, with $r \in [3, 4, 5]$, yielding the noisy maps $\tilde{P}^r_{12}$. The figure shows the 10-th eigenfunction of M_2, which we denote by f, on the target shape (left), the same function mapped to M_1^{NI} using the ground truth map $\hat{P}_{12}f$, using the 3-ring noisy map $\tilde{P}^3_{12}f$ and using the noisy map and projected to the span of Ψ_1 (center). The solution to the optimization problem (2.5) P^*_{12} has a smoothing effect on the input. See the resulting reconstructed function $P^*_{12}f$ on the right.

Figure 2.5 illustrates the result of denoising these maps, by showing the total average error as a function of k_1 (left), and the error graph as a function of the noise r (right). Note that the error decreases as k_1 is increased, until a minimum is reached and then the error increases back, since for larger k_1 values we have enough eigenfunctions to reconstruct the noisy input. Furthermore, the optimal k_1 depends on the amount of noise: higher noise requires a smaller k_1 to increase the smoothing effect.

2.1.3 Relation to FMaps 2012

Ovsjanikov et al. [OBCS+12] suggested the following objective for map deblurring:

$$\underset{P_{12}}{\text{minimize}} \quad \|(\Psi_1^\dagger)^\top - P_{12}(C_{12}\Psi_2^\dagger)^\top\|_F^2 \quad , \tag{2.6}$$

subject to the constraint that P_{12} is binary row stochastic. The rationale was that the columns of $\Psi_i^\dagger$ represent the coefficients of delta functions on M_i in the reduced basis,

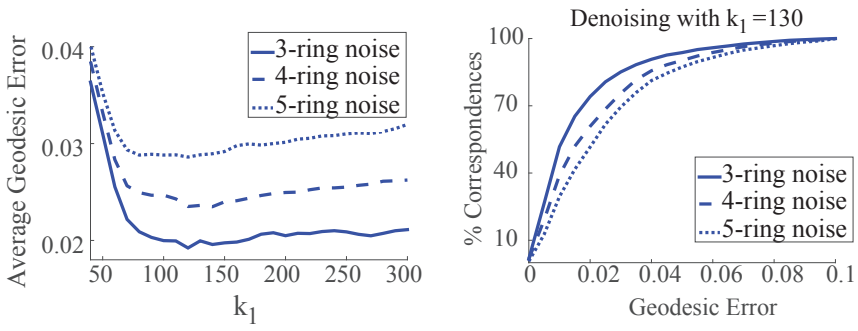


Figure 2.5: Map denoising of r-ring noise, varying parameters. (left) varying k_1, total average error, (right) error graph for fixed $k_1 = 130$ for different r. See the text for details.

and the optimal vertex-to-vertex map should put them in correspondence. However, the original approach was geared towards volume preserving maps, in which case C_{12} is a rotation matrix, namely $C_{12}^\top C_{12} = Id$. Indeed, in this case, when taking orthogonal bases, equations (2.6) and (2.4) are equivalent (to see that, take $\Psi_i^\dagger = \Psi_i^\top$ in (2.6), and then multiply by C_{12} from the right).

However, when the shapes are considerably different, the projections of delta functions of corresponding vertices on Ψ_i are no longer expected to correspond. Figure 2.6 demonstrates that: we take two highly non-isometric shapes, pick a vertex $v_1 \in \mathcal{V}_1$ (left) and measure for all vertices $v_2 \in \mathcal{V}_2$ the distance $\|(\Psi_1)_{v_1 *} C_{12} - (\Psi_2)_{v_2 *}\|_2$ where C_{12} is a given ground-truth map. This distance is then shown as a function on M_2 (center). This is the distance that our approach, Equation (2.4), aims to minimize. Note that small values are achieved in a correct zone of the mesh, and the point with the minimal distance is a correct match to v_1. We additionally show the distance $\|C_{12}(\Psi_2^\dagger)_{*v_2} - (\Psi_1^\dagger)_{*v_1}\|_2$ (right), which represents the distance in $\mathbb{R}^{k_1}$ that Equation (2.6) aims to minimize. Note that now many regions are close to v_1 and the point with the minimal distance is now an incorrect match to v_1. See in addition Figure 2.7 which shows reconstruction from a

Figure 2.6: Delta functions represented in a reduced basis do not necessarily correspond under the ground truth map if the shapes are considerably different. (left) A vertex v_1 on M_1. (center) the distance in $\mathbb{R}^{k_2}$ that our formulation minimizes. (right) the distance between mapped delta functions and the delta function of v_1. Note that only our distance yields the correct match to v_1 on M_2.

computed functional map using [OBCS+12] and using our approach.

2.2 Optimization

2.2.1 Feasible set

The main challenge in solving the optimization problem (2.4) is fulfilling the constraint that P_{12} is in the feasible set $\mathcal{P}_{12}$. We use the following definitions:

Definition. Given M_1, M_2, the set F_2 of *valid rows* of P_{12} is defined as follows: $w \in F_2$ if and only if $w \in \mathbb{R}^{1\times n_2}$ has at most three non-zero entries $(\omega_1, \omega_2, \omega_3)$ at columns (c_1, c_2, c_3), respectively, the vertices c_i form a face $f \in \mathcal{F}_2$, $\omega_i \geq 0$ and $\sum_i \omega_i = 1$. The set $\mathcal{P}_{12}$ of *valid matrices* is defined as follows: $P \in \mathcal{P}_{12}$ if and only if $P \in \mathbb{R}^{n_1\times n_2}$ and every row of P is in F_2. The operator $P(T_{12})$ constructs a matrix from a map as follows: let $T_{12}(v_i) = p \in M_2$, which lies in the face f with barycentric coordinates ω. Set the i-th row of P_{12} to all zeros except at the vertices of f, and there use the values ω. It is straightforward to show that $P_{12} \in \mathcal{P}_{12}$ if and only if there exists T_{12} such that $P_{12} = P(T_{12})$.

2.2.2 Row separability

The computational advantage of the optimization problem (2.4) is that it is separable in the rows of P_{12}. To see that, note that the objective is of the form $\|B\|_M^2 = \|\sqrt{A}B\|_F^2 = \sum_{i=1}^{n}(A)_{ii}\|B_{i*}\|_2^2$. Furthermore, the constraint on P_{12} is also row separable, as a matrix is in the valid set $\mathcal{P}_{12}$ if and only if all its rows are in the valid rows set F_2. Hence, we solve n_1 small optimization problems, for the rows of P_{12}, of the form:

$$\underset{w_i\in F_2}{\text{minimize}} \quad \|(\Psi_1)_{i*}C_{12} - w_i\Psi_2\|_2^2 \,, \tag{2.7}$$

for $i \in [1, n_1]$, and then set the i-th row of P_{12} to the value of the minimizer.

2.2.3 Implementation

Vertex to vertex maps. In [OBCS+12] it was noted that map deblurring can be considered as a point-correspondence problem in $\mathbb{R}^k$. Our formulation has the same structure, and if we only need a vertex-to-vertex map, we can use the same nearest neighbor approach to solve (2.4). Note that in this case we are reducing the feasible set to binary row stochastic matrices, which are a strict subset of $\mathcal{P}_{12}$. Hence, while this approach is more efficient than extracting precise maps, it yields maps with a higher conformal distortion that are more sensitive to the triangulation (see Figure 2.11).

Precise Maps. When working with the full feasible set of precise maps $\mathcal{P}_{12}$, nearest neighbor search is not enough. In fact, we are effectively considering an embedding of

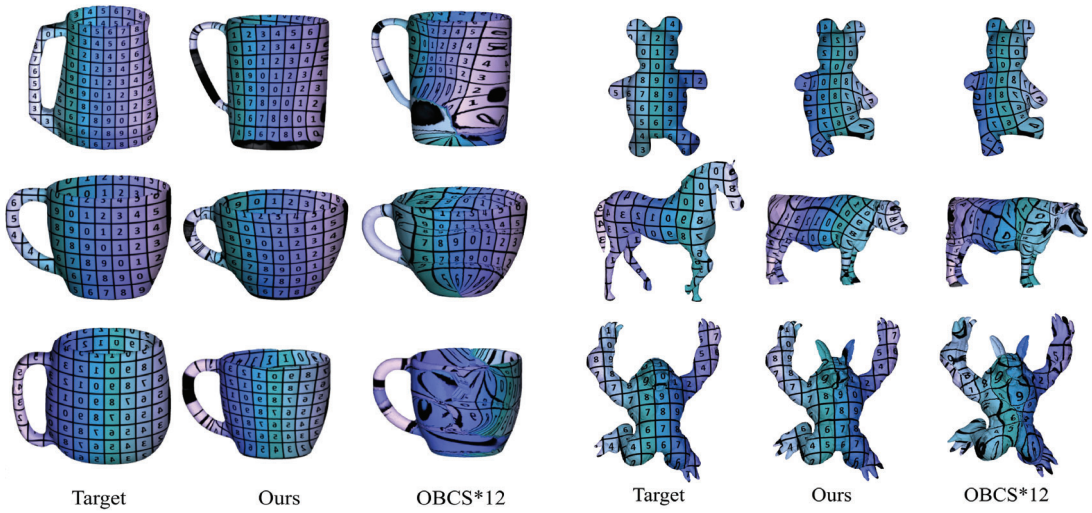


Figure 2.7: We compute functional maps using a few landmark constraints and extract a precise map to transfer texture. The figure shows the target surface M_2 (left), and the texture pulled back to the source surface M_1 using the map computed with our deblurring approach (center) and the original recovery method [OBCS$^+$12] (right). Note, that while for similar surface the original method performs well (e.g. the teddy models), for surface which undergo large deformations our recovery is considerably better.

the triangle mesh M_2 in $\mathbb{R}^{k_2}$ given by Ψ_2, and we need to project on it the n_1 points $\Psi_1 C_{12}$. To solve (2.7) for a vertex $v_i \in \mathcal{V}_1$, we need:

$$\underset{f\in\mathcal{F}_2}{\text{minimize}} \quad \underset{\omega\in\mathbb{R}_+^{1\times 3},\sum\omega=1}{\text{minimize}} \quad \|(\Psi_1)_{i*}C_{12} - \omega\,(\Psi_2)_{f*}\|_2^2\,, \tag{2.8}$$

where $(\Psi_2)_{f*} \in \mathbb{R}^{3\times k_2}$ are the rows of Ψ_2 corresponding to the vertices of the face f. From the minimizer we generate w_i, the solution to (2.7), by setting w_i at the locations given by the vertices of f to the values in ω, which leads to a feasible solution $w_i \in F_2$.

Thus, for each vertex of M_1, we iterate over the faces of M_2, solve for each face a linear least squares problem with linear constraints for ω, and pick the face and the corresponding ω which minimize the error. The least squares problem is solved using a straightforward generalization to $\mathbb{R}^k$ of the algorithm that projects a point to a triangle, see e.g. [Ebe].

We therefore need to solve m_2 constrained optimization problems for each vertex of $\mathcal{V}_1$ which is prohibitive for large meshes. We can gain a considerable speedup by using nearest neighbors queries to identify faces which *cannot* be minimizers without explicitly solving the optimization problem. For example, for FAUST meshes this procedure allowed us to solve the optimization problem only on 0.5 percent of the faces. The details of the algorithm are provided in Appendix A.2.

Limitations While we have demonstrated the relation between k_1 and the noise ratio, in practice the choice of best k_1, k_2 values is highly dependent on the two shapes and

the application. It is an interesting direction for future work to try to estimate k_1 from the noise level for different datasets and mapping methods.

Even with the reduction in the number of candidate faces, the time required for computing a precise map is still considerably higher than using k-nearest neighbors. For example, our naive Matlab implementation on the CPU takes 55s for a mesh with 10k faces on a standard laptop. Note, though, that the problem is highly parallelizable as we solve for each row of P_{12} independently. A parallel implementation could therefore potentially be used to increase the performance.

Finally, we have only considered the l_2 norm, and can thus handle only local high frequency noise. It might be beneficial to generalize to other norms to make the method more robust to outliers, e.g. if the map has concentrated noise in some region.

2.3 Applications

2.3.1 Map Deblurring

Recovery from computed functional maps. We compute functional maps for a few shapes from SHREC07 [GBP07], using the landmarks provided in the BIM benchmark. We used the Wave Kernel Map as landmark descriptors as well as Wave

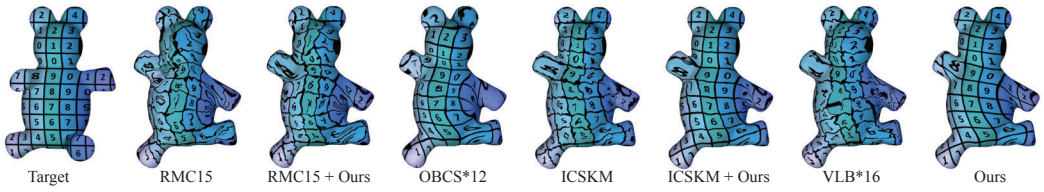


Figure 2.8: Qualitative comparison with previous methods for functional map deblurring. The input functional map was computed using a few landmarks. "+ Ours" indicates that Eq. (2.4) was used. Note, that our method yields the best results, see especially the ear and the hands.

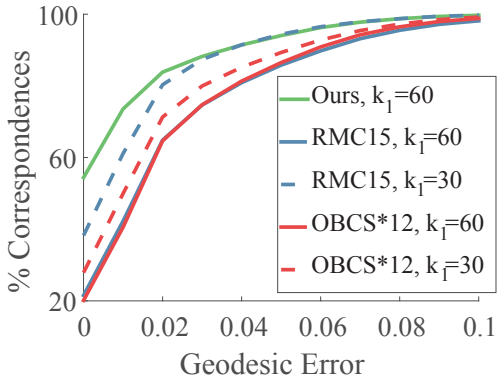


Figure 2.9: Deblurring the FAUST dataset, starting from the ground truth map. See the text for details.

Kernel Signature [ASC11], *without any additional constraints*. When computing a functional map from a sparse set of landmarks performing the iterative ICP based recovery from [OBCS+12] is essential for getting reasonable maps. We show in Figure 2.7 the deblurred map computed using our approach and the original approach, where we incorporated our reconstruction into the ICP process. In this case, we do not have ground truth data, but as the figure qualitatively shows, we obtain considerably better maps using our approach.

We additionally perform a qualitative comparison to the recovery methods ICSKM [SK14], and [RMC15, VLB+16]. The shapes were resampled to have the same number of vertices in order to accommodate the latter two methods. Figure 2.8 shows the results of the original deblurring methods as well as combinations of [RMC15] and ICSKM with our method by using Equation (2.4) during the optimization. Note that our method achieves the best results, see, e.g., the ear and the hands of the model.

Recovery from ground truth. To isolate the effect of our deblurring procedure from the map computation algorithm, we check the accuracy of our deblurring when full information is given. We use 45 non-isometric pairs of shapes from the FAUST dataset, remesh so they do not share the same connectivity, convert their ground truth maps to a functional map and then deblur them using our approach and competing approaches. We compare to the original map deblurring approach from [OBCS+12], and to the approach by Rodolà et al. [RMC15]. We use $k_2 = 30$, and k_1 as is shown in Figure 2.9. Note that our approach outperforms both methods. It is worth noting that the competing approaches in fact perform better with a *smaller* number of eigenfunctions. In following experiments we have used the best k_1, k_2 parameters for each method.

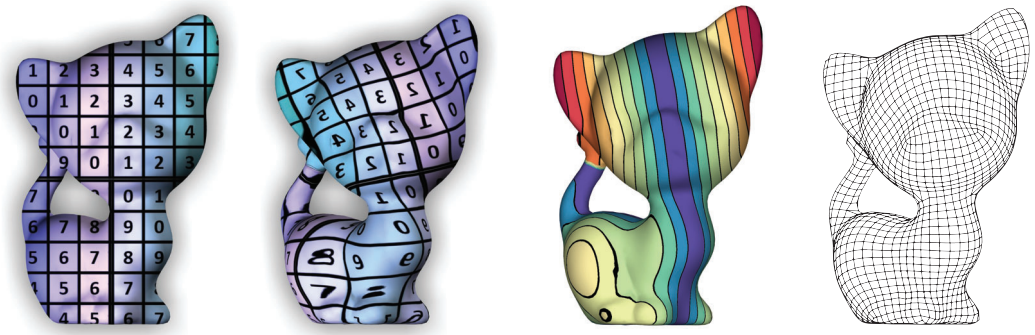


Figure 2.10: Intrinsic symmetry on a high genus surface, deblurred by our method from a computed functional map. (left) the pulled back texture through the precise map. (center) the isolines of the distance between each vertex and its image under the map, (right) the resulting quad mesh when this map was used as input to a symmetric quadrangulation method [PLPZ12].

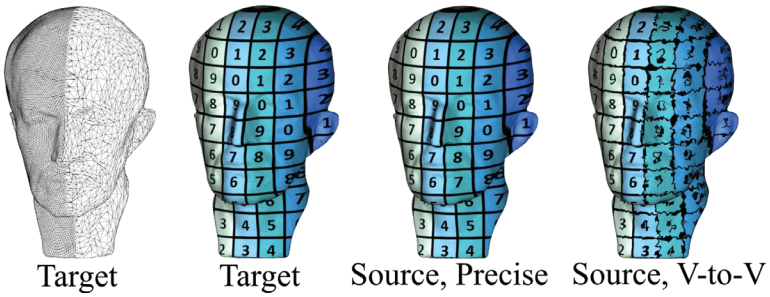


Figure 2.11: Tesselation invariance of our precise maps. (left) target model and texture, (center) pulled back texture using a precise map, and (right) using a vertex-to-vertex map, both extracted by deblurring the ground truth functional map. Note that the texture pulled back with the precise map is indistinguishable from the original.

High quality intrinsic symmetries. Generalized maps are especially useful for computing maps in difficult cases where tailored approaches are not available. For example, intrinsic symmetry can be extracted in a variety of ways, but no method exists for generating a *precise* map for intrinsically symmetric high genus surfaces which can be used in applications which require a high quality map. We use the functional map framework, with a few user chosen landmarks, to generate a functional symmetry map for the kitten model. We then apply our deblurring method to compute a high quality precise symmetry map. In Figure 2.10 we show the pointwise map recovered using our approach. The figure shows the texture pulled back through the map (left), as well as the Euclidean distance between each point and its image (center), which identifies the symmetry line. We further fed this map to the symmetric quadrangulation method by Panozzo et al [PLPZ12], and succeeded in generating a high quality symmetric quad mesh (right).

Tessellation invariance. The ability to extract precise maps is especially important to avoid tessellation dependence. To demonstrate that, we deblur the ground-truth map between a shape and its re-tessellation. Figure 2.11 shows the re-tessellated surface M_2 with its texture (left), and the texture pulled back to the original surface M_1 using the deblurred precise (center) and vertex-to-vertex (right) maps. Note the notably higher quality for the precise maps.

2.3.2 Map Denoising

Improving conformal distortion. The ICSKM method for map denoising [SK14] is highly effective for improving the ground truth error of noisy maps, yet it introduces high conformal distortion. Since it uses the same recovery method as [OBCS$^+$12], we can simply replace it with our recovery method. We used BIM to generate maps between 45 non-isometric pairs of shapes from the FAUST dataset, and then applied different denoising approaches. As Figure 2.12 shows, using our approach with ICSKM yields a high quality map, where both the ground truth error and the conformal distortion are

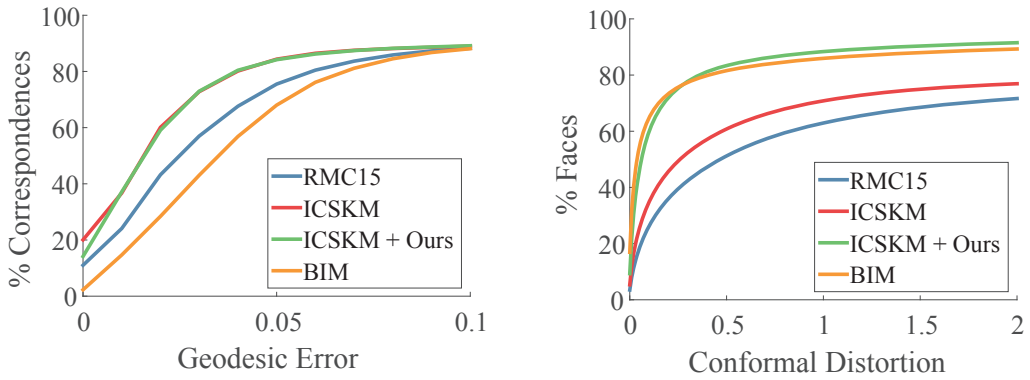


Figure 2.12: We apply our algorithm in tandem with [SK14] to extract high quality maps for FAUST by denoising maps obtained with BIM. See the text for details.

low. We use the definition by Hormann et al. [HG00] (equation 3 in their paper) for conformal distortion and subtract 2 so that the minimal conformal distortion is zero.

Chapter 3

Reversible Harmonic Maps between Discrete Surfaces

In applications such as texture transfer and shape interpolation, desirable correspondences satisfy some basic key properties: they should be *smooth* to avoid introducing geometric noise during transfer; they should *preserve semantic features* to ensure that key features are put in correspondence; and they should be *reversible*, namely invariant to which of the two shapes is chosen as the source.

Many existing approaches to shape mapping focus on generating maps with low *global* distortion (e.g. preserving pairwise distances [SY11]) at the expense of large local distortion, which reduces the quality of the correspondence and hinders downstream applications. On the other hand, approaches that minimize local distortion measures mostly require an intermediate domain and construct the final map as a composition through this domain (e.g. [AL16]). While such methods minimize distortion of the maps into the intermediate domain, the distortion of the composed map can be large. This problem is exacerbated when the input shapes have significantly different geometric features, such as four-legged animals with different dimensions, e.g. a cat and a giraffe. In this case, the *isometric distortion* of the optimal map is expected to be large, and thus minimizing the distortion of the two maps into an intermediate domain is quite different from minimizing the distortion of the composition.

We propose a novel approach for computing a smooth and reversible map between surfaces that are not isometric to each other, without requiring an intermediate domain. We incorporate semantic information by starting from some user guidance given in the form of sparse landmark constraints or a functional correspondence. Our main contribution is the formulation of an optimization problem whose objective is to minimize the *geodesic Dirichlet energy* of the forward and backward maps, while maximizing their reversibility. We compute an approximate solution to this problem using a high-dimensional Euclidean embedding and an optimization technique known as *half-quadratic splitting* [GY95]. We demonstrate that our maps have considerably lower local distortion than those from state-of-the-art methods for the difficult case of non-isometric deformations. We

further show that our maps are semantically accurate by measuring their adherence to self-symmetries of the input shapes, their agreement with ground-truth when the deformation is known, and their compatibility with human-generated segmentations.

3.0.1 Contributions

We present an algorithm for shape correspondence between non-isometric triangular meshes, that has the following advantages:

- The algorithm is widely applicable, and the resulting maps are semantic and exhibit low conformal distortion.
- The formulation is simple and efficient to optimize and thus can be combined with additional energy terms and various initializations.
- The maps are accurate enough for downstream applications, such as shape interpolation and quad mesh transfer.

3.1 Background: Harmonic Maps and Local Distortion

Suppose $M_1, M_2 \subseteq \mathbb{R}^3$ are smooth, compact surfaces with or without boundary. Given a map $T_{12} : M_1 \to M_2$ from one into another, a natural task is to measure the distortion of M_1 as it is mapped via T_{12} onto M_2; this distortion measure eventually will serve as an objective function for optimization problems whose unknown is the correspondence T_{12}. The basic role of these distortion measures is to evaluate whether nearby points are mapped to nearby points under T_{12}, at least differentially, a common proxy for the quality of the map.

In the theory of differential geometry, a key distortion measure is the *Dirichlet energy* $E[\cdot]$ (defined below) of T_{12}; minimizers of $E[\cdot]$ are called *harmonic maps*. Intuitively, if we think of M_1 as a rubber sheet, a harmonic map represents an equilibrium position of the sheet after stretching it over M_2 and letting it compress. The Dirichlet energy and its minimizers find many roles in the geometry processing literature, most prominently in surface parameterization [LPRM02b], due to its intuitive measurement of distortion and connections to notions of conformality. At the same time, theory and practice of harmonic mapping become considerably more challenging when M_2 has areas of positive curvature; intuitively these can cause the rubber sheet to slip or bunch, yielding singularities in gradient flow procedures designed to uncover harmonic maps.

In this section, we describe the basic construction of the Dirichlet energy and point out its advantages and flaws in the context of surface-to-surface correspondence; we also provide basic constructions for approximating the Dirichlet energy of a map between discrete surfaces. In §3.2, we then propose a modified notion of harmonicity designed to avoid singularities and asymmetry in the surface-to-surface correspondence pipeline.

3.1.1 Smooth Surfaces

Following [Ura93, Nis00], harmonic maps between smooth surfaces are defined as the critical points of the *Dirichlet energy*:

$$E[T_{12}] := \frac{1}{2}\int_{M_1} |\mathrm{d}T_{12}|^2 \, dv_1, \tag{3.1}$$

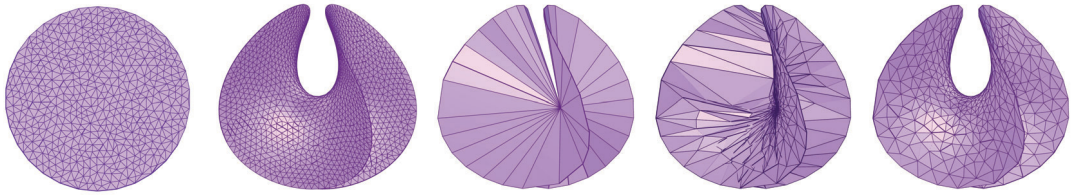
where $\mathrm{d}T_{12}$ is the *map differential* and dv_1 is the volume element of M_1. $E[T_{12}]$ measures the total stretch of M_1 after it is warped onto M_2, as measured by the integrated norm of the Jacobian $\mathrm{d}T_{12}$. Formally, given an orthonormal basis $\{e_1, e_2\}$ for T_pM_1 at $p \in M_1$, the integrand can be expanded as

$$|\mathrm{d}T_{12}|^2 = \sum_{i=1}^{2} \langle \mathrm{d}T_{12}(e_i), \mathrm{d}T_{12}(e_i) \rangle_{g_2(T_{12}(p))},$$

where g_2 is the metric of M_2.

Existence, uniqueness, and regularity of harmonic maps given assumptions on the geometry/topology of M_1 and M_2 as well as the homotopy class of T_{12} is a key theme in the twentieth-century differential geometry literature. A landmark paper by Eells and Sampson [ES64] proves existence of a harmonic map in each homotopy class under the assumption that M_2 has non-positive curvature. The proof technique in this paper is attractive from a computational perspective: Essentially they start with an arbitrary map in the prescribed homotopy class and use an analog of gradient descent to decrease the Dirichlet energy.

A key drawback of the Eells and Sampson proof technique from a computational perspective, however, highlights an issue with harmonic correspondence in the context of algorithmic mapping between surfaces. In particular, their gradient descent procedure can fail when the target M_2 has regions of positive curvature. Roughly, this singular behavior is explained by the fact that the objective $|\mathrm{d}T_{12}|^2$ is minimized globally by $\mathrm{d}T_{12} \equiv 0$, the constant map! This observation highlights the difference between harmonic mapping and elastic models like the ones proposed in [SA07, CPSS10], which seek $\mathrm{d}T_{12}$



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

Figure 3.1: Limitations of the piecewise linear discretization of the Dirichlet energy. (a) Source M_1: a flat disk embedded in $\mathbb{R}^3$. (b) Target M_2: enneper. (c) Initial piecewise linear map. (d,e) Final maps that minimize the energies in Eqs. (3.3). and Eq. (3.4), respectively. See the text for details.

to be close to a rotation matrix rather than to the zero matrix. We will address this issue in our "reversible harmonic" formulation by adding the Dirichlet energy of T_{12}^{-1} for the case of diffeomorphic correspondence; this has the added benefit of making forward maps T_{12} and reverse maps T_{21} critical points of the same objective function.

3.1.2 Triangle Meshes

For surfaces that are discretized as triangle meshes M_i, represented by their vertex, edge and face sets $(\mathcal{V}_i, \mathcal{E}_i, \mathcal{F}_i)$, a pointwise vertex map T_{12} assigns a point on a face of M_2 to each vertex of M_1. The extension of the vertex map to the interior of faces of M_1 determines the corresponding Dirichlet energy.

If the map is assumed to be *affine* on every face $f \in \mathcal{F}_1$, then the Dirichlet energy is given by

$$E\left[T_{12}\right] = \frac{1}{2} \sum_{f \in \mathcal{F}_1} \left|\mathrm{d}T_{12}\left(f\right)\right|_2^2 a_f, \tag{3.2}$$

where $\mathrm{d}T_{12}\left(f\right) \in \mathbb{R}^{2\times 2}$ is the unique linear transformation between f and its image triangle $T_{12}(f)$, and a_f is the area of f [PP93]. Equivalently, the energy can also be written as:

$$E\left[T_{12}\right] := \frac{1}{4} \sum_{(u,v)\in\mathcal{E}_1} w_{uv} \left\|T_{12}\left(u\right) - T_{12}\left(v\right)\right\|_2^2, \tag{3.3}$$

where w_{uv} is the *cotangent weight* of the edge (u, v). This energy is convex and quadratic in the images of the vertices of M_1, and is therefore straightforward to minimize efficiently when T_{12} is unrestricted, e.g. for planar parameterization [LPRM02b].

When M_2 is a non-Euclidean space, T_{12} should be restricted to lie on M_2, leading to a constrained optimization problem that is harder to solve. In addition, a more serious issue is the linearity assumption itself. When T_{12} does not sample the target surface M_2 well, the linear extension of $T_{12}(\mathcal{V}_1)$ can be far from M_2. In this case, minimizing the Dirichlet energy of the piecewise-affine map can lead to incorrect results.

Consider for example, as in Figure 3.1, mapping a disk M_1 (a) to an enneper surface M_2 (b) with Dirichlet boundary conditions; since the target has negative curvature, in the smooth case [ES64] gradient flow will reach a harmonic map $T_{12} : M_1 \to M_2$. An initial map (c) maps all the interior vertices of M_1 to a single interior vertex on M_2, and the boundary of the disk is mapped to the boundary of the enneper. Minimizing Eq. (3.3) using gradient descent, the analog of Eells & Sampson's heat flow, with the side constraint that $T_{12}(\mathcal{V}_1)$ is restricted to lie on M_2, leads to a map (d) that is clearly not smooth. Effectively, Eq. (3.3) aims to place the image of every vertex of M_1 in the Euclidean weighted average of the image of its neighbors. When the affine map samples the target poorly, this strategy fails to generate an approximation of a smooth map.

Alternatively, [IN05] suggest an intrinsic formulation, the *geodesic harmonic energy*, which replaces the Euclidean distances in Eq. (3.3) with the geodesic distances $d_{M_2}(\cdot,\cdot)$,

as follows:

$$E_D\left[T_{12}\right] := \sum_{(u,v)\in\mathcal{E}_1} w_{uv}\, d^2_{M_2}(T_{12}\left(u\right), T_{12}\left(v\right)). \tag{3.4}$$

As shown in Figure 3.1(e), minimizing this energy instead of the Euclidean one yields a significantly better result at the cost of having to compute geodesic distances. Motivated by this idea, we propose to use this energy as the main building block in a shape mapping algorithm. We reformulate it to allow efficient optimization and combination with other terms that address the case of positively-curved target surfaces M_2, as described in the following section.

3.2 Reversible Harmonic Maps

Notation. We represent a triangle mesh M by its vertex, edge and face sets $(\mathcal{V}, \mathcal{E}, \mathcal{F})$, respectively, where we denote $n = |\mathcal{V}|$, and its given embedding by $V \in \mathbb{R}^{n\times 3}$. We denote scalar functions $g : M \to \mathbb{R}$ by a vector of coefficients of piecewise linear hat functions, with $g \in \mathbb{R}^n$. The squared l_2 norm of a function on the surface is given by $\|g\|^2_M = g^T A g$, where $A \in \mathbb{R}^{n\times n}$ is the diagonal (lumped) mass matrix of the vertices. The total area of the mesh is denoted by $s = \mathrm{Tr}(A)$. The squared gradient norm is given by $\|g\|^2_W = g^T W g$, where W is the matrix of cotangent weights. Similarly, for matrices $G \in \mathbb{R}^{n\times k}$ whose columns are scalar functions, we use the matrix trace: $\|G\|^2_M = \mathrm{Tr}(G^T A G)$, $\|G\|^2_W = \mathrm{Tr}(G^T W G)$. When two meshes are involved we use a subscript, e.g. A_i is the mass matrix of M_i.

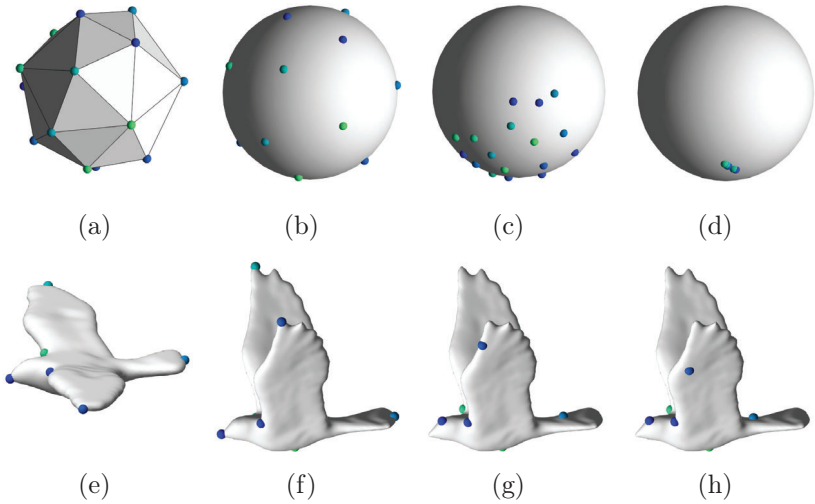


Figure 3.2: Collapse of a harmonic map. Top row: mapping from a low resolution sphere (a) to a high resolution sphere, starting from the identity map (b). The map quickly "slides" to a single hemisphere (c) and then degenerates (d). Bottom row: the same phenomenon with more complex shapes from SHREC'07 [GBP07], where we use a sparse set of landmarks for initialization and visualization. The final result (h) does not map any points to the upper part of the wings and to most of the tail.

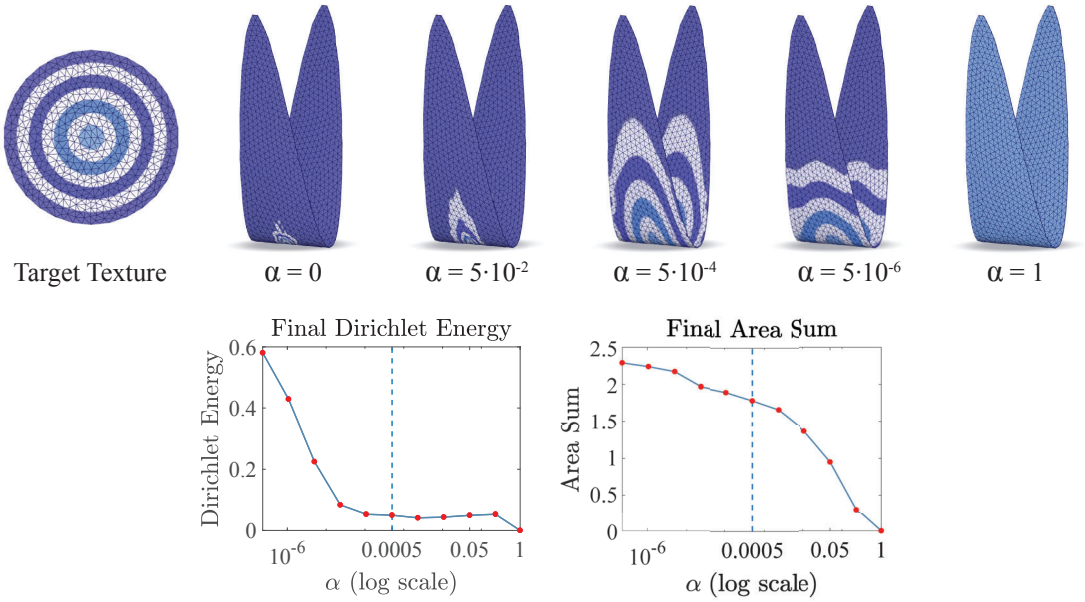


Figure 3.3: Preventing collapse with reversibility. For different α values, we measure the discrete geodesic Dirichlet energy and the sum of relative mapped area (ideally 2). We visualize some of the results using texture transfer (left), and show the final values as a function of α (right). Note that when α is small the Dirichlet energy is high, and when α is large the map collapses, as is evident by the zero total area. Finally, taking $\alpha = 5 \cdot 10^{-4}$ leads to a good balance between the energy components.

3.2.1 Energy

While the geodesic harmonic energy can be effective, harmonic maps in general can become degenerate and map large regions to a single point. Indeed, the map taking all the points on M_1 to a single point on M_2 is harmonic. An example of such behavior is demonstrated in Figure 3.2. On the top row, the source M_1 is a sphere with a small number of triangles (a), that is mapped to a high-resolution target sphere M_2. Even if the initial map is the ground-truth map (b) between the spheres, during the optimization the map quickly "slides over" to a single hemisphere of the target sphere (c), and then degenerates and collapses to a single point (d). The same phenomenon occurs for complex shapes, as shown in the bottom row. Here, we initialized a map between two bird shapes from SHREC'07 [GBP07] (e,f) using landmarks as specified in section 3.3.2. Initially (f), the tips of the wings and tail are mapped correctly, but again they gradually slide (g) until no vertices are mapped to most of the target's wings or tail (h).

Intuitively, in the discrete case, we can think of M_1 as an elastic fishnet instead of a continuous rubber sheet, stretched over M_2 and allowed to compress. Then, in addition to the usual degeneracies in the smooth case, the target surface can effectively "slip through" one of the holes in the net, allowing the map to degenerate to a single point. To prevent this, we minimize the geodesic harmonic energies of both the *forward* and *backward* maps, together with a *reversibility* constraint relating both maps. As we later

demonstrate, this approach is highly effective in generating non-degenerate harmonic maps.

Smoothness. Given two triangle meshes M_1, M_2, and maps T_{12}, T_{21}, the total harmonic energy of both forward and backward maps is given by

$$E_D\left[T_{12}, T_{21}\right] = \sum_{\substack{i,j\in\{1,2\}\\ i\neq j}} \frac{1}{s_j} E_D\left[T_{ij}\right], \tag{3.5}$$

where E_D is given in Equation (3.4).

Reversibility. We define the reversibility energy similarly to [KBB+13, EBC17] as:

$$E_R\left[T_{12}, T_{21}\right] = \sum_{\substack{i,j\in\{1,2\}\\ i\neq j}} \frac{1}{s_i^2} \sum_{p_i\in\mathcal{V}_i} d^2_{M_i}\left(T_{ji}\left(T_{ij}(p_i)\right), p_i\right) A_i(p_i). \tag{3.6}$$

The reversibility energy prevents the maps from collapsing. In the smooth case, if the reversibility energy is bounded pointwise, it easily follows that the maps are close to being injective and surjective, as we show in the Appendix. For both energies, care is required to handle correctly meshes of different scales, hence the normalization by s_i, the total area of M_i.

Finally, the full energy is given by:

$$E\left[T_{12}, T_{21}\right] = \alpha E_D\left[T_{12}, T_{21}\right] + (1-\alpha) E_R\left[T_{12}, T_{21}\right], \tag{3.7}$$

where the parameter $\alpha \in [0,1]$ controls the trade-off between smoothness and reversibility. Note that while in the continuous setting exact reversibility might be desired, this will not be the case in the discrete setting described in section 3.2.2. Thus we enforce reversibility as a soft constraint controlled by the parameter α.

To demonstrate the effect of the different components of the energy we compute a map between a disk and a folded disk, using different values of α. In the initial map all the interior vertices of the disk are mapped to a single interior vertex of the folded disk, and the boundary of the source is mapped to the boundary of the target. During the optimization the mapped boundary vertices are not constrained to lie on the boundary of the target.

Figure 3.3 shows the results, where we visualize the maps using texture transfer (left), and quantitatively evaluate them using the discrete geodesic Dirichlet energy from Eq. (3.4) (center right) as well as the sum of the total area of the images of the forward and backward maps (right). The graphs show the final values of these quantities as a function of the parameter α, where the energy is normalized with respect to its value in the first iteration.

The figure demonstrates the trade-off that the parameter α controls, e.g. taking

$\alpha=0$ models reversibility only, leading to a high Dirichlet energy. On the other hand, taking $\alpha=1$ models harmonicity only, and leads to a map that collapses the image to a single point, as is evidenced by the final total area which is zero. In general, if α is too small, the map has a high Dirichlet energy and thus is more distorted locally, and if α is too large the map collapses. Taking $\alpha=5\cdot10^{-4}$, as we did for all the experiments except for Figure 3.3, leads to a balance between the harmonic energy and reversibility.

Minimizing the energy in Eq. (3.7) requires computing the gradient of the geodesic distances with respect to the forward and backward maps, as well as tracing vector fields on the surface, which are both computationally heavy. We therefore apply two approximations to address these issues.

3.2.2 Energy approximation

Notation

Any point $p\in M$ can be represented uniquely using its barycentric coordinates $\omega_l(p), l\in\{1,..,3\}$ with respect to the face $f(p)=(v_1,v_2,v_3)\in\mathcal{F}$ it lies on. We denote by $\lambda(p)\in\mathbb{R}^{1\times n}$ the row vector that is zero everywhere except at the vertices of $f(p)$, where we have $\lambda(p)[v_l]=\omega_l(p)$. In addition, we denote the *feasible row set* of M, i.e. the set of all possible such vectors, by $\mathcal{P}=\{\lambda(p)\,|\,p\in M\}$. Finally, the *feasible set* of all possible precise maps from M_1 to M_2 is given by $\mathcal{P}_{12}=\{P_{12}\in\mathbb{R}^{n_1\times n_2}\,|\,P_{12}(l,:)\in\mathcal{P}_2,\ \forall l\in\{1..n_1\}\}$, where $P(l,:)$ denotes the l-th row of the matrix P. Thus, any map T_{12} can be represented using a matrix P_{12}, by setting $P_{12}(l,:)=\lambda_2(T_{12}(v_l)),\ \forall l\in\{1..n_1\}$, which, by definition, is in the feasible set $\mathcal{P}_{12}$. Furthermore, the matrix $P_{12}V_2\in\mathbb{R}^{n_1\times3}$ represents the images of the vertices $\mathcal{V}_1$ under the map T_{12}.

High-dimensional embedding

As we have seen, if the target space is *Euclidean* then the geodesic distances are Euclidean distances, and the optimization is simple and efficient. Following similar ideas in the literature [BBK06], we therefore suggest to use a *high-dimensional Euclidean embedding* as a proxy for fast geodesic distance computation.

Given a mesh M, we seek an embedding $x:\mathcal{V}\to\mathbb{R}^m$, for $m\ll n$ such that the Euclidean distance $\|x(u)-x(v)\|$ approximates well the geodesic distance $d_M(u,v)$, for all $u,v\in\mathcal{V}$. The literature on such embeddings is quite vast, and we chose to leverage the method suggested by [PBDSH13] that relies on multidimensional scaling [CC00]. Any other embedding method could be used as well, as long as the geodesic distances are well approximated. We took $m=8$ in all our experiments, following [PBDSH13].

Our goal now is to compute a harmonic map between the high-dimensional Euclidean embeddings. We denote by $X\in\mathbb{R}^{n\times m}$ the matrix whose rows are the embeddings $x(v),\forall v\in\mathcal{V}$. Rewriting the harmonic and reversibility energies in terms of the high-

dimensional embeddings, and in matrix form, leads to:

$$E\left(P_{12}, P_{21}\right) = \sum_{\substack{i,j\in\{1,2\} \\ i\neq j}} \alpha \frac{1}{s_j} \|P_{ij} X_j\|_{W_i}^2 + (1-\alpha) \frac{1}{s_i^2} \|P_{ij} P_{ji} X_i - X_i\|_{M_i}^2. \tag{3.8}$$

Note that the weights in the harmonic energy, given now in matrix form in W_i, remain the same for the high-dimensional embedding, since the embedding is nearly isometric.

Figure 3.4 demonstrates the importance of the high dimensional embedding. For shapes where the geodesic distances are considerably different than Euclidean distances in $\mathbb{R}^3$, e.g., the spring shapes from SHREC'07 [GBP07], using the three-dimensional input Euclidean embedding in the optimization leads to a highly distorted map (center). Specifically, neighboring vertices on the source shape are mapped to different coils of the target spring, which are extrinsically close but intrinsically far. On the other hand, by using the high dimensional embedding, the geodesic Dirichlet energy is well approximated leading to an improved map (right).

Half quadratic splitting

While Equation (3.8) can be minimized using gradient descent, we found, similarly to [EBC17], that it is more efficient to use the *half quadratic splitting* optimization method [GY95] (see also e.g. [WYYZ08, ZW11]). We introduce auxiliary variables $X_{ij} \in \mathbb{R}^{n_i \times m}$, which estimate the images of the vertices $\mathcal{V}_i$ given by $P_{ij} X_j$. Substituting, the energies are:

$$\bar{E}_D(X_{ij}) = \frac{1}{s_j} \|X_{ij}\|_{W_i}^2, \quad \bar{E}_R(P_{ij}, X_{ji}) = \frac{1}{s_i^2} \|P_{ij} X_{ji} - X_i\|_{M_i}^2, \tag{3.9}$$

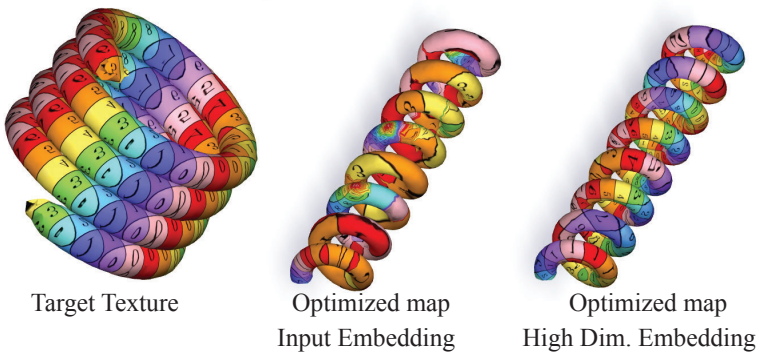


Figure 3.4: The importance of the high dimensional embedding. For these shapes, geodesic and Euclidean distances are significantly different, thus using the input vertex positions in $\mathbb{R}^3$ during the optimization results in a highly distorted map (center). The high dimensional embedding that approximates the geodesic distances leads to a better map (right).

for $i,j \in \{1,2\}, i \neq j$. In addition, we need soft constraints for the auxiliary variables:

$$\bar{E}_Q(P_{ij}, X_{ij}) = \frac{1}{s_i s_j} \|X_{ij} - P_{ij} X_j\|_{M_i}^2, \tag{3.10}$$

where we again normalize by $s_i s_j$ to retain scale invariance.

The full energy is now:

$$\begin{aligned} &\bar{E}\left(P_{12}, P_{21}, X_{12}, X_{21}\right) = \\ &\sum_{\substack{i,j \in \{1,2\} \\ i \neq j}} \alpha \bar{E}_D(X_{ij}) + (1-\alpha)\bar{E}_R(P_{ij}, X_{ji}) + \beta \bar{E}_Q(P_{ij}, X_{ij}), \end{aligned} \tag{3.11}$$

where β controls the accuracy of the auxiliary variables and functions as a step size. When using the half-quadratic splitting optimization scheme, the update schedule for β is often tailored per application, with the general guideline of increasing β as the iterations advance [WYYZ08]. In our case, we often initialize the optimization with a highly degenerate map, e.g. as obtained from a sparse set of landmarks, and therefore the value of β during the first iterations should be small enough so that the map can change significantly. As the iterative solution approaches a local optimum, β can increase, as less modification is required. The final value of β should be large enough to ensure P_{ij} and X_{ij} correspond. In all of our experiments, we took $\beta = 5 \cdot 10^{-3} k$ where k is the optimization iteration number for the first 100 iterations, and then kept the value of β fixed until convergence.

3.2.3 The optimization problem

Our optimization problem is now given by:

$$\begin{aligned} \underset{P_{12}, P_{21}, X_{12}, X_{21}}{\text{minimize}} \quad & \bar{E}(P_{12}, P_{21}, X_{12}, X_{21}) \\ \text{subject to} \quad & P_{12} \in \mathcal{P}_{12},\ P_{21} \in \mathcal{P}_{21}, \end{aligned} \tag{3.12}$$

where $\mathcal{P}_{ij}$ is the feasible set of precise maps from M_i to M_j. Despite the two approximations that we used, solving this optimization problem succeeds in decreasing the total discrete Dirichlet energy from Eq. (3.4) while preventing the map from collapsing, as is illustrated in Figure 3.5. In addition to the energy, we show the initial map that was created from landmarks as described in section 3.3.2, and the intermediate map at a few iterations.

3.3 Optimization

Our optimization problem has block structure, in the sense that if some of the variables are kept fixed it becomes a linear least squares problem. We therefore chose to use block coordinate descent (see e.g. [XY13]) as the optimization algorithm.

3.3.1 Block coordinate descent

In each sub-iteration we solve for one of the matrices X_{ij}, P_{ij}, while keeping the others fixed. Since the energy is quadratic in all the variables, every sub-iteration involves a relatively simple optimization problem, with the only complication arising because of the non-convex feasible sets $\mathcal{P}_{ij}$.

Optimizing for X_{ij}. When P_{12}, P_{21} are fixed, the optimization problem is a linear least squares minimization of the form $\|AX_{ij} - B\|_2^2$, with known matrices A and B, where A is sparse, which we solve using a direct method. The system is highly over-constrained, as even if P_{ji} degenerates, the system is well-conditioned due to the term $\bar{E}_Q$, as long as the vertex areas of the mesh M_i do not vanish.

Optimizing for P_{ij}. When X_{12}, X_{21} are fixed, the energy has the form $\|P_{ij}A - B\|_2^2$, where A, B are known, with the constraints that $P_{ij} \in \mathcal{P}_{ij}$. Following [EBC17], the optimization is done by solving for every row of P_{ij} separately. Intuitively, we can think of A as a high-dimensional embedding of the faces of M_j, and of B as a high-dimensional point cloud. The optimal P_{ij} projects each point in B to its closest point on the faces given by A. As shown in [EBC17], this process is guaranteed to find P_{ij} which are globally optimal when X_{ij} are kept fixed.

Stopping criterion. The alternating descent guarantees that the energy is reduced at every iteration, since the sub-iterations find a global optimum of the reduced optimization problems. In practice, we stopped the optimization when the change of energy was less than 10^{-9}, or after a maximum of $N = 200$ iterations. In most cases, we have observed convergence of the energy to high precision even when early stopping after N iterations was used.

We provide the details of the alternating descent in Algorithm 1.

Algorithm 3.1 Alternating minimization.

Input: Two triangles meshes M_1, M_2, initial $P_{12}, P_{21}, X_{12}, X_{21}$
Output: P_{12}, P_{21}

For $k = 1 \dots N$
  For $i = 1, 2$
    $j = 3 - i$
    $P_{ij} \leftarrow \underset{P \in \mathcal{P}_{ij}}{\operatorname{argmin}} \bar{E}_R(P, X_{ji}) + \bar{E}_Q(P, X_{ij})$
    $X_{ij} \leftarrow \underset{X \in \mathbb{R}^{n_i \times m}}{\operatorname{argmin}} \bar{E}_D(X) + \bar{E}_R(P_{ji}, X) + \bar{E}_Q(P_{ij}, X)$
  end
end

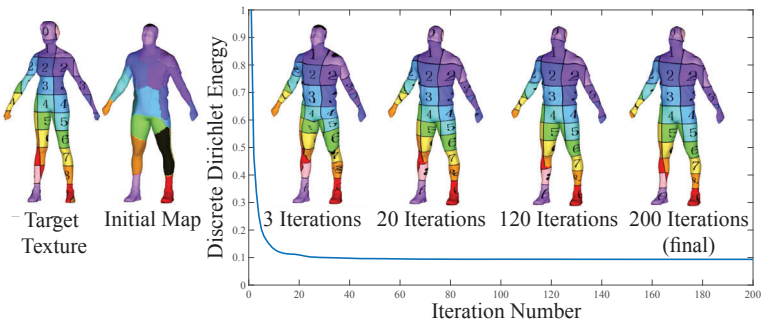


Figure 3.5: Optimization of Eq. (3.12), starting from landmarks. The discrete Dirichlet energy in Eq. (3.4) decreases, and the final map, visualized by texture transfer, is semantic and not distorted locally.

3.3.2 Initialization

Our method is general and can receive as input various initial data. Depending on the input, we describe the initialization of the variables P_{ij}, X_{ij}. Given a pointwise map $T_{12} : \mathcal{V}_1 \to M_2$ its corresponding matrix representation is given by $P_{12}(l,:) = \lambda_2(T_{12}(v_l))$ for $l \in \{1,..,n_1\}$. Similarly, given a matrix representation $P_{ij} \in \mathcal{P}_{ij}$, we have that $T_{ij}(v_l) = (P_{ij}V_j)(l,:) \in M_j$. Therefore, in the following refer to P_{ij} or T_{ij} according to which notation is more convenient.

Pointwise map. Given a pointwise map P_{12} we approximate an inverse map P_{21} by taking $T_{21}(v_2) = \text{argmin}_{v \in V_1} \|T_{12}(v) - v_2\|$. Then, the initialization of X_{ij} is $P_{ij}X_j$.

Functional map. The term *functional map* [OBCS+12] denotes a map between scalar functions. It is a linear operator that can be represented using a matrix when scalar functions are represented in a linear basis. Let $\Psi_i \in \mathbb{R}^{n_i \times k_i}$ be a matrix whose columns are basis functions of a subspace of scalar functions on M_i, where each function is piecewise linear and is defined by values assigned to vertices. Given a pointwise map that maps vertices of M_1 to points on M_2, the corresponding functional map $C_{12} \in \mathbb{R}^{k_1 \times k_2}$ maps functions on M_2 to functions on M_1, represented in the reduced basis. Therefore, given two functional maps C_{12} and C_{21}, we initialize $X_{ij} = \Psi_i C_{ij} \Psi_j^\dagger X_j$. Optimizing P_{ij} does not require initialization. Figure 3.10 shows results where functional maps were used for initialization.

Landmarks. Given r input landmark pairs $\{(p_i, q_i)\}$ where $p_i \in \mathcal{V}_1, q_i \in \mathcal{V}_2$ and $i \in \{1,..,r\}$, we first construct a rough initial pointwise map P_{12} and then use it to initialize the rest of the variables, as previously described. We first compute the geodesic Voronoi diagram on M_1 with centers p_i, and then set $T_{12}(v) = q_i$, $\forall v \in C_i$, where C_i is the geodesic cell corresponding to the center p_i. Note that this initialization is highly degenerate, as all the points on M_1 are mapped to the landmarks q_i on M_2, yet it is enough for our needs. In Figure 3.5 initialization using input landmarks is visualized.

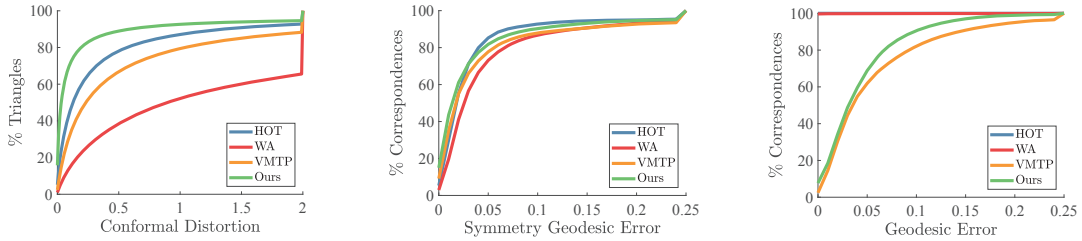


Figure 3.6: Quantitative comparison on the SHREC dataset, measuring, from left to right: conformal distortion, compatibility with symmetries and distance from ground truth landmarks. Note that we achieve a better conformal distortion, and comparable symmetry geodesic error. Furthermore, note that WA and HOT do not modify their input landmarks, while our method and VMTP do. Compared to VMTP we achieve a better landmark geodesic error.

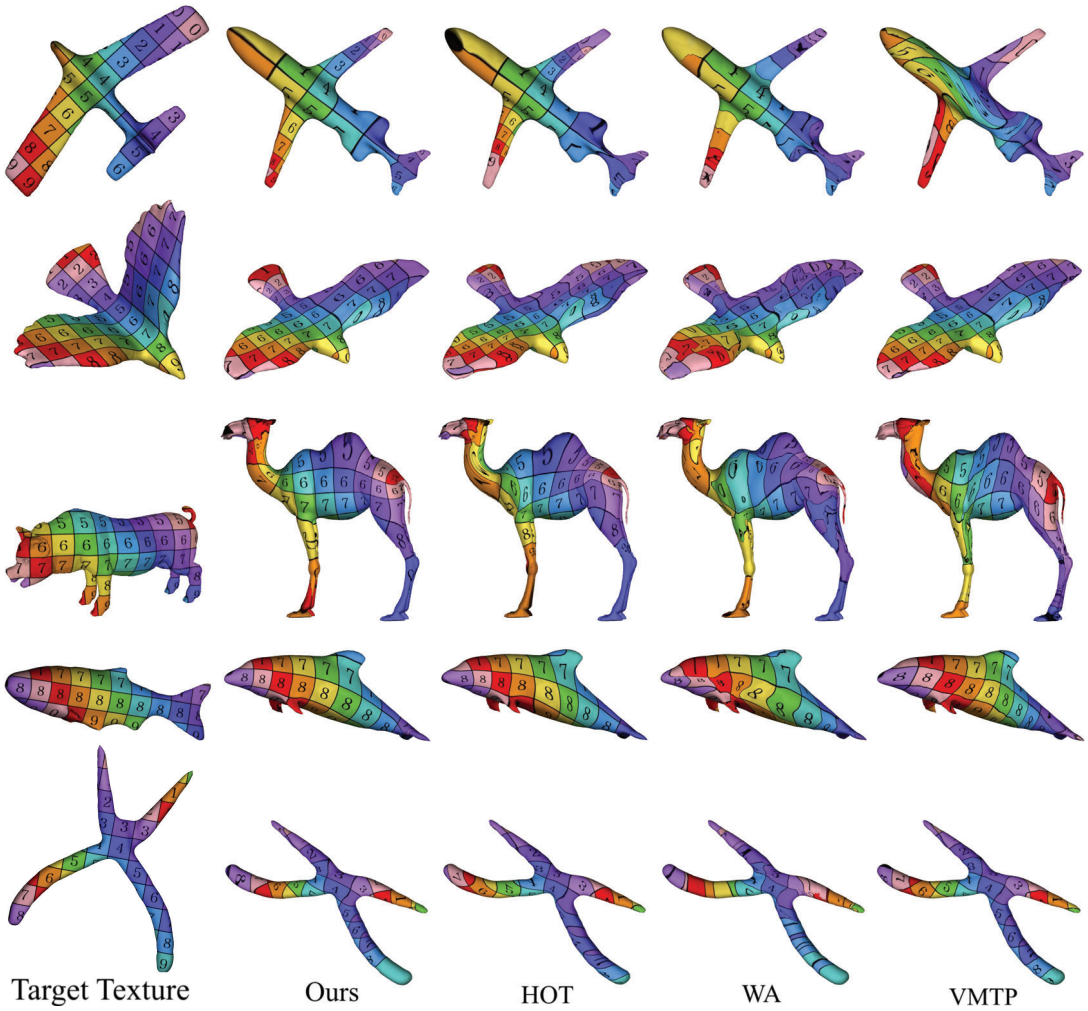


Figure 3.7: Qualitative results, from input landmarks. From left to right: target texture, our method, HOT [AL16], WA [PBDSH13] and VMTP [MCSK+17]. See the text for details.

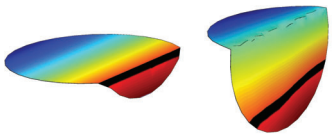


Figure 3.8: Minimizing the reversible harmonic energy does not align extrinsic features.

3.3.3 Limitations

Using the projection step when optimizing for P_{ij} has some limitations. First, as discussed in [PBDSH13], such a projection is not smooth. Furthermore, the closest point on the high-dimensional embedding of the triangle mesh might not be unique, therefore the solution of the optimization for P_{ij} might alternate between two configurations with the same energy. Thus, while the energy is guaranteed to converge, we do not have a similar guarantee for the convergence of the solution. In practice, we have not encountered a case where these limitations posed a practical problem. In future work it could be possible to handle the first issue using a Phong projection, as in [PBDSH13], and the second issue using an additional regularization that penalizes diverting from the current solution.

Since we optimize for both harmonicity and reversibility we cannot guarantee convergence to a smooth harmonic map; this is likely a fruitful avenue for future work. In addition, our method does not consider extrinsic features such as edges and corners, and hence such features will not necessarily be mapped to each other. An example is shown in figure 3.8, where our map between the two folded disks is smooth, but the edge features do not correspond.

3.3.4 Timing

The most expensive step in the optimization process is the projection on a triangle mesh for optimizing P_{ij}. However, this procedure is highly parallelizable since the projection of each point is independent of the other points. We used CUDA 8 to implement the projection in parallel, while the rest of the optimization method was written in MATLAB. On a desktop machine with a TITANX GPU and an Intel Core i7 processor, 200 optimization iterations of our method, for shapes with 5K vertices, took around 115 seconds.

3.4 Results

To validate our method we have compared with a variety of state-of-the-art mapping techniques, in accordance with the type of input they can accept. In addition, we show applications to shape interpolation and quad mesh transfer.

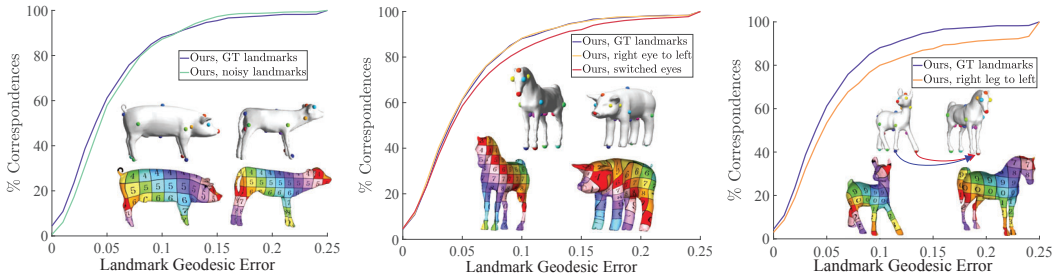


Figure 3.9: Mapping SHREC quadrupeds by our method, starting from noisy landmarks. Our method is only slightly affected by the noise even when the landmark modification is severe.

3.4.1 Quality metrics

To evaluate the quality of a map we measure its smoothness through its conformal distortion and its semantic accuracy, using the distance to the ground truth, when given. We also use alternative measures, such as symmetry and compatibility with ground truth segmentations, when no dense ground truth map is available.

Conformal distortion. We use the definition by Hormann and Greiner [HG00, Eq. (3)] for the conformal distortion of a single triangle $f \in \mathcal{F}_1$: $\kappa(f) = \frac{\sigma_1}{\sigma_2} + \frac{\sigma_2}{\sigma_1}$, where $\sigma_1 \geq \sigma_2$ are the singular values of the linear transformation which maps f from M_1 to M_2. We subtract 2 so that the minimal conformal distortion is zero and visualize the result as a cumulative graph showing the percentage of triangles with less than a certain distortion value.

Distance from ground truth. When a ground truth map is given, we measure the distance from the ground truth using the protocol suggested by [KLF11, Section 8.2]. For every mapped vertex, we measure its geodesic distance from the ground truth location, relative to the square root of the total area of M_2, and visualize the percent of vertices whose distortion is less than a given value.

Compatibility with segmentations. For some datasets ground truth labeled segmentations are available. In this case, for every pair of shapes and a given map we measure the consistency of the segmentation with respect to the map. This is done by computing the relative vertex area of vertices that are mapped to a face that belongs to the same segment as the source vertex.

Compatibility with symmetry. For some datasets a ground truth map is only known for a subset of the points, yet a full intrinsic symmetry can be computed for every shape separately. We assume that a good map should respect the intrinsic symmetries of the source and target shapes, given by S_1, S_2, respectively. We therefore use these symmetries as input, and measure the compatibility of the map T_{12} with the symmetries,

given by the geodesic distance $d_{M_2}(S_2(T_{12}(v_1)), T_{12}(S_1(v_1))), \forall v_1 \in \mathcal{V}_1$. We visualize the result using a cumulative graph, similarly to the ground truth error. To compute the symmetries we use the method by Kim et al. [KLF11]. We also manually filtered the results to use only the accurate symmetries.

3.4.2 Dataset: SHREC, input: landmarks

We use the BIM benchmark [KLF11] that provides more than 200 pairs of highly non-isometric shapes from the SHREC dataset [GBP07] with user-verified landmarks. We compare our method with a state of the art parameterization based method [AL16] (HOT) and the weighted averages method [PBDSH13] (WA). Both receive as input landmark points, which are not modified during the optimization, and generate precise maps. In addition we compare to the recent method by [MCSK+17] (VMTP), that similarly gets as input landmark points, yet can modify them during the optimization. Since VMTP requires uniform isotropic meshes, we recursively add edges using the longest edge bisection method to meshes with less than 10K vertices, before applying VMTP. All the methods we compare with produce *precise* maps, as vertex-to-vertex maps induce high local distortion. As input to our method we also use the user defined landmarks, and extend them to a full initial map as described in section 3.3.2. The landmarks are not used after the initialization.

Quantitative results are shown in Figure 4.11, where we measure conformal distortion, compatibility with symmetries and distance from the ground truth landmarks. Note

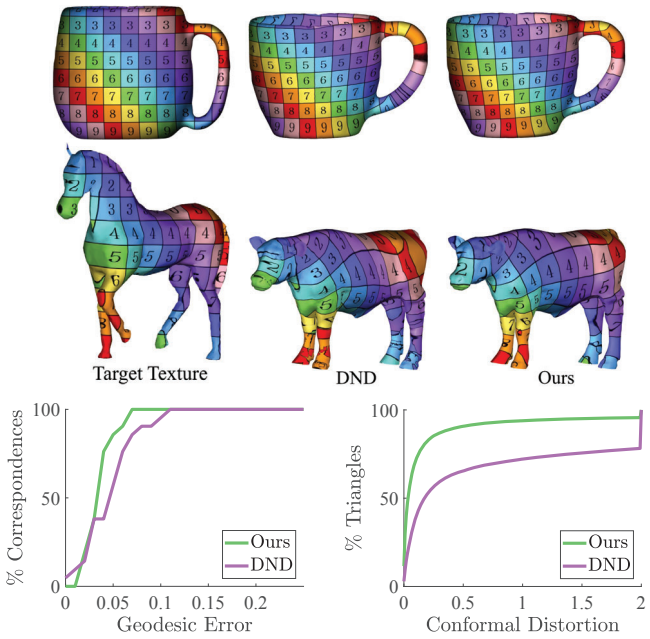


Figure 3.10: Qualitative and quantitative comparison starting with a functional map computed from landmarks. From left to right: target texture, [EBC17] (DND), our method. Notice the difference at the cup handle and the legs.

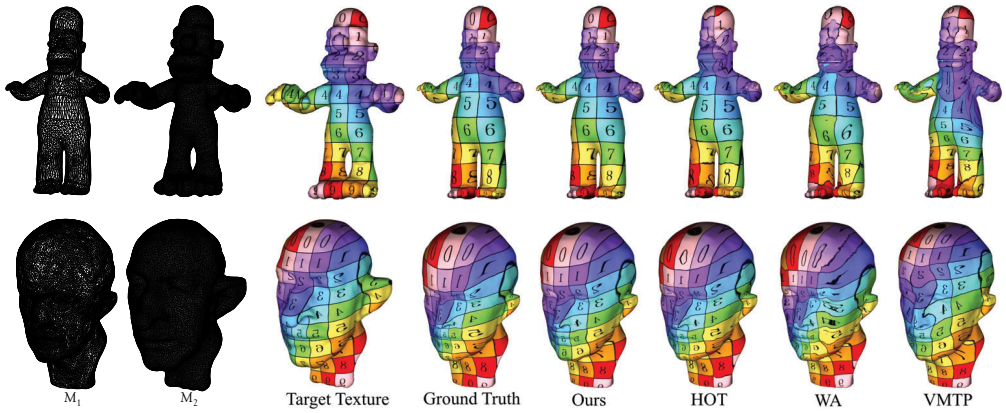


Figure 3.11: Qualitative result, caricatures. M_2 was generated by deforming M_1 [SAK15], and the deformation defines a ground truth map. M_2 was remeshed so that the source and target shapes do not have the same connectivity.

that our method achieves the best conformal distortion. In addition, the distance from the ground truth landmarks is also improved when compared to the other method which modifies them (VMTP). As shown in section 3.4.3, the option to modify the input landmarks is valuable when the input is not completely reliable. In terms of compatibility with symmetry, our method is comparable with existing techniques, notably achieving a better ratio of perfect matches with about 15% of the vertices exactly symmetric for our method, where the next best method has less than 10% exactly symmetric vertices. On this dataset ground truth segmentations are also available [KHS10a, CGF09], and measuring the relative mapped area which is compatible with the segmentations we have HOT: 90.39%, WA: 90.35%, VMTP: 81.8%, our method: 89.62%. Therefore, this measure also demonstrates that our maps are as compatible semantically as existing techniques, while being considerably more conformal.

Qualitative results are shown in Figure 3.7, where we have selected a subset of pairs to visually show the differences between the maps. In every row we show, from left to right, the target texture, and the results of our method, HOT, WA and VMTP.

3.4.3 Dataset: SHREC quadrupeds, input: noisy landmarks

In many cases, the selection of the landmarks by the user has some variability (see, e.g. [CSPF12]), and it might be better to treat these landmarks as guidelines rather than exact ground truth. Our approach is compatible with this notion, since our method only uses the landmarks for initialization, and their final location will, in most cases, vary from their initial one. To check the sensitivity of our approach to the landmark locations, we repeated the experiment from section 3.4.2 with various landmark modifications. Figure 3.9 shows the landmark geodesic error of the output maps when starting from the noisy landmarks compared to starting from the original landmarks. We ran the experiment on the "quadrupeds" class (20 pairs) from the SHREC dataset and did the following modifications: (a) moved every landmark randomly to a vertex in its 5-ring

neighborhood, (b) switched between the two eyes, or mapped both eyes on M_1 to a single eye on M_2, and (c) mapped both feet on M_1 to the same foot on M_2. In addition to the error graph we show some example maps, as well as the input noisy landmarks. As the figure shows, our results are not sensitive to landmark noise, and even a relatively severe modification, such as mapping both feet to the same foot, yields good qualitative and quantitative results.

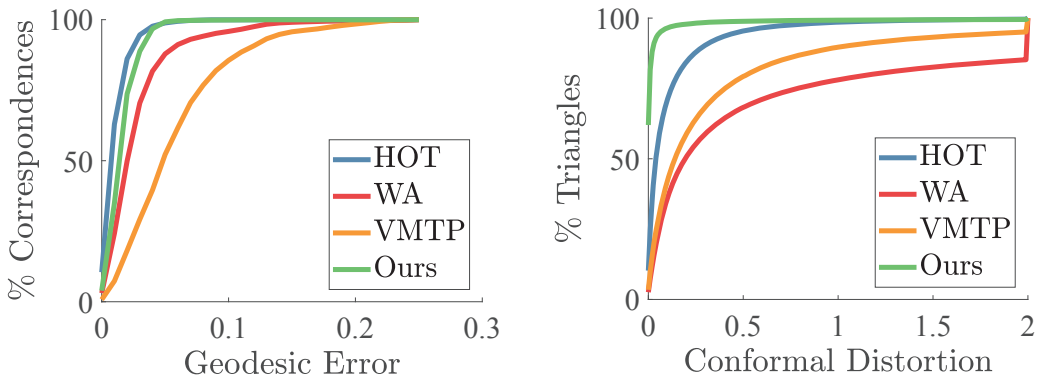


Figure 3.12: Quantitative measures for the meshes in Figure 3.11. Note that our method achieves a considerably better conformal distortion, while maintaining ground truth error comparable to existing methods.

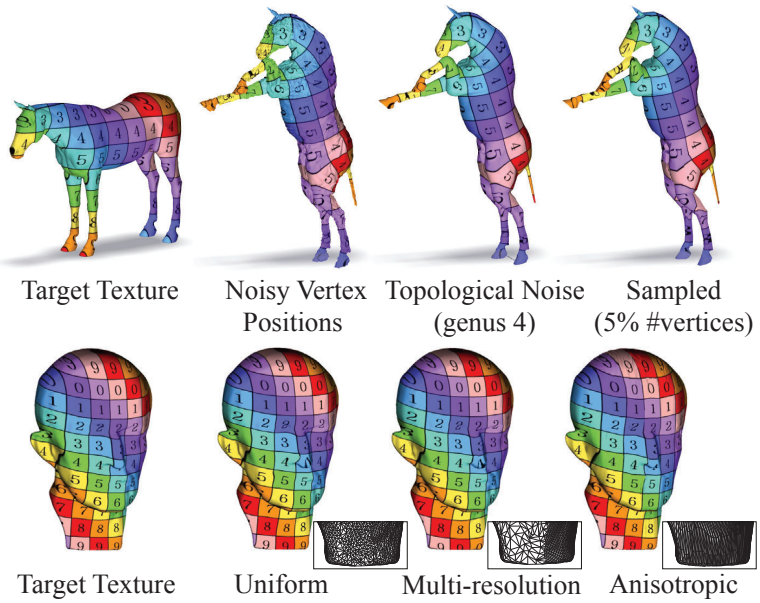


Figure 3.13: Robustness to noise and sampling. Top row: a shape with various transformations from SHREC'10 [BBB$^+$10]. Bottom row: the left shape is mapped to the same geometry with different tessellations.

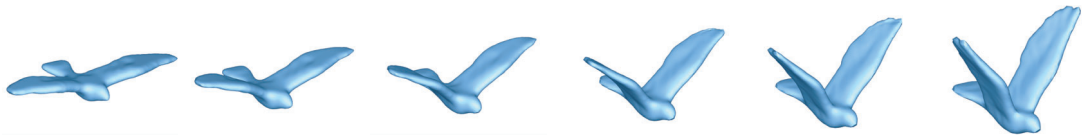


Figure 3.14: Shape interpolation using our computed correspondence as input for [HRS+14].

3.4.4 Dataset: SHREC two pairs, input: functional map

The functional map [OBCS+12] machinery is quite versatile, and allows to compute generalized maps in a variety of cases. Our method can also be used to extract a precise pointwise map from a given functional map. We use the SHREC dataset with its landmark data from the BIM benchmark [KLF11], and use the landmarks to compute a functional map using the Wave Kernel Map and the Wave Kernel Signature [ASC11]. Any other recent method for computing functional maps could be used as well. We provide the functional map as input to our approach and the recent map deblurring approach [EBC17] (DND), which is the only other method that recovers *precise* maps from a functional map. Specifically, we used the consistency extension of DND with $\alpha = 0.8$. Figure 3.10 qualitatively visualizes the difference between the methods for two pairs of shapes. Note the map improvement on the handle of the cup and the legs of the cow. We also show graphs of the conformal distortion and ground-truth error of the landmarks.

3.4.5 Dataset: caricatures, input: landmarks

One of the advantages of our formulation is its simplicity, that leads to flexibility in adding additional components to the energy. For example, in some cases it can be beneficial to add *weak* landmark constraints, to encourage feature points to remain in the neighborhood of the input landmarks.

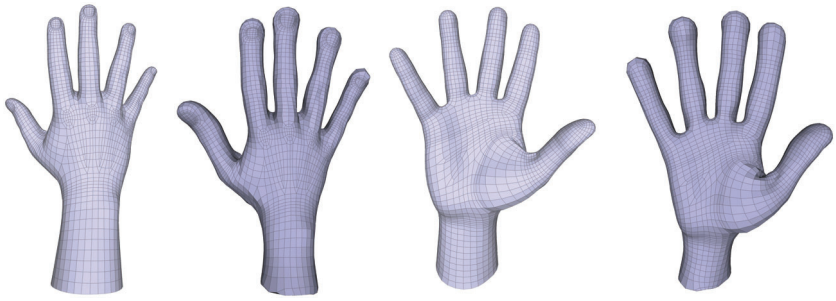


Figure 3.15: Quad mesh transfer using our computed correspondence. Left: input quad mesh, right: output quad mesh. Note the preservation of the prominent edge flows in the quad mesh, such as the fingernails and knuckles.

Weak landmark constraints. Given pairs of matching landmarks $p_i \in \mathcal{V}_1, q_i \in \mathcal{V}_2, i = 1 \dots r$, we add the following term to the energy:

$$\gamma \sum_{i=1}^{r} A_1(p_i) \| X_{12}(p_i, :) - X_2(q_i) \|_{M_2}^2 + A_2(q_i) \| X_{21}(q_i, :) - X_1(p_i) \|_{M_1}^2. \tag{3.13}$$

The value of γ depends on the reliability of the landmarks, if the landmarks are not accurate then γ should be small. To demonstrate the effectiveness of this approach, we used as input the Homer and Max Planck models, and caricatures of these models generated using the method by [SAK15]. The caricatures have the same triangulation as the original shapes. We remesh the caricatures to avoid bias, and project the original caricature to the remeshed model to generate the ground truth for validation. As input to all approaches, we picked 28 and 14 landmarks from the input map for the homer and Max models respectively. For our algorithm we added the energy in Equation (3.13) with $\gamma = 1$. The qualitative and quantitative results are shown in figures 3.11 and 3.12, respectively. As in previous experiments, we achieve considerably better conformal distortion, with comparable ground truth error.

3.4.6 Robustness

We test our method on horse shapes from the SHREC'10 dataset [BBB$^+$10], which contains various shapes with geometric and topological noise. The results are shown in the top row of Figure 3.13, where we compute a correspondence between the horse in a neutral pose (left) and a noisy pose: (center left) noise is added to the vertex positions; (center right) topological noise is introduced, e.g. between the rear feet; and (right) the mesh is sampled to 5% of the number of vertices. We initialized our method using 19 landmarks and obtained semantically correct results in all cases. The bottom row shows our results for different tessellations of a shape of a head, where one shape has uniform triangles, a second shape has larger triangles on one half and smaller triangles on the other half, and a third shape has highly anisotropic triangles. For initialization, we used the identity map with additional noise. The resulting mapped texture is smooth and similar for the different tessellations.

3.4.7 Application: Shape Interpolation

Existing shape interpolation methods require the input shapes to share the same connectivity, while real data rarely satisfies this requirement. Our mapping can be used to *remesh* the target surface M_2 using the image of the vertices of M_1, given by $P_{12}V_2$, and the connectivity of M_1. We used our map between two birds from SHREC to demonstrate this application, starting from the BIM landmarks as described in section 3.4.2. After remeshing M_2, as a few faces had a zero area, we iteratively moved vertices of the degenerate faces to an average of their 1-ring neighbors until there were no degenerate faces. We then used the shape interpolation method by [HRS$^+$14] to interpolate between

the shapes, as shown in Figure 3.14. Note that we correctly mapped the wings, head and tail of the birds, as is evident from the natural interpolation results.

3.4.8 Application: Quad Mesh Transfer

Finally, we demonstrate a potential application of our correspondence to quad mesh transfer of artist-generated quad meshes to scanned meshes. In this experiment, we start from a set of 41 landmark points, and use weak landmark constraints with $\gamma = 5 \cdot 10^{-5}$. We choose to have a larger number of landmarks in this example to preserve the fine features such as fingernails and joints. The results are shown in Figure 3.15, with two views of the input and output quad meshes on the left and right, respectively. Note, that the edge flow of the output quad mesh closely follows the features of the hand, and the special structures in the input, such as the fingernails and knuckles, are nicely preserved in the output mesh. Such a high quality transfer can only be achieved if the computed map has a low conformal distortion, which leads to well preserved quad shapes.

Chapter 4

Elastic Correspondence between Triangle Meshes

We consider the case of correspondence between two manifold triangle meshes that have the same topology, but are not necessarily isometric. In such cases, the strong geometric prior of the preservation of geodesic distances is no longer available, and there exists a huge set of smooth, valid maps between such surfaces. A high quality map should have a low *angle and area distortion*, facilitating downstream applications such as texture and deformation transfer. However, intrinsic geometric information alone is often not enough to yield a semantically correct map. An important *extrinsic* geometric property of semantically correct maps, is the correct alignment of prominent *curvature features*, such as the crease of an airplane wing (see Fig. 4.1). Yet, achieving both low metric distortion and crease alignment is difficult using existing techniques. Some mapping approaches are fully intrinsic [KLF11, AL16, MCSK$^+$17], and therefore are not aware of extrinsic curvature dependent features. Alternatively, classic extrinsic approaches (e.g. [LSP08]) can match extrinsic features, but often focus on global rather than local distortion, and in addition are sensitive to the global orientation of the input shapes, since they optimize for the extrinsic deformation matrices.

We bridge this gap and achieve extrinsic feature matching, as well as low local

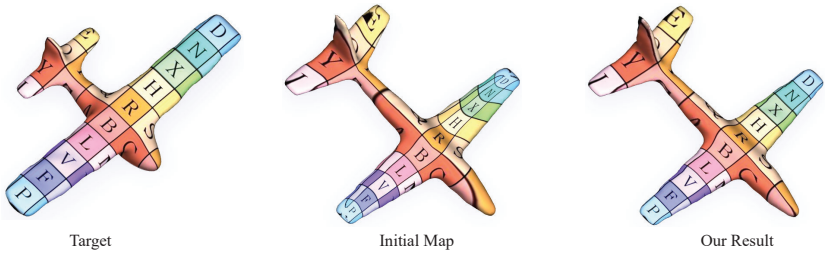


Figure 4.1: Visualization of our results using texture transfer from the target shape (left) to the source shape with the initial map (middle) and our final map (right). Note the texture distortion in the initial map due to the incorrect matching of the airplane wing creases, which is aleviated using our crease-aware method.

distortion, while remaining parameterization-free and invariant to global rigid transformations of the input shapes. We accomplish this using a novel combination of a discrete *thin-shell* energy that is often used for shape deformation [HRS+14] with a recent projection-based, parameterization-free, optimization technique for *local distortion* of maps between triangle meshes [EBC17].

Starting from a set of sparse landmarks provided by the user, we initialize using an *intrinsic* map computed using existing methods [AL16]. Then we simultaneously optimize for an *elastic* deformation of the input shape while penalizing the distance from its projection on the target shape. Our energy is composed of a non-linear *membrane* energy that favors isometry, and a *bending* energy that is rotation invariant and promotes feature alignment. It is a novel modification of the classic discrete thin-shell energy, that is robust to the extreme deformations that may arise in a projection-based optimization framework, while remaining faithful to the physical behavior of the classic energy.

Our scheme yields high-quality, crease preserving maps between non-isometric shapes, that far surpass state-of-the-art methods on both the FAUST [BRLB14] and SHREC07 [GBP07] datasets. We show quantitative improvement of various error measures, specifically geodesic error from the dense ground truth on FAUST, and angular distortion, area distortion, mean curvature error and geodesic error from sparse ground truth on SHREC07. Further, our crease preserving maps are highly useful in downstream applications, as we demonstrate by applying our computed maps for generating consistent cross-fields [ACBCO17], for shape interpolation [HRWW12] and for consistent quadrangular remeshing of a set of shapes.

Our contributions. We propose a novel matching algorithm to compute a high quality correspondence map between two triangle meshes. The main characteristics are:

- Our algorithm combines a physical thin shell deformation model with a parameterization-free projection-based correspondence scheme.
- Our method is initialized with the output of an existing correspondence algorithm, which is smooth yet might have substantial local distortion and lacks feature alignment.
- Our output is a low-distortion and feature-aligned correspondence, which is highly effective for downstream applications such as texture transfer.

4.1 Related Work

This related work section focuses on elasticity and specifically its use for shape correspondence.

4.1.1 Elastic Shape Modeling

Physically-based elastic energy models have been widely used for computer graphics and geometry processing applications [RW14]. The classical model for elastically deformable surfaces is the shell model, originally introduced in a graphics context by Terzopoulos et al. [TPBF87], for thin, flexible materials. Grinspun et al. [GHDS03] introduced the discrete shell model in which a triangle mesh is a spatially-discrete representation of the mid-surface of a shell. The model was used for simulation of deformable materials under physical forces. Similar thin-shell energies have been used by Botsch et al. [BPGK06] for interactive shape deformation. Heeren and coworkers [HRWW12, HRS$^+$14] have used the same physical model for time-discrete Riemannian analysis of shapes. In the direction of improving efficiency, the as-rigid-as-possible (ARAP) framework [SA07] is based on alternating minimisation over vertex positions and local rotations of an energy that measures deviation from rigidity. Von Radziewsky et al. [vRESH16] recently showed how model reduction can be used to efficiently evaluate elastic deformation models, including the discrete shell energy. This enables elastic models to be used in realtime applications.

4.1.2 Elastic Shape Correspondence

Already in [LDRS05] a non-linear elastic deformation energy between thin shells has been investigated for surface matching. In this approach, the matching problem is formulated on disc type parameter domains of the surface patches to be matched. This renders the surface matching problems as a classical elastic image registration problem and the membrane energy takes the role of the regularization energy, whereas the bending energy turns into a fidelity energy with respect to the matching of mean curvature functions on the parameter domains. The surface matching method in [WSSC11] picked up the matching energy from [LDRS05] and investigated matches as surfaces in the product space of the source and template geometry. Using relaxation methods from linear programming Windheuser et al. were able to robustly compute bijective triangle matching deformations with vertex to vertex correspondence. They do not allow for general deformations of source vertices onto the target surface, and the approach is quite heavy computationally due to the use of the product manifold. The elastic matching of volumetric shapes from the perspective of shape optimization has been investigated by Buhan et al. [dBDFN16]. Finally, Iglesias et al. [IRS18] have applied elastic energies for computing correspondences between level-set surfaces.

Many early deformation-based approaches, that find, e.g., sparse [ZSCO$^+$08] or dense [LSP08], correspondences, solve for an *extrinsic* deformation, namely the optimization variables depend on the local rotation that is applied to each face or vertex of the mesh to obtain the deformed mesh. Two more recent deformation-based correspondence approaches [AXZ$^+$15, ZYL$^+$17] target mostly man made shapes consisting of parts that can be represented using simple geometric primitives, and are therefore less appropriate

for non-isometric manifold models.

4.2 The elastic matching model

In a very general setup we consider two discrete surfaces, i.e. a *source* surface M_1 and a *target* surface M_2 having $n_1, n_2 \in \mathbb{N}$ vertices, respectively. In general, we assume $n_1 \neq n_2$, hence M_1 and M_2 do *not* share the same connectivity. The meshes are represented by the coordinate matrices with rows containing vertex positions, i.e. $V_1 \in \mathbb{R}^{n_1,3}$ and $M_2 \in \mathbb{R}^{n_2,3}$, respectively. Since the connectivities of M_1 and M_2 are supposed to be fixed throughout our algorithm, there are unique relationships $M_1 \leftrightarrow V_1$ and $M_2 \leftrightarrow M_2$, respectively, and we interchange notation if appropriate.

We aim at studying deformations Φ_{12} of the source surface M_1 that are constrained to the target surface M_2 by means of a soft penalty. In detail, we consider two distinct situations. First, we only require Φ_{12} to be locally *injective*. This model is suitable for the matching of almost isometric shapes and, in particular, for partial matching. In a second step, we expand the model to be suitable for non-isometric matching problems. To this end, we additionally consider a reverse deformation Φ_{21} of M_2 and study pairs of deformations (Φ_{12}, Φ_{21}), which are approximately inverses of each other.

4.2.1 Locally injective matching

In the locally injective matching case the inclusion

$$\Phi_{12}(M_1) \subset M_2\,, \tag{4.1}$$

should hold approximately and the deformation Φ_{12} should induce as little distortion as possible. We therefore define Φ_{12} such that V_1 and $\Phi_{12}(V_1)$ share the *same* connectivity, and use well established elastic deformation energies to control this distortion. In addition, to establish (4.1) on discrete surfaces, i.e. on their nodal positions V_1 and M_2, we use a projection mapping from $\mathbb{R}^{n_1,3}$ to $\mathbb{R}^{n_2,3}$ (see Fig. 4.2). These aspects are explained in detail in the following two paragraphs.

Deformation energy. We assume the deformation Φ_{12} to be defined on vertices - the deformed values of interior points are obtained by piecewise linear interpolation on the faces. In particular, $\Phi_{12}(V_1) \in \mathbb{R}^{n_1,3}$ and the resulting mesh has the same connectivity as M_1. To simplify notation, we introduce an auxiliary variable $V_{12} := \Phi_{12}(V_1)$ that contains the deformed vertex positions of M_1 and represents Φ_{12} uniquely. Now we build on the vast literature on physical deformation energies defined between two triangle meshes sharing the same connectivity, e.g. [TPBF87, GHDS03, BS08, FB11, HRWW12]. In these approaches the distortion induced by an *elastic deformation* is separated into two distinct contributions, i.e. *membrane distortion* and *bending distortion*. To this end,

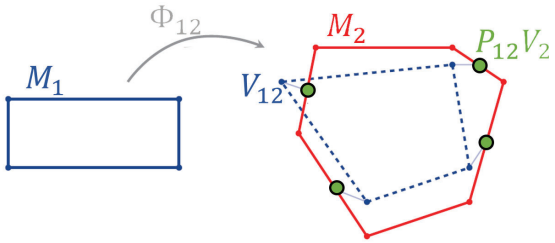


Figure 4.2: A sketch of the locally injective matching configuration for 1D simplicial meshes in $\mathbb{R}^2$. The source mesh M_1 with coordinate matrix V_1 (blue) is deformed via Φ_{12} to a mesh with coordinate matrix V_{12} (dashed blue). The deformed vertices are projected based on the matrix P_{12} onto the target mesh M_2 (red).

a generic elastic energy in the context of thin shell deformations can be written as

$$\mathcal{W}_{\text{def}}(V_1, V_{12}) = \alpha\, \mathcal{W}_{\text{mem}}(V_1, V_{12}) + \eta\, \mathcal{W}_{\text{bnd}}(V_1, V_{12}) \tag{4.2}$$

with weights α and η. While $V_1 \in \mathbb{R}^{n_1,3}$ is fixed throughout the algorithm, $V_{12} \in \mathbb{R}^{n_1,3}$ is a primal variable that we optimize for. The definition of the two energy components will in particular ensure that (4.2) is invariant with respect to rigid deformations and isometric deformations minimize the membrane energy. More details also on the physical background on this energy will be given below in Section 4.3.

Projection mapping. To establish a suitable approximation of condition (4.1) on triangular surfaces we have to consider mappings between the involved discrete spaces. To this end, we consider a linear projection map $P_{12} : \mathbb{R}^{n_2} \to \mathbb{R}^{n_1}$ such that $P_{12}V_2$ is a projection of V_{12} onto the target surface M_2. In particular, the mapping P_{12} represents another degree of freedom. Then (4.1) can be achieved in a discrete setup by penalizing $\|P_{12}V_2 - V_{12}\|^2_{V_1}$ (similarly to (3.10)), where the (squared) norm on the source surface is given as the lumped L^2-norm

$$\|B\|^2_{V_1} = Tr\left(B^\top A_1 B\right) \tag{4.3}$$

with $A_1 \in \mathbb{R}^{n_1,n_1}$ being the lumped mass matrix of M_1 (with the vertex areas on the diagonal).

The linear projection $P_{12} : \mathbb{R}^{n_2} \to \mathbb{R}^{n_1}$ can be represented as a (sparse) matrix $P_{12} \in \mathbb{R}^{n_1,n_2}$ which contains barycentric coordinates for triangles of M_2. In detail, the i-th row of the matrix P_{12} has at most three nontrivial entries $0 \leq P_{12ij}, P_{12ik}, P_{12il} \leq 1$, such that (jkl) represents a triangle in M_2 and $P_{12ij} + P_{12ik} + P_{12il} = 1$. Let $\mathcal{P}_{12} \subset \mathbb{R}^{n_1,n_2}$ the set of all matrices fulfilling these properties which is denoted as the set of admissible maps. This definition implies that $(P_{12}M_2)_i^T \in \mathbb{R}^3$ is a point on the discrete surface M_2 for $i = 1, \ldots, n_1$. In particular, $(P_{12}M_2)_i^T$ will be a good approximation of the deformed position of the i-th vertex in M_1 whenever $\|P_{12}V_2 - V_{12}\|_{V_1}$ is small. Note that we only project vertices, i.e. corresponding edges/faces might not be mapped onto the target

surface.

Total energy. Altogether we obtain the following variational problem: For fixed $V_1 \in \mathbb{R}^{n_1,3}$ and $M_2 \in \mathbb{R}^{n_2,3}$ we minimize the energy

$$\mathcal{E}_{\text{inj}}(V_{12}, P_{12}) = \mathcal{W}_{\text{def}}(V_1, V_{12}) + \beta \, \|P_{12}V_2 - V_{12}\|_{V_1}^2 \tag{4.4}$$

for $V_{12} \in \mathbb{R}^{n_1,3}$ and $P_{12} \in \mathcal{P}_{12}$, where β is a penalty parameter and $\mathcal{W}_{\text{def}}$ a generic elastic deformation energy as in (4.2) that will be further specified in Section 4.3. Note that for an optimal energy (4.4) the resulting deformation Φ_{12} defined by the optimal V_{12} will be (close to) an isometry and, in particular, locally injective. Due to the matching term we expect that there are no local overfolds in the projection $P_{12}V_2$, which is close to V_{12} in an L^2-sense. However, Φ_{12} is not necessarily surjective which is obvious for instance for partial matching problems.

4.2.2 Bijective matching

In the case of strongly non-isometric surfaces we additionally favor surjectivity, i.e. we seek for one-to-one matching deformations. To this end, we expand our matching model in a two step procedure. First, we introduce a reverse deformation Φ_{21} of M_2 and *symmetrize* our model (4.4). In particular, we consider an additional deformation energy related to Φ_{21} and establish a suitable approximation of the condition

$$\Phi_{21}(M_2) \subset M_1 \, . \tag{4.5}$$

Then, in a second step, we ensure *reversibility* with additional energy terms to imply the opposite inclusions of (4.1) and (4.5), i.e. $M_2 \subset \Phi_{12}(M_1)$ and $M_1 \subset \Phi_{21}(M_2)$, respectively.

Symmetry. We introduce an auxiliary variable $V_{21} \in \mathbb{R}^{n_2,3}$ which contains the deformed vertex positions of the target surface M_2 and represents Φ_{21} uniquely, i.e. $V_{21} = \Phi_{21}(V_2)$ (see Fig. 4.3). Furthermore, we consider a projection map $P_{21} : \mathbb{R}^{n_1} \to \mathbb{R}^{n_2}$ to represent projections of V_{21} onto M_1. Analogously to the definition of $\mathcal{P}_{12} \subset \mathbb{R}^{n_1,n_2}$ above, we define a set $\mathcal{P}_{21} \subset \mathbb{R}^{n_2,n_1}$ of admissible maps, which are sparse matrices containing rows with barycentric coordinates for triangles of M_1. To symmetrize (4.4) we finally consider the additional terms $\mathcal{W}_{\text{def}}(V_2, V_{21})$ and $\tilde{\beta} \, \|P_{21}V_1 - V_{21}\|_{V_2}^2$ where $\tilde{\beta} \in \mathbb{R}$ is a suitable penalty parameter and $\|\cdot\|_{V_2}^2$ is defined analoguously to (4.3). In particular, V_{21} and P_{21} are additional degrees of freedom in our optimization algorithm.

Reversibility. Solely symmetrizing our model the new variables V_{21} and P_{21} are not yet coupled with the variables V_{12} and P_{12} of the original model (4.4). In particular, for a one-to-one matching we have to ensure suitable approximations of the inclusions $M_2 \subset \Phi_{12}(M_1)$ and $M_1 \subset \Phi_{21}(M_2)$. So far, $P_{12}V_2$ is considered as the projection of V_{12} onto M_2 and the difference is penalized in a lumped L^2 sense in the energy. Now, we assume that the same map P_{12} represents the projection of V_1 onto $\Phi_{21}(M_2)$, given

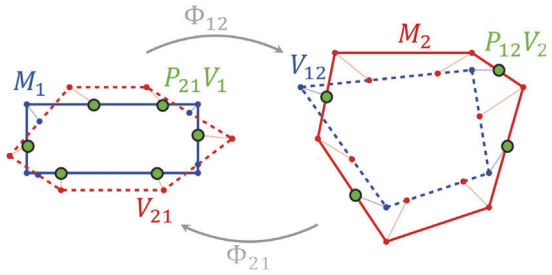


Figure 4.3: To symmetrize the locally injective matching model sketched in Fig. 4.2 we consider a deformation Φ_{21} in the opposite direction, which maps M_2 with coordinate matrix V_2 to a mesh with coordinate matrix V_{21} (dashed red), and a matrix P_{21} encoding the projection onto the source mesh M_1 as additional degrees of freedom.

by $P_{12}V_{21} \in \mathbb{R}^{n_1,3}$ (see Fig. 4.3). The corresponding projection error $P_{12}V_{21} - V_1$ is also measured in the lumped L^2-norm (4.3) to define a corresponding penalty energy. Likewise, $P_{21}V_1$ was defined to reflect the projection of V_{21} onto M_1. Thus, we analogously consider the same map P_{21} as the projection of V_2 onto $\Phi_{12}(M_1)$ with the projection error $P_{21}V_{12} - V_2$, and a resulting penalty energy $\|P_{21}V_{12} - V_2\|^2_{V_2}$ (similarly to (3.9)).

Total energy. Altogether we obtain the following variational problem for the one-to-one matching of strongly non-isometric surfaces: For fixed $V_1 \in \mathbb{R}^{n_1,3}$ and $M_2 \in \mathbb{R}^{n_2,3}$ we minimize the energy

$$\begin{aligned}\mathcal{E}_{\text{bij}}(V_{12}, V_{21}; P_{12}, P_{21}) =& \mathcal{W}_{\text{def}}(V_1, V_{12}) + \beta\|P_{12}V_2 - V_{12}\|^2_{V_1} \\ &+ \mathcal{W}_{\text{def}}(V_2, V_{21}) + \tilde{\beta}\|P_{21}V_1 - V_{21}\|^2_{V_2} \\ &+ \gamma\|P_{12}V_{21} - V_1\|^2_{V_1} + \tilde{\gamma}\|P_{21}V_{12} - V_2\|^2_{V_2} \end{aligned} \tag{4.6}$$

for $V_{12} \in \mathbb{R}^{n_1,3}$ and $V_{21} \in \mathbb{R}^{n_2,3}$ as well as $P_{12} \in \mathcal{P}_{12}$ and $P_{21} \in \mathcal{P}_{21}$. Here, $\beta, \tilde{\beta}, \gamma, \tilde{\gamma} \in \mathbb{R}$ are penalty parameters to be chosen appropriately.

The difference between $\mathcal{E}_{\text{inj}}$ and $\mathcal{E}_{\text{bij}}$ for non isometric shapes is demonstrated in Fig. 4.4. Evidently minimizing $\mathcal{E}_{\text{bij}}$ leads to better results. Note that minimizing (4.6) implies $P_{21}P_{12}M_2 \approx M_2$ and $P_{12}P_{21}V_1 \approx V_1$. Indeed, the triangle inequality yields that $\|P_{21}P_{12}M_2 - M_2\|_{V_2}$ and $\|P_{12}P_{21}V_1 - V_1\|_{V_1}$ become arbitrary small for large enough penalty parameters β, $\tilde{\beta}$, γ, and $\tilde{\gamma}$.

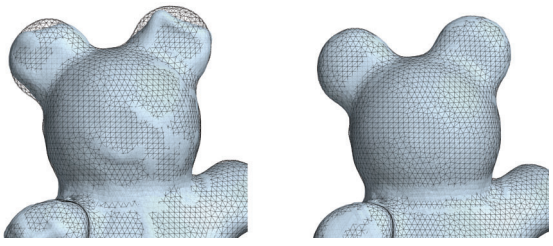


Figure 4.4: The difference between minimizing $\mathcal{E}_{\text{inj}}$ (left) and $\mathcal{E}_{\text{bij}}$ (right) for non isometric shapes from SHREC07 [GBP07]. The final deformation V_{12} is shown as a solid shape, and the target shape is rendered as wireframe. Here, $\mathcal{E}_{\text{bij}}$ significantly improves the matching in the ear region, where $\mathcal{E}_{\text{inj}}$ leads to unwanted artifacts.

4.3 Elastic deformation energy

In this section we focus on the deformation energy associated with a matching deformation Φ_{12} for two discrete surfaces sharing the *same* mesh connectivities. As above, let $V_1 \in \mathbb{R}^{n_1,3}$ and $V_{12} = \Phi_{12}(V_1) \in \mathbb{R}^{n_1,3}$ be the geometries of the *undeformed* and *deformed* configuration, respectively. It will often be convenient to denote quantities living on the undeformed/reference surface with a hat – in particular, those objects are not subject to optimization since V_1 is fixed.

As mentioned before we deploy a model of thin shell elasticity in our approach. If the triangle mesh is considered to be an approximation of a middle layer of a thin elastic material of finite thickness $\delta > 0$, the corresponding membrane energy $\mathcal{W}_{\text{mem}}$ scales as δ, whereas isometric deformations induce a bending energy $\mathcal{W}_{\text{bnd}}$ that scales as δ^3. Here the non-linear membrane energy takes into account stretching and shearing and encodes a preference for isometric deformations, while the bending energy (theoretically in the limit for $\delta \to 0$ only observable for isometric deformations) compares curvature related quantities such as shape operators or mean curvatures and is in particular suitable for feature matching.

4.3.1 Non-linear membrane energy

Considering a simple quadratic energy functional to model membrane deformations a degenerate mesh with vanishing edge length naturally appears as a minimizer. Although a total collapse can in principle be prevented by additional forcing or boundary terms, the favoring of short edges induces a bias in the optimization. To this end, we make use of the non-linear membrane energy

$$\mathcal{W}_{\text{mem}}(V_1, V_{12}) = \sum_{t \in \mathcal{T}} \hat{a}_t W(\mathcal{G}_t)\,,$$

where $\hat{a}_t$ denotes the area of triangle t in V_1 and $\mathcal{G}_t = \hat{g}_t^{-1} g_t \in \mathbb{R}^{2,2}$ is the geometric distortion tensor. Here $\hat{g}_t, g_t \in \mathbb{R}^{2,2}$ are the discrete first fundamental forms of V_1 and V_{12}, respectively, i.e. if $e_0, e_1, e_2 \in \mathbb{R}^3$ are the edges of triangle t we have $g_t = [e_1|-e_2]^T[e_1|-e_2]$ which is invertible if t contains no parallel edges. The hyperelastic energy density $A \mapsto W(A)$ for $A \in \mathbb{R}^{2,2}$ is given by

$$W(A) = \frac{\mu}{2}\text{tr}A + \left[\frac{\lambda}{4}\det A - \left(\frac{\mu}{2} + \frac{\lambda}{4}\right)\log\det A - \left(\mu + \frac{\lambda}{4}\right)\right]\,, \tag{4.7}$$

where $\mu, \lambda \geq 0$ are physical parameters, i.e. the Lamé coefficients of linear elasticity (see [LDRS05, HRWW12] and for mathematical details [Cia00]). Note that we choose $\lambda = \mu = 1$ in all our experiments.

Since we have $W(A) \geq 0$, $W(\mathbb{1}) = 0$ and $W_{,A}(\mathbb{1}) = 0$, the identity map is a minimizer of the energy, and the gradient of the energy is zero at the identity. Furthermore W is

invariant wrt. rigid body motions, i.e., if t is deformed by means of a rigid body motion we have $\mathcal{G}_t = \mathbb{1}$ and hence $W(\mathcal{G}_t) = 0$.

The first term $\sum_{t\in\mathcal{T}} \hat{a}_t \operatorname{tr} \mathcal{G}_t$ is sensitive to changes in edge lengths and coincides with the Dirichlet energy used in [ESBC19]. The second term penalizes variations in triangle areas. In particular, growth of area is penalized quadratically whereas shrinkage is penalized logarithmically to prevent the degeneration of faces, i.e., $W(A) \to \infty$ for $\det A \to 0$. This property is indeed crucial to prevent the collapsing of triangles by a matching deformation.

But, our algorithm relies on a good initialization given e.g. by the output of an established correspondence method (as will be discussed in Sec. 4.4.1). This initialization might come with degenerate faces and led to an undefined membrane energy density due to a zero argument in the log term in (4.7). In order to deal with these situations we slightly modify W in (4.7) by redefining the negative log term. In detail, we extend the function $A \mapsto -\log\det A$ via a linear profile $\log\epsilon + \frac{x-\epsilon}{\epsilon}$ below a small threshold $\epsilon > 0$ such that the compound function is still differentiable. This way $W(A)$ is well-defined even if $\det A \leq 0$.

4.3.2 Bending energy

As the initial choice for our bending energy one might consider the *Discrete Shells* bending energy introduced in [GHDS03], i.e.

$$\mathcal{W}_{\mathrm{DS}}(V_1, V_{12}) = \sum_{e\in\mathcal{E}} \frac{(\hat{\theta}_e - \theta_e)^2}{\hat{d}_e} \hat{l}_e^2 \,, \tag{4.8}$$

where $\hat{\theta}_e, \theta_e$ denote the dihedral angle at some edge e in V_1 and V_{12}, respectively. If $e = t \cap t'$ we have $d_e = \frac{1}{3}(a_t + a_{t'})$, and l_e denotes the edge length of e. This energy has the smallest possible stencil (i.e. two adjacent triangles) to capture bending in a mesh and has been used extensively in the computer graphics community, e.g. [BMF03, TW06, GGWZ07, FB11, HRWW12, HSTP11]. However, when fixing t and rotating t' around the common edge e the dihedral angle θ_e jumps by 2π when the faces intersect each other (see Fig. 4.5). Unfortunately, this is a serious drawback when facing extreme situations, e.g. large dihedral angles and faces penetrating each other, that can appear in the initial phases of the optimization. In particular, the lack of continuity of the above energy density leads to a failure of line-search methods in the optimization. To resolve this we propose a modified version of (4.8) that rather penalizes deviations in the *cosine* of the dihedral angles and is continuous for all pairs of dihedral angles:

$$\mathcal{W}_{\mathrm{bnd}}(V_1, V_{12}) = \sum_{e\in\mathcal{E}} \frac{(\cos\hat{\theta}_e - \cos\theta_e)^2}{\hat{d}_e} \hat{l}_e^2 \,. \tag{4.9}$$

For $e = t \cap t'$ we have $\cos\theta_e = \langle n_t, n_{t'} \rangle$ where $n_t, n_{t'} \in \mathbb{R}^3$ denote the unit triangle normals of t and t', respectively. This energy density is periodic when rotating t' around the common edge e and has proven to be very robust even for degenerated situtations (see Fig. 4.5). Furthermore, the evaluation of (4.9) is cheaper than (4.8), since the latter requires the costly calculation $\theta_e = \arccos\langle n_t, n_{t'} \rangle$.

This modification of the bending energy provides robustness and does not affect the physical or semantic correctness of the results in all of the conducted experiments. This can be verified by comparing the results obtained with both energies, for a pair of meshes where the Discrete Shells bending energy does not cause numerical problems. Such an example is shown in Fig. 4.6.

Combining $\mathcal{W}_{\text{mem}}$ and $\mathcal{W}_{\text{bnd}}$ we define the total elastic energy (4.2) with weights

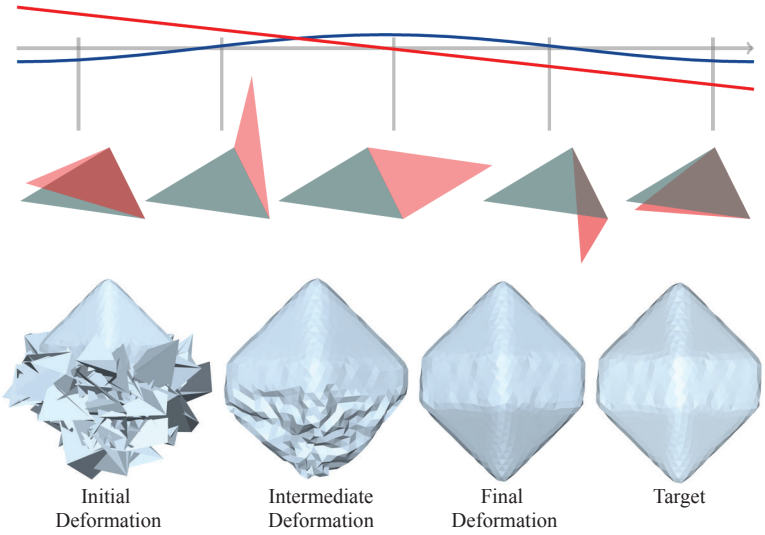


Figure 4.5: Robustness of bending energy. Top: different rotation angles for two adjacent triangles and corresponding graphs of discontinuous dihedral angle (red) and its periodic cosine (blue). Bottom: Reconstruction of degenerated mesh by minimizing $Y \mapsto \mathcal{W}_{\text{def}}(X, Y)$. Starting from a degenerated state (left), we first optimize with $\eta = 10^{-5}$ (middle left) and finally with $\eta = 10^{-3}$ (middle right) to restore X (right).

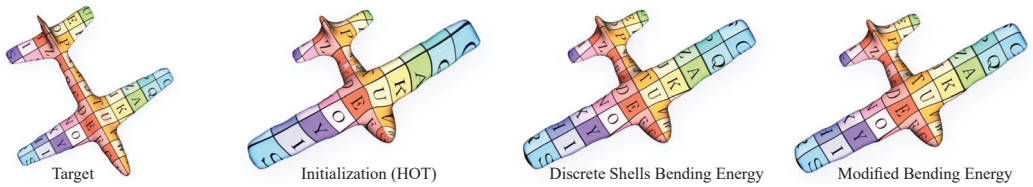


Figure 4.6: Comparison of the Discrete Shells (DS) bending energy (Eq. (4.8)) and our modified bending energy (Eq. (4.9)). From left to right: the target texture, the initial map (HOT), our optimized map using the DS bending energy (4.8), and our optimized map using the modified bending energy (4.9). Note that the modified energy remains faithful to the DS energy, leading to very similar results, yet with the advantage of increased robustness (see Fig. 4.5).

$\alpha, \eta > 0$. Note that due to the scaling properties discussed above we have $\eta/\alpha \sim \delta^2$, where $\delta > 0$ is the physical thickness of the approximated thin shell. Obviously, the total elastic energy (4.2) is invariant with respect to rigid deformations.

4.4 Optimization

In this section we describe the optimization pipeline in order to solve the bijective matching problem, i.e. minimizing (4.6). The corresponding algorithm to optimize (4.4) can be considered as a special case, i.e. with certain parameters set to zero and, in particular, less degrees of freedom. The degrees of freedom (DOFs) of the energy (4.6) are given by $V_{12} \in \mathbb{R}^{n_1,3}$ and $V_{21} \in \mathbb{R}^{n_2,3}$ as well as $P_{12} \in \mathcal{P}_{12}$ and $P_{21} \in \mathcal{P}_{21}$. The numerical minimization of (4.6) is based on an alternating optimization strategy, i.e. we sequentially minimize one of the above matrix valued DOFs while fixing the other three. In the following, we discuss the main features of this algorithm as well as additional implementation details.

4.4.1 Alternating optimization

Given suitable initial mappings $P_{12} \in \mathcal{P}_{12}$ and $P_{21} \in \mathcal{P}_{21}$, which represent projection maps of V_1 onto M_2 and V_2 onto M_1, we initialize $V_{12} := P_{12}V_2$ and $V_{21} := P_{21}V_1$. Then we perform the alternating optimization algorithm described in Algorithm 4.1. Each of the K *outer* iterations of the algorithm consists of a sequential solution of four *inner* optimization problems, two optimizing for the mappings P_{12}, P_{21}, and two optimizing for the deformations V_{12}, V_{21}.

Mapping optimization. The minimization of $\mathcal{E}_{\text{bij}}$ with respect to P_{12} and with fixed V_{12}, V_{21}, P_{21} is performed by solving small, constrained, linear least squares systems for each row of P_{12}, similarly to the approach suggested by Ezuz et al [EBC17]. The terms that contain P_{12} are

$$\beta\|P_{12}M_2 - V_{12}\|_{V_1}^2 + \gamma\|P_{12}V_{21} - V_1\|_{V_1}^2,$$

thus we need to balance the quality of the projection of V_{12} onto M_2 with the quality of the projection of V_1 onto $\Phi_{21}(M_2)$. As all variables except for P_{12} are fixed, this term can be written as $\|P_{12}A - B\|_{V_1}^2$, where A, B collect the fixed terms. Note that instead of considering this as the balancing of two projections in $\mathbb{R}^3$, we can think of A and B as coordinates in $\mathbb{R}^6$. Therefore the optimal P_{12} will give the orthogonal projection of the rows of B on a triangle mesh with the triangulation of M_2, that is embedded in $\mathbb{R}^6$ and its vertex positions are given by A. This projection can be computed naively by finding for each row of B the distance to its orthogonal projection on each such 6 dimensional triangle. Let (j_0, j_1, j_2) be the triangle that realizes the smallest distance to the i-th row of B. Then, the three entries $P_{12_{ij_k}}$ for $k = 0, 1, 2$ are set to the corresponding barycentric coordinates of the closest point on the triangle. To improve efficiency, it

is enough to project each point only to a relevant subset of the triangles [EBC17]. In addition, this process can be parallelized as the projection can be done in parallel for each point and triangle, and therefore it is performed on the GPU.

Deformation optimization. Minimizing $\mathcal{E}_{\text{bij}}$ with respect to V_{12} is done with a Quasi-Newton method while fixing all other variables. Hence, we only need to evaluate the energy and the energy gradient with respect to V_{12}. In detail, we employ a standard BFGS-method using Armijo's inexact line-search (as described in [NW99] for example) with a stopping criterion $\epsilon = 10^{-8}$ and a maximum number of 50 BFGS iterations.

Finally, analogous strategies are used for optimizing with respect to P_{21} and V_{21}. Note that $P_{12} \in \mathcal{P}_{12}$ and $P_{21} \in \mathcal{P}_{21}$ by construction. For the locally injective matching problem the only degrees of freedom are P_{12} and V_{12}, hence there are only two inner problems.

4.4.2 Implementation details

Global scale. Our deformation energy favors rigid body motions as solutions, where the geometric distortion tensor is the identity. We therefore scale the shapes so that the total area of each is one. This ensures that the argument of the membrane energy density (4.7) is not globally shifted away from its local minimum at the identity, and thus improves the robustness of our method.

Initialization. We chose to initialize our method using the Hyperbolic Orbifold Tutte Embedding (HOT) method [AL16], that produces bijective maps given a sparse set of landmarks. As HOT is intrinsic, it cannot take into consideration extrinsic features such as crease lines, which our method can align and thus significantly improve the map, as we demonstrate in the next Section.

In Fig. 4.7 we investigate the robustness of our method with respect to the initializa-

Algorithm 4.1 Alternating optimization for the bijective matching problem with $K = 300$ outer iterations and four inner variational problems to be solved sequentially via different optimization methods.

Input: Two triangle meshes M_1, M_2, initial $P_{12} \in \mathbb{R}^{n_1,n_2}$ and $P_{21} \in \mathbb{R}^{n_2,n_1}$
Output: Optimized P_{12} and P_{21}

Scale M_1, M_2 such that each has unit total area
Get vertex positions matrices $V_1 \in \mathbb{R}^{n_1,3}$ and $V_2 \in \mathbb{R}^{n_2,3}$
Initialize $V_{12} \leftarrow P_{12}V_2$ and $V_{21} \leftarrow P_{21}V_1$
Until (Energy increment $<$ Threshold) or (IterationCount $>$ K)
$P_{12} \leftarrow \underset{P_{12}\in\mathcal{P}_{12}}{\text{argmin}}\ \mathcal{E}_{\text{bij}}(V_{12}, V_{21}; P_{12}, P_{21})$ (projection)
$V_{12} \leftarrow \underset{V_{12}\in\mathbb{R}^{n_1,3}}{\text{argmin}}\ \mathcal{E}_{\text{bij}}(V_{12}, V_{21}; P_{12}, P_{21})$ (BFGS)
$P_{21} \leftarrow \underset{P_{21}\in\mathcal{P}_{21}}{\text{argmin}}\ \mathcal{E}_{\text{bij}}(V_{12}, V_{21}; P_{12}, P_{21})$ (projection)
$V_{21} \leftarrow \underset{V_{21}\in\mathbb{R}^{n_2,3}}{\text{argmin}}\ \mathcal{E}_{\text{bij}}(V_{12}, V_{21}; P_{12}, P_{21})$ (BFGS)
end

tion. We first modify the initial HOT map by composing it with an area-preserving map generated by the flow of a divergence free vector field. Such a transformation introduces variation in the initial map while keeping it bijective. We show the initializations and results for such a modification where the vector field is flown for shorter (VFx5) and longer (VFx10) times. The smaller distortion has no effect on our results, while the larger distortion leads to convergence to a different final map, where locally the result is smooth. Additionally, we distort the initial map by snapping each mapped point to the closest target vertex (NN). Such a distortion causes many degeneracies, therefore we use a smaller $\eta = 0.002$ to successfully handle it.

Masking edge sets. While HOT is theoretically bijective, it can generate faces which have numerically zero area due to finite numerical precision. Therefore, our optimization should be robust to such occurrences of collapsed triangles during the iterations. We have already described in Section 4.3 how we modify the non-linear membrane energy to be well-defined for degenerate triangles. However, such a simple extension is not appropriate for the bending energy (4.9), where the evaluation of the energy density at an edge requires the existence of two adjacent triangle normals. Therefore, the definition (4.9) is slightly modified such that we only sum over *admissible* edges, where an edge is admissible if its two adjacent triangles are not degenerate. Of course, the definition of the gradient is modified accordingly. The set of admissible edges is updated after each inner iteration step of the BFGS-method. In most experiments, we observe that almost all edges are admissible after a few outer optimization iterations, and stay admissible throughout the optimization process.

Parameters. The parameter values can be tuned according to the properties of the shapes and the desired properties of the output, e.g. whether we expect the result to be

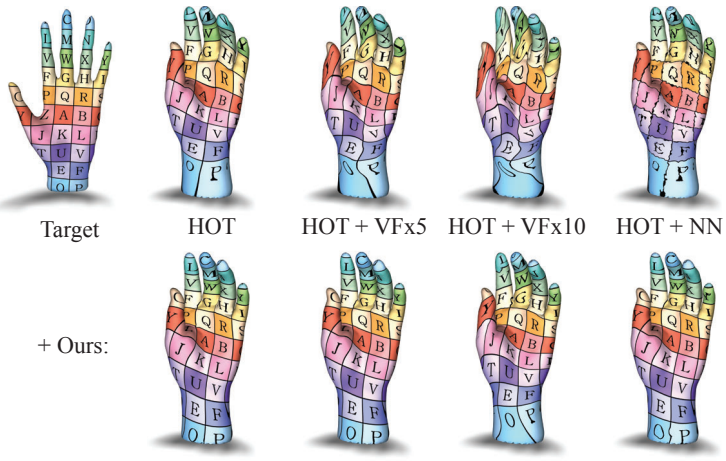


Figure 4.7: Effect of the initialization. We apply our method to the HOT initialization, and to three different distorted HOT initializations (see the text for the distortion details). Moderate perturbations (VFx5, NN) have no effect on our final map, whereas a strongly amplified perturbation (VFx10) leads to convergence to a different final map, that has a moderate coarse scale twist in cylindrical regions, yet is still locally smooth.

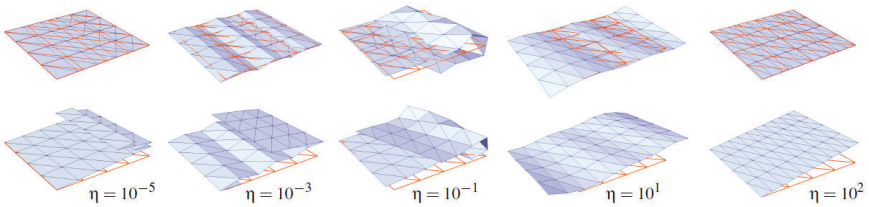


Figure 4.8: Impact of bending weight. A flat, rectangular source domain $[0, 2] \times [0, 1]$ is mapped to the flat unit square (orange wireframe). We show the optimal deformation Y (blue solid, first row) as well as Y with a linear scaling in z-direction for a better visualization (blue solid, second row). The injective matching model was used with $\alpha = 10$, $\beta = 1$ (resp. $\beta = 100$ in the rightmost column) and increasing bending weights η. For the initialization of Y we added some noise in z-direction (at most 1%) to the flat source shape. The optimal deformation ranges from an isometry (left) to a uniform compression without any bending distortion (right).

close to isometric or not.

α The weight of the non-linear membrane energy, mainly controls the area distortion. In all our experiments we used $\alpha = 20$ along with $\lambda = \mu = 1$ for the Lamé coefficients in (4.7).

η The weight of the bending energy, controls the extrinsic feature alignment. For isometric shapes, a moderate value of η is sufficient to regularize the non-linear membrane energy, and prevent undesired folding. Non-isometric shapes require a larger value of η, to favor feature matching over minimizing the amount of stretch. On the other hand, if flipped, or nearly degenerate, triangles appear in the initial map, η should have a smaller value, to allow the membrane energy to recover a smooth map. Finally, η can be used to either enforce an isometric deformation or oppress any bending distortion (see Fig. 4.8).

β The weight of the mismatch energy, controls the correspondence between P_{12} and V_{12}. It is a standard penalty coefficient, and we follow the standard guideline of penalty methods [WYYZ08] that suggests increasing β throughout the optimization process. A low value of β at the beginning of the optimization enables a significant deviation from the initial map, and β should increase to ensure the final map P_{12} corresponds to the deformation V_{12} whose elastic energy is minimized. For all the experiments we start with $\beta = 10^3$ and linearly increase until $\beta = 2 \cdot 10^5$.

γ The weight of the reversibility energy term, controls bijectivity. It should be high for non-isometric shapes to ensure bijectivity. We use $\gamma = 2 \cdot 10^4$ in all our experiments where reversibility was employed.

We use the same parameters for the symmetrized version of the energy, i.e. $\tilde{\beta} = \beta$ and $\tilde{\gamma} = \gamma$.

Timing. We measured example timings of our method applied on shapes with 5K vertices. The average duration of a single iteration is 1.5 seconds, and the complete optimization took 7.5 minutes. These measurements are the same order of magnitude as recent shape correspondence methods. All of our experiments were executed on a desktop machine with a TITANX GPU and an Intel Core i7 processor.

4.4.3 Limitations

The proposed method relies on the computation of an initial correspondence map, which is allowed to contain degenerate triangles but should not have too complicated folds and self-penetrations. To this end, our method can also be seen as a post-processing step to ensure extrinsic feature alignment on top of a fair intrinsic matching result. Unfortunately, for highly complex meshes the minimization algorithm might get stuck in a local minimum far from the global optimum. Here, a multiscale approach is required, which couples the mesh resolution on the source and the target and adapts the bending energy with an appropriate notion of scale dependent curvature information. Finally, our method is oblivious to the orientation of the resulting map, as we optimize for the first fundamental form of the deformation and not the Jacobian. On the one hand this makes our approach rotation invariant, but on the other hand we cannot incorporate into the energy a penalty for inverted triangles.

4.5 Results

In this section we present the results of our method for datasets that contain various types of shapes. We show quantitative and qualitative results that demonstrate improvement over the state of the art. In addition we show the applicability of our method for the applications such as consistent quadrangular remeshing and shape interpolation.

4.5.1 Benchmark evaluation

We measure the error with respect to the ground truth using the standard cumulative error graph [KLF11], that shows the percentage of points on the source shape (y axis) whose corresponding point on the target shape is closer than a certain geodesic distance (x axis) to the ground truth correspondence. The geodesic distances are normalized by $\sqrt{s_2}$ where s_2 is the total face area of M_2. When only a sparse ground truth is given, the percentage of points is relative to the number of landmarks. Similarly, we show cumulative error graphs of the conformal and area distortion. We compute the conformal distortion per triangle as defined by Hormann et al. [HG00], and subtract 2 so that the minimal conformal distortion is zero. Area distortion per triangle is computed similarly to Nadeem et al. [NSZ+17] as $\left|\log \frac{a_t}{\hat{a}_t}\right|$ where $\hat{a}_t$ is the area of the source triangle of M_1 and a_t is the area of the corresponding deformed triangle.

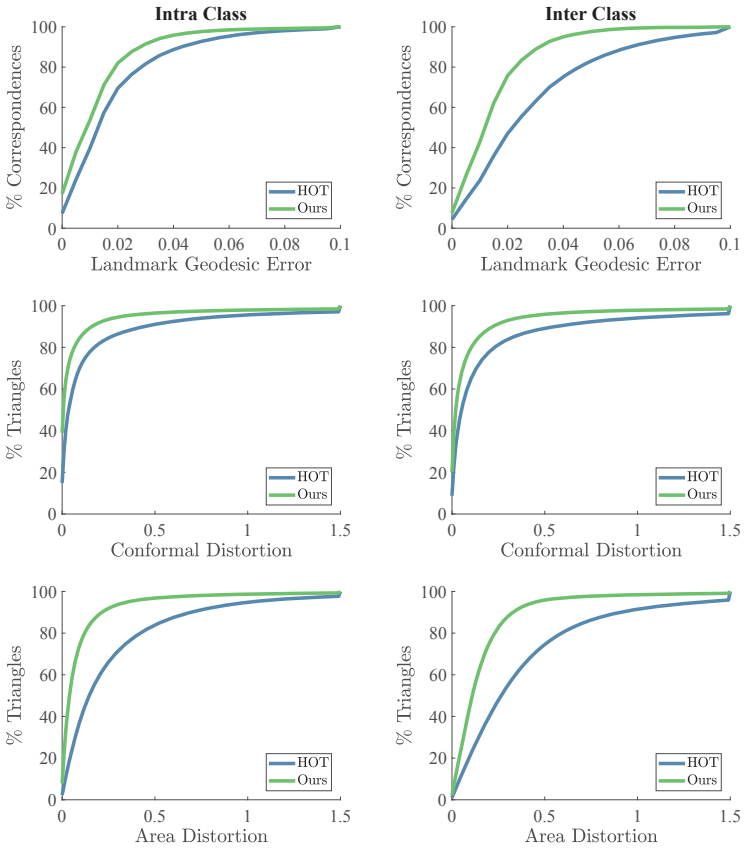


Figure 4.9: Quantitative results for the FAUST dataset. Our method significantly outperforms the initialization according to all distortion measures.

For the qualitative evaluation we use texture transfer. We compute the texture coordinates of the target shape by projecting V_2 on a plane. We then transfer them to M_1 by applying the projection map P_{12} to the texture coordinates. As we use a texture with fine details, this visualization enables the detection of local distortion, and the assessment of semantic correctness. As the texture coordinates are computed by projection on a plane, texture discontinuities indicate regions that are parallel to the projection plane. In addition we visualize the vertex normals of the target surface by color, where the RGB values correspond to the $[x, y, z]$ coordinates of the normals. Again we transfer the normals to the source shape using P_{12}. This visualization emphasizes regions of the surface where features were not mapped correctly.

FAUST Dataset. The FAUST dataset [BRLB14] contains shapes of 10 different humans, each of them in 10 different poses, and a ground truth correspondence is given. Since the shapes in FAUST have the same triangulation, which might bias matching algorithms, we uniformly remesh the dataset to avoid this bias. We recover the ground truth correspondence between the remeshed shapes by composing the projections of the remeshed shapes on the original corresponding shapes.

We employ the standard subsets of intra and inter classes. Note, however, that we use different parameters for the intra class and inter class, as the locally injective model (4.4) is sufficient for isometric shapes. For the inter class, we deploy symmetrization

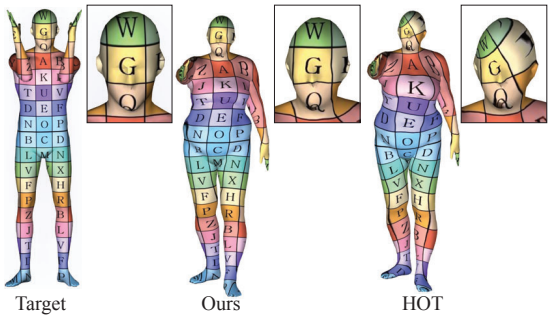


Figure 4.10: Visualization of the initial map and our results for a pair of non-isometric shapes from FAUST using texture transfer. We zoom in on the head to highlight the effect of our crease preserving map, that is semantically correct despite the distorted initialization.

and reversibility, i.e. make use of the bijective model (4.6). We use the same values of $\alpha = 20$ and $\beta = 10^3$ initially and linearly increase until $\beta = 2 \cdot 10^5$. However, we use a larger bending weight $\eta = 0.01$ for the intra class than $\eta = 0.001$ for the inter class. The bending and reversibility terms both prevent "foldings" of the deformed surface, therefore the bending weight is larger when the reversibility term is not used. For the inter class we use $\gamma = 2 \cdot 10^4$. We use 17 landmarks as input to HOT[AL16].

The quantitative results are shown in Fig. 4.9. We measure the error with respect to the ground truth maps, the conformal and area distortion for each class, as specified in section 4.5.1. Our method significantly improves the similarity to the ground truth map, as well as the conformal and area distortion, especially for the more challenging inter class. In Fig. 4.10 we show the qualitative improvement due to our crease-aware method. Note especially the correct mapping of the forehead, the ears and the nose region.

SHREC'07 Dataset. We additionally test our method on 71 pairs of shapes from the SHREC'07 dataset [GBP07], that contains various classes that are mostly non-isometric, and the BIM benchmark [KLF11], which provides a sparse set of corresponding landmarks for each class. Since hard landmark constraints are used in HOT, we only use a subset of the landmarks given by the benchmark to evaluate the ground truth error on the remaining landmarks. For the SHREC'07 dataset we minimize the symmetric energy (4.6) with $\alpha = 20$, $\eta = 0.1$, $\beta = 1000$ initially and linearly increasing for the first 200

Class	VMTP	HOT	RHM	DDM	Ours
glasses	76.12	97.64	100.87	98.11	**54.18**
airplanes	263.70	175.16	183.05	188.63	**105.30**
ants	116.07	136.63	248.56	713.89	**59.44**
tables	290.57	225.59	282.18	237.05	**141.57**
teddies	58.76	85.26	119.68	81.44	**51.90**
hands	**55.94**	122.39	142.93	129.37	89.49
pliers	24.24	36.11	41.43	38.76	**20.27**
fish	59.62	87.69	84.91	89.55	**48.48**
birds	166.88	199.50	158.34	201.23	**84.32**

Table 4.1: Mean curvature error for each class of SHREC'07 [GBP07]. Our method produces the lowest error for all the classes except for the "hands" class.

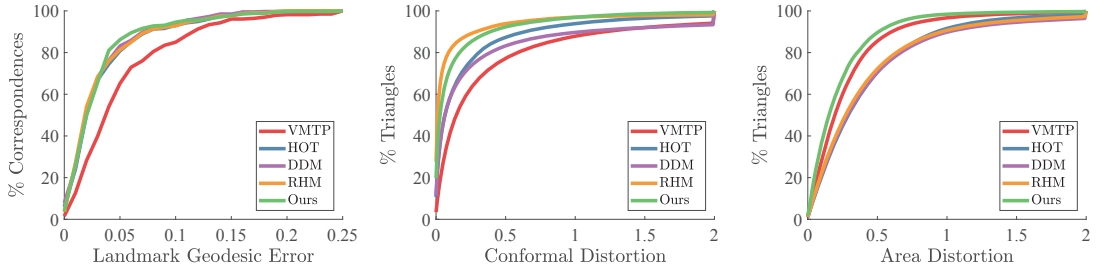


Figure 4.11: Quantitative results for the SHREC07 dataset. Our results have a significantly lower geodesic and conformal distortions compared to VMTP [MCSK+17], and a significantly lower area distortion compared to HOT [AL16], DDM [EBC17] and RHM [ESBC19].

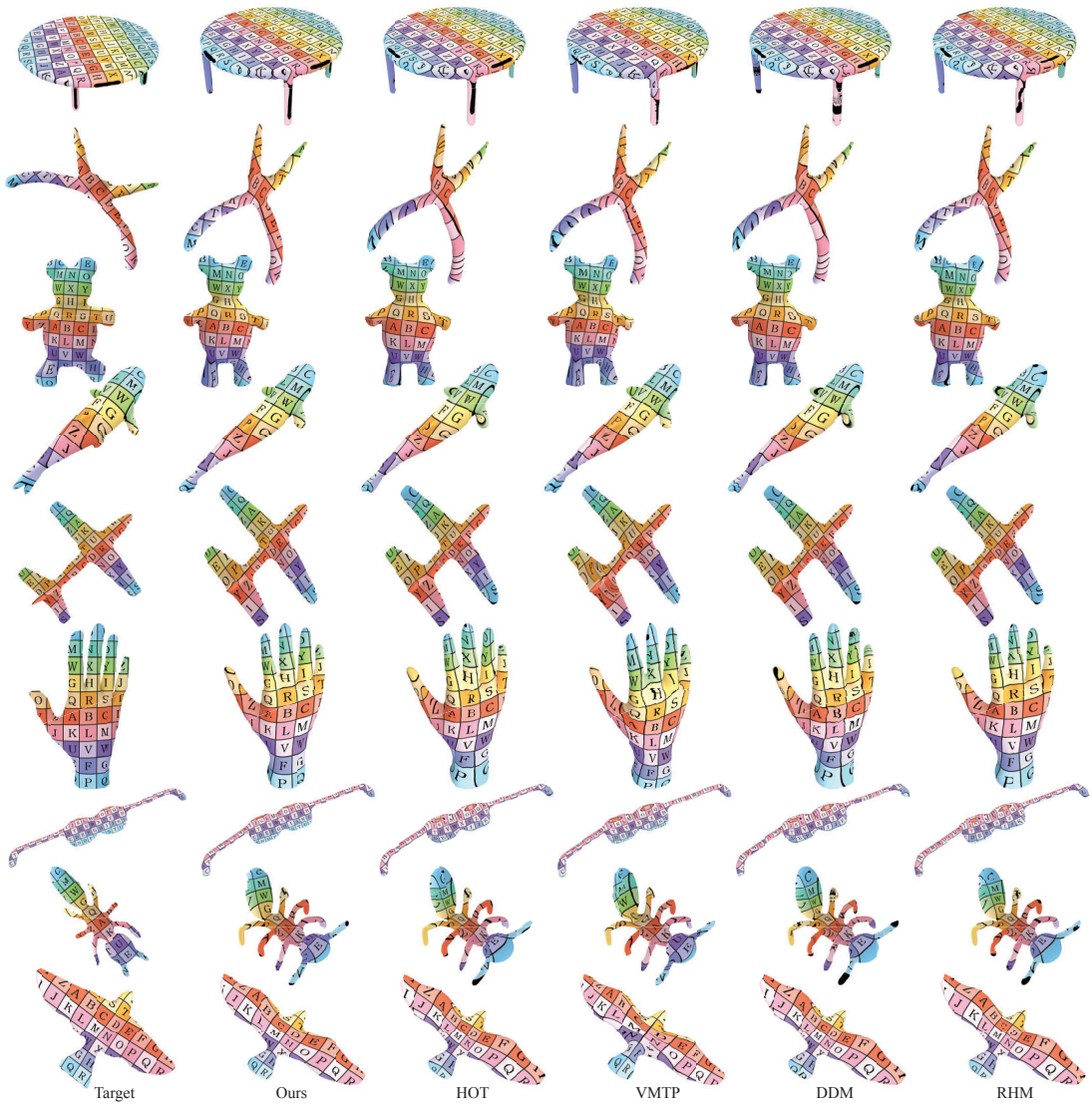


Figure 4.12: Visualization by texture transfer of our results compared to HOT [AL16], VMTP [MCSK+17], DDM [EBC17] and RHM [ESBC19] for a pair from each class we used from the SHREC07 dataset. See the text for details.

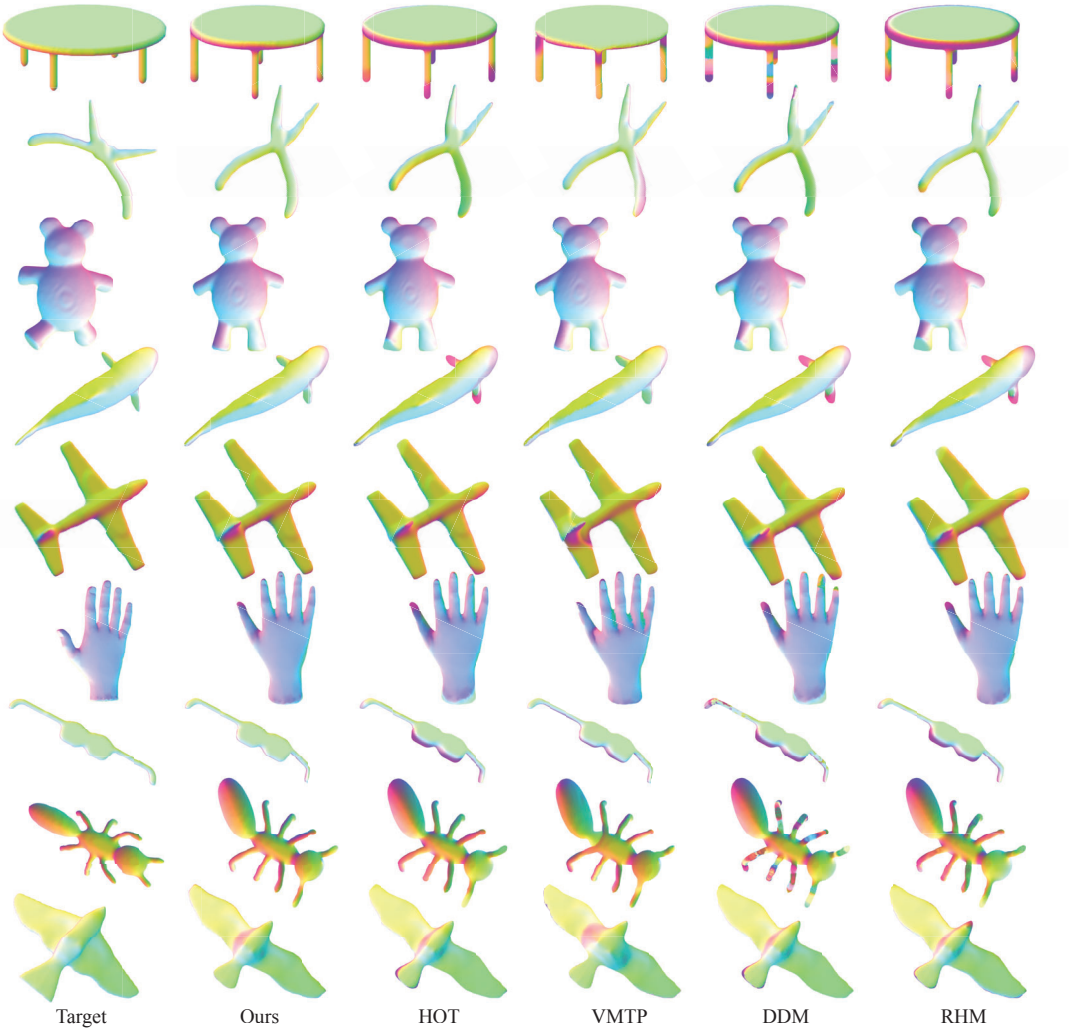


Figure 4.13: Visualization by normal transfer of our results compared to HOT [AL16], VMTP [MCSK+17], DDM [EBC17] and RHM [ESBC19] for a pair from each class we used from the SHREC07 dataset. See the text for details.

iterations and $\gamma = 2 \cdot 10^4$. These parameters are almost identical to the parameters we used for FAUST, with the exception of a larger bending weight η, which is required for the significantly non-isometric shapes of SHREC'07.

We compare our results with the initial map computed using HOT and with the shape correspondence method that was recently suggested by Manded et al. [MCSK+17] (VMTP). Additionally, we compare with two other methods that improve input correspondences, where we use the same initialization as our method (HOT): one method is based on functional maps [EBC17] (DDM) and another optimizes the reversible harmonic energy [ESBC19] (RHM). The quantitative comparison is shown in Fig. 4.11, and visualizations using texture mapping and normal transfer are shown in Fig. 4.12 and 4.13. Note that compared to VMTP our method yields considerably lower geodesic and conformal distortions, as well as a significantly lower area distortion compared to

HOT, DDM and RHM. While RHM has the lowest conformal distortion, it does not align extrinsic features, as is especially notable in the "tables" and "glasses" examples. Thus our approach yields maps which are semantic, and have both low area and conformal distortions.

Additionally, we compare the error of mean curvature alignment. Semantic maps are expected to align features, such that corresponding points should have similar mean curvature values. We compute the mean curvature per vertex of the source and target shapes using [KSNS07], and compute the error per mesh as $\|P_{12}H_2 - H\|_{V_1}$ where H, H_2 denote the vector with mean curvature values per vertex of the source and target shapes respectively. We sum this error across each class we used, and compare the results in Table 4.1. Our mean curvature error is the lowest for almost all classes, with the exception of the "hands" class, where we believe that more landmarks are needed for a better initialization and results.

The advantage of matching mean curvature values is additionally evident in Fig. 4.13. The texture correspondence (left) visualizes our correct mapping of the table crease, the airplane and bird wings, and the sunglasses top crease. The normal mapping (right) serves as an additional visualization of the advantage of our approach. Note that using our result, the pushed-forward normals from the target shape match the creases of the source shape (see e.g., the handles of the plier, the right leg of the teddy, and the bottom crease of the sunglasses).

More results are shown in Fig. 4.14, where we used our method (with $\alpha = 20$, $\eta = 0.002$, $\beta = 1000$ initially and linearly increasing for the first 200 iterations, $\gamma = 2{\cdot}10^4$) to compute correspondence between the left human shape and other human shapes from FAUST, and between the Giraffe and other quadrupeds from SHREC'07. The transferred texture shows that our results are highly accurate even in the case of significant non-isometric deformations.

4.5.2 Partial matching

We demonstrate our method for partial matching, using a simple test case, as shown in Fig. 4.15. We map a part of the tail of an airplane to the whole airplane shape. An initial distorted map is computed by rotation of the partial shape and projection on the target surface. While this initialization method is not robust generally, we use it as a proof of concept to show that our method can be extended to partial matching.

Slight adjustments needed to be made for the partial case. First, rescaling both shapes to have the same total area is not be suitable for partial matching, as it would cause the desired map to have a high area distortion which our method penalizes. In this case we do not scale the shapes. Second, since the total area of the shapes is different, the parameter values also depend on the total area of the shapes; α is divided by s the total area of M_1, and β is divided by $s \cdot s_2$ the multiplication of the total areas of M_1, M_2 to ensure robustness across shapes in different scales. Indeed, using this normalization,

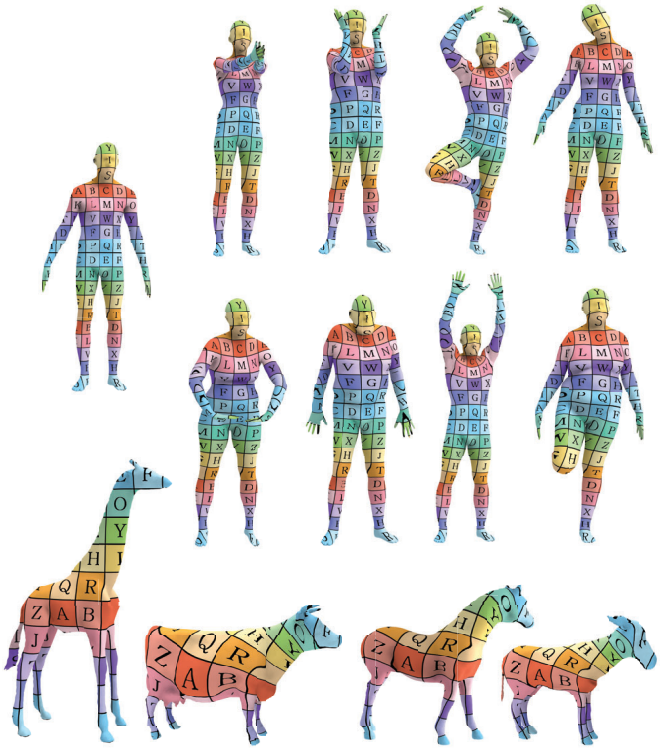


Figure 4.14: Visualization of our correspondence between various non-isometric shapes from the FAUST and SHREC'07 datasets, using texture transfer.

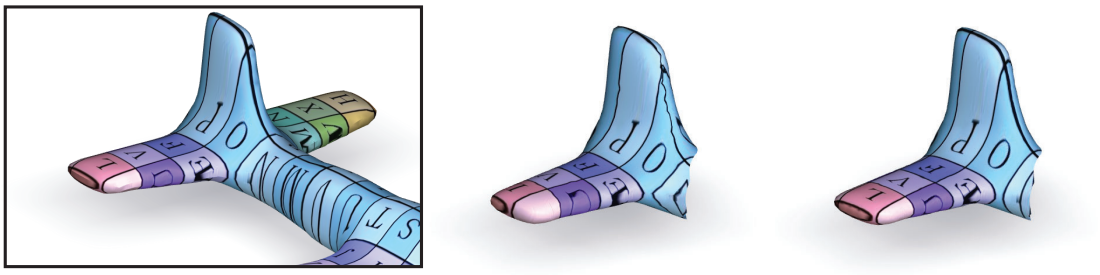


Figure 4.15: Demonstration of our method for partial matching of a part of the tail of an airplane to the whole airplane shape. An initial distorted map (middle) was computed by rotation of the partial shape and projection on the target surface. Our method successfully computed a map (right) that aligns the features correctly.

we used the same $\alpha = 20, \eta = 0.1, \beta = 1000$ for the partial matching example as we used for the full matching of shapes from SHREC07.

4.5.3 Applications

Consistent cross-field design. High-quality feature aligned correspondences are especially important in the context of computing consistent cross fields for a pair of shapes. Consistent cross fields are used for generating (approximately) consistent quad meshes, and are therefore required to align to the principal curvature directions of the shape. If the correspondence does not align correctly the creases of the shapes, a solution

where both cross fields are aligned with the curvature directions, and also correspond under the map, will not exist. We show in Fig. 4.16 the quadrangulations resulting from employing the method of Azencot et al. [ACBCO17], where we either use maps computed with HOT (top row) or with our method (bottom row). Notice that HOT maps induce a significant stretch on one of the plier legs (shown both for the front and back views in the top row). In comparison, our result does not have such a shear, and, in addition, the quad edges are aligned with the curvature directions.

Consistent quadrangulation. Ultimately, shape correspondence can be used to consistently remesh a collection of shapes, for applications such as shape blending and shape analysis. To demonstrate that such an application is feasible with our mapping

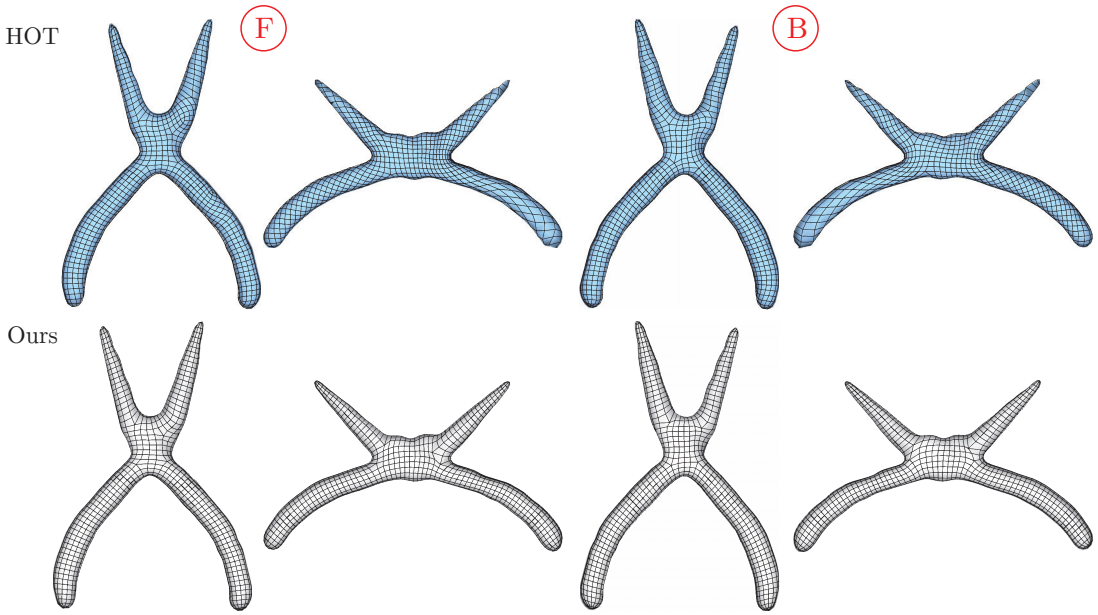


Figure 4.16: We compute consistent cross fields to quadrangulate a pair of shapes and we provide views from the front and the back. Using maps computed with HOT (top row) yields a significant shear on the plier leg. This shear is not observed when our correspondences are used (bottom row).

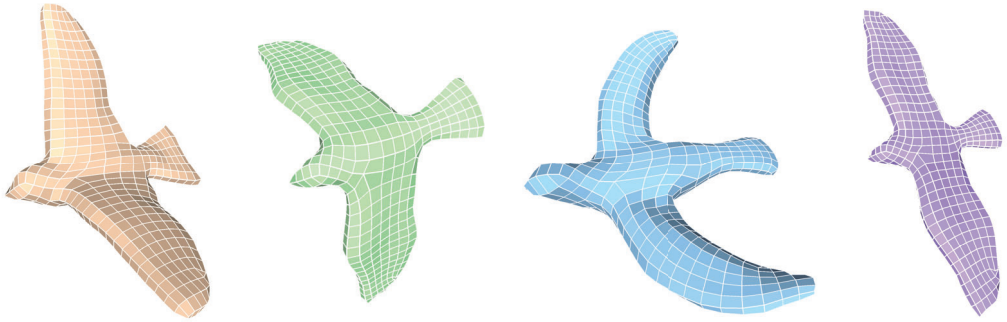


Figure 4.17: Consistent quadrangulation of a set of shapes by computing a curvature aligned quadrangulation of one shape (left) and mapping it to the other shapes (right) using our computed map. Note that the mapped quadrangulations are curvature aligned, although they were not computed as such explicitly.

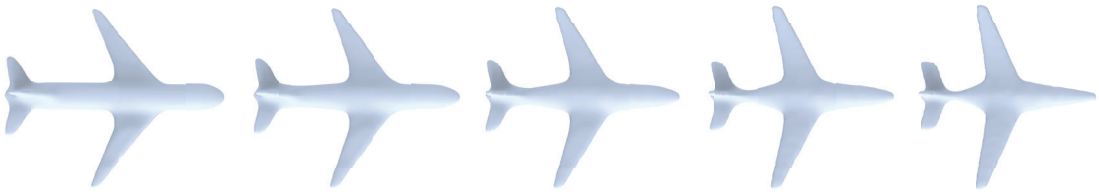


Figure 4.18: Shape interpolation using [HRS+14]. The target shape is remeshed using our result (right) so that the triangulation of the source (left) and target shapes correspond. The resulting interpolation (three middle shapes) is smooth, which indicates a correct semantic mapping of the airplane's features.

approach, we used our results to map a curvature-aligned quadrangulation of a source shape, shown in Fig 4.17 (left), to a set of other shapes, shown in Fig 4.17 (right). Note that the resulting quadrangulations are similarly curvature aligned, although we have not explicitly computed them as such.

Shape interpolation. We demonstrate the usage of our method for shape interpolation. As shape interpolation methods mostly require consistent triangulation of the source and target shapes, a correspondence between them is required. We remesh the target shape according to a map P_{12} by setting the new vertex positions to $P_{12}M_2$ and using the triangulation of the source shape. Then we use [HRS+14] to interpolate between the shapes. The results are shown in Fig. 4.18, and the supplementary video. Note the smooth interpolation of the shapes, which indicates the semantically correct mapping of features.

Chapter 5

GWCNN: A Metric Alignment Layer for Deep Shape Analysis

Recent advances in acquisition and modeling tools for 3D geometry, as well as the rising popularity of user-facing applications such as VR, have led to unprecedented growth in geometric data. This necessitates effective shape analysis algorithms that predict semantic attributes of shapes, essential for organizing, searching, and using geometric datasets for content creation. Deep neural networks hold significant promise for these tasks, providing a fundamental tool for learning a mapping from low-level geometric features to high-level semantic attributes. Unfortunately, these networks usually operate on regular inputs such as n-D grids, which are not natural representations for geometric data. Thus, most existing network architectures pre-process the data by converting unstructured point or triangle surface samples to regular representations.

One common way to do this conversion is to rasterize surfaces to volumetric grids, where each cell either stores a binary occupancy function [QSN$^+$16] or a distance to the closest surface point [SX16]. This leads to surface representations that only capture coarse geometry. One can also project 3D models to multiple external camera planes and analyze the corresponding images [SMKLM15]; this leads to redundancy in representation and is only suitable for cases when the majority of the surface is visible from a small number of pre-defined cameras. The main limitation of the extrinsic approaches described above is that they are sensitive to articulation and deformation. A common way to circumvent this problem is to analyze the surfaces intrinsically. To this end, previous work explored flattening techniques such as geometry images [SBR16] and local geodesic polar coordinates [BMRB16]. While these pre-processing steps typically embed surfaces to a regular domain enabling application of convolutional neural networks, existing embedding procedures stay fixed across all problems and cannot be tailored specifically to the task.

Our main contribution is a *parametric* and *differentiable* mapping layer that can be optimized for a specific problem and dataset. Our key idea is to leverage an efficient algorithm that optimizes the regularized Gromov–Wasserstein (GW) objective [SPKS16]

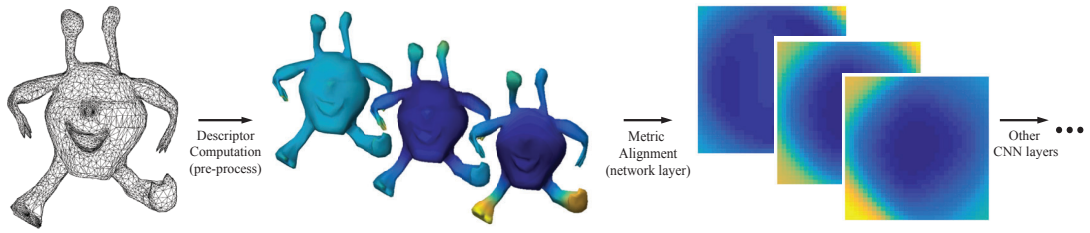


Figure 5.1: Our pipeline. We first compute point-wise surface features that are mapped to a stack of 2D functions over a canonical domain via our novel metric alignment layer (see Fig. 5.5). This provides a natural input for subsequent CNN layers.

to map from unstructured data to a regular representation. Unlike most correspondence algorithms, this regularized technique is differentiable in the geometries of the mapped domains, making it amenable to gradient-based optimization techniques.

Our pipeline is visualized in Figure 5.1. Given an input shape and scalar geometric features, our layer maps the features to a common 2D grid. This yields a multi-channel image over the grid that we feed into standard deep architectures. As we optimize the GW objective [SPKS16], our input is given in the form of a *pairwise distance matrix* and thus our layer can handle polygonal meshes, point clouds or general graphs as long as pairwise distances are computable. Further, the GW map minimizes distortion in pairwise distances between the source and the target, leading to mapped features that are *consistent* under isometric deformation of the source geometry. Finally, we include the geometry of the 2D grid as a variable during the learning stage, *learning* task-optimal mappings from unstructured domains to the regular domain. We implement our layer with the Torch library [CKF11] and use it within a deep architecture for shape classification. Results demonstrate that our approach outperforms state-of-the-art methods on standard benchmarks for non-rigid shape classification and retrieval.

5.1 Related Work

A wide range of fundamental shape analysis problems such as classification [BBGO11], segmentation [KHS10b], and correspondence [COC14] have been addressed with machine learning techniques (see [XKH+16] for a survey). Due to recent developments and success of deep neural networks, researchers have focused on developing appropriate shape representations suitable for deep learning. In this section, we overview relevant extrinsic and intrinsic representations as well as other work related to our approach.

5.1.1 Representations for Deep Learning

Extrinsic. One straightforward representation for geometry is a 3D voxel grid, where each voxel stores a binary occupancy function [MS15] or a truncated distance to the surface [SX16]. Deep CNNs have been used to analyze these grids for shape classification [WSK+15] and geometric modeling [BLRW16, WZX+16]. This representation

is inefficient since typical surfaces occupy only a small fraction of the volume. Riegler et al. [ROUG17] suggested to use an *octree* instead of a dense grid to allow a more compact representation. Another alternative is to project surfaces to external camera planes. One can analyze rendered images [SMKLM15, KAMC17] or depth images from multiple viewpoints [WHC+16] or from panoramic views [SBZB15]. These techniques only work for shapes for which all relevant geometric details are visible from a fixed set of external cameras. Qi et al. [QSN+16] proposed view-dependent volumetric analysis with anisotropic kernels, closing the gap between multi-view and volumetric approaches.

One can also directly analyze features of unordered surface elements such as mesh faces [GZC15], but this approach requires powerful features that provide additional contextual information. Concurrently with our work, Qi et al. [QSMG17] proposed a novel network architecture that directly analyzes coordinates of unordered point sets by using symmetric order-independent max pooling functions. This concept was later extended in an hierarchical model [QYSG17]. Atzmon et al. [AML18] performed convolution on point clouds by performing an Euclidean convolution on the volumetric extension of point cloud functions.

Extrinsic techniques are sensitive to articulation and non-rigid deformation, a common problem in shape analysis addressed via intrinsic shape representations. Such representations often map the shape to a common domain in a way that is invariant to nearly-isometric deformation of the input.

Spectral. Several methods have been proposed for spectral analysis of functions on a graph [BZSL14, HBL15, DBV16, KW17]. While these methods are invariant to isometry, they are limited to analyzing functions on a *specific* non-Euclidean domain (e.g. a graph with fixed connectivity) and thus cannot be used to analyze different geometries. Concurrent to our approach, Yi et al. [YSGG17] propose to synchronize the spectral domains of the graphs to enable cross-shape analysis. They create canonical shape domains (described by their graph Laplacian eigenbases) as a pre-process, and use extrinsic alignments between shapes to initialize functional maps [OBCS+12] to the common domain. While potentially applicable to non-rigid shapes, they focus their analysis on rigid man-made objects and use the consistent extrinsic alignment provided in the ShapeNet dataset for the initialization.

Local mapping. Masci et al. [MBBV15] use geodesic polar coordinates to parameterize a surface locally around every point. Boscaini et al. [BMRB16] improve this approach with anisotropic patches. These techniques operate on relatively small geodesic patches which limits their ability to incorporate global context.

Global mapping. To remedy this, Sinha et al. [SBR16] proposed to use geometry images [GGH02], a global shape parameterization technique for manifold genus zero surfaces. To parameterize point clouds or polygon soups, Sinha et al. use α-shapes [EM94]

to create a manifold input, and topological processing to ensure the input is genus zero. Unfortunately, this pre-processing does not preserve the original geometric details, such as the interior structure of the model. Furthermore, the same surface can be parameterized in multiple ways depending on the placement of cuts. Thus, this method also suffers from inconsistent mappings across similar surfaces. Maron et al. [MGA+17] used seamless parameterization to a planar torus, that allows a translation invariant convolution operator on genus 0 surfaces.

In contrast to existing approaches, our metric alignment layer is based on minimizing the regularized GW objective and thus explicitly optimizes for consistency in aligning surfaces to the regular domain. Moreover, we learn the geometry of the canonical domain specifically for the task at hand.

5.1.2 Shape Parameterization and Mapping.

Mapping a shape to another domain is a common task in shape analysis with applications to texture mapping [LPRM02a], modeling [PSS01], correspondence [AL15], and retrieval [SK16]. Surface parameterization techniques [HLS08] typically require manifold surfaces and do not aim to embed similar shapes consistently. Spectral methods [ZvKD10] usually rely on the first few eigenfunctions of a Laplacian operator and thus only encode low-frequency attributes of shapes. Correspondence techniques map between arbitrary geometric domains [AL15, OBCS+12] while minimizing some distortion metric, thus offering a certain degree of consistency when mapping similar shapes. However, it is not immediately clear how to optimize the embedding process for these methods with respect to a back-propagated loss function of the neural network.

Typical deep learning algorithms require the gradient of the loss function with respect to all parameters of the learned network. This is problematic for shape parameterization as a customizable unit in a network, since at some level the output is a permutation. To overcome this issue, we use the metric alignment method of Solomon et al. [SPKS16] because its use of entropic regularization makes the objective differentiable in the input metric spaces. This differentiability is also leveraged by Peyré et al. [PCS16] to compute barycenters of sets of metric spaces.

5.2 Metric Alignment Layer

5.2.1 Problem Setup

The role of the metric alignment layer is to map scalar geometric features given on an input shape Σ to features on a canonical domain Σ_0. In our design of the layer we have the following goals:

Applicability. The layer should be applicable to *multiple shape representations*, e.g. point clouds and triangle meshes, and its output should be usable with standard network

architectures.

Consistency. Given geometric features that are invariant to isometries, the output of the layer should be *invariant to isometries*.

Learn-ability. The layer should be specified by a set of *parameters*, which can be learned and tuned within a deep learning network. Consequently, the layer should be *differentiable* with respect to these parameters.

With these goals in mind, we make a few design choices. First, since many network architectures are available for images, we define the canonical domain Σ_0 to be n_0 points laid out on the 2D grid, and encode the k output features as a multi-channel image $f_0 : \Sigma_0 \to \mathbb{R}^k$ on this grid. Second, to allow for diverse input representations, we use the Gromov–Wasserstein (GW) generalized mapping algorithm [SPKS16]. The GW algorithm represents a map as a "soft correspondence," namely given two geometries Σ, Σ_0 with n and n_0 points respectively, the algorithm constructs a matrix $\Gamma \in \mathbb{R}^{n_0 \times n}$ such that a pair of points $(p, p_0) \in \Sigma \times \Sigma_0$ is assigned a high probability if they should be matched. We thus define the output of the layer to be $f_0 = \Gamma f$, where $f : \Sigma \to \mathbb{R}^k$ are the features defined on the input geometry. This mapping technique can be applied to point clouds, polygon soups, or any other geometric domain equipped with a metric.

Another advantage of using the GW algorithm is that it encourages consistency as it tries to find an embedding that preserves the metric of the source shape. Specifically, given the pairwise distance matrix $D \in \mathbb{R}^{n \times n}$ between points on the source domain Σ, and the pairwise distance matrix $D_0 \in \mathbb{R}^{n_0 \times n_0}$ on the target domain Σ_0, the matrix Γ is constructed such that if a high probability is given to (x, x_0) and (y, y_0), both in $\Sigma \times \Sigma_0$, then the distances $D(x, y)$ and $D_0(x_0, y_0)$ are similar. Figure 5.2 shows a GW map between a human in different poses and a 2D grid. We visualize the map by selecting

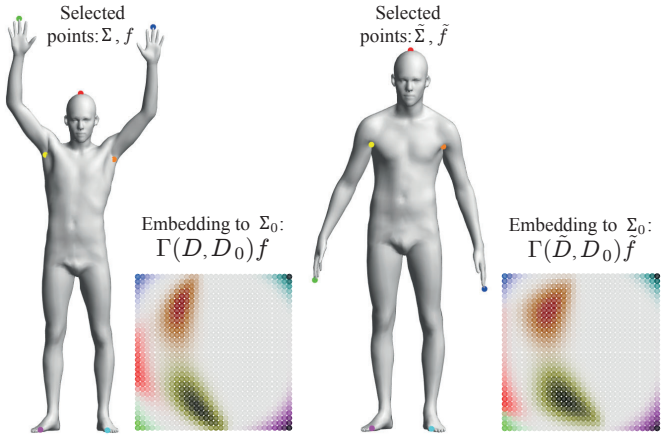


Figure 5.2: Visualization of the GW fuzzy map for two nearly isometric shapes from FAUST [BRLB14]. (left) two shapes and corresponding color coded points, (right) distributions on the grid according to the GW map, high intensity indicates high probability for a match.

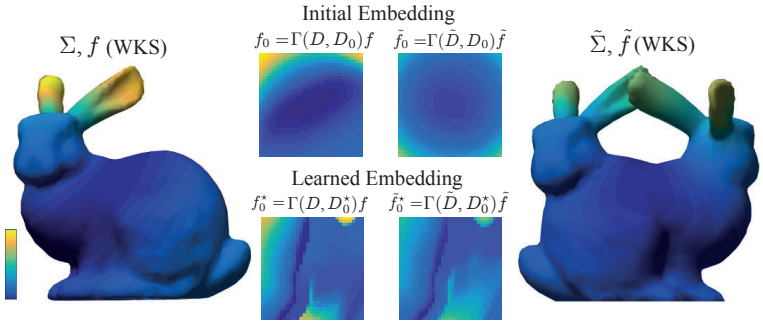


Figure 5.3: Toy example, learning a target metric to optimize consistency. Given input descriptors $f, \tilde{f}$ on two shapes we show embeddings to canonical domain (center) before optimization (top) and after optimization (bottom).

a few feature points on the human, assigning them consistent colors and mapping the colors to the grid with the soft correspondence produced by GW optimization. Note the similarity between the resulting representations on the grid. We use a slightly distorted target grid to avoid symmetric ambiguities in the target metric (see Section 5.3).

Finally, the GW algorithm is differentiable with respect to the geometry of the target domain, represented by the metric D_0. Hence, we can set up optimization problems over the distance matrix D_0 to modify the resulting map, which we demonstrate using the following toy experiment visualized in Figure 5.3. Given two shapes $\Sigma, \tilde{\Sigma}$ of a one-headed and a two-headed bunny, we compute a Wave Kernel Signature [ASC11] descriptor to define the source functions $f, \tilde{f}$ (left and right) respectively. We embed them to the 2D grid Σ_0 with the GW mapping which yields functions over the grid $f_0, \tilde{f}_0$ (center-top). As the surfaces are not isometric these mappings are not consistent and the resulting images are dissimilar. This can be amended by optimizing over D_0 to minimize the image dissimilarity:

$$D_0^\star = \arg\min_{D_0} \|\Gamma(D, D_0) f - \Gamma(\tilde{D}, D_0) \tilde{f}\|^2.$$

We use gradient descent to find a local optimum $D_0^\star$, and the resulting mapped images $f_0^\star$ and $\tilde{f}_0^\star$ are more consistent (center-bottom). Figure 5.4 further demonstrates the improvement in consistency when using the optimized domain. We map a function that highlights the heads of the two-headed bunny (left) to the one-headed bunny via the original and the optimized domains. When using the original domain the two heads are mapped to the tail (center) whereas with the optimized domain the heads are mapped to the head (right).

We use a similar idea within a convolutional neural network architecture, by constructing the metric alignment layer depicted in Figure 5.5. The layer receives as input the distance matrices of the source shapes D, as well as area weights μ. It then learns the target metric D_0 during training so that the mapped descriptors minimize the loss function when plugged into the next layers. Therefore, we effectively tune the regular

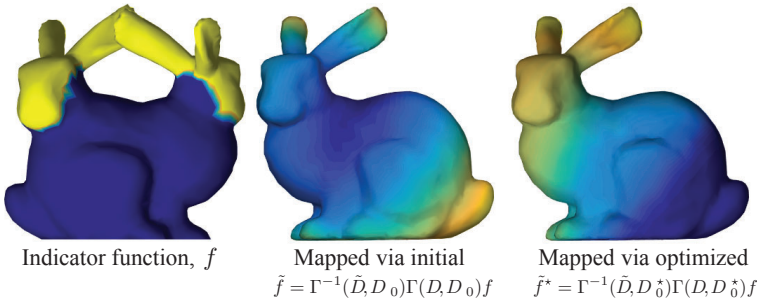


Figure 5.4: Mapping an indicator of the heads (left) to the single-headed bunny before (center) and after (right) the optimization of the target domain. See the text for details.

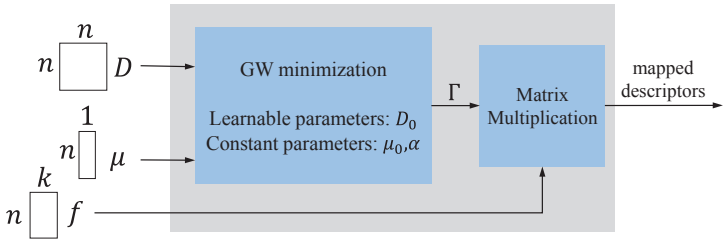


Figure 5.5: Our metric alignment layer. The inputs are a pairwise distance matrix D, a measure μ, and k per-point descriptors. The layer maps the descriptors to a common domain by learning its metric D_0, and generates a mapping matrix Γ. Finally, the input descriptors are multiplied by Γ to generate a stack of 2D images to feed to the following layers.

embedding to produce mapped descriptors that best suit our task.

5.2.2 Implementation

GW minimization. The computation of the mapping matrix Γ between the input domain Σ and the regular domain Σ_0 is the main building block of our metric alignment layer. We use the method proposed by Solomon et al. [SPKS16, Algorithm 1] that requires two distance metrics D and D_0 for two domains and nonnegative area measures μ, μ_0. The output is a matrix Γ that locally minimizes the regularized Gromov–Wasserstein distance measure:

$$GW_2^2\left(\mu_0, \mu, D_0, D\right) := \min_{\Gamma \in \bar{\mathcal{M}}(\mu_0, \mu)} \left[\sum_{ijkl} \left(D_{0ij} - D_{kl}\right)^2 \Gamma_{ik} \Gamma_{jl} \mu_{0i} \mu_{0j} \mu_k \mu_l - \alpha H\left(\Gamma\right) \right] \tag{5.1}$$

where $H\left(\Gamma\right) = -\sum_{ik} \Gamma_{ik} \ln\left(\Gamma_{ik}\right) \mu_{0i} \mu_k$ is the *entropy* of Γ, and

$$\bar{\mathcal{M}}\left(\mu_0, \mu\right) := \left\{ \Gamma \in \mathbb{R}_+^{n_0 \times n} : \Gamma\mu = 1,\, \Gamma^T \mu_0 = 1 \right\}$$

is the set of possible fuzzy maps, also known as *measure couplings*. Intuitively, Γ_{ij} represents the probability that the i^{th} point of Σ_0 corresponds to the j^{th} point of Σ. The parameter α controls the entropy, where larger values create "fuzzier" maps. We set $\alpha = 0.005$ throughout all experiments and normalize each distance matrix by its maximal value.

Derivatives. Typically, neural network parameters are optimized using stochastic gradient descent, in which the chain rule is used to differentiate the loss function with respect to the parameters in a process called backpropagation. Standard implementations of this procedure provide us with the gradient of the loss L with respect to the map Γ, denoted by $\nabla_\Gamma L$; from this matrix, our goal is to compute the gradient $\nabla_{D_0} L$ of the loss L with respect to the metric D_0, which determines Γ through Gromov–Wasserstein optimization.

Recall that [SPKS16] alternates between a closed-form exponential formula and Sinkhorn projection onto the cone of doubly stochastic matrices to compute Γ. For every iteration $i = 1, ..., I$ of their algorithm, we denote the map by Γ^i and the input to Sinkhorn projection as K^i. Given $\nabla_{\Gamma^I} L$ from backpropagation through the later stages of our network, we obtain $\nabla_{D_0} L$ by iteratively computing $\nabla_{\Gamma^i} L$, $\nabla_{K^i} L$ in reverse order from $i = I$ to $i = 1$.

Given $\nabla_{\Gamma^i} L$, $\nabla_{K^i} L$ is computed using partial derivatives of Sinkhorn projection of K^i onto the doubly stochastic cone. While [BPC16] differentiates individual iterations of the Sinkhorn algorithm for this task, for efficiency and storage reasons we choose to compute the derivative of the converged term directly using an implicit linear system derived from the following stationarity conditions for Γ^i:

$$\begin{aligned} \Gamma^i &= [\![v]\!] K^i [\![w]\!] & \Gamma^{i\top} \mu_0 &= [\![w]\!] K^{i\top} (v \otimes \mu_0) = \mathbf{1} \\ \Gamma^i \mu &= [\![v]\!] K^i (w \otimes \mu) = \mathbf{1} & \mathbf{1}^\top v &= \mathbf{1}^\top w, \end{aligned} \tag{5.2}$$

where $[\![v]\!] \in \mathbb{R}^{n \times n}$ is a diagonal matrix with v on the diagonal. Here, v and w are vectors of length n_0, n, respectively, computed during Sinkhorn projection. The first equation is the explicit computation of Γ^i, while the second and third equations define a valid fuzzy map; the last equation ensures that we have unique solution for v, w. All are satisfied when Sinkhorn projection converges. Differentiating these four equations with respect to each entry of K^i yields a linear system whose solution gives the derivatives of Γ^i with respect to the entries of K^i.

Using the chain rule and $\nabla_{\Gamma^i} L$ we iteratively compute $\nabla_{K^i} L$ and $\nabla_{\Gamma^{i-1}} L$; chaining these computations together leads to $\nabla_{D_0} L$. Full derivations and formulas are provided in Appendix C.

Mapped descriptors. Once Γ is computed, we map the k point-wise input features $f \in \mathbb{R}^{n \times k}$ to the regular domain Σ_0 obtaining an image $f_0 = \Gamma [\![\mu]\!] f$. We then reshape

the k mapped images f_0 into k matrices of size $\sqrt{n_0} \times \sqrt{n_0}$ (with $\sqrt{n_0} = 32$ for all experiments), to represent a multi-channel image over the 2D grid. We then feed the resulting k matrices of fixed size to a neural network that may contain convolutional layers, which is expected to learn appropriate kernels for each feature independently.

Parameters. For a given shape Σ we compute n evenly distributed point samples, typically around 1000. To evenly sample n points on a triangle mesh, we first randomly sample $10n$ triangles and barycentric coordinates for each sampled triangle, where each triangle is chosen with probability proportional to its area. Then we cluster the sampled points to n clusters, and select the point closest to the centroid of each cluster. The pairwise distances D are computed using Dijkstra's algorithm on a graph constructed by connecting every sampled point to its 5 nearest neighbors and iteratively connecting closest disconnected components. The measures μ are taken to be one third of the area of neighboring faces for manifold meshes, and unit for point clouds.

For intrinsic per-point features f of deformable manifold meshes we use Gaussian curvature [BKP+10], conformal factor [BCG08] and the first entry of the Wave Kernel Signature [ASC11]. For extrinsic per-point features of polygon soups we use PCA-based features in the local point neighborhoods [KHS10b], height, shape diameter function [SSCO08], and absolute curvature using the publicly available implementation [KCGF14]. The descriptors are normalized to have zero mean and unit variance on the training set.

Computing the metric alignment matrix and its gradients requires solving a large system of linear equations, and can become a major bottleneck in training. To remedy this, we run our metric alignment algorithm for a fixed number of iterations $I = 5$, and use Γ from the previous epoch as the initial solution.

We implement this layer using the Torch library [CKF11] and execute the metric alignment algorithm and its differentiation on GPU using NVIDIA's cuDNN [CWV+14] and MAGMA [TDB10].

5.3 Shape Classification

Given the multi-channel 2D grid as the output of metric alignment layer we can take advantage of standard CNN layers designed for image analysis. In particular, for classification task we use a stack of convolutional layers, batch normalization, ReLU and dropout layers [SHK+14], depicted in Figure 5.6.

While the differentiable metric alignment layer enables us to train this network end-to-end, we found that pairwise distances of the target domain are difficult to optimize directly. Thus, we created a different mini-network dedicated to learning the common domain. Our mini-network only contains the metric alignment layer, and follows the *Siamese architecture* [BBB+93]. We construct our loss function to favor images of shapes within the same category to be as-similar-as-possible, and images of shapes from different

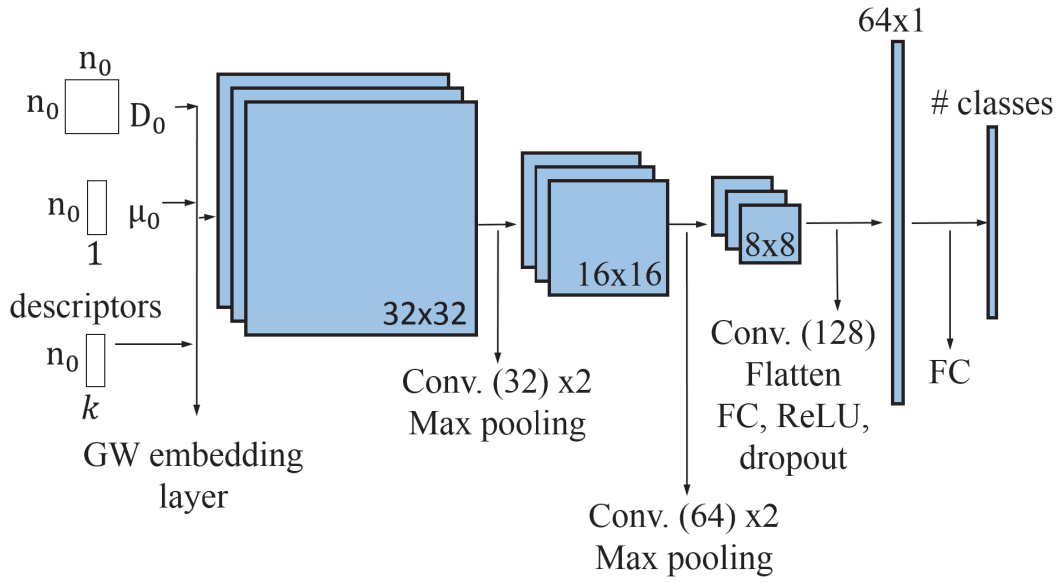


Figure 5.6: Our convolutional neural network for classification and shape retrieval.

categories to be separated by a margin. The input is a pair of shapes Σ_1, Σ_2 with their category labels y_1, y_2 represented using pairwise distance matrices D_1, D_2, measures μ_1, μ_2 and a set of k pointwise descriptors f, g. The features of each shape are mapped to the common domain Σ_0 using the two copies of the metric alignment layer with shared parameters. The embedding creates images f_0, g_0 and we use L2 hinge loss function:

$$L(f_0, g_0) = \begin{cases} \|f_0 - g_0\|_2 & y_1 = y_2 \\ \max(0, m - \|f_0 - g_0\|_2) & y_1 \neq y_2 \end{cases} \tag{5.3}$$

where m is the *margin*. This loss function penalizes *different* embeddings of shapes of the same category, as well as *similar* embeddings of shapes of different categories.

We initialize the pairwise distances of the 2D grid D_0 to the Euclidean distances between the points of a distorted 2D grid Σ_0. We distort the metric in the 2D plane to

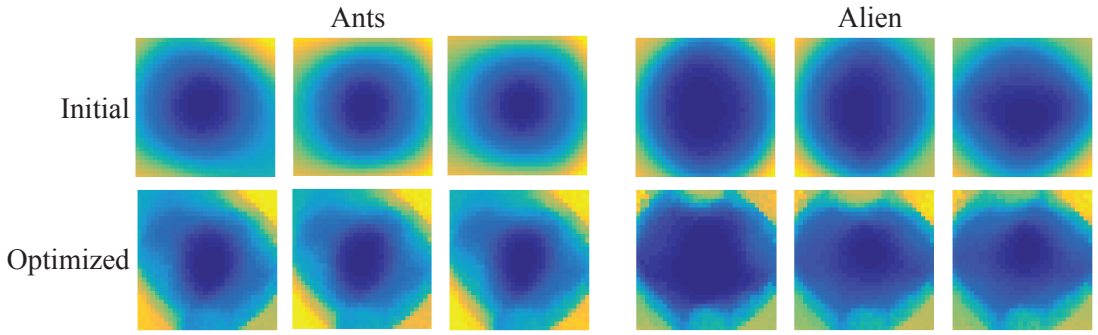


Figure 5.7: Embedding of a single descriptor before (top) and after (bottom) metric learning, for two categories of SHREC'11, 3 meshes from each category.

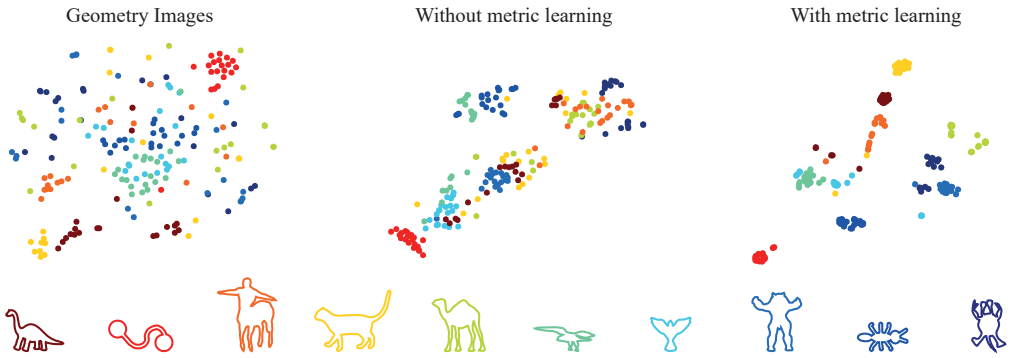


Figure 5.8: tSNE embedding [MH08] of mapped descriptors corresponding to different mapping methods: (left) Geometry images [SBR16], (center) GW mapping to a fixed 2D grid, (right) GW mapping with a learned metric. We show the first 10 classes of SHREC'11 [LGB+11], where each point represents a shape, and shapes from the same class are in the same color. Note that after metric learning our mapped descriptors are nicely clustered, aiding the classification and retrieval tasks we optimized the embedding for.

avoid perfect symmetries that lead to unnecessary ambiguity in the mapping. Since our metric alignment optimization is fairly robust to random perturbations, we introduce a consistent bias in the metric. In particular, we stretch the rows and columns of the grid using the following formula for each point of the grid with initial coordinates (x, y):

$$x_{\text{distorted}} = x\left(1 + (\alpha_x - 1)\, y\right).$$

We apply a similar transformation for y coordinate, and set $\alpha_x = 1.1, \alpha_y = 1.2$ for all the experiments. This formula was chosen arbitrarily to generate a common domain that is similar to a grid and is not symmetric. At each epoch we compute a random permutation of shapes, and take pairs of subsequent examples for training. We use the Siamese network with the smallest training error.

Figure 5.7 illustrates the embedded features before (top) and after (bottom) the metric learning step. Note how the images that correspond to the meshes from the same category are similar to one another after metric learning, and dissimilar across categories. In Figure 5.8 we visualize the proximity between these feature images via a tSNE embedding [MH08]. Each dot corresponds to a model from the SHREC'11 dataset [LGB+11] and the color corresponds to the ground truth class label. We use three methods to map the shape features to an image: the method used by Sinha et al. [SBR16] (Geometry Images) for deep learning (left), GW mapping directly to a 2D grid (middle), and GW mapping based on the learned metric (right). Note that even without learning, the GW mapping to a 2D grid provides more consistency in the embedding, which is then enhanced after metric learning.

After we train the metric alignment layer, we keep the learned parameters D_0 fixed, and train the other parameters of the classification network depicted in Figure 5.6. We

	Classification				Retrieval			
Data \ method	SG	SN	GI	Ours	SG	SN	GI	Ours
SHREC, 10	62.6	52.7	88.6	**90.3**	0.65	0.1	0.65	**0.87**
SHREC, 16	70.8	48.4	**96.6**	**96.6**	0.74	0.13	0.72	**0.96**

Table 5.1: Performance on the SHREC'11 classification benchmark, percentage of correct classification / mAP. We compare with Shape Google [BBGO11] (SG), 3D ShapeNets [WSK+15] (SN) and Geometry Images [SBR16] (GI).

tried fine-tuning our D_0 parameters in the whole network and train it end-to-end, but it significantly slowed down the training step and did not yield any significant improvement in accuracy.

We next evaluate the performance of our network on classification and retrieval tasks on commonly used benchmarks.

5.4 Results

We test our method for classification and retrieval tasks on several existing benchmarks, and also demonstrate the effect of different design choices for the features and the metric.

Classification and retrieval of deformable shapes. We use the SHREC'11 benchmark [LGB+11] to test our method for classification of articulated shapes. The dataset contains 600 shapes containing both rigid (e.g., furniture) and non-rigid (e.g., humans, animals) shapes with significant articulations. We compute the intrinsic shape descriptors: Gaussian curvature (GC), Conformal Factor (CF) and Wave Kernel Signature (WKS). We start by learning the metric using the Siamese architecture described in Section 5.3 for 100 epochs. We set the margin to 50 (approximately the average distance between pairs from different classes). Then we train the classification network depicted in Figure 5.6. Due to the small dataset size we did not use a validation set and stopped training after a fixed number of epochs (200), similarly to Sinha et al. [SBR16]. We used l_1 and l_2 regularization with weights 10^{-4}, 10^{-5} respectively, as well as weight decay 10^{-5}.

We evaluate our classification network by measuring the percentage of correct classifications of shapes in the test set. We also use the output of a penultimate fully connected layer as a global shape descriptor, and evaluate the quality of retrieval using Mean Average Precision (mAP). We use the metrics and the protocol established in Sinha et al. [SBR16] and similarly run on two data splits: 10 training samples from each category (and 10 test shapes), and 16 training samples from each category (4 test shapes).

We compare our results with three other methods: Shape Google [BBGO11] (SG), which uses a bag of features representation, 3D ShapeNets [WSK+15] (SN), which uses a volumetric representation, and Geometry Images [SBR16] (GI), which uses a deep network trained over a flattened shape. Note that GI use substantially larger images in

Metric	Classification	Retrieval
Ours (metric learning)	**90.3**	**0.87**
Distorted grid (no learning)	88.66	0.85
Augmented grid (no learning)	86.66	0.87

Table 5.2: Results of our method with and without metric learning on SHREC'11 with 10 training samples per class.

their representation (32×32 pixels). We present the accuracy and retrieval results in Table 5.1. Our method produces more accurate classifications than state-of-the-art tools using only 10 training examples, and performs comparably to geometry images with 16 training examples. Our retrieval mAP score is consistently higher for all experiments.

Visualization of learned features. After mapping the shapes to the common domain each shape is represented by a multi channel image. The output of the layers of the classification network is therefore challenging to interpret directly. To visualize properties of the classification network we map the output of each convolution layer back to the shape using the GW map, generating a shape descriptor with 320 values for each vertex. Interestingly, the features of the first layer highlight relatively low resolution properties, while the features of the last layer capture finer details, as has been shown for natural images classification. This is visualized in Figure 5.9 using distances in *feature space*. We pick a point p on a bird model from SHREC, and plot the l_2 distance between the features of the first convolution layer of p and all the other points on the shape (top), and similarly for the last convolution layer (bottom) . We also show the distance between the features of the same point p and all the points on two other shapes, one is another shape from the same class and the second is from a different class of a different kind of bird. The features of the first convolution layer are symmetric and similar for points with similar functionality, even for birds from different classes. However, the features of the last convolution layer can distinguish between the two wings of the same bird, and are different for birds of different classes.

Effect of learning the metric of the common domain. We next investigate how important is it to learn the metric of the common domain in comparison to simpler alternatives. We run experiments with 10 training examples on the SHREC'11 dataset. First, we use the initial metric of the grid, computed using distorted Euclidean distances between grid cells (Section 5.3). Another option is to use the undistorted metric of the grid, which will yield some symmetric ambiguities. To resolve these ambiguities, we can augment the data at training time and feed all the elements of the symmetry group to the classification CNN (i.e., rotations and reflections of the 2D square). Both options are presented in Table 5.2 and yield inferior results to our method.

Effect of input features. In Table 5.3 we compare our method when using Euclidean distances for computing the distance matrices for the input shapes, and with different geometric features. Note that since SHREC'11 has many models with severe articulations the use of an extrinsic (Euclidean) distance metric significantly decreases the quality of

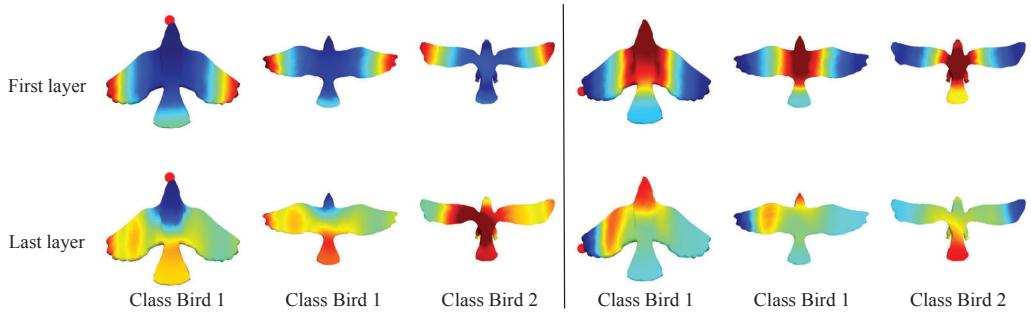


Figure 5.9: Features learned at the first convolution layer (top row) vs. last convolution layer (bottom row) of the classification network for SHREC'11. We visualize the distance between a selected point (tip of the nose or tip of the wing) and other points on the same or different shapes. Note that features from the last convolution layer provide more details and differentiate between birds from different classes.

Features	Classification	Retrieval
GC,CF,WKS, Geodesic	**90.3**	**0.87**
GC,CF,WKS, Euclidean	84.6	0.81
GC, CF, Geodesic	89.3	0.87
GC, Geodesic	89	0.84

Table 5.3: Comparison of our method with different geometric features on SHREC'11 with 10 training samples per class. The features are Gaussian curvature (GC), Conformal factor (CF), Wave Kernel Signature (WKS), Geodesic distances (Geodesic), Euclidean distances (Euclidean).

the results. Our retrieval results are higher than the other methods (Table 5.1) even with very simple features.

Classification and retrieval of rigid shapes. We also evaluate our method on the ModelNet40 and ModelNet10 benchmarks [WSK$^+$15] that contain mostly rigid shapes. For these datasets we use as features the distance histogram, PCA in the local neighborhood of a point and the height, computed using the publicly available code [KCGF14]. Due to the size of this dataset we do the initial metric optimization with the Siamese network only for 10 epochs. We set the margin to 120 (approximately the average distance between pairs from different classes). We do 5-fold cross validation during training and use l_2 regularization with the weight 10^{-3}, as well as weight decay 10^{-4}. The results are presented in Table 5.4. Our method provides competitive retrieval results with respect to GI, but classification suffers from lack of resolution (we use 32×32 images to represent a shape, whereas GI uses images of size 64×64). Our method under-performs on ModelNet40 when compared to multi-view representation methods, e.g. MVCNN [SMKLM15] (classification and retrieval rates of 90.1 and 0.79, respectively), or [QSN$^+$16] (classification rate 91.4). Therefore, it appears that some of our design choices lead to non-optimal performance on rigid shapes, however as our main focus is non-rigid shapes, we leave further investigation of this direction for future work.

Timing. The new metric alignment layer takes about 1s to compute an *initial* matrix Γ

	Classification			Retrieval		
Data \ method	SN	GI	Ours	SN	GI	Ours
ModelNet10	83.5	88.4	85.8	0.68	0.75	0.74
ModelNet40	77	83.9	74.6	0.49	0.51	0.59

Table 5.4: Performance on ModelNet. We compare with 3D ShapeNets [WSK$^+$15] (SN) and Geometry Images [SBR16] (GI).

(50 internal GW iterations) for shapes with approximately 1000 points. After computing the initial maps we only use 5 internal GW iterations and computing Γ takes about 0.2s. Computing the gradient with respect to D_0 takes about 1s. Our approach took 12h to train the Siamese network for SHREC'11 classification (100 epochs), and 6m to train the classification network. Training the Siamese network for ModelNet40 took 25h and 20m to train the classification network.

Limitations. While learning the optimal embedding is beneficial, the number of parameters grows quadratically with the number of discrete elements in the target domain; this affects training complexity. While the GW computation is robust to uniformly distributed noise in the input pairwise distances, topological noise might drastically change pairwise geodesic distances and undermine the quality of near-isometric mappings. Finally, the user's choice of point-wise shape descriptors affects the applicability of the method, and thus requires some prior knowledge about the analyzed dataset. We found our method and design choices to be more suitable for nonrigid shape analysis, as is evident from poor performance of our method on the ModelNet40 dataset.

Chapter 6

Conclusion and Future Work

This thesis describes a few approaches for semi-automatic shape correspondence. Our deblurring and denoising approach (section 2) is based on the prior that the eigenfunctions of the LB operator on the target mesh are mapped to functions in the span of the source eigenfunctions. This assumption was made implicitly when taking the same number of eigenfunctions for isometric shapes, and we have shown that enforcing it explicitly as a prior in the optimization problem yields an efficient deblurring method. We have further demonstrated the use of this idea for map denoising, and used it to generate high quality symmetry maps. The method presented in section 3 can also be used to deblur generalized representations of correspondence, and can get input landmarks as well. Since it optimizes the reversible Dirichlet energy directly, it produces maps with lower conformal distortion than other methods, but its computational cost is higher than the method in section 2. Optimizing the *discrete thin shell energy* rather than the Dirichlet energy, aligns extrinsic features, and leads to highly accurate results (section 4).

These methods produce results with low conformal distortion, that is necessary for downstream applications in computer graphics such as shape interpolation, texture transfer, joint cross field design and joint quadrangulation. However, shape classification and retrieval using deep learning require different properties rather than low conformal distortion. We therefore designed a novel metric alignment layer that is used to learn the correspondence between an input shape and a common domain, such that the classification results are optimal. Our layer works with a range of geometric representations such as point clouds and polygon soups and can capture intrinsic as well as extrinsic geometric structure, as long as it is possible to compute a metric over the input shape. There are many directions to explore in the future using metric alignment. Incorporating automatic feature learning can also potentially improve results and reduce user intervention. Training deep networks for other geometry analysis tasks, such as keypoint detection and segmentation, is also an interesting future direction.

While non isometric shape correspondence methods have significantly improved in recent years, there are still many challenges for future research. State of the art methods are semi automatic, and designing a fully automatic pipeline for non isometric matching

is a challenging task. The task of *volumetric* matching, where the entire volume should be matched (rather than just the surface), is especially relevant to the medical imaging community and was not thoroughly explored from the geometry point of view. Partial matching is another difficult problem, especially in the non isometric case.

To conclude, this thesis describes methods for non isometric shape correspondence and its applications. Accurate shape correspondence is an instrumental component of 3D data analysis and generation methods, such as shape interpolation and deformation transfer. Such methods can be used to synthesize *labeled* 3D data, that is valuable as deep learning methods become more and more common, until finally the data revolution reaches the realm of 3D.

Appendix A

Appendix of Chapter 2

A.1 Equivalence of optimization problems

Proposition 1. The optimization problems (2.1) and (2.4) are equivalent when using the regularizer (2.3).

Proof. Since the constraints of the optimization problems are the same, it suffices to show that the objectives are equal. Our objective from Equation (2.4) has the form

$$\|\Psi_1 X - Y\|_{M_1}^2, \tag{A.1}$$

where $X = C_{12}$ and $Y = P_{12}\Psi_2$. We will show that:

$$\|\Psi_1 X - Y\|_{M_1}^2 = \|X - \Psi_1^\dagger Y\|_F^2 + \|(\Psi_1\Psi_1^\dagger - Id)Y\|_{M_1}^2,$$

which is exactly the objective in Equation (2.1) when plugging in the regularizer from Equation (2.3).

Adding and subtracting $\Psi_1\Psi_1^\dagger Y$ from the expression inside the norm in (A.1) we have:

$$\begin{aligned}&\|\Psi_1 X - \Psi_1\Psi_1^\dagger Y + \Psi_1\Psi_1^\dagger Y - Y\|_{M_1}^2 = \\ &= \|\Psi_1(X - \Psi_1^\dagger Y)\|_{M_1}^2 + \|(\Psi_1\Psi_1^\dagger - Id)Y\|_{M_1}^2 \\ &+ 2\,\mathrm{Tr}((\Psi_1(X - \Psi_1^\dagger Y))^\top A(\Psi_1\Psi_1^\dagger - Id)Y).\end{aligned} \tag{A.2}$$

First, note that for any $X \in \mathbb{R}^{k\times l}$ we have $\|\Psi_1 X\|_{M_1}^2 = \mathrm{Tr}(X^\top\Psi_1^\top A\Psi_1 X) = \mathrm{Tr}(X^\top X) = \|X\|_F^2$, where we used the fact that $\Psi_1^\top A\Psi_1 = Id$. Therefore, the first term in (A.2) is equal to $\|X - \Psi_1^\dagger Y\|_F^2$. For the third term, note that $\Psi_1^\top A(\Psi_1\Psi_1^\dagger - Id) = \Psi_1^\top A\Psi_1\Psi_1^\dagger - \Psi_1^\top A = \Psi_1^\dagger - \Psi_1^\dagger = 0$, where we used the facts that $\Psi_1^\top A\Psi_1 = Id$ and $\Psi_1^\dagger = \Psi_1^\top A$. Together, these give the result.

A.2 Eliminating candidate faces

Given a vertex $v_1 \in \mathcal{V}_1$ and a face $f \in \mathcal{F}_2$ with vertices (c_1, c_2, c_3), we denote $A = \Psi_2, b = (\Psi_1)_{v_1 *} C_{12}$, and:

$$\Delta_{\min} = \min_{v_2 \in \mathcal{V}_2} \|A_{v_2 *} - b\|_2 \quad , \delta_{\min} = \min_{i \in 1 \ldots 3} \|A_{c_i *} - b\|_2$$
$$l_{\max} = \max_{i,j \in 1 \ldots 3} \|A_{c_i *} - A_{c_j *}\|_2 \quad .$$

Assume that the minimizer lies on the face f at the point q, and take $w(q)$ to be its corresponding vector in the valid rows set F_2. By the triangle inequality, we have:

$$\delta_{\min} \leq \|w(q)A - b\|_2 + \|A_{c_i *} - w(q)A\|_2 \leq \|w(q)A - b\|_2 + l_{\max},$$

and therefore: $\delta_{\min} - l_{\max} \leq \|w(q)A - b\|_2$. Since q is a minimizer, we have $\|w(q)A - b\|_2 \leq \Delta_{\min}$, hence f is a face which contains the minimizer only if $\delta_{\min} - l_{\max} \leq \Delta_{\min}$.

Appendix B

Appendix of Chapter 3

Proposition 2. Let $(M_1, g_1), (M_2, g_2)$ be two smooth compact Riemannian surfaces, with geodesic distance functions $d_{M_i}(\cdot,\cdot)$ given by the metrics g_i, respectively, and let $\phi_{12}: M_1 \to M_2$, $\phi_{21}: M_2 \to M_1$ be smooth maps. If there exists $\epsilon \geq 0$ such that:

$$d_{M_1}\left(p_1, \phi_{21}\left(\phi_{12}\left(p_1\right)\right)\right) \leq \epsilon, \quad \forall p_1 \in M_1,$$

then:

1. $\phi_{12}\left(p_1\right) = \phi_{12}\left(q_1\right) \Rightarrow d_{M_1}\left(p_1, q_1\right) \leq 2\epsilon, \quad \forall p_1, q_1 \in M_1.$
2. $\forall p_1 \in M_1 \ \exists p_2 \in M_2 \quad \text{s.t.} \quad d_{M_1}\left(p_1, \phi_{21}\left(p_2\right)\right) \leq \epsilon.$

As a corollary of Proposition 1 we have that if the reversibility energy defined in Equation (3.6) is zero, then the maps ϕ_{12}, ϕ_{21} are both injective and surjective.

Proof. Let $p_1, q_1 \in M_1$, and set $p_2 = \phi_{12}\left(p_1\right), q_2 = \phi_{12}\left(q_1\right)$. Further, set $\hat{p}_1 = \phi_{21}\left(p_2\right)$, and $\hat{q}_1 = \phi_{21}\left(q_2\right)$.

1. From the triangle inequality we have that $d_{M_1}(p_1, q_1) \leq d_{M_1}(p_1, \hat{q}_1) + d_{M_1}(\hat{q}_1, q_1)$. From the assumption (1) we have that $p_2 = q_2$ and therefore $\hat{p}_1 = \hat{q}_1$. Thus, we have $d_{M_1}(p_1, \hat{q}_1) = d_{M_1}(p_1, \hat{p}_1) \leq \epsilon$ and $d_{M_1}(\hat{q}_1, q_1) \leq \epsilon$, which gives the required result.
2. This follows trivially from the assumption of the Proposition if we set $p_2 = \phi_{12}(p_1)$.

Appendix C

Appendix of Chapter 5

Computation of Gromov–Wasserstein (GW) distances alternates between two steps, as specified in Algorithm C.1. The algorithm is adjusted to our purpose by fixing the number of GW iterations and constraining v, w. The first step is a closed-form exponential formula applied independently to every element of the matrix variable. The other is projection onto the cone of doubly-stochastic matrices. We differentiate the result of GW distance computation by providing derivatives of the two steps independently and composing the formulas during alternation. The exponential is differentiable using formulaic techniques; we work out the derivative of doubly-stochastic projection below.

Algorithm C.1 Iteration for finding regularized Gromov-Wasserstein distances. $\otimes, \oslash$ denote elementwise multiplication and division.

function GROMOV-WASSERSTEIN($\mu_0, D_0, \mu, D, \alpha, \eta, \Gamma^0$)
  // Computes a local minimizer Γ *of GW distance*
  for $i = 1, 2, \ldots, I$
    $K^i \leftarrow \exp(D_0[\![\mu_0]\!]\Gamma^{i-1}[\![\mu]\!]D^\top \eta/\alpha) \otimes \left(\Gamma^i\right)^{\wedge(1-\eta)}$
    $\Gamma^i, v^i, w^i \leftarrow$ SINKHORN-PROJECTION($K^i; \mu_0, \mu$)
  return Γ^I,

function SINKHORN-PROJECTION($K; \mu_0, \mu$)
  // Finds Γ *minimizing* $KL(\Gamma|K)$ *subject to* $\Gamma \in \mathcal{M}(\mu_0, \mu)$
  $v, w \leftarrow \mathbf{1}$
  for $j = 1, 2, 3, \ldots$
    $v \leftarrow \mathbf{1} \oslash K(w \otimes \mu)$
    $w \leftarrow \mathbf{1} \oslash K^\top(v \otimes \mu_0)$
  $v \leftarrow v/\sqrt{\frac{\mathbf{1}^\top v}{\mathbf{1}^\top w}}$
  $w \leftarrow w\sqrt{\frac{\mathbf{1}^\top v}{\mathbf{1}^\top w}}$
  return $[\![v]\!]K[\![w]\!], v, w$

C.1 Derivative with respect to K_{ij}

Suppose we wish to rescale a kernel matrix K to be doubly stochastic (Sinkhorn projection step). We can think of this as solving the following quadratic system of equations for Γ and dual vectors v, w (we don't use superscript here for simplicity, we assume all the variables were computed at the same iteration):

$$\Gamma = [\![v]\!] K [\![w]\!] \tag{C.1}$$

$$\Gamma\mu = [\![v]\!] K (w \otimes \mu) = \mathbf{1} \tag{C.2}$$

$$\Gamma^\top \mu_0 = [\![w]\!] K^\top (v \otimes \mu_0) = \mathbf{1} \tag{C.3}$$

$$\mathbf{1}^\top v = \mathbf{1}^\top w \tag{C.4}$$

We differentiate these expressions with respect to an element K_{ij}. Then,

$$\frac{d\Gamma}{dK_{ij}} = [\![\frac{dv}{dK_{ij}}]\!] K [\![w]\!] + v_i w_j (e_i e_j^\top) + [\![v]\!] K [\![\frac{dw}{dK_{ij}}]\!] \tag{C.5}$$

$$0 = \frac{dv}{dK_{ij}} \otimes [K(w \otimes \mu)] + (v_i w_j \mu_j) e_i + [\![v]\!] K [\![\mu]\!] \frac{dw}{dK_{ij}} \tag{C.6}$$

$$0 = \frac{dw}{dK_{ij}} \otimes [K^\top (v \otimes \mu_0)] + (v_i w_j \mu_{0i}) e_j + [\![w]\!] K^\top [\![\mu_0]\!] \frac{dv}{dK_{ij}} \tag{C.7}$$

$$\mathbf{1}^\top \frac{dv}{dK_{ij}} = \mathbf{1}^\top \frac{dw}{dK_{ij}} \tag{C.8}$$

Let's organize the second two relationships as a matrix equation:

$$\begin{pmatrix} [\![K(w \otimes \mu)]\!] & [\![v]\!] K [\![\mu]\!] \\ [\![w]\!] K^\top [\![\mu_0]\!] & [\![K^\top (v \otimes \mu_0)]\!] \\ \mathbf{1}^\top & -\mathbf{1}^\top \end{pmatrix} \begin{pmatrix} {}^{dv}/_{dK_{ij}} \\ {}^{dw}/_{dK_{ij}} \end{pmatrix} = \begin{pmatrix} -v_i w_j \mu_j e_i \\ -v_i w_j \mu_{0i} e_j \\ 0 \end{pmatrix} \tag{C.9}$$

We can simplify this by leveraging the fact that the diagonal elements of this block 2×2 matrix appear in the Sinkhorn conditions. In particular, $K(w \otimes \mu) = \mathbf{1} \oslash v$ and $K^\top (v \otimes \mu_0) = \mathbf{1} \oslash w$. Hence, we can write the expression as:

$$\begin{pmatrix} [\![v]\!]^{-1} & [\![v]\!] K [\![\mu]\!] \\ [\![w]\!] K^\top [\![\mu_0]\!] & [\![w]\!]^{-1} \\ \mathbf{1}^\top & -\mathbf{1}^\top \end{pmatrix} \begin{pmatrix} {}^{dv}/_{dK_{ij}} \\ {}^{dw}/_{dK_{ij}} \end{pmatrix} = \begin{pmatrix} -v_i w_j \mu_j e_i \\ -v_i w_j \mu_{0i} e_j \\ 0 \end{pmatrix} \tag{C.10}$$

We factor to make it look symmetric:

$$\begin{pmatrix} [\![v]\!] & 0 & 0 \\ 0 & [\![w]\!] & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} [\![v \otimes v \otimes \mu_0]\!]^{-1} & K \\ K^\top & [\![w \otimes w \otimes \mu]\!]^{-1} \\ (\mathbf{1} \oslash \mu 0)^\top & -(\mathbf{1} \oslash \mu)^\top \end{pmatrix} \begin{pmatrix} [\![\mu_0]\!] & 0 \\ 0 & [\![\mu]\!] \end{pmatrix} \begin{pmatrix} {}^{dv}/_{dK_{ij}} \\ {}^{dw}/_{dK_{ij}} \end{pmatrix} = \begin{pmatrix} -v_i w_j \mu_j e_i \\ -v_i w_j \mu_{0i} e_j \\ 0 \end{pmatrix} \tag{C.11}$$

The two vectors on the right-hand side are sparse except element i in the first half and element j in the second half. So, we can invert the first matrix and put the diagonal

matrix into the unknowns:

$$\begin{pmatrix} [\![v \otimes v \otimes \mu_0]\!]^{-1} & K \\ K^\top & [\![w \otimes w \otimes \mu]\!]^{-1} \\ (\mathbf{1} \oslash \mu 0)^\top & -(\mathbf{1} \oslash \mu)^\top \end{pmatrix} \begin{pmatrix} [\![\mu_0]\!]^{dv/dK_{ij}} \\ [\![\mu]\!]^{dw/dK_{ij}} \end{pmatrix} = \begin{pmatrix} -w_j \mu_j e_i \\ -v_i \mu_{0i} e_j \end{pmatrix} \tag{C.12}$$

Denote the left inverse of the left matrix as

$$\begin{pmatrix} [\![v \otimes v \otimes \mu_0]\!]^{-1} & K \\ K^\top & [\![w \otimes w \otimes \mu]\!]^{-1} \\ (\mathbf{1} \oslash \mu 0)^\top & -(\mathbf{1} \oslash \mu)^\top \end{pmatrix}^+ := \begin{pmatrix} A & B \\ C & E \end{pmatrix} \tag{C.13}$$

Then,

$$\begin{pmatrix} {}^{dv}/_{dK_{ij}} \\ {}^{dw}/_{dK_{ij}} \end{pmatrix} = \begin{pmatrix} [\![\mathbf{1} \oslash \mu_0]\!] & 0 \\ 0 & [\![\mathbf{1} \oslash \mu]\!] \end{pmatrix} \begin{pmatrix} A & B \\ C & E \end{pmatrix} \begin{pmatrix} -w_j \mu_j e_i \\ -v_i \mu_{0i} e_j \end{pmatrix} \tag{C.14}$$

$$= \begin{pmatrix} [\![\mathbf{1} \oslash \mu_0]\!] & 0 \\ 0 & [\![\mathbf{1} \oslash \mu]\!] \end{pmatrix} \begin{pmatrix} -w_j \mu_j A_{\text{column } i} - v_i \mu_{0i} B_{\text{column } j} \\ -w_j \mu_j C_{\text{column } i} - v_i \mu_{0i} E_{\text{column } j} \end{pmatrix} \tag{C.15}$$

C.2 Gradient with respect to K

Returning to our original derivative, we extract a single element

$$\frac{d\Gamma_{k\ell}}{dK_{ij}} = \frac{dv_k}{dK_{ij}} K_{k\ell} w_\ell + v_i w_j \delta_{ik} \delta_{j\ell} + v_k K_{k\ell} \frac{dw_\ell}{dK_{ij}} \tag{C.16}$$

In the end, we know $\frac{dL}{d\Gamma_{k\ell}}$ for some function L and want $\frac{dL}{dK_{ij}}$. Write the gradient of L with respect to Γ as $\nabla_\Gamma L$. We start computing

$$\begin{aligned} \frac{dL}{dK_{ij}} &= \sum_{k\ell} (\nabla_\Gamma L)_{k\ell} \frac{d\Gamma_{k\ell}}{dK_{ij}} \\ &= \sum_{k\ell} (\nabla_\Gamma L)_{k\ell} \left[\frac{dv_k}{dK_{ij}} K_{k\ell} w_\ell + v_i w_j \delta_{ik} \delta_{j\ell} + v_k K_{k\ell} \frac{dw_\ell}{dK_{ij}} \right] \end{aligned}$$

Breaking this down term by term,

$$\begin{aligned} \sum_{k\ell} (\nabla_\Gamma L)_{k\ell} \frac{dv_k}{dK_{ij}} K_{k\ell} w_\ell &= \left(\frac{dv}{dK_{ij}} \right)^\top (K \otimes \nabla_\Gamma L) w \\ &= (-w_j \mu_j A_{\text{column } i} - v_i \mu_{0i} B_{\text{column } j})^\top [\![\mathbf{1} \oslash \mu_0]\!] (K \otimes \nabla_\Gamma L) w \\ \sum_{k\ell} (\nabla_\Gamma L)_{k\ell} v_i w_j \delta_{ik} \delta_{j\ell} &= (\nabla_\Gamma L)_{ij} v_i w_j \\ \sum_{k\ell} (\nabla_\Gamma L)_{k\ell} v_k K_{k\ell} \frac{dw_\ell}{dK_{ij}} &= v^\top (K \otimes \nabla_\Gamma L) \frac{dw}{K_{ij}} \\ &= v^\top (K \otimes \nabla_\Gamma L) [\![\mathbf{1} \oslash \mu]\!] (-w_j \mu_j C_{\text{column } i} - v_i \mu_{0i} E_{\text{column } j}) \end{aligned}$$

Getting rid of the ij index (and applying symmetry of A and C) shows

$$\boxed{\begin{aligned}\nabla_K L = \; & -A[\![\mathbf{1} \oslash \mu_0]\!](K \otimes \nabla_\Gamma L) w (w \otimes \mu)^\top - (v \otimes \mu_0)[(C[\![\mathbf{1} \oslash \mu_0]\!](K \otimes \nabla_\Gamma L) w]^\top \\ & + [\![v]\!] \nabla_\Gamma L [\![w]\!] \\ & - B[\![\mathbf{1} \oslash \mu]\!](K \otimes \nabla_\Gamma L)^\top v (w \otimes \mu)^\top - (v \otimes \mu_0)[E[\![\mathbf{1} \oslash \mu]\!](K \otimes \nabla_\Gamma L)^\top v]^\top \end{aligned}} \tag{C.17}$$

C.3 Exponential formula

Given $\nabla_{K^i} L$ we would like to compute $\nabla_{\Gamma^{i-1}} L$, the derivatives with respect to Γ from the previous iteration. In the rest of this section K will be used for K^i and Γ for Γ^{i-1}.

$$K = exp\left(D_0[\![\mu_0]\!]\Gamma[\![\mu]\!]D^\top \cdot \eta/\alpha\right) \otimes \Gamma^{\wedge 1-\eta} \tag{C.18}$$

$$\frac{dL}{d\Gamma_{ij}} = \sum_{kl} \frac{dL}{dK_{kl}} \frac{dK_{kl}}{d\Gamma_{ij}} \tag{C.19}$$

$$\begin{aligned}\frac{dK_{kl}}{d\Gamma_{ij}} = \; & \delta_{i==k}\delta_{j==l}\,(1-\eta)\,(\Gamma)_{kl}^{-\eta} \exp\left(D_0[\![\mu_0]\!]\Gamma[\![\mu]\!]D^\top \cdot \eta/\alpha\right)_{kl} + \\ & \exp\left(D_0[\![\mu_0]\!]\Gamma[\![\mu]\!]D^\top \cdot \eta/\alpha\right)_{kl} \frac{\eta}{\alpha} \left[D_0[\![\mu_0]\!]\right]_{ki} \left[[\![\mu]\!]D^\top\right]_{jl} \Gamma^{\wedge 1-\eta}\end{aligned} \tag{C.20}$$

$$\begin{aligned}\nabla_\Gamma L = \; & (1-\eta)\,\nabla_K L \otimes (\Gamma)^{\wedge -\eta} \otimes \exp\left(D_0[\![\mu_0]\!]\Gamma[\![\mu]\!]D^\top \cdot \eta/\alpha\right) + \\ & \frac{\eta}{\alpha} \nabla_K L \otimes \left([\![\mu_0]\!] D_0^\top \left(\Gamma^{\wedge 1-\eta} \otimes \exp\left(D_0[\![\mu_0]\!]\Gamma[\![\mu]\!]D^\top \cdot \eta/\alpha\right)\right) D[\![\mu]\!]\right)\end{aligned} \tag{C.21}$$

C.4 Gradient with respect to D

Given $\nabla_{K^i} L$, we can compute $\nabla_D L$ by cumulating the following gradients throughout the iterations:

$$\frac{dK_{kl}}{dD_{ij}} = \Gamma_{kl}^{1-\eta} exp\left(D_0[\![\mu_0]\!]\Gamma[\![\mu]\!]D^\top \cdot \eta/\alpha\right)_{kl} \delta_{i==l} \frac{\eta}{\alpha} \left[D_0[\![\mu_0]\!]\Gamma[\![\mu]\!]\right]_{kj} \tag{C.22}$$

$$\frac{dL}{dD_{ij}} = \sum_k \frac{dL}{dK_{ki}} \frac{\eta}{\alpha} \Gamma_{ki}^{1-\eta} exp\left(D_0[\![\mu_0]\!]\Gamma[\![\mu]\!]D^\top \cdot \eta/\alpha\right)_{ki} \left[D_0[\![\mu_0]\!]\Gamma[\![\mu]\!]\right]_{kj} \tag{C.23}$$

$$\nabla_D L = \frac{\eta}{\alpha} \left(\nabla_K L \otimes \Gamma^{\wedge 1-\eta} \otimes exp\left(D_0[\![\mu_0]\!]\Gamma[\![\mu]\!]D^\top \cdot \eta/\alpha\right)\right)^\top D_0[\![\mu_0]\!]\Gamma[\![\mu]\!] \tag{C.24}$$

We also add the transposed matrix of derivatives to enforce symmetry.

Bibliography

[ACBCO17] Omri Azencot, Etienne Corman, Mirela Ben-Chen, and Maks Ovsjanikov. Consistent functional cross field design for mesh quadrangulation. *ACM Trans. Graph.*, 36(4):92:1–92:13, 2017.

[AKL17] Noam Aigerman, Shahar Z Kovalsky, and Yaron Lipman. Spherical orbifold Tutte embeddings. *ACM Transactions on Graphics (TOG)*, 36(4):90:1–90:13, 2017.

[AL15] Noam Aigerman and Yaron Lipman. Orbifold Tutte embeddings. *ACM Transactions on Graphics (TOG)*, 34, 2015.

[AL16] Noam Aigerman and Yaron Lipman. Hyperbolic orbifold tutte embeddings. *ACM Trans. Graph.*, 35(6):217–1, 2016.

[AML18] Matan Atzmon, Haggai Maron, and Yaron Lipman. Point convolutional neural networks by extension operators. *arXiv preprint arXiv:1803.10091*, 2018.

[APL14] Noam Aigerman, Roi Poranne, and Yaron Lipman. Lifted Bijections for Low Distortion Surface Mappings. *ACM Transactions on Graphics (TOG)*, 33, 2014.

[APL15] Noam Aigerman, Roi Poranne, and Yaron Lipman. Seamless surface mappings. *ACM Transactions on Graphics (TOG)*, 34(4):72, 2015.

[ASC11] Mathieu Aubry, Ulrich Schlickewei, and Daniel Cremers. The wave kernel signature: A quantum mechanical approach to shape analysis. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 1626–1633. IEEE, 2011.

[AVBC16] Omri Azencot, Orestis Vantzos, and Mirela Ben-Chen. Advection-based function matching on surfaces. In *Computer Graphics Forum*, volume 35, pages 55–64. Wiley Online Library, 2016.

[AXZ+15] Ibraheem Alhashim, Kai Xu, Yixin Zhuang, Junjie Cao, Patricio Simari, and Hao Zhang. Deformation-driven topology-varying 3D

shape correspondence. *ACM Trans. Graph.*, 34(6):236:1–236:13, 2015.

[BBB+93] Jane Bromley, James W. Bentz, L'eon Bottou, Isabelle Guyon, Yann LeCun, Cliff Moore, Eduard S"ackinger, and Roopak Shah. Signature Verification Using a "Siamese" Time Delay Neural Network. *IJPRAI*, 7(4):669–688, 1993.

[BBB+10] Alexander M Bronstein, Michael M Bronstein, Benjamin Bustos, Umberto Castellani, Marco Crisani, Bianca Falcidieno, Leonidas J Guibas, Iasonas Kokkinos, Vittorio Murino, Ivan Sipiran, Maks Ovsjanikov, Giuseppe Patane, Michuela Spagnuolo, and Jian Sun. Shrec 2010: robust feature detection and description benchmark. *Proc. 3DOR*, 2(5):6, 2010.

[BBGO11] Alexander M. Bronstein, Michael M. Bronstein, Leonidas J. Guibas, and Maks Ovsjanikov. Shape Google: Geometric Words and Expressions for Invariant Shape Retrieval. *TOG*, 30(1), February 2011.

[BBK06] Alexander M Bronstein, Michael M Bronstein, and Ron Kimmel. Generalized multidimensional scaling: a framework for isometry-invariant partial surface matching. *Proceedings of the National Academy of Sciences*, 103(5):1168–1172, 2006.

[BCG08] Mirela Ben-Chen and Craig Gotsman. Characterizing Shape Using Conformal Factors. In *3DOR*, pages 1–8, 2008.

[BDK17] Oliver Burghard, Alexander Dieckmann, and Reinhard Klein. Embedding shapes with green's functions for global shape matching. *Computers & Graphics*, 68:1–10, 2017.

[BKP+10] M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, and B. Levy. *Polygon Mesh Processing*. Ak Peters Series. Taylor & Francis, 2010.

[BLRW16] Andrew Brock, Theodore Lim, JM Ritchie, and Nick Weston. Generative and Discriminative Voxel Modeling with Convolutional Neural Networks. *arXiv preprint arXiv:1608.04236*, 2016.

[BMF03] Robert Bridson, Sebastian Marino, and Ronald Fedkiw. Simulation of clothing with folds and wrinkles. In *Proc. ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 28–36, 2003.

[BMRB16] Davide Boscaini, Jonathan Masci, Emanuele Rodolà, and Michael Bronstein. Learning shape correspondence with anisotropic convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 3189–3197, 2016.

[BPC16] Nicolas Bonneel, Gabriel Peyr'e, and Marco Cuturi. Wasserstein Barycentric Coordinates: Histogram Regression Using Optimal Transport. *TOG*, 35(4), 2016.

[BPGK06] Mario Botsch, Mark Pauly, Markus H Gross, and Leif Kobbelt. PriMo: coupled prisms for intuitive surface modeling. In *Proc. Eurographics Symposium on Geometry Processing*, pages 11–20, 2006.

[BRLB14] Federica Bogo, Javier Romero, Matthew Loper, and Michael J Black. FAUST: Dataset and evaluation for 3d mesh registration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3794–3801, 2014.

[BS08] Mario Botsch and Olga Sorkine. On linear variational surface deformation methods. *IEEE Trans. Vis. Comput. Graph.*, 14(1):213–230, 2008.

[BZSL14] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral Networks and Locally Connected Networks on Graphs. In *Proc. ICLR*, 2014.

[CC00] Trevor F Cox and Michael AA Cox. *Multidimensional scaling*. CRC Press, 2000.

[CGF09] Xiaobai Chen, Aleksey Golovinskiy, and Thomas Funkhouser. A benchmark for 3D mesh segmentation. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 28(3):73:1–73:12, August 2009.

[Cia00] Philippe G. Ciarlet. *Mathematical elasticity. Vol. III*, volume 29 of *Studies in Mathematics and its Applications*. North-Holland Publishing Co., Amsterdam, 2000. Theory of shells.

[CKF11] Ronan Collobert, Koray Kavukcuoglu, and Cl'ement Farabet. Torch7: A Matlab-like Environment for Machine Learning. In *BigLearn, NIPS Workshop*, number EPFL-CONF-192376, 2011.

[COC14] Étienne Corman, Maks Ovsjanikov, and Antonin Chambolle. Supervised descriptor learning for non-rigid shape matching. In *Computer Vision-ECCV 2014 Workshops*, pages 283–298, 2014.

[COC15] Etienne Corman, Maks Ovsjanikov, and Antonin Chambolle. Continuous matching via vector field flow. In *Computer Graphics Forum*, 2015.

[CPSS10] Isaac Chao, Ulrich Pinkall, Patrick Sanan, and Peter Schröder. A simple geometric model for elastic deformations. *ACM Transactions on Graphics (TOG)*, 29(4):38:1–38:6, 2010.

[CSPF12] Xiaobai Chen, Abulhair Saparov, Bill Pang, and Thomas Funkhouser. Schelling points on 3d surface meshes. *ACM Transactions on Graphics (TOG)*, 31(4):29, 2012.

[CWV+14] Sharan Chetlur, Cliff Woolley, Philippe Vandermersch, Jonathan Cohen, John Tran, Bryan Catanzaro, and Evan Shelhamer. Cudnn: Efficient Primitives for Deep Learning. *arXiv preprint arXiv:1410.0759*, 2014.

[dBDFN16] Maya de Buhan, Charles Dapogny, Pascal Frey, and Chiara Nardoni. An optimization method for elastic shape matching. *C. R. Math. Acad. Sci. Paris*, 354(8):783–787, 2016.

[DBV16] Micha"el Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In *NIPS*, pages 3837–3845, 2016.

[DML17] Nadav Dym, Haggai Maron, and Yaron Lipman. Ds++: A flexible, scalable and provably tight relaxation for matching problems. *ACM Transactions on Graphics (TOG)*, 36(6), 2017.

[EBC17] Danielle Ezuz and Mirela Ben-Chen. Deblurring and denoising of maps between shapes. In *Computer Graphics Forum*, volume 36, pages 165–174, 2017.

[Ebe] David Eberly. Distance between point and triangle in 3d.

[EHA+19] Danielle Ezuz, Behrend Heeren, Omri Azencot, Martin Rumpf, and Mirela Ben-Chen. Elastic correspondence between triangle meshes. In *Computer Graphics Forum*, volume 38, 2019.

[EM94] H. Edelsbrunner and E. P. Mücke. Three-dimensional Alpha Shapes. *TOG*, 1994.

[ERGB16] Davide Eynard, Emanuele Rodola, Klaus Glashoff, and Michael M Bronstein. Coupled functional maps. In *3D Vision (3DV), 2016 Fourth International Conference on*, pages 399–407. IEEE, 2016.

[ES64] James Eells and Joseph H Sampson. Harmonic mappings of Riemannian manifolds. *American Journal of Mathematics*, 86(1):109–160, 1964.

[ESBC19] Danielle Ezuz, Justin Solomon, and Mirela Ben-Chen. Reversible harmonic maps between discrete surfaces. *ACM Trans. Graph.*, 38(2):15:1–15:12, March 2019.

[ESKBC17] Danielle Ezuz, Justin Solomon, Vladimir G Kim, and Mirela Ben-Chen. Gwcnn: A metric alignment layer for deep shape analysis. In *Computer Graphics Forum*, volume 36, pages 49–57, 2017.

[FB11] Stefan Fröhlich and Mario Botsch. Example-driven deformations based on discrete shells. *Computer Graphics Forum*, 30(8):2246–2257, 2011.

[GBKS18] Anne Gehre, M Bronstein, Leif Kobbelt, and Justin Solomon. Interactive curve constrained functional maps. In *Computer Graphics Forum*, volume 37, pages 1–12, 2018.

[GBP07] Daniela Giorgi, Silvia Biasotti, and Laura Paraboschi. SHREC: Shape retrieval contest: Watertight models track, 2007.

[GGH02] Xianfeng Gu, Steven Gortler, and Hugues Hoppe. Geometry Images. In *SIGGRAPH*, 2002.

[GGWZ07] Akash Garg, Eitan Grinspun, Max Wardetzky, and Denis Zorin. Cubic shells. In *Proc. ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 91–98, 2007.

[GHDS03] Eitan Grinspun, Anil N Hirani, Mathieu Desbrun, and Peter Schröder. Discrete shells. In *Proc. ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 62–67, 2003.

[GWC+04] Xianfeng Gu, Yalin Wang, Tony F Chan, Paul M Thompson, and Shing-Tung Yau. Genus zero surface conformal mapping and its application to brain surface mapping. *Trans. on Medical Imaging*, 23(8):949–958, 2004.

[GY95] Donald Geman and Chengda Yang. Nonlinear image recovery with half-quadratic regularization. *IEEE Transactions on Image Processing*, 4(7):932–946, 1995.

[GZC15] Kan Guo, Dongqing Zou, and Xiaowu Chen. 3D Mesh Labeling Via Deep Convolutional Neural Networks. *TOG*, 35(1), 2015.

[HBL15] Mikael Henaff, Joan Bruna, and Yann LeCun. Deep Convolutional Networks on Graph-structured Data. *arXiv preprint arXiv:1506.05163*, 2015.

[HG00] Kai Hormann and Günther Greiner. Mips: An efficient global parametrization method. Technical report, DTIC Document, 2000.

[HKC+17] Haibin Huang, Evangelos Kalogerakis, Siddhartha Chaudhuri, Duygu Ceylan, Vladimir G. Kim, and Ersin Yumer. Learning local shape descriptors from part correspondences with multiview convolutional networks. *ACM Trans. Graph.*, 37(1):6:1–6:14, 2017.

[HLR+18] Oshri Halimi, Or Litany, Emanuele Rodolà, Alex Bronstein, and Ron Kimmel. Self-supervised learning of dense shape correspondence. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[HLS08] Kai Hormann, Bruno L'evy, and Alla Sheffer. Mesh Parameterization: Theory and Practice. *SIGGRAPH Asia, Course Notes*, 2008.

[HO17] Ruqi Huang and Maks Ovsjanikov. Adjoint map representation for shape analysis and matching. In *Computer Graphics Forum*, volume 36, pages 151–163, 2017.

[HRS+14] Behrend Heeren, Martin Rumpf, Peter Schröder, Max Wardetzky, and Benedikt Wirth. Exploring the geometry of the space of shells. *Computer Graphics Forum*, 33(5):247–256, 2014.

[HRWW12] Behrend Heeren, Martin Rumpf, Max Wardetzky, and Benedikt Wirth. Time-discrete geodesics in the space of shells. *Computer Graphics Forum*, 31, 2012.

[HSTP11] Klaus Hildebrandt, Christian Schulz, Christoph von Tycowicz, and Konrad Polthier. Interactive surface modeling using modal analysis. *ACM Trans. Graph.*, 30(5):119:1–11, 2011.

[HWG14] Qixing Huang, Fan Wang, and Leonidas Guibas. Functional map networks for analyzing and exploring large shape collections. *ACM Transactions on Graphics (TOG)*, 33(4):36, 2014.

[IN05] Hiroyasu Izeki and Shin Nayatani. Combinatorial harmonic maps and discrete-group actions on Hadamard spaces. *Geometriae Dedicata*, 114(1):147–188, 2005.

[IRS18] José A. Iglesias, Martin Rumpf, and Otmar Scherzer. Shape-aware matching of implicit surfaces based on thin shell energies. *Found. Comput. Math.*, 18(4):891–927, 2018.

[KAMC17] Evangelos Kalogerakis, Melinos Averkiou, Subhransu Maji, and Siddhartha Chaudhuri. 3D Shape Segmentation with Projective Convolutional Networks. *Proc. CVPR, IEEE*, 2017.

[KBB+13] Artiom Kovnatsky, Michael M Bronstein, Alexander M Bronstein, Klaus Glashoff, and Ron Kimmel. Coupled quasi-harmonic bases. *Computer Graphics Forum*, 32:439–448, 2013.

[KBBV15] Artiom Kovnatsky, Michael M Bronstein, Xavier Bresson, and Pierre Vandergheynst. Functional correspondence by matrix completion. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, 2015.

[KCGF14] Vladimir G. Kim, Siddhartha Chaudhuri, Leonidas Guibas, and Thomas Funkhouser. Shape2Pose: Human-Centric Shape Analysis. *TOG*, 33(4), 2014.

[KGB16] Artiom Kovnatsky, Klaus Glashoff, and Michael M Bronstein. Madmm: a generic algorithm for non-smooth optimization on manifolds. In *European Conference on Computer Vision*, pages 680–696, 2016.

[KHS10a] Evangelos Kalogerakis, Aaron Hertzmann, and Karan Singh. Learning 3D mesh segmentation and labeling. *ACM Transactions on Graphics*, 29(3):102:1–102:12, 2010.

[KHS10b] Evangelos Kalogerakis, Aaron Hertzmann, and Karan Singh. Learning 3D Mesh Segmentation and Labeling. *TOG*, 29(3), 2010.

[KKBL15] Itay Kezurer, Shahar Z Kovalsky, Ronen Basri, and Yaron Lipman. Tight relaxation of quadratic matching. In *Computer Graphics Forum*, volume 34, pages 115–128, 2015.

[KLF11] Vladimir G Kim, Yaron Lipman, and Thomas Funkhouser. Blended Intrinsic Maps. *ACM Transactions on Graphics (TOG)*, 30, 2011.

[KLM+12] Vladimir G Kim, Wilmot Li, Niloy J Mitra, Stephen DiVerdi, and Thomas Funkhouser. Exploring collections of 3d models using fuzzy correspondences. *ACM Transactions on Graphics (TOG)*, 31(4):54, 2012.

[KSNS07] Evangelos Kalogerakis, Patricio Simari, Derek Nowrouzezahrai, and Karan Singh. Robust statistical estimation of curvature on discretized surfaces. In *Proc. Eurographics Symposium on Geometry Processing*, 2007.

[KW17] Thomas N Kipf and Max Welling. Semi-supervised Classification with Graph Convolutional Networks. In *Proc. ICLR*, 2017.

[LB14] Roee Litman and Alexander Bronstein. Learning spectral descriptors for deformable shape correspondence. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 36(1):171–180, 2014.

[LDRS05] Nathan Litke, Mark Droske, Martin Rumpf, and Peter Schröder. An image processing approach to surface matching. In *Proc. Eurographics Symposium on Geometry Processing*, pages 207–216, 2005.

[LGB+11] Z Lian, A Godil, B Bustos, M Daoudi, Jeroen Hermans, S Kawamura, Y Kurita, G Lavou'e, HV Nguyen, R Ohbuchi, et al. SHREC'11 Track: Shape Retrieval on Non-rigid 3D Watertight Meshes. In *3DOR*, pages 79–88, 2011.

[LPRM02a] B. Levy, S. Petitjean, N. Ray, and J. Maillot. Least Squares Conformal Maps. *SIGGRAPH*, 2002.

[LPRM02b] Bruno Lévy, Sylvain Petitjean, Nicolas Ray, and Jérome Maillot. Least squares conformal maps for automatic texture atlas generation. In *ACM Transactions on Graphics (TOG)*, volume 21, pages 362–371. ACM, 2002.

[LRB+16] O. Litany, E. Rodola, A. M. Bronstein, M. M. Bronstein, and D. Cremers. Non-rigid puzzles. *Computer Graphics Forum*, 35(5), 2016.

[LRBB17] Or Litany, Emanuele Rodol'a, Alex M Bronstein, and Michael M Bronstein. Fully spectral partial shape matching. In *Computer Graphics Forum*, volume 36, pages 247–258, 2017.

[LRR+17] Or Litany, Tal Remez, Emanuele Rodolà, Alex Bronstein, and Michael Bronstein. Deep functional maps: Structured prediction for dense shape correspondence. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[LSP08] Hao Li, Robert W Sumner, and Mark Pauly. Global correspondence optimization for non-rigid registration of depth scans. *Comput. Graph. Forum*, 27(5):1421–1430, 2008.

[LVB+17] Zorah Lähner, Matthias Vestner, Amit Boyarski, Or Litany, Ron Slossberg, Tal Remez, Emanuele Rodolà, Alexander M. Bronstein, Michael M. Bronstein, Ron Kimmel, and Daniel Cremers. Efficient deformable shape correspondence via kernel matching. In *3D Vision (3DV)*, 2017.

[MBBV15] Jonathan Masci, Davide Boscaini, Michael Bronstein, and Pierre Vandergheynst. Geodesic convolutional neural networks on riemannian manifolds. In *The IEEE International Conference on Computer Vision (ICCV)*, pages 37–45, 2015.

[MBM+17] Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodola, Jan Svoboda, and Michael M Bronstein. Geometric deep learning on graphs and manifolds using mixture model cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5115–5124, 2017.

[MCSK+17] Manish Mandad, David Cohen-Steiner, Leif Kobbelt, Pierre Alliez, and Mathieu Desbrun. Variance-minimizing transport plans for inter-surface mapping. *ACM Transactions on Graphics*, 36:14, 2017.

[MDK+16] Haggai Maron, Nadav Dym, Itay Kezurer, Shahar Kovalsky, and Yaron Lipman. Point registration via efficient convex relaxation. *ACM Transactions on Graphics (TOG)*, 35(4), 2016.

[MDW08] Brent C Munsell, Pahal Dalal, and Song Wang. Evaluating shape correspondence for statistical shape analysis: A benchmark study. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):2023–2039, 2008.

[MGA+17] Haggai Maron, Meirav Galun, Noam Aigerman, Miri Trope, Nadav Dym, Ersin Yumer, Vladimir G Kim, and Yaron Lipman. Convolutional neural networks on surfaces via seamless toric covers. *ACM Trans. Graph.*, 36(4):71–1, 2017.

[MH08] Laurens van der Maaten and Geoffrey Hinton. Visualizing Data Using t-SNE. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[MS15] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d Convolutional Neural Network for Real-time Object Recognition. In *IROS*, pages 922–928. IEEE, 2015.

[Nis00] Seiki Nishikawa. *Variational Problems in Geometry*. Iwanami Series in Modern Mathematics. AMS, 2000.

[NO17] Dorian Nogneng and Maks Ovsjanikov. Informative descriptor preservation via commutativity for shape matching. In *Computer Graphics Forum*, volume 36, pages 259–267, 2017.

[NSZ+17] Saad Nadeem, Zhengyu Su, Wei Zeng, Arie Kaufman, and Xianfeng Gu. Spherical parameterization balancing angle and area distortions. *IEEE Trans. Vis. Comput. Graph.*, 23(6):1663–1676, 2017.

[NW99] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer-Verlag, New York, 1999.

[OBCCG13] Maks Ovsjanikov, Mirela Ben-Chen, Frederic Chazal, and Leonidas Guibas. Analysis and visualization of maps between shapes. In *Computer Graphics Forum*, volume 32, pages 135–145, 2013.

[OBCS+12] Maks Ovsjanikov, Mirela Ben-Chen, Justin Solomon, Adrian Butscher, and Leonidas Guibas. Functional Maps: a Flexible Representation of Maps between Shapes. *ACM Transactions on Graphics (TOG)*, 31, 2012.

[OCB+16] Maks Ovsjanikov, Etienne Corman, Michael Bronstein, Emanuele Rodolà, Mirela Ben-Chen, Leonidas Guibas, Frederic Chazal, and Alex Bronstein. Computing and processing correspondences with functional maps. In *SIGGRAPH ASIA 2016 Courses*, page 9. ACM, 2016.

[PBB+13] Jonathan Pokrass, Alexander M Bronstein, Michael M Bronstein, Pablo Sprechmann, and Guillermo Sapiro. Sparse modeling of intrinsic correspondences. In *Computer Graphics Forum*, volume 32, pages 459–468, 2013.

[PBDSH13] Daniele Panozzo, Ilya Baran, Olga Diamanti, and Olga Sorkine-Hornung. Weighted averages on surfaces. *ACM Transactions on Graphics (TOG)*, 32(4):60, 2013.

[PCS16] Gabriel Peyr'e, Marco Cuturi, and Justin Solomon. Gromov–Wasserstein Averaging of Kernel and Distance Matrices. In *ICML*, 2016.

[PLPZ12] Daniele Panozzo, Yaron Lipman, Enrico Puppo, and Denis Zorin. Fields on Symmetric Surfaces. *ACM Transactions on Graphics (TOG)*, 31, 2012.

[PP93] Ulrich Pinkall and Konrad Polthier. Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics*, 2(1):15–36, 1993.

[PSS01] Emil Praun, Wim Sweldens, and Peter Schroder. Consistent Mesh Parameterizations. *SIGGRAPH*, 2001.

[QSMG17] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. *Proc. CVPR, IEEE*, 2017.

[QSN+16] Charles Ruizhongtai Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas Guibas. Volumetric and Multi-View CNNs for Object Classification on 3D Data. In *Proc. CVPR, IEEE*, 2016.

[QYSG17] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, pages 5099–5108, 2017.

[RCB+16] Emanuele Rodolà, Luca Cosmo, Michael M Bronstein, Andrea Torsello, and Daniel Cremers. Partial Functional Correspondence. *Computer Graphics Forum*, 2016.

[RMC15] Emanuele Rodolá, Michael Moeller, and Daniel Cremers. Point-wise Map Recovery and Refinement from Functional Correspondence. In David Bommes, Tobias Ritschel, and Thomas Schultz, editors, *Vision, Modeling and Visualization*, 2015.

[RMC17] Emanuele Rodolà, Michael Möller, and Daniel Cremers. Regularized pointwise map recovery from functional correspondence. In *Computer Graphics Forum*, volume 36, pages 700–711. Wiley Online Library, 2017.

[RO18] Jean-Michel Roufosse and Maks Ovsjanikov. Unsupervised deep learning for structured shape matching. *arXiv preprint arXiv:1812.03794*, 2018.

[ROA+13] Raif M Rustamov, Maks Ovsjanikov, Omri Azencot, Mirela Ben-Chen, Frédéric Chazal, and Leonidas Guibas. Map-based exploration of intrinsic shape differences and variability. *ACM Transactions on Graphics (TOG)*, 32(4):72, 2013.

[ROUG17] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3577–3586, 2017.

[RPWO18] Jing Ren, Adrien Poulenard, Peter Wonka, and Maks Ovsjanikov. Continuous and orientation-preserving correspondences via functional maps. In *SIGGRAPH Asia 2018 Technical Papers*, SIGGRAPH Asia '18, pages 248:1–248:16, 2018.

[RRBW+14] Emanuele Rodola, S Rota Bulò, Thomas Windheuser, Matthias Vestner, and Daniel Cremers. Dense non-rigid shape correspondence using random forests. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 4177–4184. IEEE, 2014.

[RW14] Martin Rumpf and Max Wardetzky. Geometry processing from an elastic perspective. *GAMM-Mitteilungen*, 37(2):184–216, 2014.

[SA07] Olga Sorkine and Marc Alexa. As-rigid-as-possible surface modeling. In *Computer Graphics Forum*, volume 4, 2007.

[Sah18] Yusuf Sahillioğlu. A genetic isometric shape correspondence algorithm with adaptive sampling. *ACM Trans. Graph.*, 37(5), October 2018.

[SAK15] Matan Sela, Yonathan Aflalo, and Ron Kimmel. Computational caricaturization of surfaces. *Computer Vision and Image Understanding*, 141:1 – 17, 2015.

[SBC14] Nitzan Shapira and Mirela Ben-Chen. Cross-collection map inference by intrinsic alignment of shape spaces. In *Computer Graphics Forum*, volume 33, pages 281–290. Wiley Online Library, 2014.

[SBR16] Ayan Sinha, Jing Bai, and Karthik Ramani. Deep Learning 3D Shape Surfaces Using Geometry Images. In *ECCV*, pages 223–240. Springer, 2016.

[SBS06] Florian Steinke, Volker Blanz, and Bernhard Schölkopf. Learning dense 3d correspondence. In *Advances in Neural Information Processing Systems*, pages 1313–1320, 2006.

[SBZB15] Baoguang Shi, Song Bai, Zhichao Zhou, and Xiang Bai. Deeppano: Deep Panoramic Representation for 3-d Shape Recognition. *IEEE Signal Processing Letters*, 22(12):2339–2343, 2015.

[SHK+14] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[SK14] Alon Shtern and Ron Kimmel. Iterative closest spectral kernel maps. In *3D Vision (3DV)*. IEEE, 2014.

[SK16] Yusuf Sahillioğlu and Ladislav Kavan. Detail-preserving Mesh Unfolding for Non-rigid Shape Retrieval. *TOG*, 35(2), 2016.

[SMKLM15] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view Convolutional Neural Networks for 3d Shape Recognition. In *Proc. ICCV*, pages 945–953, 2015.

[SNB+12] Justin Solomon, Andy Nguyen, Adrian Butscher, Mirela Ben-Chen, and Leonidas Guibas. Soft Maps between Surfaces. *Computer Graphics Forum*, 31, 2012.

[SP04] Robert W Sumner and Jovan Popović. Deformation transfer for triangle meshes. In *ACM Transactions on Graphics (TOG)*, volume 23, pages 399–405, 2004.

[SPKS16] Justin Solomon, Gabriel Peyré, Vladimir G Kim, and Suvrit Sra. Entropic metric alignment for correspondence problems. *ACM Transactions on Graphics (TOG)*, 35(4):72, 2016.

[SSB05] Bernhard Schölkopf, Florian Steinke, and Volker Blanz. Object correspondence as a machine learning problem. In *Proc. ICML*, pages 776–783. ACM, 2005.

[SSCO08] Lior Shapira, Ariel Shamir, and Daniel Cohen-Or. Consistent Mesh Partitioning and Skeletonisation Using the Shape Diameter Function. *Vis. Comput.*, 24(4):249–259, 2008.

[SX16] Shuran Song and Jianxiong Xiao. Deep Sliding Shapes for Amodal 3D Object Detection in RGB-D Images. In *Proc. CVPR, IEEE*, 2016.

[SY11] Yusuf Sahillioğlu and Yücel Yemez. Coarse-to-fine combinatorial matching for dense isometric shape correspondence. In *Computer Graphics Forum*, volume 30, pages 1461–1470, 2011.

[SZS+17] Rui Shi, Wei Zeng, Zhengyu Su, Jian Jiang, Hanna Damasio, Zhonglin Lu, Yalin Wang, Shing-Tung Yau, and Xianfeng Gu. Hyperbolic harmonic mapping for surface registration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(5):965–980, 2017.

[TCL+13] Gary KL Tam, Zhi-Quan Cheng, Yu-Kun Lai, Frank C Langbein, Yonghuai Liu, David Marshall, Ralph R Martin, Xian-Fang Sun, and Paul L Rosin. Registration of 3d point clouds and meshes: a survey from rigid to nonrigid. *IEEE Transactions on Visualization and Computer Graphics*, 19(7):1199–1217, 2013.

[TDB10] Stanimire Tomov, Jack Dongarra, and Marc Baboulin. Towards Dense Linear Algebra for Hybrid GPU Accelerated Manycore Systems. *Parallel Computing*, 36(5-6):232–240, June 2010.

[TFV+13] Alex Tsui, Devin Fenton, Phong Vuong, Joel Hass, Patrice Koehl, Nina Amenta, David Coeurjolly, Charles DeCarli, and Owen Carmichael. Globally optimal cortical surface matching with exact landmark correspondence. In *Information Processing in Medical Imaging*, volume 23, page 487. NIH Public Access, 2013.

[TPBF87] Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. Elastically deformable models. In *Proc. SIGGRAPH*, volume 21, pages 205–214, 1987.

[TW06] Bernhard Thomaszewsk and Markus Wacker. Bending models for thin flexible objects. In *WSCG Short Comm.*, 2006.

[Ura93] Hajime Urakawa. *Calculus of Variations and Harmonic Maps*. Translations of Mathematical Monographs. AMS, 1993.

[VKZHCO11] Oliver Van Kaick, Hao Zhang, Ghassan Hamarneh, and Daniel Cohen-Or. A survey on shape correspondence. In *Computer Graphics Forum*, volume 30, pages 1681–1707, 2011.

[VLB+16] Matthias Vestner, Roee Litman, Alex Bronstein, Emanuele Rodolà, and Daniel Cremers. Bayesian inference of bijective non-rigid shape correspondence. *arXiv preprint arXiv:1607.03425*, 2016.

[VLR+17] Matthias Vestner, Roee Litman, Emanuele Rodola, Alex Bronstein, and Daniel Cremers. Product manifold filter: Non-rigid shape correspondence via kernel density estimation in the product space. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6681–6690. IEEE, 2017.

[vRESH16] Philipp von Radziewsky, Elmar Eisemann, Hans-Peter Seidel, and Klaus Hildebrandt. Optimized subspaces for deformation-based modeling and shape interpolation. *Computers & Graphics*, 58:128–138, 2016.

[WHC+16] Lingyu Wei, Qixing Huang, Duygu Ceylan, Etienne Vouga, and Hao Li. Dense human body correspondences using convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1544–1553, 2016.

[WHG13] Fan Wang, Qixing Huang, and Leonidas J Guibas. Image co-segmentation via consistent functional maps. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 849–856. IEEE, 2013.

[WSK+15] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d Shapenets: A Deep Representation for Volumetric Shapes. In *Proc. CVPR*, pages 1912–1920, 2015.

[WSSC11] Thomas Windheuser, Ulrich Schlickewei, Frank R Schmidt, and Daniel Cremers. Geometrically consistent elastic matching of 3d shapes: A linear programming solution. In *Proc. IEEE International Conference on Computer Vision*, pages 2134–2141, 2011.

[WYYZ08] Yilun Wang, Junfeng Yang, Wotao Yin, and Yin Zhang. A new alternating minimization algorithm for total variation image reconstruction. *SIAM Journal on Imaging Sciences*, 1(3):248–272, 2008.

[WZX+16] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a Probabilistic Latent Space of Object Shapes Via 3d Generative-adversarial Modeling. In *NIPS*, pages 82–90, 2016.

[XKH+16] Kai Xu, Vladimir G. Kim, Qixing Huang, Niloy J. Mitra, and Evangelos Kalogerakis. Data-Driven Shape Analysis and Processing. *SIGGRAPH Asia Course notes*, 2016.

[XY13] Yangyang Xu and Wotao Yin. A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion. *SIAM Journal on Imaging Sciences*, 6(3):1758–1789, 2013.

[YSGG17] Li Yi, Hao Su, Xingwen Guo, and Leonidas Guibas. SyncSpecCNN: Synchronized Spectral CNN for 3D Shape Segmentation. *Proc. CVPR, IEEE*, 2017.

[ZSCO+08] Hao Zhang, Alla Sheffer, Daniel Cohen-Or, Quan Zhou, Oliver Van Kaick, and Andrea Tagliasacchi. Deformation-driven shape correspondence. *Comput. Graph. Forum*, 27(5):1431–1439, 2008.

[ZvKD10] Hao Zhang, Oliver van Kaick, and Ramsay Dyer. Spectral Mesh Processing. *Computer Graphics Forum*, 29(6):1865–1894, 2010.

[ZW11] Daniel Zoran and Yair Weiss. From learning models of natural image patches to whole image restoration. In *IEEE International Conference on Computer Vision (ICCV)*, pages 479–486. IEEE, 2011.

[ZWL+17] Xiaopeng Zheng, Chengfeng Wen, Na Lei, Ming Ma, and Xianfeng Gu. Surface registration via foliation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 938–947, 2017.

[ZYL+17] Chenyang Zhu, Renjiao Yi, Wallace Lira, Ibraheem Alhashim, Kai Xu, and Hao Zhang. Deformation-driven shape correspondence via shape recognition. *ACM Trans. Graph.*, 36(4):51:1–51:12, 2017.

בין מאפיינים כגון פינות וכפלים. במקרים רבים, התאמת מאפיינים כאלה חיונית על מנת חישוב מיפוי סמנטי. ניתן לבצע זאת על ידי החלפת אנרגית דיריכלה באנרגיה אלסטית, בה נעשה שימוש נרחב עבור דפורמציה של צורות. השימוש באנרגיית דפורמציה לחישוב מיפויים הוא טבעי, שכן מיפוי בין צורות למעשה שקול לדיפורמציה של צורת המקור אשר מוגבלת לצורת המטרה. כדי לאפשר אופטימיזציה של האנרגיה האלסטית ללא מגבלות על המיפוי ההתחלתי, נעשו מספר שינויים באנרגיה המקורית. מכיוון שהאנרגיה אינה ליניארית במקרה זה, האופטימיזציה שלה איטית יותר מהאופטימיזציה של אנרגיית דיריכלה, אך התוצאות מדויקות יותר.

לבסוף, מתוארת שיטה המשתמשת בהתאמה בין צורות עבור למידה עמוקה. בעוד שיש שימוש נרחב בשיטות למידה עמוקה המקבלות תמונות כקלט, יישום טכניקות למידה עמוקה עבור נתונים תלת־ממדיים מאתגר יותר. הסיבה להבדל היא כי בעוד תמונות ניתן לייצג בקלות על ידי מבנה אחיד)מערך של פיקסלים(, נתונים תלת מימדיים בדרך כלל לא מובנים באופן אחיד. אם נעשה שימוש בייצוג אחיד של נתונים תלת־ממדיים (כגון סריג תלת מימדי), ניתן להשתמש בטכניקות למידה עמוקה, אך ייצוג כזה בדרך כלל אינו עושה שימוש בתכונות הצורה. לדוגמה, סריג תלת מימדי מקודד באופן שונה לחלוטין צורות של אותו אובייקט במנח אחר. כדי להתגבר על בעיה זו, ניתן להשתמש בהתאמה בין צורות כדי למפות פונקציות גאומטריות מהצורה לסריג דו ממדי. התוצאה היא תמונה המייצגת את הצורה, ויכולה לשמש כקלט לרשתות עצביות סטנדרטיות שפועלות על תמונות. אם ההתאמה משמרת מרחקים בין נקודות, צורות איזומטריות יהיו מיוצגות באופן דומה. הצענו שיטה המחשבת ייצוג כזה באופן אופטימלי עבור יישום נתון, כאשר במחקר זה התמקדנו בסיווג ואחזור של צורות תלת מימדיות.

לסיכום, תזה זו מתארת מספר שיטות עבור התאמה לא איזומטרית בין צורות, אשר נבדלות בסוג הקלט וביישום המיועד. מאחר ויישומים שונים עשויים לדרוש התאמה שונה משמעותית בין צורות, יש לקחת את היישום בחשבון כאשר מתכננים שיטות התאמה. השיטות המתוארות מתאימות ליישומים שונים, כגון העברת טקסטורה, אינטרפולציה, וסיווג צורות.

תקציר

התאמה בין צורות היא בעיה בסיסית בתחום ניתוח צורות, אשר משמשת ליישומים רבים בגרפיקה וראייה ממוחשבת, כגון ניתוח סטטיסטי של צורות, העברה של טקסטורה ואינטרפולציה בין צורות. המטרה היא לחשב לכל נקודה על צורת המקור, נקודה תואמת על צורת המטרה. קיימות דרכים שונות לסווג התאמה בין צורות, לפי התכונות של הצורות ושל התוצאה הרצויה. סיווג נפוץ מבדיל בין התאמה איזומטרית, אשר לרב מאפיינת התאמה בין צורות של אותו אובייקט במנח שונה, לבין התאמה לא איזומטרית, אשר מתאימה בין אובייקטים שונים. ניתן להגדיר מתמטית התאמה איזומטרית על־ידי שימור מרחקים גאודזיים בין זוגות נקודות, בעוד שאין מוסכמה על הגדרה מתמטית שמאפיינת התאמה במקרה הכללי (לא איזומטרית). למעשה קיימים מקרים בהם יישומים שונים דורשים תוצאת התאמה שונה. לכן האתגר הראשי במחקר זה הוא להגדיר מתמטית תכונות רצויות של התאמה בין אובייקטים שונים, ולתכנן אלגוריתמים שיחשבו התאמות בעלות התכונות שהוגדרו. כאשר מתאימים בין אובייקטים שונים, לא ניתן להניח שניתן להסיק התאמה על סמך הגאומטריה של הצורות בלבד. לכן אנו מניחים שפרט לצורות נתון מידע נוסף, כגון קבוצה דלילה של נקודות תואמות, או התאמה התחלתית.

עידון התאמה התחלתית משמש כאמצעי למינוף שיטות מיפויים קיימות, המניבות תוצאות טובות עם אי דיוקים מקומיים. לדוגמה, שיטות קיימות רבות משתמשות במיפויים פונקציונליים: אופרטורים ליניאריים שמתאימים בין פונקציות חלקות במקום נקודות. למיפויים פונקציונלים יש יתרונות רבים; ייצוגם הבדיד הוא קומפקטי, וחישובו יעיל מאחר וניתן לנסח מאפיינים רבים כאילוצים ליניאריים. עם זאת, בסופו של דבר יש להמיר את המיפוי הפונקציונלי למיפוי בין נקודות, וחילוץ מיפוי טוב בין נקודות אינו טריויאלי. לכן פיתחנו שיטה שממירה ביעילות ובדיוק גבוה מיפויים פונקציונליים למיפויים בין נקודות. שיטה זו שימושית עבור מגוון שיטות המסתמכות על מיפויים פוקציונליים. השיטה יכולה לשמש גם לעידון מיפוי התחלתי בין נקודות, על ידי המרתו למיפוי פונקציונלי (הסרת תדרים גבוהים) והמרה חזרה למיפוי בין נקודות.

כאשר נתונה כקלט קבוצה דלילה של נקודות תואמות, ברצוננו להרחיבה למיפוי מלא חלק. אנרגיית דיריכלה היא פונקציה ידועה אשר מודדת את מידת החלקות של המיפוי, ואף קיימת נוסחה בדידה של אנרגיה זו בין משטחי משולשים. בעוד שהאופטימיזציה של אנרגית דיריכלה משפרת את החלקות המקומית, היא גם מצמצמת את המיפוי במובן שאזורים שלמים של צורת המטרה עשויים להישאר ללא התאמה. לכן פיתחנו לאחרונה אנרגיה המשלבת קירוב דיסקרטי של אנרגית דיריכלה וביטוי המקדם הופכיות ומונע את צמצום המיפוי. הדיסקרטיזציה שלנו מאפשרת אופטימיזציה יעילה, כמו גם דיוק מקומי גבוה.

בעוד אופטימיזציה של אנרגית דיריכלה מובילה לתוצאות חלקות, התוצאות לא בהכרח יתאימו

המחקר בוצע בהנחייתה של פרופסור מירלה בן־חן, בפקולטה למדעי המחשב.

חלק מן התוצאות בחיבור זה פורסמו כמאמרים מאת המחבר ושותפיו למחקר בכנסים ובכתבי־עת במהלך תקופת מחקר הדוקטורט של המחבר, אשר גרסאותיהם העדכניות ביותר הינן:

Danielle Ezuz and Mirela Ben-Chen. Deblurring and denoising of maps between shapes. *Computer Graphics Forum*, volume 36, pages 165-174, 2017.

Danielle Ezuz, Justin Solomon, Vladimir G. Kim and Mirela Ben-Chen. GWCNN: A metric alignment layer for deep shape analysis. *Computer Graphics Forum*, volume 36, pages 49-57, 2017.

Danielle Ezuz, Justin Solomon and Mirela Ben-Chen. Reversible harmonic maps between discrete surfaces. *ACM Transactions on Graphics (TOG)*, volume 38, pages 15:1-15:12, 2019.

Danielle Ezuz, Behrend Heeren, Omri Azencot, Martin Rumpf and Mirela Ben-Chen. Elastic Correspondence between Triangle Meshes. *Computer Graphics Forum*, volume 38, 2019.

אני מודה לטכניון, לפקולטה למדעי המחשב, למלגת ג'ואן ואירוין ג'ייקובס, ולמענק נסיעה ע"ש פרופ רחמימוף עבור חוקרים צעירים של הקרן הדו־לאומית ישראל־ארה"ב למדע על התמיכה הכספית הנדיבה בהשתלמותי.

התאמה לא איזומטרית בין צורות

חיבור על מחקר

לשם מילוי חלקי של הדרישות לקבלת התואר
דוקטור לפילוסופיה

דניאל עזוז

הוגש לסנט הטכניון – מכון טכנולוגי לישראל
אייר התשע"ט חיפה מאי 2019

התאמה לא איזומטרית בין צורות

דניאל עזוז