Technische Universität München
Fakultät für Informatik

Doctoral Thesis

Continuous Correspondence of Non-Rigid 3D Shapes

Zorah Lähner

Supervised By

Prof. Dr. Daniel Cremers

September 2020

TECHNISCHE UNIVERSITÄT MÜNCHEN

Fakultät für Informatik
Lehrstuhl für Bildverarbeitung und Künstliche Intelligenz

Continuous Correspondence of Non-Rigid 3D Shapes

Zorah Lähner

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender:	Prof. Dr. Stephan Günnemann
Prüfer der Dissertation:	1. Prof. Dr. Daniel Cremers
	2. Prof. Dr. Emanuele Rodolà

Die Dissertation wurde am 23.09.2020 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 09.03.2021 angenommen.

To David.

Writing a book is a horrible, exhausting struggle, like a long bout with some painful illness. One would never undertake such a thing if one were not driven on by some demon whom one can neither resist nor understand.

– George Orwell

Abstract

THE non-rigid 3D shape correspondence problem is an important part of many algorithms for geometry processing, and applications in virtual and augmented reality. However, non-rigidity increases the degrees of freedom in comparison to the rigid problem and leads to algorithms that need to compromise between runtime, and guaranteeing desirable properties, like continuity, in the solution. In this thesis, we explore several efficient algorithms solving for continuous correspondences between non-rigid shapes. The first considers taking a 2D contour and a 3D shape as input. Trans-dimensional settings are especially hard, because common descriptors are not comparable between different dimensions and a lot of methods rely on projecting the higher dimensional shape down instead. This does not work for non-rigid deformations on the 3D shapes, but we show that popular spectral descriptors for non-rigid cases can be transferred to the 2D-3D setting with minimal adjustment. Using the special 1-dimensional structure of the solution for contour shapes, we pose the correspondence as a shortest path problem on the product graph. This can be solved efficiently by Dijkstra's algorithm and a branch-and-bound strategy. In the case of two 3D input shapes the solution is 2-dimensional, so it is a minimal surface instead of a shortest path. This can be formulated as a quadratic assignment problem (QAP) between kernels, and we show that using positive-definite heat kernels has superior theoretical properties to previously used gaussian kernels. We solve the QAP through difference of convex functions programming in a series of linear assignment problems. Additionally, we introduce a multi-scale approach which separates the problem into solvable subsets but can still propagate global information throughout. Furthermore, we analyze the properties of maps on the product manifold to prove that conventional algorithms do not make use of the optimal representation in the separable Laplace-Beltrami eigenbasis. Based on this observation we show what the optimal representation is and proprose a novel, not separable, localized basis that is better suited for correspondences, and we propose a framework to refine correspondences directly on the product manifold. Finally, we introduce a method that produces continuous correspondences based on a smooth, volume-preserving deformation field. We argue that for most real-world objects not only is the correspondence smooth, but there also exists a sequence of intermediate shapes with the same properties transforming the source into the target. To this end, our algorithm solves for the correspondences and the deformation jointly using an expectation-maximization approach. Because we represent the deformation in a closed-form, frequency ordered basis, we can perform the optimization efficiently on a subsampling but still retrieve a solution and interpolation for shapes of any resolution with only linear overhead, and without discretization artifacts.

Zusammenfassung

DAS Korrespondenzproblem zwischen elastisch verformten, drei-dimensionalen Formen spielt eine zentrale Rolle in vielen Algorithmen in der Geometrieverarbeitung und Anwendungen in Virtual und Augmented Reality. Allerdings wächst die Anzahl der Freiheitsgrade im Vergleich zur starren Problemstellung enorm, was dazu führt, dass Algorithmen Kompromisse zwischen Laufzeit und wünschenswerten Eigenschaften, wie Kontinuität, finden müssen. In dieser Dissertation beschäftigen wir uns mit mehreren effizienten Algorithmen, die verschiedene Variationen des kontinuierlichen Korrespondenzproblems für elastische Formen lösen. Die erste Methode untersucht eine Situation, in der eine 2D Kontur und eine deformierbare 3D Form gegeben sind. Transdimensionale Probleme sind besonders schwierig, weil die meisten Deskriptoren nicht zwischen unterschiedlichen Dimensionen vergleichbar sind. Viele Methoden lösen dies, indem sie die höher dimensionale Form in eine Niedrigere projezieren. Das funktioniert für elastisch deformierte Formen nicht, aber wir zeigen in dieser Arbeit, dass die für elastische Formen populären, spektralen Deskriptoren auch mit wenig Aufwand für Korrespondenzen zwischen 2D und 3D angepasst werden können. Diese neuen Deskriptoren und der Fakt, dass die Lösungsstruktur von Konturen eindimensional ist, nutzen wir, um das Korrespondenzproblem als Kürzeste-Wege-Problem auf einem Graphen zu formulieren. Diese können durch Dijkstras Algorithmus und einen Branch-and-Bound Ansatz effizient gelöst werden. Für zwei 3D Formen hat die Lösung eine zweidimensionale Struktur und ist deswegen ein Minimale-Oberflächen-Problem anstatt eines Kürzeste-Wege-Problem. Dieses kann durch ein quadratisches Zuordnungsproblem (QAP) zwischen Kerneln gelöst werden. Wir zeigen, dass die positiv-definiten Heat-Kernel, die wir einsetzen, überlegene theoretische Eigenschaften gegenüber den vorher benutzten GauSS-Kerneln haben. Wir lösen das QAP durch Difference-of-Convex-Functions Programmierung in einer Serie von linearen Zuordnungsproblemen. Zusätzlich schlagen wir einen Multiskalen-Ansatz, der das Problem in kleinere Lösbare zerlegt. Dadurch funktioniert unsere Methode für jede Auflösung, aber kann gleichzeitig globale Informationen in alle Teilprobleme übertragen. Zusätzlich analysieren wir die Eigenschaften von Abbildungen auf der Produktmannigfaltigkeit, um zu zeigen, dass konventionelle Algorithmen oft nicht die optimale Representation in der separierbaren Laplace-Beltrami Basis nutzen. Basierend auf dieser Beobachtung stellen wir die optimale Representation und eine neue, nicht-separierbare, lokalisierte Basis vor, die sich besser zum Darstellen von Korrespondenzen eignet. Diese nutzen wir, um ein neues Framework zur Verfeinerung der Korrespondenzen direkt auf der Produktmannifaltigkeit. Zuletzt stellen wir eine Methode vor, die kontinuierliche Korrespondenzen basierend auf einem glatten, volumenerhaltenden

Deformationsfeld berechnet. Die Idee entstammt der Beobachtung, dass die meisten echten Objekte nicht nur eine kontinuierliche Korrespondenz haben, sondern auch durch eine Sequenz von Zwischenzuständen mit den gleichen Eigenschaften wie den Eingangsformen verbunden sind. Dafür entwickeln wir einen Algorithmus, der durch einen Expectation-Maximization Ansatz gleichzeitig die Korrespondenz und die Deformation optimiert. Da wir die Deformationen in einer Basis mit analytischer, frequenz-geordneter Form darstellen, können wir die Optimierung effizient auf einer kleinen Auswahl von Punkten ausführen. Trotzdem können wir die Lösung und Interpolation ohne Diskretisierungsartefakte mit linearem Aufwand auf beliebige Auflösungen übertragen.

Acknowledgments

My journey to this thesis would have not been possible without the continuous support and inspiration of the amazing people around me.

First of all, I would like to thank my doctoral advisor Prof. Daniel Cremers for giving me the great opportunity to do a PhD under his guidance and the freedom to pursue the research I was interested in. I am incredibly grateful for his excellent advice and all his support of my ideas and decisions. Second, I thank Prof. Emanuele Rodolà for raising my interest in Shape Analysis many years ago, encouraging me to pursue a PhD and giving me countless advice since then. I would probably not be where I am today without him. Further, I would like to thank Prof. Stephan Günnemann, Prof. Daniel Cremers and Prof. Emanuele Rodolà for agreeing to serve as members of my examination committee.

I want to thank Sabine Wagner and Quirin Lohr for always keeping our lab running smoothly, never being annoyed by my problems, and probably saving me days full of frustration.

All my publications have been joint work with truly exceptional people and would not have been possible without their contribution: Amit Boyarski, Alex Bronstein, Michael Bronstein, Daniel Cremers, Marvin Eisenberger, Ron Kimmel, Or Litany, Tal Remez, Emanuele Rodolà, Frank Schmidt, Ron Slossberg, Justin Solomon, Tony Tung and Matthias Vestner. It was a pleasure to work with all of you. Special thanks goes to Ioannis Chiotellis, Viktor Kaszian, Jennifer Kostenzer, Lukas Köstler and David Lähner for proof-reading this thesis.

I had a really amazing time during my PhD and this is mostly due to my great colleagues from the Computer Vision and Artificial Intelligence group at TUM. I learned a lot from all of you, about research and many other things, and probably had more fun than you are supposed to have during a PhD. Additionally, I thank Dr. Tony Tung and Dr. Roberto Mecca for supervising me during my fascinating internships I did at Facebook Reality Labs and Toshiba Research Europe.

And last but not least, I want to thank my family and friends for always believing in me and making me feel like I could achieve anything I want. I could have not done it without you. Thank you, David, for all your love and constantly telling me not to sell myself short.

List of Frequently Used Symbols

Subscripts may be dropped if they are irrelevant or clear from context to improve readibility.

$\mathcal{M}, \mathcal{N}$	(Riemannian) manifolds with arbitrary dimensions
$\mathcal{X}, \mathcal{Y}$	(Riemannian) manifolds with the same dimensions
ξ_i	coordinate map
$\mathcal{M} \times \mathcal{N}, \mathcal{X} \times \mathcal{Y}$	Product manifolds
$\pi_{\mathcal{M}}$	projection from $\mathcal{M} \times \mathcal{N}$ to $\mathcal{M}$
$\Delta_{\mathcal{X}}$	Laplace-Beltrami operator on $\mathcal{X}$
A, S	the discrete mass and stiffness matrix of the LBO
$(\Phi, \Lambda), (\Psi, \mu)$	Pairs of eigenfunctions and eigenvalues
$\phi_i, \lambda_i, \psi_i, \mu_i$	the i-th eigenfunction and value
$\Pi \in \mathcal{P}_n$	a permutation matrix from the set of $n \times n$ permutation matrices

Contents

PART I

Introduction

Nobody ever figures out what life is all about, and it doesn't matter. Explore the world. Nearly everything is really interesting if you go into it deeply enough.

– Richard P. Feynman

CHAPTER 1

Introduction

The properties of shapes have been studied since ancient times. For example the relationships in **Euclid** (300BC) 's Elements describe how simple geometric objects can be constructed out of each other. Within geometry, three dimensional shapes have a special meaning for humans, because the world we live in and its objects are three dimensional. Deeper understanding of those shapes comes natural to us, but it is also necessary to navigate our world. On the other hand, the majority of research in computer vision has been done on 2D data, either images or image collections, like videos. This has multiple reasons. First, capturing the entire surface of a 3D object requires at least multiple view points. Second, storing information on a 2D surface, like paper, was much more storage efficient before computers were invented. Furthermore, through our vast experience in the 3D world, humans have enough intuition about every day objects and our environment to imagine the correct 3D object from 2D, or our stereo vision. It is even possible to play with our intuition of object projections by drawing physically impossible objects that still look natural; a skill M.C. Escher became very famous for. However, the projection of a 3D object on a 2D plane always comes with loss of information, *i.e.* it can only show one view point and depth information is ambiguous, or it is flattened out and metric distortion is unavoidable.

Some advantages of images also apply in digital representation. While images require less space and are faster to process, both in reality as well as on computers, knowing the exact 3D geometry is beneficial, or even a requirement, in many applications. One important application, in which digital 3D representation has led to tremendous advances, is protein prediction. The spatial structure of proteins is important for their effect on body functions, and simulations of protein interactions can be done computationally to find new treatments (**Knoverek et al.** 2019). This is impossible without an accurate and complete representation of the 3D structure as well as efficient, stable 3D geometry processing algorithms. Other disciplines, like architecture, have automated many previously tedious and complicated tasks with only minimal user guidance. This allows to process small changes in the requirements within seconds – instead of an expert manually redrawing plans –, and the result to be optimal in terms of material and space consumption. Again, this is only possible with exact 3D representation to simulate static effects precisely. While these applications rely on accurate 3D models, they are most likely directly modeled on a computer. This guarantees a certain quality and properties in the representation. This is not always the case when 3D models are scans from the real world.

The capture of real-world objects comes with many challenges, and, as a result, raw 3D scans are often noisy, have missing parts, and fail to represent fine geometric details. Even clean, artificial 3D models need to be discretized on the computer somehow, and different instances can differ widely in resolution, structure and quality. While images are always represented as a regularly spaced grid, 3D data has more flexibility. This adds expressiveness but reduces efficiency in both processing and memory consumption. Furthermore, acquisition hardware cannot capture the entire surface of a 3D object at once. Instead cameras or scanners are fixed to one (or multiple) point(s) around the object and detect the surface from these view points. This leads to a collection of 2D images which need to be merged into a 3D reconstruction; a process that can introduce noise and errors. It is especially hard to reconstruct highly non-convex surfaces, because more view points are needed to detect every point on the surface. Thus, real-world 3D scans are always already processed versions of 2D data, and their quality depends heavily on the acquisition setting and reconstruction algorithm. At the same time, this shows how 3D data can be superior to images in applications where a single view is not sufficient, because it cannot represent the full geometry. If multiple images are needed to reconstruct the entire object, one image obviously does not contain all information, but the 3D model is a compact representation of a collection of images all showing the same object.

3D geometry and its processing are the backbone of Virtual and Augmented Reality (VR/AR) applications. The illusion of virtual environments can only be maintained if a robust estimation of the surroundings exists and objects are processed accordingly. In the beginning VR and AR were mostly focused on entertainment applications, and, being restricted in their robustness and flexibility, this made sense. In recent years medical and industry applications have taken over. Here, virtual environments can provide safe and cheap training, or give visual feedback for crucial operations in real-time. Many of these applications require correspondences between seen objects to function properly. These correspondences can be used to highlight differences, transfer information, improve the reconstruction, or alter the appearance of an object. The correspondence problem for rigid objects has been extensively studied on images and (partial) 3D objects (**Tam et al.** 2013). The solution in this setting has only six degrees of freedom in 3D (three for rotation, three for translation) and optimizing, even in the presence of severe noise, can be done efficiently.

The non-rigid correspondence problem, which we will focus on in this thesis, includes any pair of shapes that cannot be aligned by a single rigid transformation. Non-rigid deformations can include basically anything but there are sub-classes with more restrictions, for example isometric shapes. This includes humans who can move into a wide variety of poses and whose tracking is an entire research direction (**Leal-Taixé et al.** 2017). Other examples of non-rigid deformations preserve properties like angles or volume only. What makes non-rigidity complex is that a low-dimensional description of the deformation is usually not possible. Instead every point on the surface can potentially deform in a different direction. This increases the size of the solution space exponentially and makes finding the optimal solution hard, *e.g.* the aforementioned isometric case was shown to be NP-hard. While isometric is a reasonable assumption for deformable objects, real-world deformation are never perfect. Deformations will stretch certain areas, scans induce noise by self-touching, or the matching is between the same type not the same instance. Nevertheless, the majority of solid, real-world objects still preserves neighborhood information on the surface when moving. Even a balloon that is filled with air and completely changes shapes does not change the neighborhood of particles on the surface. This thesis will focus on this kind of deformations, how to formalize them, and how to develop efficient algorithms in spite of the high complexity of the problem.

1.1 Thesis Outline

The remainder of this thesis is organized as follows. The next sections will give an overview over different categories and sub-fields in shape analysis and shape correspondence, along with the most influential work in these areas. Additionally, the methodology chapters have their own related work sections with a survey of publications directly related to the chapter. Chapter 2 lines out the major contributions in this thesis and in which original papers these were published. Additionally, Section 2.2.2 contains summaries of my publications not used in this thesis. Part II introduces mathematical background and popular methods that are referenced throughout the thesis and can be skipped by readers familiar with the topic. The background includes basic differential geometry in Chapter 3, spectral shape analysis in Chapter 4 and product spaces in Chapter 5. An introduction into 3D correspondences is given in Chapter 6. It includes formal definitions and corresponding discretization. The main part of this thesis is Part III, where each chapter goes into detail about the methodology of one shape correspondence algorithm. In Chapter 7 we look at non-rigid correspondence between 2D and 3D shapes. Solving the problem globally between two 3D shapes is provably harder, and in Chapter 8 we propose an efficient multi-scale method aligning kernels for this case. Chapter 9 considers the same problem and describes properties of the optimal solution on the product manifold as well as how it can be represented in the best way. Last, we look at continuity from a different angle in Chapter 10. There, we propose to solve for a smooth, volume-preserving deformation field jointly with the correspondence. The thesis finishes with a discussion of the previous parts in Chapter 11.

1.2 Shape Analysis

Shape analysis, as used in this thesis, refers to the at least partially automatic analysis of geometric objects. This includes any processing of geometric objects, or theoretical proofs, but also the computation of properties, and discrete representation within a computer. These questions have been tackled for a long time (**Loncaric** 1998), but recent advances in computing power and 3D scanning hardware have brought the field to a new level. While some properties and relationships are purely mathematical and timeless, it has become more important how to efficiently calculate them and under which assumptions of the data. Different shape representations can have a huge effect on computation time, and what the best representation is may vary between applications. The topology of an object is important for many mathematical theories, and many algorithms fail when comparing two objects with different topology (**Lee and Kazhdan** 2019). Semantics and human perception, on the other hand, are rarely influenced by topology but instead depend on details that are hard to put in mathematical formulas.

1.2.1 Representation

The first decision to make in shape analysis is how to define and represent a shape. Because this thesis includes mostly 3D and 2D shapes, these will be our focus here. They have the additional advantage of being easily visualized. Most real-world objects are solid, and a complete representation would model the interior of them. However, acquisition of the interior of an object is even more complicated than of the surface and requires special hardware, like a CT-scanner. We call a representation that models the interior *volumetric*. It is possible to infer certain information about the interior and physical properties of a volumetric shape by observing its interactions with the physical world (**Weiss et al.** 2020), but the process is very sophisticated. Moreover, the interior is not of interest in many application. 3D artists usually just model the surface only, because this is the visible part, both in most acquisition settings and during rendering. Therefore, in this thesis we focus our attention on shapes that are represented as their surfaces. The usual assumption is that the surface is a Riemannian manifold (in our case a 2D manifold embedded in 3D), more details on this can be found in Chapter 3.

1.2.2 Intrinsic

Intrinsic in shape analysis refers to methods and properties that can be derived without relying on any embedding information. This means they only rely on how the surface behaves relatively to itself but not within the embedding space. On Riemannian manifolds this is captured through the metric tensor attached at each point. The metric tensor defines an inner product between vectors on the surface and, as an example, applying the same translation to each point on the surface of an manifold does not change the inner product of vectors. This can also work for more complicated, non-uniform deformations, and those are called isometries. Additionally, it holds approximately for many deformable, real-world shapes. For example, humans can come in many different poses, but these do not have a significant influence on distances measured on the surface. Only relying on extrinsic properties, *e.g.* Euclidean distances, is problematic when severe pose changes happen. Imagine the distance between a fingertip and tip of the nose when standing normally versus when scratching the nose. Therefore, intrinsic properties, which are comparable when the inputs are two shapes of the same class but in different poses, are advantageous for optimization, because they can be used without regularization or prior knowledge. On the other hand, there exist intrinsic symmetries, *e.g.* the left and right side of most objects, which cannot be captured by intrinsic features. Therefore, purely intrinsic algorithms often mix up the left and right side.

Spectral A huge subfield of intrinsic methods is concerned with spectral shape analysis. Spectral refers to the eigenfunction and -values (also called spectrum, hence the name) of the Laplace-Beltrami operator (LBO) (**Chavel** 1984), a general version of the Laplace operator on manifolds. Since the LBO is purely intrinsic, its eigendecomposition is also completely invariant under isometries. This makes it predestined to be used for those classes of deformable objects, like humans. An additional advantage is the frequency order of the eigenfunctions which allow bandpass filters and basis construction similar to Fourier analysis (**Canzani** 2013). Chapter 4 gives a detailed introduction into this topic, including how to discretize the LBO and a list of its properties.

1.2.3 Extrinsic

On the other hand, extrinsic is the complement of intrinsic and includes everything that cannot be derived thorough intrinsic information only. This means coordinates of surface points and properties that are derived from that, like normals, most curvatures, or the Euclidean distance between surface points. Some of these are invariant under special deformations, e.g. curvature is invariant under translation and rotation, but in general this is not the case. Specifically, isometries and near-isometries can change extrinsic properties drastically on different parts of the shapes. However, extrinsic properties are often capable of capturing fine geometric details. This class includes popular descriptors like SHOT (**Tombari et al.** 2010), the as-rigid-as-possible regularization (**Sorkine and Alexa** 2007) and mean curvature.

1.2.4 Distances

Distances between points are an important measurement used to calculate properties, similarity, or to be preserved during optimization. The Euclidean distance is the most known and popular distance between points but has only limited use in the non-rigid deformation case we focus on in this thesis, because large distortions can occur with pose changes. For isometries and non-rigid cases the geodesic on the surface of shapes is often used. It describes the shortest path between two points restricted to the surface of the manifold instead of the Euclidean embedding space. The geodesic distance is very robust against pose changes but computationally expensive (**Surazhsky et al.** 2005). As a result, several approximations of the geodesic distance exist with different trade-offs between accuracy and speed. Fast Marching (**Kimmel and Sethian** 1998) brings a considerable speed-up with small numerical errors as long as the mesh fulfills certain properties. Less accuracy but a huge speed-up, including reusable pre-factorization, can be achieved with Geodesics in Heat (**Crane et al.** 2017) which uses the relationship between the heat equation and geodesics. Geodesics in Heat can be applied to all representations that allow a few operators like the divergence to be computed, and differentiable which makes it interesting, especially for deep learning (**Cosmo et al.** 2020). However, they are not guaranteed to be symmetric or to fulfill the triangle inequality, properties that are crucial for some applications. Instead of distances between points, the Wasserstein or earth movers distance calculates distances between probability distributions on the metric spaces. If the space is the surface of a shape (**Solomon et al.** 2014b), this can be useful for soft maps, or, if the space is the embedding space, used to interpolate between shapes (**Solomon et al.** 2015).

1.3 Correspondence

Shape correspondence describes the problem of deciding which parts or points of two shapes belong to each other according to some meaningful criterion. The definition of meaningful is highly dependent on the application and could be purely semantic, depend on mathematical relations, or learned properties. We will look at the concrete definitions and properties used in this thesis both in terms of mathematics and intuition in Chapter 6. We mostly concern ourselves with shapes that are related, at least partially, through diffeomorphism, but completely different definitions, *e.g.* based on functionality, are possible (**Kaick et al.** 2013),

1.3.1 Images

The correspondence problem also appears in many applications in image processing. In order to reconstruct a shape from a collection of images without known camera positions, it is necessary to find points which are visible in multiple images to determine how the images overlap (**Hartley and Zisserman** 2004). Since images are bound to include a lot of clutter, sparse descriptors are more popular than dense ones, *e.g.* most famously SIFT (**Lowe** 1999) and SURF (**Bay et al.** 2006). However, other applications like optical flow aim to calculate a dense displacement field, which is a kind a representation for correspondence, instead of a sparse solution (**Dosovitskiy et al.** 2015; **Lucas and Kanade** 1981).

1.3.2 2D Shapes

The information on images comes purely from the color function on an equidistant grid. This makes many operations easy but also adds pixels which do not actually depict a part of the object and clutter to the result. Segmenting out the object of interest can help reduce the complexity for point correspondence in images (**Schoenemann and Cremers** 2007) and allows practical applications, like template matching, to work without much regard for color or lighting inconsistencies as well as a certain robustness against occlusion (**Y. Su et al.** 2015). The closed silhouettes of (connected) objects are guaranteed to be 1D-manifolds and as such any geometry processing tool can be applied to them (**F. R. Schmidt et al.** 2007).

Additionally, silhouettes can be used as guidance for (real-time) posing, or for animating 3D models based on a video stream (**Dibra et al.** 2017; **Natsume et al.** 2019).

1.3.3 3D Shapes

An additional complexity in 3D is added by the fact that the neighborhood size can vary a lot. Additionally to not every point necessarily having the same amount of neighbors, the neighborhood size can differ due to different discretization in between instances. This makes the definition of consistency a lot harder when looking only at point-wise correspondence. Nevertheless, putting only vertices in correspondence – as opposed to allowing sub-vertex matchings – reduces the size of the solution space. Furthermore, assuming a bijection between shapes gives many formulation beneficial properties and makes the problem manageable.

Rigid Deformations that apply the same translation and rotation onto the entire shape are called rigid. They have six degrees of freedom in 3D. Therefore, the optimization for the optimal rigid motion between twice the same rigid object has to find six parameters. Problems may arise from scanning noise or partiality but the dimension of the solution is always fixed. The most famous algorithm for rigid alignment is Iterative Closest Point (ICP) (**Besl and McKay** 1992) which alternates between finding correspondences via nearest neighbor, and solving for the optimal rigid motion to align these correspondences via the Procrustes problem. However, ICP is prone to fail without a good initialization. More robust methods have been proposed by using 3D descriptor information (**Guo et al.** 2016) and global optimization (**Zhou et al.** 2016). Due to the low dimensionality, robust algorithms for rigid registration have existed for years while other classes still pose open problems (**Tam et al.** 2013).

Non-Rigid Non-rigid deformations include anything that is not rigid. However, there are more well-defined sub-classes of non-rigidity which preserve certain properties or add specific classes of deformation to rotation and translation, *e.g.* shearing or conformal (which preserves angles on the surface) (**Yoshiyasu et al.** 2014). This thesis is mostly concerned with the case of isometries and near-isometric deformations. While rigid motion preserves the Euclidean distance between all pairs of points, isometric deformations preserve the geodesic distance. The next sections give an overview of different directions in near-isometric correspondence.

1.3.3.1 3D Descriptors

Similar to image correspondence, the majority of 3D correspondence methods rely on descriptors to provide a similarity measure between two points in order to guide or initialize the optimization. The two dominant types of descriptors are pointwise and pairwise.

Pointwise Descriptors In contrast to image descriptors, 3D correspondence methods rely mostly on dense, pointwise descriptors. One reason is that occlusion and partiality are usually less prominent in 3D data and local neighborhood information is a lot more significant due to this.

One line of descriptors commonly used in non-rigid correspondence is based on the spectral properties of shapes. The most basic variant is the Global Point Signature (GPS) introduced by **Rustamov** (2007) which consists of a series of the scaled eigenfunction values of the Laplace-Beltrami operator at each point. The heat kernel signature (HKS) (**J. Sun et al.** 2009) and wave kernel signature (WKS) (**Aubry et al.** 2011) are based on physical phenomena (heat diffusion and quantum particle movement respectively) and can be shown to have closed form solutions using the spectral eigenfunctions and values. However, as they are based on the spectral decomposition, which is invariant under isometries including self-symmetries, all of these descriptors are unable to distinguish between the left and right part of a symmetric shape.

Descriptors that are not purely intrinsic are able to capture differences between intrinsic symmetries but may also be inconsistent for different poses, especially when a lot of bending happens. These descriptors are able to capture very fine geometric information, but, because the finest scales are the most sensitive to noise, robust optimization is needed when processing them. Some examples are Spin Images (**Johnson and Hebert** 1999) which store the orientation information of a neighborhood points in an image like representation, Fast Point Feature Histogram (**Rusu et al.** 2009) which collects the geometric properties of each points neighborhood in histogram, and the SHOT descriptor (**Tombari et al.** 2010) which builds a consistent reference system at each point and collects a histogram of normals in the neighborhood of each point.

The previously mentioned works are all hand-crafted with some geometric interpretation. While having an interpretation can be useful in some scenarios, it is equally important that the descriptor is also discriminative for points with only subtle differences in geometry. **Litman and A. M. Bronstein** (2014) aimed at learning the best combination of spectral descriptors for certain classes of shapes using Mahalanobis metric learning. Similar, **Windheuser et al.** (2014) introduce Mercer kernels to find the provable optimal combination of input descriptors using large margin nearest neighbor optimization. Learning optimal descriptors has been refined by **Boscaini et al.** (2015) and **Boscaini et al.** (2016a). Not restricted to isometries is **Corman et al.** (2014) which learns optimal descriptors for correspondence between different classes of shapes if some ground-truth maps between these are given for training. This is an example of a line of work which does not only look at descriptor properties during optimization but includes the pipeline they will actually be used in in the energy functional. Functional Maps are the most popular choice in this direction because they are differentiable and low dimensional (**Litany et al.** 2017a). Instead of relying on optimizing input features, **Donati et al.** (2020) learns descriptors to be used for Functional Maps directly from raw geometry.

Pairwise Descriptors Instead of just characterizing one point, pairwise descriptors describe the relationship between pairs of points. This is more powerful than pointwise information which is normally not unique, e.g. the pointwise descriptor of two fingertips is very similar in any case, and therefore often leads to better results. On the other hand, the amount of information increases from n pointwise to n^2 pairwise values which increases the complexity of any optimization. See Section 1.3.3.2 for an introduction to Quadratic Assignment Problems that operate on pairwise descriptors.

Distance functions are the most common type of pairwise descriptors (also see Section 1.2.4). The Euclidean distance between two points is quick and easy to compute but non-rigid correspondence often relies on the geodesic distance instead (**Surazhsky et al.** 2005). Geodesics are preserved under isometric deformations and are more reliable on non-isometric, deformable shapes, because the Euclidean distance can change drastically with pose changes, but their computation is computationally heavy. Another type of pairwise descriptors are kernel functions, for example the Gaussian kernel or the heat kernel. An advantage of the heat kernel is that it can be approximated by a finite sum of eigenfunctions and values of the Laplace-Beltrami operator. This makes them efficient to compute. There is also a direct relation between heat kernels and geodesic distances which was explored in **Crane et al.** (2017).

Global Descriptors Global descriptors are meant to describe the geometry of a shape in its entirety and not distinguish parts within. As such they are not suited for correspondence but more for retrieval of similar shapes. ShapeDNA describes a shape through its sequence of eigenvalues (**Reuter et al.** 2006). While in theory this is not a unique signature for the geometry of a shape (**Canzani** 2013), it was shown in practice that many shapes can be reconstructed faithfully from its ShapeDNA (**Cosmo et al.** 2019), and the area of partiality determined **Rampini et al.** (2019). The condition number of shapes captures its difficulty to be robustly matched with other shapes based on symmetry information (**Ovsjanikov et al.** 2011).

1.3.3.2 Quadratic Assignment Problems

The Quadratic Assignment Problem (QAP) arises when looking for a permutation that preserves pairwise descriptors in the optimal way. Assuming both shapes are samples with the same number of vertices and given a pairwise descriptor matrix $D \in \mathbb{R}^{n \times n}$, the QAP can be written the following way:

$$\min_{P \in \mathcal{P}_n} \|D_{\mathcal{X}} - P^\top D_{\mathcal{Y}} P\|_2^2 \tag{1.1}$$

QAPs have been studied extensively in literature, both in general and for shape correspondence (**Berg et al.** 2005). There exist many variants, for example the Quadratic Assignment Matching (**Kezurer et al.** 2015), but all of them have been shown to be NP-hard (**Burghard**

and Klein 2017). Furthermore, even finding an ϵ-approximation of any QAP can only be done in polynomial time if P=NP (**Sahni and Gonzalez** 1976).

QAPs are especially popular for isometric shape correspondence because setting D to be the geodesic distance matrix has a zero energy solution exactly when the inputs are isometric. Since 3D shapes are normally discretized with more vertices than feasible for solving a QAP exactly, there exist a variety of relaxations and approximation algorithms. The most straightforward relaxation is replacing the permutation with a doubly-stochastic matrix (DS) for which holds $P\mathbf{1} = P^\top\mathbf{1} = \mathbf{1}$ (**Gold and Rangarajan** 1996). The set of doubly-stochastic matrices is the convex hull of permutation matrices, and variants that are provably tight exist as shown by **Dym et al.** (2017) and **Bernard et al.** (2018). Another popular relaxation called spectral relaxation was proposed in **Leordeanu and Hebert** (2005) and reduces the solution to an eigenvector problem but is prone to produce noisy solutions. **Rodolà et al.** (2013) added a L_1-norm constraint to this approach which favors sparse solutions, leading to few but robust matches.

Instead of relaxing the permutation constraint, another line of work aims at approximating the solution to the QAP in different ways. **Vestner et al.** (2017) proposed to solve the QAP including Gaussian kernels with a series of Linear Assignment Problems (LAPs) leading to very smooth results. However, even if LAPs are not NP-hard, their complexity is still close to cubic for general matrices and, therefore, not feasible for high resolution meshes. In **Vestner* et al.** (2017) we extended this work to a multi-scale approach that can handle high resolutions and inconsistent meshes, and, in addition, has improved theoretic properties. **Sahillioglu** (2018) and **Edelstein et al.** (2020) apply variants of genetic algorithms for QAPs which are specifically designed to tackle shape correspondence problems.

1.3.3.3 Functional Maps

The functional map framework was introduced in **Ovsjanikov et al.** (2012) and formulated the correspondence problem as low-dimensional linear system which maps functions on functions instead of points to points. Given two sets of compatible pointwise descriptor functions F, G and two function bases Φ, Ψ on $\mathcal{X}, \mathcal{Y}$ respectively, the optimal functional map C can be found by minimizing:

$$C^* = \arg\min_C \|C\Phi^{-1}F - \Psi^{-1}G\|_F^2. \tag{1.2}$$

Assuming the sets Φ, Ψ each contain k basis functions, C is a $k \times k$ matrix and $\Phi^{-1}F, \Psi^{-1}G$ are each $\mathbb{R}^{k\times l}$ where l is the number of descriptor functions. If Φ, Ψ are pointwise indicator functions, which can be represented by a identity matrix (ignoring area weights for a moment), their inverse is also the identity and this formulation is equivalent to a Linear Assignment Problem because C becomes a permutation matrix. Instead of indicator functions **Ovsjanikov et al.** (2012) proposed to use the first k Laplace-Beltrami (LB) eigenfunctions for Φ, Ψ. In principle functional maps simply do a basis change from an indicator basis to the LB eigenfunctions in the linear assignment:

$$CA - B = C\Phi^{-1}F - \Psi^{-1}G = \underbrace{\Psi C \Phi^{-1}}_{=P} F - G \tag{1.3}$$

If Φ, Ψ are suitable the permutation P can be represented as C without any information loss. If $k << n$ there is likely to be some loss, but this is the trade-off for having a lot less dimensional optimization problem. Part of the problem is then shifted to extracting the correct pointwise correspondence from C (**Rodolà et al.** 2015), but there are also applications were a functional map is as useful as a pointwise map (**Rustamov et al.** 2013). Additionally, the LBO eigenfunctions are invariant (up to sign) under isometries. In theory results in only needing to find the sign flips in the basis functions, i.e. C is a diagonal matrix with 1 and -1 as entries. Pure isometries basically never exist, even on artificial data, but the result will still be approximately diagonal which is used for penalization during optimization (**Ovsjanikov et al.** 2012; **Rodolà et al.** 2016).

The low dimension and linearity of functional maps has made it a popular tool for shape correspondence, and the framework been extended with various regularizations and adapted for different applications and settings. **Rodolà et al.** (2016) used the special behavior of the LBO eigenfunctions on partial shapes, namely that C has a slanted instead of a straight diagonal, to improve results on partial isometries. This gave rise to applications like non-rigid puzzles where several non-rigidly deformed parts of an object are assembled to a whole (**Litany et al.** 2016). In **Q.-X. Huang et al.** (2014) functional maps are used to consistently match large collections of shapes; again, a setting that greatly benefits from dimensionality reduction. Functional maps have even been applied to image segmentation problems in **F. Wang et al.** (2013) although this did not take over. Due to the low dimension and differentiability, functional maps are also one of the key building blocks for non-rigid correspondences in deep learning (see Section 1.3.3.5).

1.3.3.4 Physical Deformation Models

Physical models are often used in shape interpolation (**Heeren et al.** 2012) or simulation (**Bender et al.** 2015) to determine how shapes behave realistically and what are low energy movements. If two shapes in different poses are given, it is reasonable to assume that the deformation that leads to the correspondences with the least energy needed to align both is a good correspondence. This was explored in **Windheuser et al.** (2011a) and **Windheuser et al.** (2011b) where the authors discretize the notion of diffeomorphism for triangular meshes, and use bending and stretching energy to find the lowest energy diffeomorphism that results in the final correspondence. A similar energy was used by **Ezuz et al.** (2019) with a sub-vertex placement of the correspondence instead of edge collapse. Also using a physical deformation energy, **Bernard et al.** (2020) formulated the correspondence problem as a convex integer linear program which results in a sparse set of matches. Instead of bending and stretching energy, **Eisenberger et al.** (2019) defined a basis for all volume-preserving deformation fields and solved for the optimal correspondence within this constraint.

1.3.3.5 Learning Methods

Learning has found its way into 3D non-rigid correspondences lately. As discussed in Section 1.3.3.1 a large collection of works focus on learning optimal descriptors. One of the first papers to apply learning as the matching pipeline itself was **Rodolà et al.** (2014a) which utilizes random forests to learn how to form correspondences between different classes of shapes. In **Groueix et al.** (2018) the correspondence is learned through learning the optimal parameters of a template. Functional maps has been incorporated into many learning pipeline because it is low-dimensional and fully differentiable. This is done by providing example maps in **Litany et al.** (2017a) but has been extended to an unsupervised approach using geodesic distances in **Halimi et al.** (2019), using properties of the functional map matrices in **Roufosse et al.** (2019) and cycle-consistency (**Ginzburg and Raviv** 2019). A completely different approach is taken by **Zhu et al.** (2017) where a shape space is learned to judge the plausibility of a correspondence, and then sub-parts of the shapes are hierarchically matched through deformation energies.

CHAPTER 2

Contribution

This chapter will line out the contributions made in this thesis, my research contributions in publication during my PhD candidacy and how these topics relate to each other.

2.1 Major Contributions

The main topic of this thesis focuses on calculating continuous correspondences between non-rigid shapes of varying dimensionality. The standard assumption for such cases is one of a diffeomorphism between the given inputs. Unfortunately, there are few properties or descriptors that are preserved under general diffeomorphisms, and even very restrained subclasses like isometries still pose a NP-hard problem. The major contributions in thesis are several efficient algorithms tackling variations of this continuous non-rigid correspondence problem and are laid out in Chapters 7-10. Except in Chapter 10 the optimal solution is explicitly or implicitly formulated through minimal surface submanifolds in product space, an interpretation that is known to represent a diffeomorphism between the inputs. Chapter 10 looks at the same problem from the viewpoint of deformation fields and how in many cases a continuous correspondences comes from a continuous deformation.

Chapter 7 A very challenging case considers input shapes that are of different dimensionality, *i.e.* the source shape is assumed to be a slice of the target shape. Theoretically this can be modeled through a local diffeomorphism but in practice more problems arise. First, a non-local diffeomorphism implies a bijective relation which is often implicitly or explicitly modeled but does not transfer to this case. Second, a set of comparable descriptors on both shapes is the backbone of basically any correspondence method, either to guide the optimization or at least to obtain a good initialization. We solve the setting with a given 2D and 3D shape, the most common dimensions for geometric data, and provide a provably continuous solution utilizing a shortest path algorithm on the product graph. The global optimum can found in polynomial time, but we propose an additional approximation scheme that improves the runtime in practice even more. Additionally, we show that spectral descriptors, which are popular for non-rigid shape analysis due to their invariant under isometric deformations, can also be used for comparison of 2D and 3D shapes. Because the 2D shape is a slice of the 3D shape, we can treat it as a nearly isometric part and show that it can be used to match 2D and 3D shapes in which the 2D shape is not a projection of the same pose into 2D. This is the first method that can tackle this setting.

Chapter 8 introduces an efficient non-rigid correspondence method for 3D shapes. The method combines a theoretically sound relaxation of the underlying quadratic assignment problem (QAP) with an efficient multi-scale approach. We show that that our relaxation preserves the global optimum of the original problem using the positive definite property of heat kernels and can be interpreted as an approximation of a minimal surface correspondence on the product manifold which represents a diffeomorphism. The QAP in general poses a NP-hard problem and its optimization usually does not scale to high resolutions. However, our proposed multi-scale approach separates the problem into smaller ones while propagating the same continuity information through all scales. Each scale solves a smaller QAP through difference of convex functions (DC) programming in a series of linear assignment problems. The algorithm produces state-of-the-art results on several isometric datasets but can also tackle non-isometric pairs and topological changes, cases that most state-of-the-art isometric methods fail in.

Chapter 9 considers the same setting between two 3D shapes but the focus lies on analyzing the relationship between maps and their representation on the product manifold further. We show that the coefficients used in the functional map framework are not optimal in terms of low-dimensional representation by looking at the corresponding product eigenfunctions and their coefficients. The separable relationship of product properties from their source manifolds can be used to transfer many traditional methods into an equivalent representation on the product manifold and look at their properties there. This observation leads to our derivation of non-separable, localized harmonics in the product space that are better suited to represent accurate map information with as few parameters as possible. We show the applicability of the new basis in a framework for refining correspondences directly on the product manifold.

Chapter 10 approaches the same problem from a different angle. Instead of preserving properties in the correspondence or explicitly modeling continuity in correspondence space, we relate the correspondence to a volume-preserving deformation aligning both shapes. This is based on the observation that pose changes in the real world do not happen through arbitrary deformation but through a sequence of intermediate shapes that all similar to the source and target. To that end, we jointly solve for the correspondence and a smooth, volume-preserving deformation field aligning these correspondences. This can be done with an expectation-maximization optimization alternating between the correspondence and the deformation parameters which come from a low-dimensional basis representing all volume-preservation deformation fields. We show that this basis has a closed-form solution, which means our deformations do not have any discretization artifacts. Volume-preservation is related to but not the same as isometric, however in the real world most objects approximately preserve their volume when moving. The resulting algorithm is efficient even for very high resolution shapes, because the optimization does not need to be done on all points, and the closed-form deformation field can still produce a correspondence and an interpolation sequence on the full resolution with only linear computational overhead.

2.2 List of Publications

This is a list of all my publications that were published during my PhD. Section 2.2.1 lists all publications that were used in this thesis and Section 2.2.2 contains all additional publications including short summaries.

2.2.1 Publications in this Thesis

The following publications are the basis for this thesis.

- **Z. Lähner**, **E. Rodolà**, **F. R. Schmidt**, **M. M. Bronstein**, and **D. Cremers** (May 2016b). Efficient Globally Optimal 2D-to-3D Deformable Shape Matching. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*
- **M. Vestner***, **Z. Lähner***, **A. Boyarski***, **O. Litany**, **R. Slossberg**, **T. Remez**, **E. Rodolà**, **A. M. Bronstein**, **M. M. Bronstein**, **R. Kimmel**, and **D. Cremers** (Oct. 2017). Efficient Deformable Shape Correspondence via Kernel Matching. In: *International Conference on 3D Vision (3DV).*
  Authors with * contributed equally.
- **E. Rodolà**, **Z. Lähner**, **A. M. Bronstein**, **M. M. Bronstein**, and **J. Solomon** (2019). Functional Maps Representation on Product Manifolds. In: *Computer Graphics Forum (CGF)* 38.1.
- **M. Eisenberger**, **Z. Lähner**, and **D. Cremers** (2019). Divergence-Free Shape Correspondence by Deformation. In: *Computer Graphics Forum (CGF)* 38.5.

Chapter 7 is based on **Lähner et al.** (2016b). There we solve for a provable continuous correspondence between a 1D and a 2D manifold using spectral descriptors that are both suited for non-rigid deformations as well as the different dimensions between the inputs. In **Vestner* et al.** (2017), the basis for Chapter 8, we calculate correspondences between two 2D manifolds and incentivize smoothness in the correspondences through heat kernels which are efficient to compute and have beneficial theoretical properties when solving a QAP. Chapter 9 explores the relation between representing maps and the product manifold of the inputs based on **Rodolà et al.** (2019). Lastly, we look at continuity from a slightly different perspective in Chapter 10. This is based on **Eisenberger et al.** (2019) where we calculate divergence-free deformation fields between shapes in order to compute a correspondence.

2.2.2 Other Publications

I published the following additional publications throughout the duration of my PhD, but they are not part of this thesis due to having a different focus.

- **Z. Lähner**, **E. Rodolà**, **M. M. Bronstein**, **D. Cremers**, **O. Burghard**, **L. Cosmo**, **A. Dieckmann**, **R. Klein**, and **Y. Sahillioglu** (May 2016a). SHREC16: Matching of Deformable Shapes with Topological Noise. In: *Eurographics Workshop on 3D Object Retrieval (3DOR).*
- **Z. Lähner**, **D. Cremers**, and **T. Tung** (Sept. 2018). DeepWrinkles: Accurate and Realistic Clothing Modeling. In: *European Conference on Computer Vision (ECCV).*
- **M. Eisenberger**, **Z. Lähner**, and **D. Cremers** (2020). Smooth Shells: Multi-Scale Shape Registration with Functional Maps. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

In **Lähner et al.** (2016a), we published a dataset on non-rigid correspondence with topological noise and organized a SHREC contest evaluating the performance of submitted methods under these perturbations. This noise is modeled after scanning noise in real-world object that occurs when two parts of it that are actually separated are touching. Since the scanning hardware can only perceive the visible surface, it is nearly impossible to deduct that touching parts should be separated in the 3D model. The dataset is based on artificial models and therefore we have very accurate ground-truth correspondences, but the topological changes are more severe than in other datasets of this kind. The results showed a huge gap to the performance of state-of-the-art methods on clean non-rigid datasets. To this day no method was shown to achieve the same quality of results on this dataset as is state-of-the-art on non-topologically perturbed, artificial data. The effects of topological perturbations are discussed in Section 6.2.2, but this paper was not included in the thesis because it focuses on the dataset and contest only.

Modeling and learning very fine wrinkles on clothing is the topic of **Lähner et al.** (2018). Previous methods focused on learning wrinkles as part of the 3D mesh. This can be done accurately and cleanly by physical simulation, but representing really fine, realistic wrinkles requires a very high resolution mesh which in turn leads to high computation costs and demanding large amounts of training data. Instead, our approach is to separate the wrinkles in low and high frequency. The lower frequency wrinkles are represented as vertex offsets of the 3D triangular mesh, and learned as a statistical model. This is a standard representation in clothing animation and similar to **Pons-Moll et al.** (2017). In contrast to other methods, we represent the fine, high frequency wrinkles as normal maps. Normal maps are images that are mapped onto the surface via a UV map and each pixel gives a normal direction that is used for rendering instead of the mesh normal. The advantages of normal maps are that images can be processed efficiently at a much higher resolution than meshes, and applying neural networks to images is also more advanced than on 3D data. Using this representation we were able to train a LSTM (**Hochreiter and Schmidhuber** 1997) to predict the coefficients for the low frequency wrinkles and a cGAN (**Isola et al.** 2017) generating a normal map with fitting high frequency details. Because we moved the high frequency details to an image, we were able to produce much more detailed results than state-of-the-art at nearly real-time speed.

In **Eisenberger et al.** (2020) we explore the idea of aligning shapes by starting with very coarse geometric cues, and then adding and aligning more fine-scale details. To this end, we introduce a concept called Smooth Shells which can produce a sequence of smoothed versions of an input shape. The two main properties of Smooth Shells are: 1) the amount of detail increases constantly within the sequence, and 2) the gap between two consecutive shapes in the sequence is minimal. This is incredibly useful for iterative methods which use the result of the previous iteration as initialization for the next. Additionally, we represent the alignment of two input shapes as a deformation function projected onto the Laplace-Beltrami eigenfunctions. This allows us to efficiently sample from the space of deformations, and keep the optimization fast when high frequency information in the deformation is not needed. Based on this we propose a framework that is able to match isometric and non-isometric shapes as long as their rough geometric shapes are similar.

PART II

Theoretical Background

One geometry cannot be more true than another; it can only be more convenient. Geometry is not true, it is advantageous.

– Robert M. Pirsig

CHAPTER 3

Differential Geometry

Although the world is filled with volumetric objects, normally only the surface is perceptible through vision. Therefore, a reasonable and compact representation of objects is based on their surface with the interior being implied if the surface is closed. Another advantage is that if objects are only seen from one view, it is easier to represent partial surfaces instead of having to guess volumetric properties. Two-dimensional manifolds are often used to describe physical objects, because they exactly model this surface. This chapter gives an introduction into (differentiable) manifolds and basic differential geometry, which is the standard tool for handling these manifolds. In this work we restrict ourselves to Riemannian manifolds, differentiable and with a metric attached. Additionally, the theory is only discussed for manifolds without boundaries, except for very special cases in later chapters, to keep the math simple. **do Carmo** (1976) and **do Carmo** (1992) serve as the basis for this chapter and further details can be found in the books.

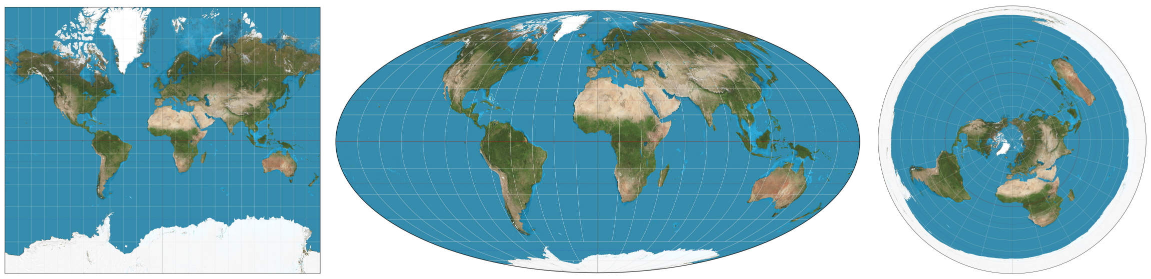


Figure 3.1: Three different world maps. The white grids indicate how the elements are distorted from one map to the other, however, none of them completely preserves the real distance between all places on earth. Images by **Strebe** (2011).

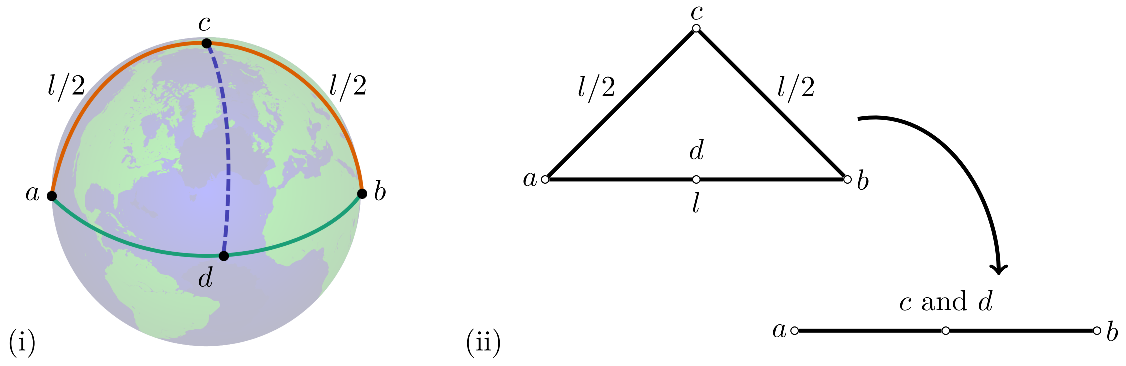


Figure 3.2: Exampe of the Cartographer's Dilemma. (i) The points a, b lie on opposite sides of the equator and c on the northpole. The shortest path between a, b has length l, but there are multiple (in fact infinite many) shortest paths between them. One of those goes through the north pole c which lies exactly in the middle, and one goes over the equator with d in the middle. Therefore, the distance from c and d to both a and b is $l/2$ and the distance between c and d is greater zero. (ii) Embedding a, b, c, d into 2D (a map) is impossible while maintaining all distances measured on the sphere.

3.1 Manifolds

A manifold of dimension d is a topological space that is locally identical to the Euclidean space $\mathbb{R}^d$. Locally identical means that the close neighborhood of each point on the surface is homeomorphic to an open subset of $\mathbb{R}^d$, but the similarity usually fades with increased distance. The implications become clear from looking at different world maps: The surface of our planet is 2-dimensional and parts of it can be well represented on a piece of paper. Small excerpts of the globe can be projected onto a flat map with limited distance distortions, but the entire world can only be represented by introducing a cut somewhere (often at the poles). Additionally, for most pairs of places the distance measured with a ruler on a flat map will not be consistent with the distance measured on a globe. See Figure 3.1.

This leads to what is known as the *Cartographer's Dilemma*. It is impossible to preserve the surface distance between all points on the globe when it is converted into a flat map. This is most obvious when looking at the following four points on a stylized, spherical globe: two of them a, b on opposite sides of the equator, d on the equator in the middle of a and b, and the last point c on the north pole. The shortest path between a and b can go along the equator as well as through the north pole. Both have the same length, let it be l. See Figure 3.2 for a visualization. Since c and d are exactly in the middle of a and b along a shortest path with length l, their distance to both a and b is $\frac{l}{2}$. Furthermore, c does not lie on the equator, but d does, so their distance is at least greater than zero. When trying to embed a, b, c, d into a plane while maintaining their original distances to each other, the uniqueness of the shortest path in Euclidean space leads to the fact that both c and d have to lie in the middle of the straight line connecting a and b. This means they have to have the same coordinates and, therefore, distance zero. This violates the assumption that they are not the same point and have positive distance. On standard world maps (Figure 3.1 left) the highest distortion is close to the north and south poles – probably the places that are the least relevant to most people (**Milnor** 1969). This visualizes that, while Euclidean geometry is suitable to represent manifolds locally, their behavior is not entirely the same and motivates why differential geometry is important.

One solution for calculating accurate distances from maps is to use multiple maps instead of just one and keeping track of which distortions are to expected in each map. If it is known how and where a map is distored, the distortion can be inverted in order to retrieve the original distance. Traditional globes are made by drawing maps on long, thin slices of paper which are glued on the globe. In order to get all pieces to fit together perfectly onto the globe, the slight curving of the slices has to be taken into account while drawing. The same idea is applied in differential geometry. Properties are calculated in d-dimensional Euclidean space, but a mapping to a curved d-dimensional surface is known such that – when done correctly – a calculation taking into account the distortion of the mapping will return the result as if it were done on the curved surface directly.

3.1.1 Coordinate Maps

In accordance to the intuition given in the last section, d-dimensional manifolds are defined by mappings from $\mathbb{R}^d$ to parts of the surface of the manifold. These are called coordinate maps and are the building blocks of manifolds. Each describes a part of the surface of the manifold. A coordinate map can be imagined as the process of applying one long paper slice on the spherical globe.

Definition 1. *A* **coordinate map** *is a function $\xi : U \subset \mathbb{R}^m \to \mathbb{R}^n$ with $m < n$ which is a diffeomorphism. U is a connected and open subset of $\mathbb{R}^m$.*

See Figure 3.3 for an illustration of the terms. The surface $S = \xi(U)$ is often called a *regular surface* if ξ is a diffeomorphism. Depth maps are a kind of coordinate map, setting $U = (1 \dots w) \times (1 \dots h) \subset \mathbb{N}^2$ as the image domain and $\xi(U) = (x, y, z)$ where z is the depth. Of course, depth maps are discretized and may not be smooth, but the principle is the same.

Example 1. *Let the unit sphere $\mathbb{S}^2 = \{(x, y, z) \in \mathbb{R}^3 | x^2 + y^2 + z^2 = 1\}$ be an approximation of the globe. A coordinate map of the nothern hemisphere can be given by $\xi_1 : U \subset \mathbb{R}^2 \to \mathbb{R}^3, \xi_1(u, v) = \left(u, v, \sqrt{1 - u^2 + v^2)}\right)$ with $U = \{u, v \in \mathbb{R}^2 | u^2 + v^2 < 1\}$.*

3.1.2 Differentiable Manifolds

One coordinate map may not be sufficient to describe the entire surface, therefore a manifold $\mathcal{M}$ is a collection of coordinate maps that cover the entire surface of $\mathcal{M}$ and are consistent with each other. We define differentiable manifolds according to **do Carmo** (1992) (except condition 3 which is redundant):

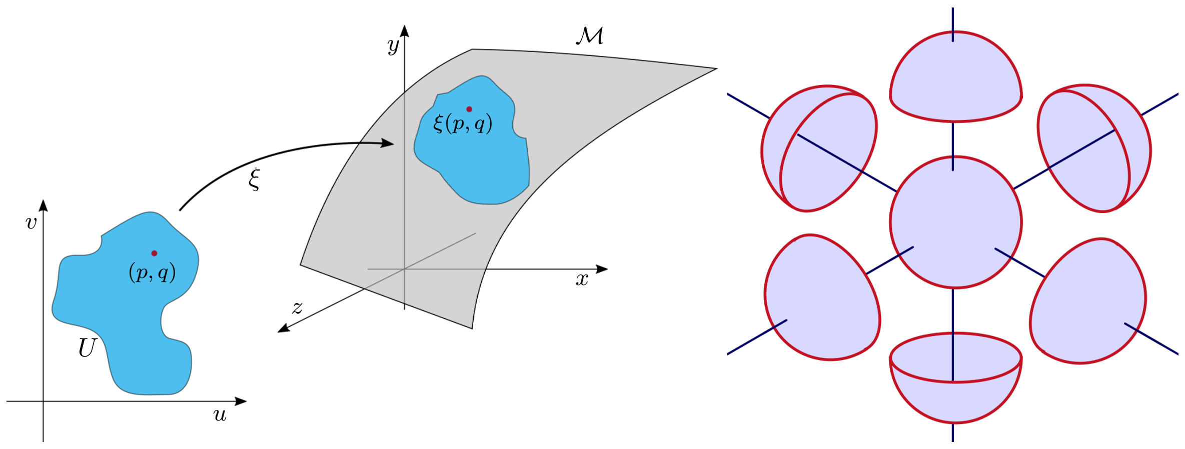


Figure 3.3: (Left) Visualization of a coordinate map. The domain U is a subset of $\mathbb{R}^2$ and $\xi : \mathbb{R}^2 \to \mathbb{R}^3$ maps a point $(p, q) \in U$ to the surface of $\mathcal{M}$ (gray). (Right) Visualization of Example 2. The globe can be covered by six cooordinate maps each covering half of the earth.

Definition 2. *A* **differentiable manifold** *of dimension d is a set $\mathcal{M}$ and a family of injective coordinate maps $x_i : U_i \subset \mathbb{R}^d \to \mathcal{M}$ such that:*

1. *$\bigcup_i x_i(U_i) = \mathcal{M}$*
2. *for any pair i, j with $x_i(U_i) \cap x_j(U_j) = W \neq \emptyset$, the sets $x_i^{-1}(W)$ and $x_j^{-1}(W)$ are open sets in $\mathbb{R}^d$ and the mapping $x_j^{-1} \circ x_i$ is differentiable*

Extending upon Example 1 the entire globe can be covered by the following coordinate maps:

Example 2. *Let the unit sphere $\mathbb{S}^2 = \big\{(x, y, z) \in \mathbb{R}^3 | x^2 + y^2 + z^2 = 1\big\}$ be an approximation of the globe and again $U = \big\{u, v \in \mathbb{R}^2 | u^2 + v^2 < 1\big\}$. Then the manifold of $\mathbb{S}^2$ can be defined through the following collection of coordinate maps, describing a different half of the sphere each:*

$$\begin{aligned} \mathbb{S}^2 = \{\xi_1 &: U \subset \mathbb{R}^2 \to \mathbb{R}^3, \xi_1(u, v) = \Big(u, v, \sqrt{1 - u^2 + v^2}\Big), \\ \xi_2 &: U \subset \mathbb{R}^2 \to \mathbb{R}^3, \xi_1(u, v) = \Big(u, v, -\sqrt{1 - u^2 + v^2}\Big), \\ \xi_3 &: U \subset \mathbb{R}^2 \to \mathbb{R}^3, \xi_1(u, v) = \Big(u, \sqrt{1 - u^2 + v^2}, v\Big), \\ \xi_4 &: U \subset \mathbb{R}^2 \to \mathbb{R}^3, \xi_1(u, v) = \Big(u, -\sqrt{1 - u^2 + v^2}, v\Big), \\ \xi_5 &: U \subset \mathbb{R}^2 \to \mathbb{R}^3, \xi_1(u, v) = \Big(\sqrt{1 - u^2 + v^2}, u, v\Big), \\ \xi_6 &: U \subset \mathbb{R}^2 \to \mathbb{R}^3, \xi_1(u, v) = \Big(-\sqrt{1 - u^2 + v^2}, u, v\Big)\} \end{aligned}$$

See Figure 3.3 for an illustration.

Notice that in this case the two coordinate maps describing the northern and southern hemispheres do not suffice to cover the entire manifold, because their domains are open and the equator is missing. However, the property of openness is needed for differentiability. The example does not use the minimum number of coordinate maps possible (it is two but using different mappings), and from a purely mathematical point of view there is no advantage in using less coordinate maps.

3.1.3 Discretization

It would be possible to actually store manifolds as collections of coordinate maps, but there are many reasons why this is impractical. Other representations that are not technically manifolds by our definition exist, e.g splines, quad meshes, polygon meshes, and we look at two of them in detail.

3.1.3.1 Triangular Meshes

A triangular mesh (V, F) represents the surface as a collection of vertices V and faces $F \subset V \times V \times V$ connecting elements of V, all of which are triangles. Normally additional constraints are put into place to make the surface reasonable and non-degenerate, *e.g.* two vertices can only be part of the same triangle twice. To get close to a manifold definition, each triangle can be interpreted as a coordinate map from a basis triangle domain $U_t = \text{convex}((0,0),(1,0),(0,1))$ to the triangle embedded in 3D. This is not sufficient for a manifold because i) U_t is not open and ii) the triangles only overlap on their edges such that their composition might not be differentiable. Technically triangle meshes are not manifolds, but most quantities can still approximated in a meaningful way on discrete structures, and a lot of operations are very efficient because the surface is a linear interpolation of a subset of vertices at all points.

Hat Functions The hat functions form a basis set on triangular meshes that are similar to indicator functions and lead to piecewise linear functions on the surface that are consistent on the edges connecting two triangles.

$$h_i(x) = \begin{cases} 1 & \text{if } x = v_i \\ 0 & \text{if } x = v_j, i \neq j \\ u \cdot h_i(v_1) + v \cdot h_i(v_2) + t \cdot h_i(v_3) & \text{if } x = u \cdot v_1 + v \cdot v_2 + t \cdot v_3, \\ & \quad u + v + t = 1, \\ & \quad (v_1, v_2, v_3) \in F \end{cases} \tag{3.1}$$

The hat functions are often used as the basis functions in finite element methods. For example, they are used to derive the discrete Laplace Beltrami operator in the next chapter.

3.1.3.2 Subdivision Surfaces

Subdivision surfaces use a series of control points to define the surface. The surface is divided and interpolated in an iterative way and the limit of this refinement defines the surface. There exists algorithms that can calculate this limit without the need for infinite iterations, or approximate the limit through parametric patches like Bezier patches. Different from triangle meshes, subdivision surfaces are actually differentiable manifolds but due to the implicit definition it is harder to process them. Even simple operations, like sampling surface points, are not straight-forward, because the control points normally do not lie on the surface. On the other hand, smooth geometry can be represented through far fewer control points than in triangular meshes, without leading to inaccuracies and numerical issues (**Estellers et al.** 2018).

3.2 The Differential and Tangent Spaces

We started the introduction of manifolds by stating that they are locally Euclidean spaces. The tangent space can be imagined as exactly that Euclidean space for each point. In order to define tangent spaces, we use the differential which is similar to coordinate maps but maps vectors to vectors instead of points to points on the surface of $\mathcal{M}$.

Definition 3. *The* **differential** *maps vectors in the domain U to vectors on the surface of $\mathcal{M}$. For a differential manifold $\mathcal{M}$ at point $p \in \mathcal{M}$ with a coordinate map $\xi : U \subset \mathbb{R}^m \to \mathbb{R}^n$ such that $p = \big(\xi^1(q), \xi^2(q), \dots, \xi^n(q)\big)$ for some $q \in U$, it can be calculated as*

$$d\xi_p = \begin{pmatrix} \frac{\partial \xi^1}{\partial d_1} & \cdots & \frac{\partial \xi^n}{\partial d_m} \\ \frac{\partial \xi^2}{\partial d_1} & \cdots & \frac{\partial \xi^2}{\partial d_m} \\ \vdots & \ddots & \vdots \\ \frac{\partial \xi^n}{\partial d_1} & \cdots & \frac{\partial \xi^n}{\partial d_m} \end{pmatrix} \tag{3.2}$$

where $d_1, \dots, d_m$ are the dimensions of the domain $\mathbb{R}^m$. The differential is a linear map.

Since all coordinate maps have to be consistent with each other, the differential will be the same no matter what coordinate map is used to calculated it. Notice that the differential is a general concept from calculus and can be calculated for other kinds of maps, for example, for maps between two manifolds. These will turn vectors from the tangent space of one manifold to the tangent space of another and are often called pushforward instead. But the basic calculations stay the same, only U needs to be replaced with a different domain. For regular surfaces the differential is always full-rank; in this case rank m. Similar to the distortion that the coordinate maps can have on the distances between points, the differential can, and likely will, introduce distortion to properties of vectors, for example the angle between them or their length.

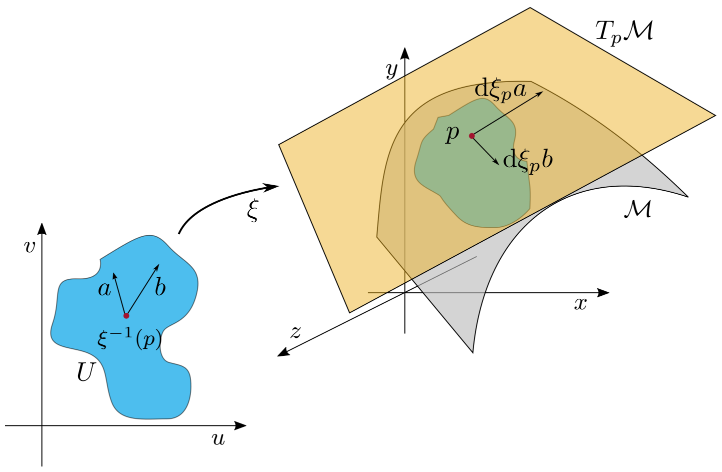


Figure 3.4: Visualization of the tangent space. Each point $p \in \mathcal{M}$ has its own tangent space $T_p\mathcal{M}$ attached. Tangent spaces are vector spaces with the same dimension as $\mathcal{M}$. Any vector $a \in U$ can be mapped to the tangent space by the differential $\mathrm{d}\xi_p(a)$. Because the differential is full-rank, if a, b span U then $\mathrm{d}\xi_p(a), \mathrm{d}\xi_p(b)$ span $T_p\mathcal{M}$.

The tangent space at a point $p \in \mathcal{M}$ is the space spanned by all vectors in the image of all differentials at p. This also means, like the differential, the tangent space is potentially different at each surface point.

Definition 4. *The* **tangent space** $T_p\mathcal{M}$ *of point* $p \in \mathcal{M}$ *in a* m*-dimensional manifold* $\mathcal{M}$ *is a* m*-dimensional vector space that contains all possible directions that can tangentially pass through* p*. If the vectors* $v_1, \ldots, v_m \in \mathbb{R}^m$ *span the domain* U*, the tangential space at* p *is spanned by the vectors* $d\xi_p v_1, \ldots, d\xi_p v_1$*. Since the differential is full-rank the tangent space has the same dimension as* U*.*

For infinitesimal neighborhoods around p the tangent space approximates the behavior of $\mathcal{M}$ very well but not for farther areas. Using the definition of tangent space, it is possible to define the gradient and vector fields on the surface of $\mathcal{M}$.

3.3 Riemannian Manifolds

Previous definitions allow the transfer of points and vectors, but most applications are concerned with properties like distances and areas. In order to be able to calculate these, we need a metric on the surface. As in Euclidean spaces, a metric can be induced by the definition of an inner product.

Definition 5. *We define a* **Riemannian manifold** $(\mathcal{M}, g)$ *to be a differentiable manifold with an inner product* $g_p : T_p\mathcal{M} \times T_p\mathcal{M} \to \mathbb{R}$ *attached to each point* $p \in \mathcal{M}$*.* g_p *can be written as a bi-form and is often called* **metric tensor***.*

As long as the metric tensor provides a proper inner product at each point any metric tensor can be assigned. However, there is a natural form of the metric tensor which arises from the inner product in Euclidean space and is used in most applications involving Riemannian manifolds. It is called the *first fundamental form* and if $u, v \in U$ are two vectors in the domain of a coordinate map ξ then $g_p(u, v) = \langle \mathrm{d}\xi_p u, \mathrm{d}\xi_p v$ where $\langle \cdot, \cdot \rangle$ is the standard inner product in $\mathbb{R}^n$. Inserting $\langle a, b \rangle = a^\top b$ it is easy to see that $g_p = \mathrm{d}\xi_p^\top \mathrm{d}\xi_p \in \mathbb{R}^{m \times m}$ should hold.

The metric tensor allows to calculate properties like angles, areas, and lengths on the surface. For example, the length of a curve $\gamma : [0, 1] \to \mathcal{M}$ can be calculated through:

$$\ell(\gamma) = \int_0^1 \|\dot{\gamma}(t)\| \mathrm{d}t \tag{3.3}$$

$$= \int_0^1 \sqrt{g_p(\dot{\gamma}(t), \dot{\gamma}(t))} \tag{3.4}$$

$\dot{\gamma}$ refers to the derivative of γ. Notice that $\dot{\gamma} \in \mathbb{R}^m U$ here, instead of $\mathbb{R}^n$, because this holds for every metric tensor. The surface $\mathcal{M}$ is defined through coordinate maps, therefore every γ can and should be written by transitioning through coordinate maps. However, since not every curve can be parameterized by a single coordinate map, we skip this part for simplicity. See **do Carmo** (1992) for details.

3.3.1 Geodesic Distance

A geodesic is the generalization of a straight line on Riemannian manifolds. As such, the geodesic distance is the length of the shortest path running over the manifold surface connecting two points. Applying the formula for the lengths of curves from the last section, we can find the geodesic distance by minimizing:

$$d_{geo}(a, b) = \min_{\gamma:[0,1] \to \mathcal{M}, \gamma(0)=a, \gamma(1)=b} \int_a^b \sqrt{g_{\gamma(t)}(\dot{\gamma}(t), \dot{\gamma}(t))} \mathrm{d}t \tag{3.5}$$

Unfortunately, unlike for the Euclidean distance, there is no closed form solution, and calculating geodesic distances is computationally heavy.

3.3.2 Isometries

There are many classes of mappings between manifolds (and we will see a more in-depth discussion in Chapter 6) but the one we use the most are isometries.

Definition 6. *A manifold $\mathcal{M}$ is called a* **isometric** *to the manifold $\mathcal{N}$ if there exists a diffeomorphism $\varphi : \mathcal{M} \to \mathcal{N}$ such that it preserves the metric tensor at each point. In this case φ is called an* **isometry***.*

It is easy to see that this is equivalent to preserving the geodesic distance between all pairs of points. We will use this class later one because many deformable objects, e.g. humans, deform in an isometric way. Additionally, the correspondence problem for isometric shapes is NP-hard, in contrast to, for example, rigid cases.

3.4 d-Dimensional Shapes

We define a *d-dimensional shape* to be the outer shell of an object in $\mathbb{R}^d$, this leads the surface of the shape to be a manifold of dimension $d-1$. The object itself will be referred to as the shape's *solid*. For example, the 3D ball $B = \{x \in \mathbb{R}^3 | \, \|x\| \leq 1\}$ is the solid of the sphere $\mathbb{S}^2 = \{x \in \mathbb{R}^3 | \, \|x\| = 1\}$. The following definition summarizes this convention:

Definition 7. *A compact set $S \subset \mathbb{R}^d$ is called a* shape *of dimension d if it is a connected, differentiable manifold and if it can be represented as the boundary $S = \partial U$ of an open subset $U \subset \mathbb{R}^d$. In this case, we call U the* solid *of S.*

Note that this definition implies that a 3D shape is a 2-dimensional manifold and a 2D shape is a 1-dimensional manifold, and shapes cannot have self-intersections or boundaries.

CHAPTER 4

Spectral Shape Analysis

Spectral shape analysis (or spectrum analysis) refers to utilizing properties derived from the spectrum or frequencies of a shape. This is closely related to Fourier analysis where functions are approximated by sums of periodic functions. The functions used in shape analysis are usually the eigenfunctions of the Laplace-Beltrami operator (LBO), which have similar properties. They form a basis for the space of L^2-functions on the surface of the manifold, are frequency ordered and can be used to do bandwidth filtering.

The Laplace-Beltrami operator has a number of useful properties for non-rigid settings. It was even called the "Swiss-army tool for Shape Analysis" (**Solomon et al.** 2014a). One of its properties is that it is invariant under isometric deformations (see Section 3.3.2). As a result methods based purely on the LBO cannot distinguish between different embeddings of the same manifold. This is useful because even extreme pose changes are invisible to the LBO, a setting other methods often struggle with, but isometries also include self-symmetries which leads to LBO based method often having symmetric flips.

4.1 The Laplace-Beltrami-Operator (LBO)

The Laplace-Beltrami operator is a generalization of the Laplace operator but on Riemannian manifolds. The curved surface of manifolds influences the behavior and has to be taken into account. This makes the computation slightly more complicated. But as in the Euclidean space, the Laplace-Beltrami operator calculates the second derivative and is defined as the divergence of the gradient of a function:

$$\Delta f = \operatorname{div}(\nabla f) \tag{4.1}$$

Both divergence and gradient are of course also generalizations for curved spaces instead of the Euclidean version.

4.1.1 Properties

The Laplace-Beltrami operator inherits many properties of the standard Laplace operator, but additionally adds some that are specific for Riemannian manifolds, or simply trivial in the Euclidean case. This section only includes properties that are used in later chapters and is not comprehensive.

Linearity. The LBO is a linear operator which means for all functions $f, g \in L^2(\mathcal{M}), a \in \mathbb{R}$ holds:

$$\Delta(a \cdot f + g) = a \cdot \Delta f + \Delta g \tag{4.2}$$

It follows that it can be written in matrix form.

Self-Adjointness. An operator is self-adjoint if it is its own adjoint operator which means

$$\langle \Delta_{\mathcal{M}} f, g \rangle_{\mathcal{M}} = \langle f, \Delta_{\mathcal{M}} g \rangle_{\mathcal{M}} \tag{4.3}$$

holds. For the LBO this holds as long as $\mathcal{M}$ has no boundary.

Positive Semi-Definiteness. The LBO is positive semi-definite which means that $\langle \Delta f, f \rangle \geq 0$ for all $f \in \text{domain}(\Delta)$.

Invariance under Isometry. Without going into too much detail the gradient and divergence are both intrinsic operators, therefore, their composition, which is the LBO, is also intrinsic. Intuitively, the LBO is a generalization of the adjacency matrix, which is completely embedding variant. Additionally, we will see in the next section that the discrete LBO can be written only using the edge lengths in a triangular mesh, and the preservation of all edge lengths follows naturally from isometries.

4.1.2 Discretization

Many different ways to discretize the LBO exist, and **Wardetzky et al.** (2007) have shown that no discretization can fulfill all properties of the continuous Laplacian. Therefore, choosing a discretization is dependent on the application and which property is the least important for it. In this thesis we will use the cotan-discretization introduced in **Pinkall and Pothier** (1993). It does not fulfill the maximum principle which means that harmonic functions with $\Delta f = 0$ in the interior of $\mathcal{M}$ have no local optima at interior points. This is not important in later applications, thus we ignore it.

The cotan LBO can be derived through finite elements with hat functions (see Section 3.1.3.1) as the basis set. This results in a mass matrix A and a stiffness matrix S, which can be combined into the Laplacian $L = A^{-1}S$. The mass and stiffness matrix take the following form, see Figure 4.1 for the notation:

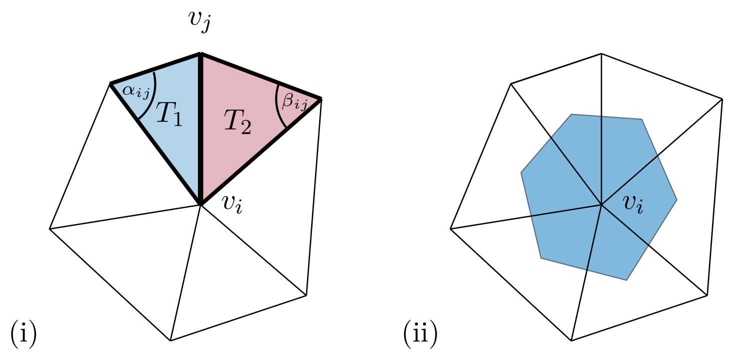


Figure 4.1: Visualization of notation for the cotan matrices. (i) The stiffness matrix takes into account the angles opposite to the edge e_{ij} and T_1, T_2 refer to the triangles on both sides of e_{ij} in the mass matrix. (ii) The lumped mass matrix weights each vertex with the area of the surrounding triangles. Each of the three vertices of a triangle gets assigned one third of the area.

$$S_{ij} = \begin{cases} \frac{\cot(\alpha_{ij})+\cot(\beta_{ij})}{2} & \text{if } (i,j) \in \mathcal{E} \\ -\sum_{k\in N(i)} C_{ik} & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \tag{4.4}$$

Using the following equality

$$\cot(\alpha_{ij}) = \frac{\ell_{jk}^2 + \ell_{ki}^2 - \ell_{ij}^2}{4\text{area}(T_1)} \tag{4.5}$$

where $\ell_{xy} = \|v_x - x_y\|_2$ the edge length between two vertices and v_k is the third vertex in T_1, we can easily write S knowing only the edge lengths.

$$A_{ij} = \begin{cases} \frac{\text{area}(T_1)+\text{area}(T_2)}{12} & \text{if } (i,j) \in \mathcal{E} \\ -\sum_{k\in N(i)} \frac{\text{area}(T_k)}{6} & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \tag{4.6}$$

Again the area can be calculated through edge lengths, making the discrete LBO completely derivable from the edge lengths only. Often the mass matrix A is replaced with a *lumped* version, in which

$$A_{ij} = \begin{cases} \sum_{k\in N(i)} \frac{\text{area}(T_k)}{3} & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}. \tag{4.7}$$

This is faster to invert and leads to localization of L but is not consistent with the finite element derivation. Other methods, for example taking into account the area of voronoi cells, exist for the lumped mass matrix but this is the fastest to compute. Some of these computations cannot be done if the mesh is non-manifold because some entries in the stiffness matrix are not properly defined. Recently, **Sharp and Crane** (2020) introduced an extension for calculating the LBO on non-manifold meshes with robust properties using a so-called tufted cover.

4.2 Eigendecomposition

The eigendecomposition (or spectral decomposition) of the LBO is what gives spectral shape analysis its name. Let $\Delta_{\mathcal{M}}$ be the LBO on $\mathcal{M}$ and

$$\Delta_{\mathcal{M}}\phi_i = \lambda_i \phi_i \tag{4.8}$$

then $\phi_i \in L^2(\mathcal{M})$ are the Laplace-Beltrami eigenfunctions with corresponding eigenvalues λ_i. We saw in Section 4.1.2 how the LBO can be written as the product $A^{-1}S$ of the mass and stiffness matrix. The eigendecomposition can also be done through the generalized eigenvalue problem, which is numerically more stable:

$$S\phi = \lambda M\phi. \tag{4.9}$$

4.2.1 Eigenvalues

As mentioned above, because of the semi-positiveness of the LBO, its eigenvalues are real and non-negative.

Real Semi-Positive Spectrum. The LBO is positive semi-definite and therefore has a real, semi-positive spectrum. Technically, there appears a minus sign in the derivation which is propagated into the operator. There exists literature with the minus left out, therefore, the LBO is negative semi-definite sometimes but we use the positive definition here.

$$0 = \lambda_0 \lambda_1 \leq \lambda_2 \leq \cdots \to \infty \tag{4.10}$$

Weyl's law. Weyl's law (**Weyl** 1911) describes how the growth of magnitude of eigenvalues is linear with the incline depending on the area of the shape:

$$\lambda_j \approx \frac{\pi}{\text{area}(\mathcal{M})} j \quad \text{for } j \to \infty \tag{4.11}$$

This is of course just an approximate relationship.

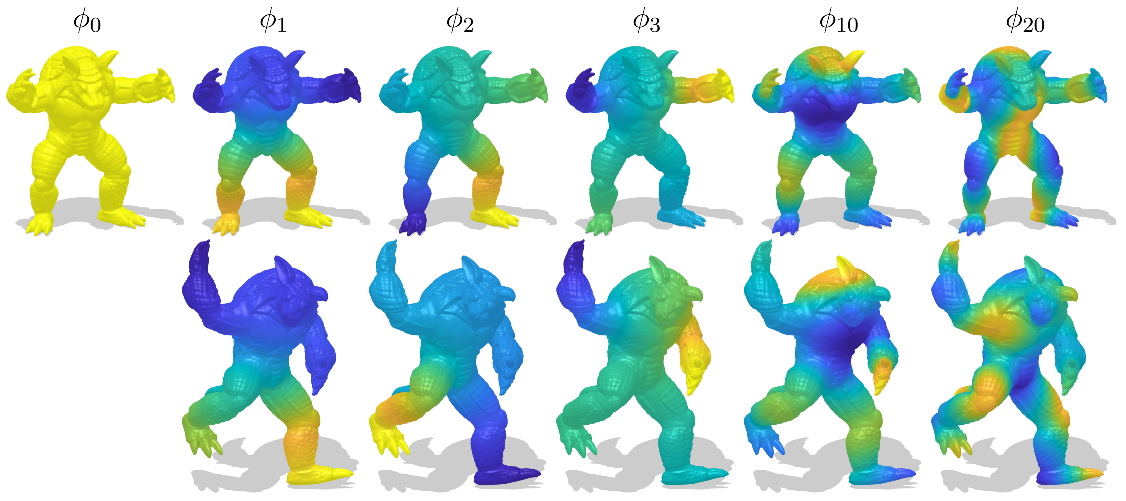


Figure 4.2: The same eigenfunctions on two approximately isometric Stanford armadillos. The eigenfunctions are the same up to sign flips (yellow and blue swapped here). Higher frequency functions are more sensitive to small inaccuracies in the isometry assumption leading to some distortions in ϕ_{10} and ϕ_{20}.

4.2.2 Eigenfunctions

The eigenfunctions inherit a natural ordering from their corresponding eigenvalues.

Orthogonality. It follows from the self-adjointness that the eigenfunctions are orthogonal to each other and form a basis set.

$$\langle \Delta\phi_i, \phi_j \rangle = \langle \Delta\phi_j, \phi_i \rangle \tag{4.12}$$

$$\Leftrightarrow \lambda_i \langle \phi_i, \phi_j \rangle = \lambda_j \langle \phi_j, \phi_i \rangle \tag{4.13}$$

$$\Rightarrow \langle \phi_i, \phi_j \rangle = \langle \phi_j, \phi_i \rangle = 0 \tag{4.14}$$

The basis is for the set of square-differentiable functions $L^2(\mathcal{M})$.

Dirichlet energy. The eigenvalue λ_i describes exactly the Dirichlet energy of each corresponding eigenfunction ϕ_i:

$$\int_{\mathcal{M}} \|\nabla\phi_i(x)\|^2 \mathrm{d}x = \int_{\mathcal{M}} \langle \nabla\phi_i(x), \nabla\phi_i(x) \rangle \mathrm{d}x \tag{4.15}$$

$$= -\int_{\mathcal{M}} \phi_i(x)(-\Delta)\phi_i(x) \mathrm{d}x \tag{4.16}$$

$$= \lambda_i \int_{\mathcal{M}} \phi_i(x)\phi_i(x) \mathrm{d}x \tag{4.17}$$

$$= \lambda_i \tag{4.18}$$

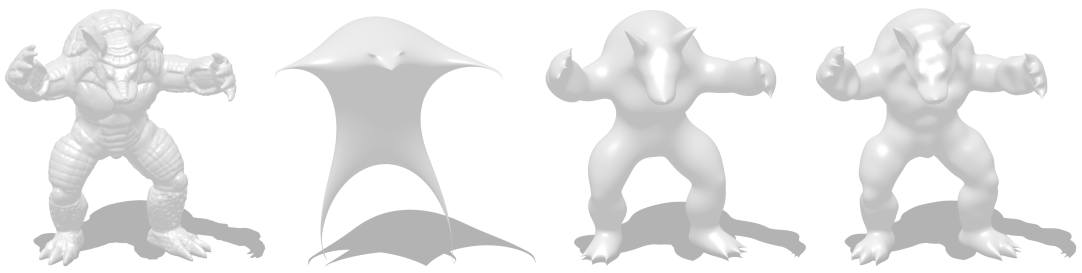


Figure 4.3: Spectral reconstruction of the left-most armadillo with an increasing number of eigenfunctions. From left to right: The original shape and using $10, 250, 1000$ eigenfunctions for the spectral reconstruction.

We have to use $-\Delta$ here because of our semi-positive definite definition above. Only constant functions have Dirichlet energy zero, it directly follows the first eigenfunction corresponding to $\lambda_0 = 0$ is constant.

Optimal Basis Set. The eigenfunctions are not only a basis for their function space $L^2(\mathcal{M})$ but it can be shown that the set of the first k eigenfunction is the optimal choice for a basis set with magnitude k in sense of representation error (**Aflalo et al.** 2015a). This explains why, when a dimensionality reduction is required, a function f is often represented in as the coefficients $a = \Phi^\top Af$ in the first k eigenfunctions.

$$\tilde{f} = \sum_{i=0}^{k} a_i \phi_i \tag{4.19}$$

If f can be completely arbitrary, $\tilde{f}$ is the optimal k-dimensional representation.

Low Pass Filter. The increase of the Dirichlet energy can also be used in applications where a low pass filter is required. Because the low frequency eigenfunctions come first, projecting f on a certain k will produce a low pass filtered version of f. The higher f the more high frequency information is added. This is, for example, applied in spectral reconstruction to produce a smoothed out version of a shape by projecting the coordinate functions onto k eigenfunctions and back, see Figure 4.3.

CHAPTER 5

Product Spaces

The product $\times$ of two structures is the combination of all their elements while preserving the intial properties, for example neighborhood information. Most commonly known is probably the *Cartesian product* of two sets

$$A \times B = \{(a, b) \mid a \in A \text{ and } b \in B\}. \tag{5.1}$$

This produces a set which includes a tuple (a, b) for any combination of elements $a \in A$ and $b \in B$. The same can be done with two graphs or two manifolds by combining all the nodes or points. Since sets are unordered collections without additional properties the Cartesian product has no special properties. But, for example, the product of two graphs is still a graph, and a product node is still connected to nodes derived from its graph neighbors. See Section 5.1 for details. This can be done with all kinds of geometric structures. In the next sections we will look at the exact definitions of product graphs (Section 5.1) and product manifolds (both continuous and discrete, Section 5.2).

5.1 Product Graphs

Let $F = (V_F, E_F), H = (V_H, E_H)$ be two graphs with sets of nodes V_F, V_H and edges $E_F \subset V_F \times V_F, E_H \subset V_H \times V_H$. The product graph $F \times H = (V_P, E_P)$ should contain all possible combinations of V_F and V_H as nodes which means $V_P = V_F \times V_H$ is the Cartesian product of both sets (see Eq. (5.1)). Therefore, every node of the product graph is a tuple of a node of F and H and every possible combination of nodes exists once in V_P. Any edge in the product graph is a tuple of two nodes, therefore $E_P \subset V_P \times V_P = (V_F \times V_H) \times (V_F \times V_H)$ contains tuples of tuples of nodes of F and H.

E_P is supposed to preserve the connectivity information of F and H but there are several ways to achieve this. **Hammack et al.** (2011) gives an overview of the three most widely used variations of product graphs which define the connectivity in different ways. Figure 5.1 shows examples of each type.

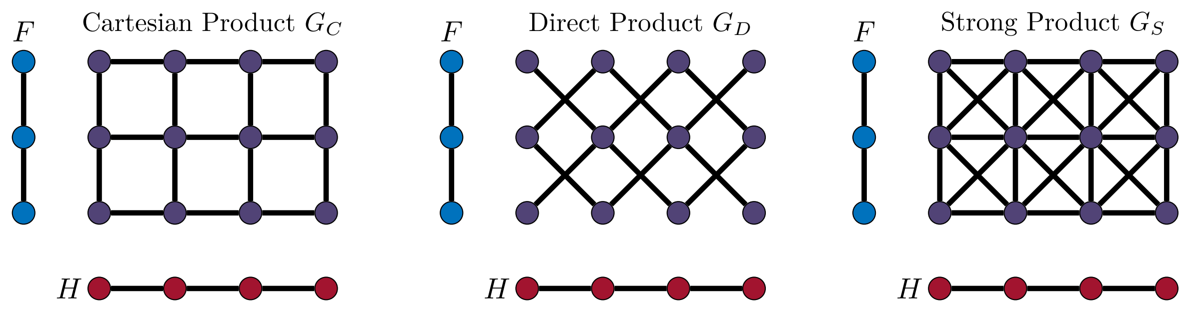


Figure 5.1: Examples of the Cartesian product (left), direct product (middle) and strong product (right) of two line graphs F and H.

Definition 8. *The* **Cartesian product** *of two graphs F, H is the graph $G_C = (V_P, E_C)$ such that $V_C = V_F \times V_H$ and*

$$E_C = \left\{((f_1, h_1), (f_2, h_2)) \in (V_F \times V_H)^2 \mid f_1 = f_2, (h_1, h_2) \in E_H \text{ or } (f_1, f_2) \in E_F, h_1 = h_2\right\}.$$

In the Cartesian product of graphs two nodes $(f_1, h_1), (f_2, h_2)$ are connected by an edge if one pair of elements are the same and the other was connected in the original graph.

Definition 9. *The* **direct product** *of two graphs F, H is the graph $G_D = (V_P, E_D)$ such that $V_D = V_F \times V_H$ and*

$$E_D = \left\{((f_1, h_1), (f_2, h_2)) \in (V_F \times V_H)^2 \mid (f_1, f_2) \in E_F \text{ and } (h_1, h_2) \in E_H\right\}.$$

The direct product of graphs only connects two nodes $(f_1, h_1), (f_2, h_2)$ if both f_1, f_2 and h_1, h_2 were connected by an edge in the original graphs.

Definition 10. *The* **strong product** *of two graphs F, H is the graph $G_S = (V_P, E_S)$ such that $V_S = V_F \times V_H$ and*

$$E_S = E_C \cup E_D.$$

The strong product of graphs combines all edges from the Cartesian and direct product.

By definition each product graph is also a normal graph, and any graph algorithm can be applied to it. However, not every graph can be decomposed into the product of two non-trivial graphs.

5.1.1 Projection

For later analysis, we define the projections of elements of the product graph (nodes and edges) back onto its source graphs. As each element is constructed by combining elements of F and H, the projection deconstructs elements of the product back into their elements of F and H.

Definition 11. *The* **node projection** π_F *of a node in the product graph* $F \times H$ *to* F *is defined as*

$$\begin{aligned}\pi_F &: V_P \to V_F \\ \pi_F\left((f, g)\right) &= f\end{aligned}$$

The definition is equivalent for π_H*.*

Definition 12. *The* **edge projection** Π_F *of an edge in the product graph* $F \times H$ *to* F *is defined as*

$$\begin{aligned}\Pi_F &: E_P \to V_F \times V_F \\ \Pi_F\left(((f_1, g_1), (f_2, g_2))\right) &= (f_1, f_2)\end{aligned}$$

The definition is equivalent for Π_H*.*

By construction $\Pi_F(e) \in E_F$ holds for any $e \in E_P$. An important property is that the projections of E_C, E_D, E_S onto F (and H) return exactly the original edge sets E_F (and E_H).

$$\begin{aligned}\Pi_F(E_C) &= E_F \\ \Pi_F(E_D) &= E_F \\ \Pi_F(E_S) &= E_F\end{aligned}$$

The same is true for Π_H and E_H. Therefore, the original connectivity of F, H is preserved through all definition of the product graph. Neither definition is inherently superior to the others. Rather, which is preferable depends on the intended application.

5.2 Product Manifolds

This section considers the product of two manifolds $\mathcal{M}, \mathcal{N}$ as defined in Definition 2. Similar to graphs, the surface of the product manifold consists of all combination of points from $\mathcal{M}, \mathcal{N}$, and the connectivity is preserved through combination of the coordinate maps. Because manifolds are continuous (in contrast to edges in graphs) there are no variations in how to combine them.

Definition 13. *Let* $\left\{x_i^{\mathcal{M}} : \mathbb{R}^m \to \mathbb{R}^n \mid i \in \{1, \ldots, M\}\right\}, \left\{x_j^{\mathcal{N}} : \mathbb{R}^k \to \mathbb{R}^l \mid j \in \{1, \ldots, N\}\right\}$ *be the set of coordinate maps defining two manifolds* $\mathcal{M}, \mathcal{N}$*. The* **product manifold** $\mathcal{M} \times \mathcal{N}$ *of* $\mathcal{M}$ *and* $\mathcal{N}$ *is described by the set of coordinate maps*

$$\left\{x_{i,j}^{\mathcal{M}\times\mathcal{N}} : \mathbb{R}^{m+k} \to \mathbb{R}^{n+l} \mid k \in i\{1, \ldots, M\}, j \in \{1, \ldots, N\}\right\}$$

such that

$$x_{i,j}^{\mathcal{M}\times\mathcal{N}}(x_1, x_2) = \left(x_i^{\mathcal{M}}(x_1), x_j^{\mathcal{N}}(x_2)\right).$$

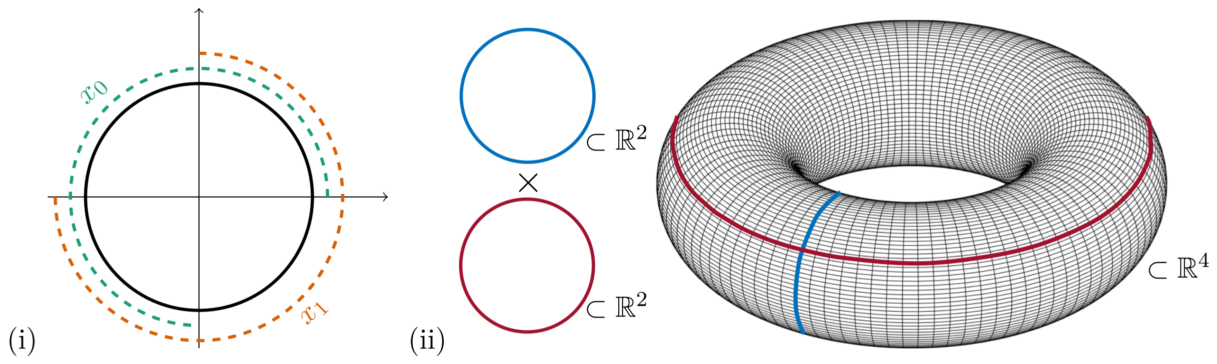


Figure 5.2: (i) Visualization which parts the two coordinate maps x_0, x_1 cover. (ii) Example of the product manifold of two circles. Each slice of the torus is a copy of a circle, the colors indicate how the circles are related to the originals. Mathematically $\mathcal{M} \times \mathcal{N}$ is a 2D manifold embedded in 4D but shown here is low dimensional embedding in 3D, topology is the same but it is not entirely metric preserving.

Again the product manifolds inherits most properties of $\mathcal{M}, \mathcal{N}$. If both are differentiable manifolds, the product manifold will also be a differentiable manifold. If $\mathcal{M}, \mathcal{N}$ are m- and k-dimensional manifolds embedded in $\mathbb{R}^n$ and $\mathbb{R}^l$ respectively, $\mathcal{M} \times \mathcal{N}$ will be a $m+k$-dimensional manifold embedded in $\mathbb{R}^{n+l}$. This is straight-forward as all coordinate maps are concatenations of the original coordinate maps.

Example 3. *Let* $\mathcal{X} = \mathcal{Y} = \{x_0 : (0, 1.5\pi) \to \mathbb{R}^2, t \mapsto (\cos(t), \sin(t)); x_1 : (\pi, 3\pi) \to \mathbb{R}^2, t \mapsto (\cos(t), \sin(t))\}$ *be the parametrization of two (identical) circles with radius* 1*. In this case, the product manifold* $\mathcal{X} \times \mathcal{Y}$ *contains four coordinate maps.*

$$\begin{aligned}\mathcal{X} \times \mathcal{Y} = \{ & x_{0\times 0} : (0, 1.5\pi) \times (0, 1.5\pi) \to \mathbb{R}^4, (t_1, t_2) \mapsto (\cos(t_1), \sin(t_1), \cos(t_2), \sin(t_2)) \\ & x_{0\times 1} : (0, 1.5\pi) \times (\pi, 3\pi) \to \mathbb{R}^4, (t_1, t_2) \mapsto (\cos(t_1), \sin(t_1), \cos(t_2), \sin(t_2)) \\ & x_{1\times 0} : (\pi, 3\pi) \times (0, 1.5\pi) \to \mathbb{R}^4, (t_1, t_2) \mapsto (\cos(t_1), \sin(t_1), \cos(t_2), \sin(t_2)) \\ & x_{1\times 1} : (\pi, 3\pi) \times (\pi, 3\pi) \to \mathbb{R}^4, (t_1, t_2) \mapsto (\cos(t_1), \sin(t_1), \cos(t_2), \sin(t_2))\}\end{aligned}$$

They describe a torus embedded in $\mathbb{R}^4$*. See Figure 5.2 for a visualization of this example.*

5.2.1 Projections

As for graphs each element of the product manifold $\mathcal{M} \times \mathcal{N}$ is constructed from elements of $\mathcal{M}, \mathcal{N}$ and therefore backprojections exist:

Definition 14. *The* **projection** $\pi_{\mathcal{M}} : \mathcal{M} \times \mathcal{N} \to \mathcal{M}$ *from the product manifold back onto* $\mathcal{M}$ *is defined as:*

$$\begin{aligned}&\pi_{\mathcal{M}} : \mathcal{M} \times \mathcal{N} \to \mathcal{M} \\ &\pi_{\mathcal{M}}(\ (m, n)\ ) = m\end{aligned}$$

The definition is equivalent for $\pi_{\mathcal{N}}$.

Again, as $\pi_{\mathcal{M}}(\mathcal{M} \times \mathcal{N}) = \mathcal{M}$ holds, the entire connectivity of each manifold is preserved in the product. Furthermore, since all elements of $\mathcal{M}$ are combined with each element of $\mathcal{N}$ exactly once the following holds: $\pi_{\mathcal{M}}(\pi_{\mathcal{N}}^{-1}(n)) = \mathcal{M}$ for any $n \in \mathcal{N}$, $\pi_{\mathcal{N}}^{-1}(n)$ maps to the set of points in $\mathcal{M} \times \mathcal{N}$ that are backprojected onto n. This means $\mathcal{M} \times \mathcal{N}$ consists of a collection of copies of $\mathcal{M}$, as well as of $\mathcal{N}$ because of the symmetric definition. See Figure 5.2 how the torus is just a lot of copies of the circles from Example 3.

5.2.2 Properties and Operators

All product manifolds inherit the properties that their origins hold in some way. For example, the product of two Riemannian manifolds $(\mathcal{M}, g_{\mathcal{M}}), (\mathcal{N}, g_{\mathcal{N}})$ will be a Riemannian manifold again and as such be equipped with a metric tensor $g_{\mathcal{M}} \oplus g_{\mathcal{N}} = \left(\begin{smallmatrix} g_{\mathcal{M}} & 0 \\ 0 & g_{\mathcal{N}} \end{smallmatrix}\right)$ (**Guillemin and Pollack** 2010). The dimension of the product manifold and its embedding are the sum of the dimensions of the original manifold. The area elements are the products of the original area elements. But since the size of the product manifold is the product of the sizes of the original manifolds, calculating certain properties and operators can become infeasible. The product manifold of two shapes with $1,000$ vertices already has $1,000,000$ vertices.

Nevertheless, one can leverage the fact that each coordinate map, and therefore the surface of the product manifold, is a pure concatenation of the original coordinate map and in many cases can be calculated just from these.

5.2.3 Laplace-Beltrami Operator

Looking at adjacency matrices of graphs and the rules from constructing product graphs, it is not hard to see that there are closed form solutions for constructing the adjacency matrix of the product graph. The Laplace-Beltrami operator (LBO, see Chapter 4) is a generalization of the adjacency information on Riemannian manifolds. It is not surprising that a closed form solution also exists for this case.

More interesting for some applications is that the solution of the eigen decomposition of the LBO can also be directly derived from the eigen decomposition of $\mathcal{M}, \mathcal{N}$. **Chavel** (1984) showed that the LBO $\Delta_{\mathcal{M} \times \mathcal{N}}$ obeys the (outer) product rule identity for $f \in \mathcal{F}(\mathcal{M}), g \in \mathcal{F}(\mathcal{N})$ in some function spaces $\mathcal{F}(\mathcal{M}), \mathcal{F}(\mathcal{N})$:

$$\Delta_{\mathcal{M} \times \mathcal{N}}(f \otimes g) = (\Delta_{\mathcal{M}} f) \otimes g + f \otimes (\Delta_{\mathcal{N}} g) \tag{5.2}$$

where $f \otimes g : (x, y) \mapsto f(x)g(x)$ denotes the outer product of two functions and $\Delta_{\mathcal{M}}, \Delta_{\mathcal{N}}$ the LBO of the manifolds $\mathcal{M}, \mathcal{N}$. Using this property leads to

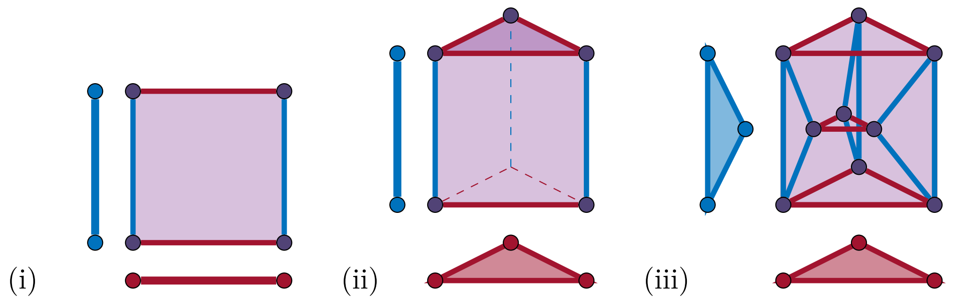


Figure 5.3: Examples of the product of base elements of meshes. The dimension of each product is colored by the shape it comes from. (i) The product of two lines is a rectangle, resulting in a quad mesh for contours. (ii) The product of a line and a triangle is a cylinder with a triangle base. (iii) The product of two triangles is 3-3 duo prism. It is a $4D$ structure, here represented by its Schlegel diagram. It has 9 vertices, 18 edges, 9 quad faces and 6 triangle faces.

$$\begin{aligned}\Delta_{\mathcal{M}\times\mathcal{N}}(\phi\otimes\psi) &= (\Delta_{\mathcal{M}}\phi)\otimes\psi + \phi\otimes(\Delta_{\mathcal{N}}\psi) && (5.3)\\ &= (\alpha+\beta)(\phi\otimes\psi) && (5.4)\end{aligned}$$

with $\Delta_{\mathcal{M}}\phi = \alpha\phi$ and $\Delta_{\mathcal{N}}\psi = \beta\psi$ the eigen decomposition of $\mathcal{M}$ and $\mathcal{N}$ respectively. This leads to the following characterization from **Berger et al.** (1971), Proposition A.II.3:

Corollary 1. *Let ξ be an eigenfunction of the product LB operator $\Delta_{\mathcal{M}\times\mathcal{N}}$ with the corresponding eigenvalue γ. Then, there exist some eigenfunctions ϕ of $\Delta_{\mathcal{M}}$ and ψ of $\Delta_{\mathcal{N}}$ with the eigenvalues α and β, respectively, such that $\xi = \phi\otimes\psi$ and $\gamma = \alpha+\beta$.*

It is also easy to check that the set of eigenfunctions $\{\phi_i\otimes\psi_j\}_{i,j}$ is orthogonal:

$$\begin{aligned}\int_{\mathcal{M}\times\mathcal{N}}(\phi_i\otimes\psi_j)(\phi_k\otimes\psi_\ell)\,\mathrm{da} &= \int_{\times}\phi_i(x)\psi_j(y)\phi_k(x)\psi_\ell(y)\,\mathrm{da}\int_{\phi}{}_i\phi_k\underset{\cdot}{\mathrm{x}}\int_{\psi}{}_j\psi_\ell\mathrm{d}y && (5.5)\\ &= \delta_{ik}\delta_{j\ell} = \begin{cases}1 & (i=k)\text{ and }(j=\ell);\\ 0 & \text{otherwise,}\end{cases} && (5.6)\end{aligned}$$

where δ_{ij} is the Kronecker delta.

5.3 Product of Discrete Meshes

Discrete meshes are not defined by coordinate maps but by sets of vertices V and faces F, see Chapter 3. As for product graphs (Section 5.1) the vertices of the product mesh are the products of the sets of vertices. The faces act like coordinate maps from a base face to the actual embedded face (see Section 3.1.3.1 for triangles). Therefore, following directly from Definition 13 the product of two discrete meshes is built by taking the product of all combination of faces.

Line Products. In the case of contours, which are collections of lines, each face of the product mesh is a rectangle, which is the product of two lines (see Figure 5.3). Mathematically the two lines are $\ell_0 = [0, 1]$ and $\ell_1 = [0, 1]$, therefore only one combination exists which is exactly $\ell_{0\times1} = [0, 1] \times [0, 1]$. This means the product of two contours is a quad mesh embedded in 4D. See Figure 5.2 where the quads are indicated on the torus. The quads are connected exactly on the product as the lines were connected on the contour. Furthermore, as we already saw for the LBO in Eq. (5.2) functions can be transferred to the product space by taking the outer product. This works for any function and, for example the natural extension of hat basis functions on contours are bi-linear hat functions on the product quads.

Triangle Products. The product of two triangle meshes consists of a 3-3 duo prisms which is a 4D shape embedded in 6D and therefore hard to visualize. Figure 5.3 shows its Schlegel diagram (**Schlegel** 1883). Nevertheless, the product follows the same rules as before and its construction is not complicated. We will revisit these products in Chapter 9. In Chapter 7 we also consider the product of lines with triangles which results in a cylinder with a triangular base as the product face.

CHAPTER 6

Correspondences

Single 3D shapes can be useful for a wide variety of applications, including but not limited to Virtual Reality, 3D printing, architectural design, or video games. The processing of single objects can be utilized for applications like 3D printing or verifying the structural analysis of a planned building. However, if multiple shapes are given – either as separate objects or as a composition – knowing the relationship between them can generate a lot of knowledge about both of them without any semantic information necessary. Knowing which parts of two objects are the same in certain aspects makes it possible to automatically replace them with parts of equal functionality, or transfer information, style and pose to a new object. This could be about finding out which parts in a collection of chairs are to be sat on, or designing new chairs with given functionality that fit the style of a room. The information which parts or points are the same is called *correspondence* and notoriously hard to find if the objects are not mere copies of each other. Section 6.1 will introduce the mathematical definition of correspondences and Section 6.2 will focus on what we call *continuous* correspondences. This roughly means that if two parts are in correspondences their immediate neighborhood should also be in correspondence. Continuity is a property that is meaningful in almost all scenarios because what is a shape except the specific arrangement of its parts? Mathematically continuity in correspondences between manifolds is related to diffeomorphism and Section 6.3 looks at how to characterize diffeomorphisms on discretized shapes.

6.1 General Correspondence

A correspondence defines which points of two shapes "belong to each other". What exactly qualifies two points to belong to each other might be up to interpretation and vary widely from application to application. However, the set that was chosen can be defined mathematically by selecting the appropriate subset of the product of both shapes which by definition includes all possible combinations of points. See Chapter 5 for an introduction into product spaces.

Definition 15. *A* **correspondence** *C between two shapes $\mathcal{X}, \mathcal{Y}$ is a subset of their product.*

$$C \subset \mathcal{X} \times \mathcal{Y} \tag{6.1}$$

We call a single element $c = (x, y) \in C$ a **match**. *If C describes a proper function[1] $\varphi : \mathcal{X} \to \mathcal{Y}$, we call $C = \Gamma_\varphi = \{(x, \varphi(x)) | x \in X\} \subset \mathcal{X} \times \mathcal{Y}$ the* **correspondence graph**.

If the pair (x, y) is contained in C, it means $x \in \mathcal{X}$ is in correspondence to $y \in \mathcal{Y}$. Normally algorithms define specific properties that C or its elements are assumed to have and the optimization tries to minimize a certain energy $E(C)$ that penalizes when this property is not fulfilled. A common scenario includes two representations of the same object, maybe scanned at two different times, and then it is obviously meaningful to put points that depict the same point on the objects surface into correspondence. Another application might include two different objects, for example two styles of chairs, but since they belong to the same class of objects some parts have the same function or semantic meaning and therefore should correspond. In the first example it is intuitive that every point is only part of exactly one match, whereas in the second example it could be multiple or none – think of a standard four-legged diner chair and an office chair with five legs and arm rests. This thesis will focus on algorithms dealing the first scenario.

Example 4. *A simplified assumption is that both input shapes are sampled with the same number of points n and each point on $\mathcal{X}$ corresponds to exactly one point on $\mathcal{Y}$. This transfers to C by imposing that C is a bijective[2] function and in this case C can be represented by a permutation matrix $P \in \{0, 1\}^{n \times n}, P^\top P = 1_n$. Given a point property on both shapes (e.g. curvature or heat kernel signature (**J. Sun et al.** 2009)) a very simple optimization would be to find the permutation between the points of $\mathcal{X}$ and $\mathcal{Y}$ that preserves this property in the best possible way:*

$$P^* = \underset{P \in \mathcal{P}_n}{\arg\min} \|FP - G\|_1 \tag{6.2}$$

Here, $\mathcal{P}_n$ is the set of $n \times n$ permutation matrices and $F, G \in \mathbb{R}^n$ are comparable, point-wise properties on $\mathcal{X}, \mathcal{Y}$ respectively. P^* can be found by solving a Linear Assignment Problem. However, a single descriptor function would probably not suffice to give a good solution to (6.2) *because as soon as F or G contain a value twice the solution is not unique anymore. We will see more sophisticated formulation in later parts of this thesis. Additionally, it is important that F and G are comparable,* i.e. *they assume approximately the same value on points that are supposed to match. For example, some notions of curvature are extrinsic and therefore not well suited for matching deformed but isometric shapes, instead an intrinsic descriptor will fit much better.*

[1] A function $f : A \to B$ is a binary relation $f \subset A \times B$ such that $\forall a \in A \; \exists \; b \in B : (a, b) \in f$ and $\forall a \in A, b_1 \in B, b_1 \in B : (a, b_1) \in f \wedge (a, b_2) \in f \implies b_1 = b_2$.

[2] A function $f : A \to B$ is bijective iff. $\forall b \in B \; \exists! \; a \in A : f(a) = b$.

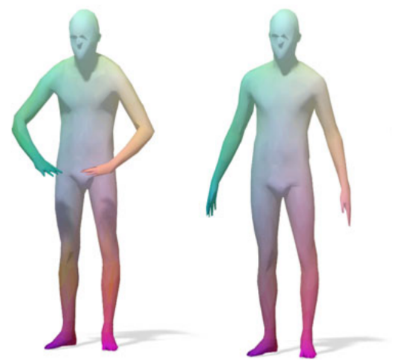
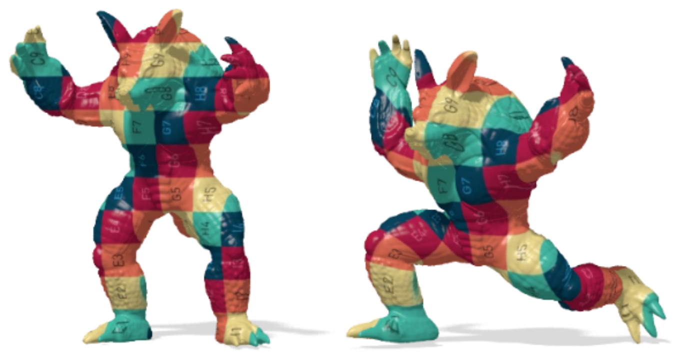
(i)

(ii)


Figure 6.1: Examples of correspondence visualization throughout this thesis. (i) Color Transfer. Each point on $\mathcal{X}$ is marked with a unique color, each point on $\mathcal{Y}$ takes it color from its match such that $\text{color}(x) = \text{color}(y)$ when $(x, y) \in C$. Small distortions are often not visible in this representation, and some information is lost should y have two matches. But it gives a good idea of the overall correctness and critical areas. (ii) Texture Transfer. $\mathcal{X}$ is textured and the texture transferred to $\mathcal{Y}$ via the correspondence. This representation makes small distortions visible when the texture has clean geometry but when larger errors occur it is not immediately obvious what exactly is wrong. Additionally, producing smooth texture maps on $\mathcal{X}$ is not a trivial problem.

Once a correspondence is found, we can visualize it in various ways to make it more easily accessible to human readers than the set of tuples. Figure 6.1 shows a few examples of correspondences and how they can be visualized. We will use these throughout this thesis.

We will begin by looking at pairs of shapes that can be deformed into each other through an isometry and later relax to unconstrained diffeomorphisms and only partially overlapping shapes as well as topological changes.

One subclass of isometries are rigid deformations (only allowing rotation and translation of the entire shape). Because a rotation and translation combined only have six degrees of freedom (in three dimensional space), the problem is not very complex and we focus on the non-rigid case here.

6.1.1 Vertex-to-Vertex Correspondence

If the shapes are discretized through point clouds or meshes, the most common representation of correspondences is to match pairs of vertices. In that case the correspondence is a subset of the vertex sets $C \subset V_M \times V_N$. This has many advantages because the vertex sets are normally already explicitly stored, most descriptors return vertex-wise values and imposing constraints like bijectivity is straight-forward. On the other hand, the underlying assumption is that both shapes are at least approximately meshed consistently. In real-world data this basically never happens due to suboptimal scanning conditions and areas that are densely sampled on one shape might only contain a few vertices on the other shape. This poses problems especially when unrealistic assumption about (near) bijectivity are made.

Soft Correspondences A variant of point-to-point correspondences are soft correspondences. Instead of a permutation the correspondences is stored in a matrix $S \in [0,1]^{n \times n}$ where the value in S_{ij} indicates how likely the vertices $x_i \in \mathcal{X}$ and $y_j \in \mathcal{Y}$ are to match. This can be extended to represent a probability by enforcing the rows or columns to sum up to one, $S\mathbf{1} = \mathbf{1}$ and $\mathbf{1}^\top S = \mathbf{1}$. If $n = m$ and $S\mathbf{1} = \mathbf{1}^\top S = \mathbf{1}$, S is called bi-stochastic. The set of bi-stochastic matrices is a popular replacement for the set of permutation matrices because it is a superset of permutations and convex. A more general form of soft correspondences, namely soft maps, have been introduced in **Solomon et al.** (2012). These maps are already capable of capturing inconsistent meshing by distributing the probabilities over a larger area.

6.1.2 Subvertex Correspondence

To overcome the problem of incompatible meshing, point-to-point correspondences can also be represented with subvertex accuracy. As the name implies, a vertex of $\mathcal{X}$ cannot only match with a vertex of $\mathcal{Y}$ but also lie in-between. On triangular meshes this is realized by letting the $C(p)$ lie on the interior of a triangle of $\mathcal{Y}$ instead of just on the corners. If an area of $\mathcal{X}$ is much more densely sampled then the corresponding area in $\mathcal{Y}$ this allows all points of $\mathcal{X}$ to match onto the same triangle without collapsing into the same vertex. This is mathematically beneficial because the differential of the correspondence does not become low rank. Similarly for point clouds the match can be a combination of points from $\mathcal{Y}$. **Ezuz et al.** (2019) demonstrated how important subvertex accuracy can be when optimizing for deformation energies. Bending energy can be very distorted when points are inaccurately placed due to meshing inconsistencies and the advantages outweigh the additional complexity here.

6.1.3 Registration

A highly related problem to correspondence is the registration problem (and sometimes the names are used interchangeably). Instead of finding a mapping $C : \mathcal{X} \to \mathcal{Y}$ a registration finds a mapping $R : \mathcal{X} \to \mathbb{R}^3$ that aligns the surface of $\mathcal{X}$ tightly with $\mathcal{Y}$. A registration R can be easily converted into a point-to-point correspondence by doing a nearest neighbor search of $R(\mathcal{X})$ in $\mathcal{Y}$ or a subvertex correspondence by projecting $R(\mathcal{X})$ onto the surface of $\mathcal{Y}$. The advantage is that points of $\mathcal{X}$ can be set in meaningful relation with $\mathcal{Y}$, even if there is no point-wise relation. For example, if $\mathcal{Y}$ has large holes, the surface of $R(\mathcal{X})$ can span over this hole whereas a correspondences would force points to be projected on $\mathcal{Y}$ somehow. On the other hand, transferring information through alignment does not always work, *e.g.* a registration cannot be used to transfer a UV map to a different surface. We use this in Chapter 10. In the end, it depends on the application and algorithm whether a correspondence or registration is more beneficial.

6.2 Continuous Correspondences

A common property of non-rigid real-world objects is that they are deformable but not dismountable. This means they can be reposed but only through a smooth movement of some parts, and not by de- and reattaching them. Humans are a good of example of deformable because they can move into a wide variety of poses, but it is obvious that limbs of humans should not be removed in the process. On the other hand, a figure made out of LEGO bricks can be reposed (even into an isometric counterpart), but it can only be done by removing some or all bricks and then assembling them in a different way. We will call a correspondence that can be achieved by posing without decomposing smooth. This can be expressed mathematically as follows:

Definition 16. *We call a correspondence $C : \mathcal{X} \to \mathcal{Y}$* **continuous** *iff. C is a diffeomorphism.*

A diffeomorphism from $\mathcal{X}$ to $\mathcal{Y}$ is a differentiable function $f : \mathcal{X} \to \mathcal{Y}$ and its inverse f^{-1} is also differentiable. This also implies that C is a bijection, which is not meaningful in all cases. We will relax this definition later. A diffeomorphism can be meaningful in cases where a full correspondence is possible. We discuss partiality and other deviations from this case in Section 6.2.1. Furthermore, diffeomorphisms imply natural properties, *e.g.* that the neighborhood of each point is preserved after applying C. However, showing that a function is a diffeomorphism for discrete $\mathcal{X}, \mathcal{Y}$ is not straightforward, and it might be easier to look for properties that are equivalent to the diffeomorphism. We discuss this in more detail in Section 6.3. The following statements are equivalent:

Theorem 1. *Given a correspondence $C : \mathcal{X} \to \mathcal{Y}$ the following properties are equivalent:*

(i) C is a diffeomorphism,

(ii) all differentiable paths $r \subset \mathcal{X}$ are mapped to differentiable paths $C(r)$ on $\mathcal{Y}$ and vice versa,

(iii) for every point $p \in \mathcal{X}$ and its ϵ-neighborhood $\mathcal{N}(p)$ holds C is a local diffeomorphism $\mathcal{N}(p) \to \mathcal{Y}$ and equivalently also for every point $p \in \mathcal{Y}$,

(iv) Γ_φ is a differentiable, connected, closed surface in the product manifold $\mathcal{X} \times \mathcal{Y}$ with the projections $\pi_\mathcal{X} : \Gamma_\varphi \to \mathcal{X}$ and $\pi_\mathcal{Y} : \Gamma_\varphi \to \mathcal{Y}$ both diffeomorphisms. It follows that $dim(\Gamma_\varphi) = dim(\mathcal{X}) = dim(\mathcal{Y})$.

The equivalence of the first three properties is straight-forward to see because (ii) and (iii) just deconstruct the differentiable property into its local components. (iv) was proposed and proven by **Windheuser et al.** (2011a).

Defining the same property on discretized surfaces is not that obvious. See Figure 6.4 for an example where neighborhood preservation does not lead to the expected behavior. This is due to the fact that approximating a continuous surface is possible through many different discretizations, which are unlikely to be neighborhood preserving everywhere.

Diffeomorphisms are not meaningful in all cases. For example, they imply a bijective between $\mathcal{X}$ and $\mathcal{Y}$ which does not hold if the shapes have a partial relationship.

6.2.1 Partiality

With a few tweaks the above theory can be applied when $\mathcal{X} \subset \mathcal{Y}$, i.e. $\mathcal{X}$ is assumed to described a strict subset of $\mathcal{Y}$. We call this a partial shape. Assuming $\varphi : \mathcal{X} \to \mathcal{Y}$ to be a diffeomorphism, enforces $\varphi^{-1} : \mathcal{Y} \to \mathcal{X}$ to be a function which implies that every $y \in \mathcal{Y}$ is mapped to some $x \in \mathcal{X}$. However, the partiality assumption means that there are points on $\mathcal{Y}$ which do not have a counterpart on $\mathcal{X}$ because they show a part that is missing there. Put differently, a diffeomorphism is bijective but in partiality φ is only injective. A function that fulfills all properties of a diffeomorphism but is injective instead of bijective is called a local diffeomorphism:

Definition 17. *A function $f : X \to Y$ between two Riemannian manifolds X, Y is called a* **local diffeomorphism** *if for each $x \in X$ exists an open set U with $x \in U$ such that $f(U)$ is open in Y and $f|_U : U \to f(U)$ is a diffeomorphism.*

This is basically a diffeomorphism between $\mathcal{X}$ and the image $C(\mathcal{X})$. A similar definition can be made when $\mathcal{Y} \subset \mathcal{X}$. The definition of local diffeomorphism is especially interesting because it allows $\mathcal{X}$ to be of lower dimension than $\mathcal{Y}$. In that case, the dimension of Γ_φ is equal to the dimension of the domain of φ. This is the exact case we explore in Chapter 7.

Two-Sided Partiality In the case of two-sided partiality neither $\mathcal{X}$ nor $\mathcal{Y}$ are a subset of the other, instead they are assumed to have a common part such that $S_X \subset \mathcal{X}$ and $S_Y \subset \mathcal{Y}$ and S_X and S_Y are diffeomorphic. Unfortunately, $C : S_X \to S_Y$ cannot be written as a function $\mathcal{X} \to \mathcal{Y}$ or vice versa, but this is an implicit assumption in almost any algorithm. To adapt for this setting, it is necessary to also optimize the subsets S_X, S_Y in addition to the matching energy $E(S_X, S_Y)$. Since the matching is never perfect E will not be zero and often improve by just removing points from S_X, S_Y with $S_X = S_Y = \emptyset$ often the minimal energy solution. This can be counteracted by either knowing the area of S_X or S_Y, which is unrealistic, or adding a penalization term for a small overlap. However, the amount of penalization will have a high effect on the solution, and the correct weight will depend heavily on the type of data, which makes this approach unstable. Another way to look at this is that many methods implicitly solve for a minimum area surface $C \subset \mathcal{X} \times \mathcal{Y}$. This is a good approximation when the final solution is constrained to have at least area $area(\mathcal{X}) \cdot area(\mathcal{Y})$, but if this constraint does not hold (because the correspondence is allowed to be partial on both sides) the minimal surface solution is always the empty set. This is a reason why there is no state-of-the-art method that can handle two-sided partiality in the general case without a template, learning or other prior information.

6.2.2 Surface Merging

When acquiring 3D data from real-world objects, the scanner is normally only able to sense the surface information. When two separate parts of the same object touch each other, despite not being physically connected, it is basically impossible to detect this from typical sensor data, e.g. camera images. Therefore, the resulting 3D scan will show the parts with a merged surface. This is especially a problem if the object is deformable, for example a human, and the merged parts vary between different scans that are to be set in correspondence. This behavior is also called topological merging, even if the topology does not necessarily change.

Let $\mathcal{Y}$ be a full model and $\mathcal{X}$ the same object but with topological merging. While for partiality a local diffeomorphism was sufficient, in this case, additionally to some parts missing on $\mathcal{X}$, there are also additional connections on the surface. These connections have to be "broken" in the correspondence which is not possible through a differentiable function.

We published a dataset and associated SHREC contest with artificial shapes containing topological noise in **Lähner et al.** (2016a). At the time no participating method was able to achieve a comparable quality of results as usual on data sets with similar structure but without topological noise. Since then some methods have shown improved results, but not close to the level that one could consider the problem solved.

6.2.3 Example: Contour-to-Contour Correspondence

The following example will explain and visualize the above properties on two contours. For two 2D shapes the product space become a 4D manifold which is hard to visualize, but the same concepts apply.

Let $\mathcal{X} = \mathcal{Y} = \{x_0 : (0, 1.5\pi) \to \mathbb{R}^2, t \mapsto (\cos(t), \sin(t)); x_1 : (\pi, 3\pi) \to \mathbb{R}^2, t \mapsto (\cos(t), \sin(t))\}$ be two circles (the same as in Example 3). While there are some obvious choices for the correspondence between them, like the identity as well as all rotations of it, these are not all possible diffeomorphisms between $\mathcal{X}$ and $\mathcal{Y}$. For example using $\nu : [0, 2\pi] \to \mathbb{R}, t \mapsto \pi \cdot \operatorname{sgn}(\frac{t}{2} - 1) \cdot \left(\frac{t}{2} - 1\right)^2$ we could take $C_1 = (\cos(t), \sin(t), \cos(\nu(t)), \sin(\nu(t)))$. This is still a diffeomorphism but the correspondences are not equally distributed over the surface. On the other hand, $C_2 : \mathcal{X} \to \mathcal{Y}, (x, y) \mapsto (0, 1)$ is not a diffeomorphism because it is not invertible.

$$C_3 : \mathcal{X} \to \mathcal{Y}, (x, y) \mapsto \begin{cases} (x, y) & \text{if } y >= 0 \\ (-x, y) & \text{if } y < 0 \end{cases}$$

is not a diffeomorphism because its not differentiable at $(-1, 0)$ and $(1, 0)$. See Figure 6.2 for a visualization of C_1 to C_3.

It is easy to see where neighborhoods and paths are preserved on 1D manifolds, but we look at the submanifold property in detail now. The product manifold $\mathcal{X} \times \mathcal{Y}$ has the form of a torus

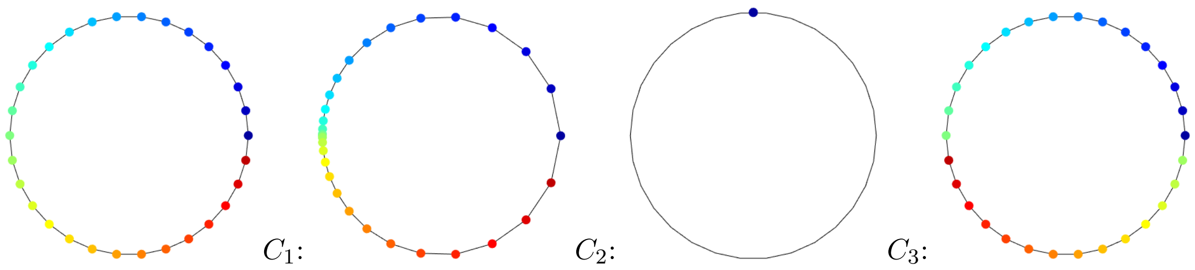


Figure 6.2: Visualization of C_1-C_3. The leftmost circle depicts the base coloring on $\mathcal{X}$, the other circles show how C_i transfers the coloring from $\mathcal{X}$ to $\mathcal{Y}$ via the correspondence. C_2 maps every point to $(0, 1)$ which maps all points, and therefore colors, over each other.

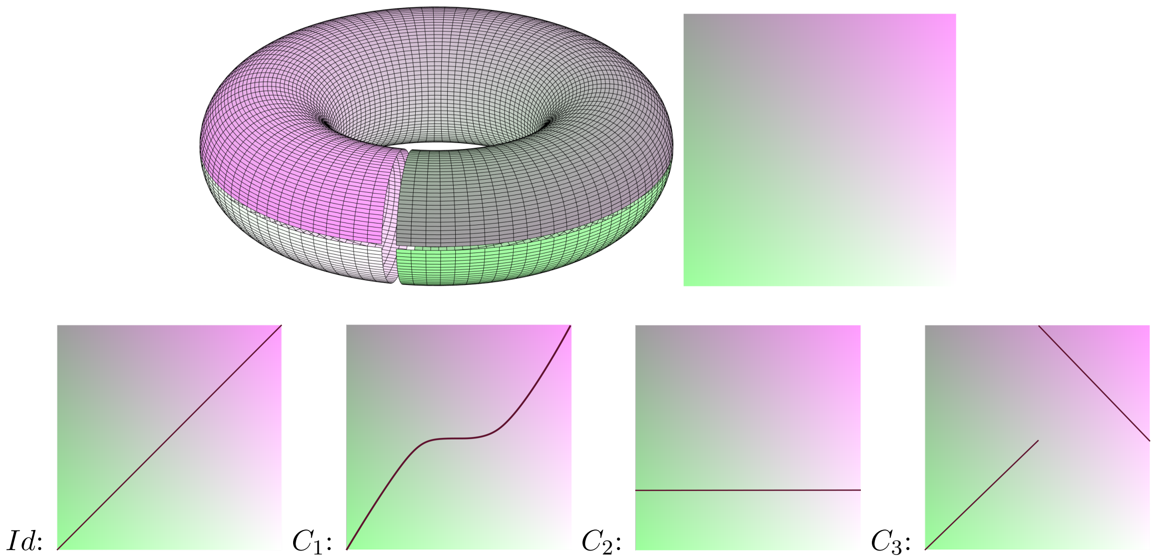


Figure 6.3: (Top) The torus can be cut open and be laied out flat for better visualization (see colors). Apart from the discontinuities, the flat patch actually preserves the metric of the 4D torus better than the 3D torus. (Bottom) The identity and each C_i and their subsets of the product manifold are shown in red. Id and C_1 are both connected, smooth and have an invertible projection on the edges which means they are diffeomorphisms. C_2 is only projected onto one point on the vertical edge which means this backprojection is not invertible. C_3 can be properly projected but is not connected, not even when the torus would be wrapped up.

and is a 2D manifold embedded in 4D. See Example 3 and Figure 5.2 for a visualization and the complete parametrization of the torus. Because each correspondence graph Γ_C is a subset of $\mathcal{X} \times \mathcal{Y}$ we can visualize each match $c \in C$ as a point on this torus. Theorem 1-(iii) states that Γ_C should be a differentiable, closed and connected surface on $\mathcal{X} \times \mathcal{Y}$ which means the collection of points in Γ_C needs to completely describe this surface. By construction $c \in \Gamma_C$ can be represented as $(u, C(u)) \subset \mathcal{X} \times \mathcal{Y}$ and assuming that $\mathcal{X}, \mathcal{Y}, C$ are differentiable transfers this property to Γ_C and vice versa. Figure 6.3 shows the correspondence graphs of each C_i and makes clear where these properties are violated.

The constraint that $\pi_\mathcal{X}, \pi_\mathcal{Y}$ need to be diffeomorphism includes that they are invertible, which means there needs to be exactly one point in each row and column of the flat torus. Due to being differentiable, the surface (or line in this example) should be smooth and not have boundaries. It becomes obvious by trying around a little that the only way to draw the line differentiable (*i.e.* without boundary points) and without drawing two points in one row or column is by doing it in a connected way (ignoring discontinuities that came from flattening the torus). Although there are infinitely many possible diffeomorphisms between two circles, from our examples only the identity and C_1 fulfill the stated criteria.

6.3 Discrete Diffeomorphisms

While it is in theory possible to represent surfaces as collections of differentiable coordinate maps on the computer, the vast majority of shapes is not actually represented as manifolds (see Section 3.1). In this section we will look at how the notion of diffeomorphism can be discretized for triangular meshes, one of the most popular representations for 3D surfaces. In particular, triangular meshes are not differentiable at their edges and our best result will therefore be a continuous correspondence but not a differentiable map.

6.3.1 Neighborhood Preservation

Property (iii) in Theorem 1 states that the neighborhood of each point should be mapped to the neighborhood of its match. This cannot be directly transferred to triangular meshes because even when there exists a perfect bijection between the vertex sets, the triangulation does not need to be consistent. See Figure 6.4.

A different approach would be to utilize the ϵ-neighborhood of a point instead of using the edge information and compare it to the ϵ-neighborhood of its match. Since diffeomorphism do not necessarily preserve metric this does not need to hold. In fact, the metric distortion can be arbitrary and different in every direction, therefore, not even scaling the neighborhood or enforcing convexness leads to an equivalent result.

6.3.2 Path preservation

Similar to the neighborhood property, the path preservation also suffers from inconsistent meshing. To check continuity the path has to be traced over the surface, *i.e.* the faces, which could potentially be split up into arbitrarily many. To check the continuity of the splitting, the new faces and their neighbors have to be aligned, This is equivalent to modeling continuity of the entire surface (as described in Section 6.3.3) so there is no advantage in looking at paths. Additionally, a swap of two vertices in the solution can still be converted into a continuous path, although the correspondence itself is not continuous anymore. The length of any path is also not necessarily preserved in general diffeomorphism, but, if this is made an assumption, it can be used on discrete meshes in a clean way.

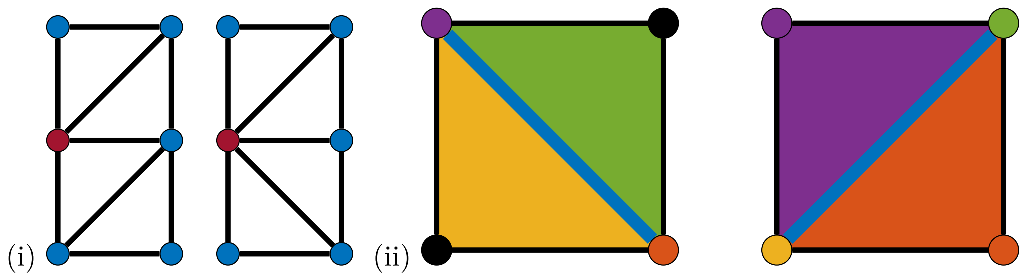


Figure 6.4: (i) Although both meshes discretize the same surface, the neighborhood of the red point changes. The neighborhood can change arbitrarily through remeshing. (ii) **Windheuser et al.** (2011a) model continuity in different meshing by allowing all elements (triangles, edges and corners) to shrink and extend into degenerated versions. Like this, neighborhood can be preserved, even though the meshing changed. For example, the blue edge is flipped, this causes the yellow and green triangle to degenerate into corners, whereas the purple and orange corners extended into triangles.

Isometries Isometries are one case that can be easily transferred to discrete meshes. The definition of an isometry is that all geodesic distances are preserved and is a popular formulation for shape correspondence through a Quadratic Assignment Problem (see Section 1.3.3.2).

$$\min_{P} \|D_{\mathcal{X}} - P^{\top} D_{\mathcal{Y}} P\|_1 \tag{6.3}$$

Since every geodesic distance is actually the length of the shortest geodesic path between two points, there is a direct relationship to Theorem 1-(ii). In fact, every isometry between two Riemannian manifolds is a diffeomorphism. Therefore, QAPs using geodesics are indirectly optimizing for a diffeomorphism. Each segment of a geodesic is also a geodesic between its endpoints and by enforcing all of them to be length-preserving, continuity is enforced implicitly. This does not work for general diffeomorphisms because they do not need to preserve geodesic distances nor geodesics themselves.

6.3.3 Product Submanifolds

The authors of **Windheuser et al.** (2011a) introduced the fourth property from Theorem 1 and transferred it onto triangular meshes. The original theorem in **Windheuser et al.** (2011a) also includes a third property about orientations, but this one is only needed for orientation preserving diffeomorphisms, which we skip here. They formulate the transformation as deformation of the triangles, edges, and corners. But in order to handle inconsistent meshes, triangles can shrink into an edge or corner, edges can extend to a triangle, or shrink into a corner and vice versa. This can model arbitrary meshing changes but still guarantees a continuous solution because neighboring elements stay neighboring. The method combines this with penalizing bending and stretching in the solution to find the least energy diffeomorphism.

In the end, these constraints are used in an Integer Linear Program (ILP) to find a discrete diffeomorphism between two shapes. This is combined with penalizing bending and stretching in the solution to find the least energy diffeomorphism. However, ILP is NP-complete and the algorithm takes a few hours for shapes with a hundred vertices.

PART III

Methodology

Equations are just the boring part of mathematics. I attempt to see things in terms of geometry.

– Stephen Hawking

CHAPTER 7

Correspondences as a Shortest Path Problem

This chapter looks at the problem of finding a correspondence between a 1D and a 2D manifold. The underlying assumption is that the 1D manifold describes the contour or a meaningful slice of the shape of the 2D manifold, and the task is to find out where this slice on the 2D manifold is (see Figure 7.1 for an example). This can be used for more intuitive interaction of humans with 3D data. When searching for a specific 2D object in a large, unlabeled collection, it is easier for the average user to produce a sketch of an object than querying with a similar 3D object. An easy way to do this for rigid objects is comparing a projection of the 3D object onto the plane to the drawing but deformable shapes can have widely varying projections which are hard to compare. If the user wants any human from the collection, they do not know which they have to draw. The natural step would be to draw a standing, neutral pose as a query because it is easy to produce. Rigid methods could only retrieve standing similar poses from this sketch but the presented method is the first that can also retrieve isometrically deformed 3D shapes.

Additionally to the complexity of the standard correspondence problem, this setting presents two more challenges. First, the inputs have different dimensionality which makes feature-based approaches hard because many features do not translate well – or at all – between different dimensional surfaces. Second, the result should be invariant to isometries in 3D and to a certain extend in 2D. In previous work about (rigid) retrieval the dimensionality problem is often solved by projecting the higher dimensional shape onto the plane which breaks down if poses vary too much. Isometric approaches fail in the case of inputs of different dimensionality because they rely on descriptors to define a notion of similarity and choosing descriptors that are invariant across dimensions is highly dependent on the application, the type of dimensionality reduction and most examples focus on multi-modal settings with images and text where geometry is irrelevant (**Masci et al.** 2013).

As pointed out in the previous chapter, the correspondence problem itself is hard but, if one

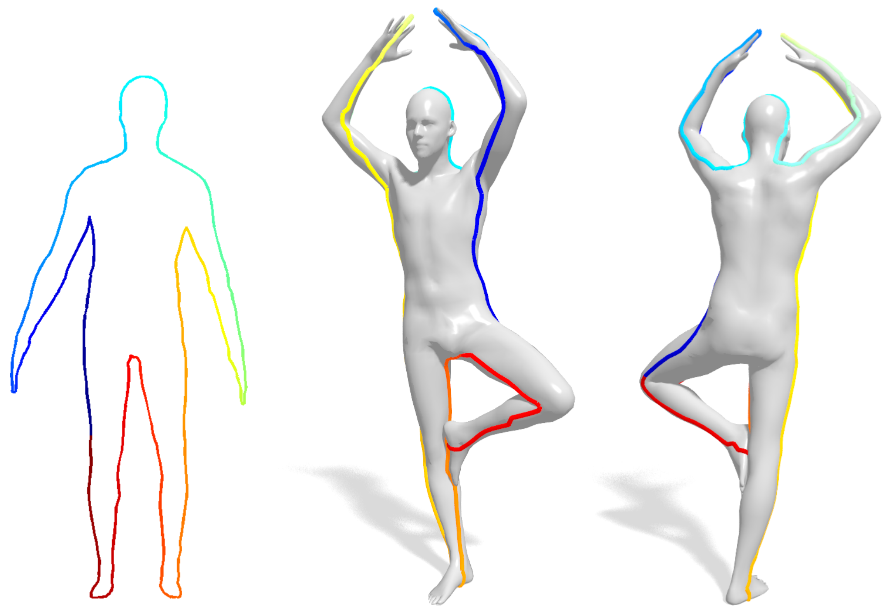


Figure 7.1: We propose a shape matching method between a 2D query shape (left) and a 3D target shape (right), both of which are allowed to deform non-rigidly. The globally optimal matching (shown on top of the 3D target) is guaranteed to be continuous.

of the manifolds involved is 1-dimensional, it becomes feasible because the solution is also 1-dimensional. Finding an optimal 1-dimensional structure can often be reformulated as a shortest path problem in a fitting embedding space, and this has been studied extensively. We will make use of this relationship to propose an efficient algorithm in this setting.

This chapter is an extended version of **Z. Lähner**, **E. Rodolà**, **F. R. Schmidt**, **M. M. Bronstein**, and **D. Cremers** (May 2016b). Efficient Globally Optimal 2D-to-3D Deformable Shape Matching. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

The remainder of the chapter is organized as follows: In Section 7.2.1 we will formulate the 2D-to-3D shape matching problem as an energy minimization problem on the product graph, which we will globally optimize and approximate in Section 7.4. To this end, we assume that descriptive features for both shapes are given and that it is possible to measure the dissimilarity between 2D and 3D features. The specific choice of such features depends on the application. For the application of *shape retrieval* that we discuss in Section 7.6.3 we use purely spectral features.

7.1 Related Work

This section gives an overview on work directly related to the results of this chapter. A more extensive and general overview can be found in Chapter 1.

7.1.1 2D and 3D shape correspondence.

In the domain of 3D-to-3D shape matching, a major challenge is to have theoretical guarantees about the optimality and quality of the correspondence. Several popular methods try to find a correspondence that minimally distorts intrinsic distances between pairs of corresponding points by approximate solution to the quadratic assignment problem (**A. M. Bronstein et al.** 2006b; **Leordeanu and Hebert** 2005; **Mémoli and Sapiro** 2005; **Rodolà et al.** 2012; **Rodolà et al.** 2013). A recent line of works builds upon the functional representation (**Ovsjanikov et al.** 2012; **Pokrass et al.** 2013a; **Rodolà et al.** 2016; **Rodolà et al.** 2015), where a point-wise map is replaced by a linear map between function spaces. While these approaches solve many challenging problems, they lack theoretical guarantees on the quality of the final solution. In particular, none of these methods yields provably *continuous* maps between the given shapes. This seems to be surprising, since the functional maps are linear. Nonetheless, linear operators between function space do not need to be continuous. Hence, finding continuous matchings is still a challenging task in the area of 3D-to-3D matching applications. A notable exception is the work of **Windheuser et al.** (2011a). Similarly to **Windheuser et al.** (2011a), our method comes with the theoretical guarantee of a continuous solution. In contrast to **Windheuser et al.** (2011a), our method can compute a matching in under half a minute instead of several hours. In our 2D-to-3D correspondence problem, the 2D shape is modeled as a closed planar curve and the 3D shape as a surface in $\mathbb{R}^3$. To find the correspondence, we look for a closed curve on the surface. From this perspective, our method can be seen as an extension of an image segmentation task that looks for a closed curve within a 2D image domain. It was shown in **Schoenemann and Cremers** (2010) that this segmentation problem can be formulated as finding a shortest path in the product graph of the 2D image domain and the 1D curve domain, where the size of the graph depends on the Lipschitz constant of the mapping. The drawback is that this constant is typically unknown in advance. Differently from **Schoenemann and Cremers** (2010), the size of the constructed graph with our method is independent of the Lipschitz constant.

Furthermore, one of the main challenges in our method is to find an initial match on the product graph. In **Schoenemann and Cremers** (2010) this problem was solved by parallelization. As a result, the overall computation time is not reduced but just distributed intelligently among several computational cores. Instead, we use a branch-and-bound approach that only computes shortest paths in those regions that are most 'promising'. This strategy reduces the runtime substantially (especially with well-chosen shape descriptors), while still converging to a global optimum.

Even in the simpler 2D-to-2D setting, the computation of a globally-optimal correspondence can be very slow if we do not know an initial match. For example, the runtime of Dynamic Time Warping methods is $\mathcal{O}(n^2)$ if an initial match is given, and $\mathcal{O}(n^3)$ if every possible initial match is tested independently, where n is the number of shape samples. It was shown that by exploiting the planarity of the involved graph, the runtime of the whole matching including an initial match can be reduced to $\mathcal{O}(n^2 \log(n))$ by using shortest circular path or graph cut approaches (**Maes** 1991; **F. R. Schmidt et al.** 2009; **F. Schmidt et al.** 2007). A competitive approach is the branch-and-bound approach of **Appleton and C. Sun** (2003). While this does not reduce the worst case time complexity of $\mathcal{O}(n^3)$, it is rather fast in practice. Since this method does not use the planarity of the involved graph, we can adapt it to our scenario in order to reduce the practical runtime substantially.

7.1.2 Sketch-based retrieval

One of the important applications of 2D-to-3D matching is shape-from-sketch retrieval. This problem has recently drawn the attention of the machine learning community as a fertile playground for cross-modal feature learning (**Eitz et al.** 2012; **Furuya and Ohbuchi** 2014; **Hueting et al.** 2015; **B. Li et al.** 2015; **H. Su et al.** 2015). **Herzog et al.** (2015) recently proposed to learn a shared semantic space from multiple annotated databases, on which a metric that links semantically similar objects represented in different modalities (namely 2D drawings and 3D targets) is learned. Although this approach yields promising results in the *rigid* setting and can address some variability of the shapes, its applicability to the *non-rigid* setting is an open question. In contrast, our method targets explicitly the setting when both the 3D target and the 2D query are allowed to deform in a non-rigid fashion. Furthermore, the method of **Herzog et al.** (2015) as well as other existing approaches mostly focus on finding *similarity* between a 2D sketch and a 3D shape while we solve the more difficult problem of finding *correspondence* (from which a criterion of similarity is obtained as a byproduct).

7.2 Continuous Problem Formulation

We consider the case of shape matching between a 2D query shape and a 3D target shape (see Definition 7). The problem to be solved is to find a correspondence between a closed 1D manifold $\mathcal{M} \subset \mathbb{R}^2$ and a 2D manifold $\mathcal{N} \subset \mathbb{R}^3$. The solution should be a continuous mapping $\varphi\colon \mathcal{M} \to \mathcal{N}$. As discussed in Section 6.2.1 a local diffeomorphism is possible between manifolds of different dimensionality. A local diffeomorphism is a special case of an immersion, *i.e.* the differential $d\varphi$ is of maximal rank at every point. While this is weaker than the diffeomorphism constraint it still means the mapping is differentiable and cannot collapse into a single point anywhere, *i.e.* , for any $x, y \in \mathcal{M}$ such that $\varphi(x) = \varphi(y) = q$ there has to exist a point $z \in \mathcal{M}$ in-between x, y for which $\varphi(z) \neq q$ holds. This local similarity may lead to a point $n \in \mathcal{N}$ to be matched multiple times but we still exclude the collapsed solution where all $m \in \mathcal{M}$ are matched with the same n. Additionally, this relaxation will allow us to apply an optimization technique with local operations which makes our entire algorithm efficient.

7.2.1 Energy formulation

The goal is to find a continuous 2D-to-3D matching that sets points that look alike into correspondence. Here similarity will be modeled by using point-wise features and a good correspondence will put points with similar features with respect to some distance function into correspondence.

Features. Constructing features that are comparable between structures of different dimension is not trivial and we will discuss our specific choice in more detail in Section 7.5. Without going into detail, we name the two feature maps on $\mathcal{M}, \mathcal{N}$ as $f_M \colon \mathcal{N} \to \mathbb{R}^{k_M}$ and $f_N \colon \mathcal{N} \to \mathbb{R}^{k_N}$ respectively. The dimensions k_M and k_N do not need to agree as long as a positive distance function $\operatorname{dist} \colon \mathbb{R}^{k_M} \times \mathbb{R}^{k_N} \to \mathbb{R}_0^+$ that can relate the 2D feature $f_M(x)$ of $x \in \mathcal{M}$ and the 3D feature $f_N(y)$ of $y \in \mathcal{N}$ exists. This distance takes care of the difficult task of comparing 2D features with 3D features and depends of course on the chosen features. The following energy and optimization is completely independent of the choice of features and distance function, except different optima, but a concrete choice that works well in our scenarios is presented in Section 7.5.

Given the two feature maps f_M and f_N as well as the distance function, we call a 2D-to-3D correspondence φ optimal if it minimizes the following energy

$$E(\varphi) \colon = \int_{\Gamma_\varphi} \operatorname{dist}(f_M(s_1), f_N(s_2)) \, \mathrm{ds}, \tag{7.1}$$

where $\Gamma_\varphi = \{(s_1, s_2) \in \mathcal{M} \times \mathcal{N} | s_2 = \varphi(s_1)\} \subset \mathcal{M} \times \mathcal{N}$ denotes the correspondence graph of φ as defined in Definition 15. The energy E accumulates the distance between the features of any pair of matched points (s_1, s_2) in φ which means a good solution will place points of the query onto points on the 3D shapes with similar features. Since φ is assumed to be continuous, this is not a simple nearest neighbor problem but takes the geometry of $\mathcal{M}$ and $\mathcal{N}$ into account. Note that Γ_φ is a simplicial complex due to the immersion property of φ. E is therefore defined as a line integral. Calculating the area elements needed for the line integral on Γ_φ is not straight-forward. Instead we substitute φ with a higher-dimensional mapping $\hat{\varphi} : \mathcal{M} \to \mathcal{M} \times \mathcal{N}$, $x \mapsto (x, \varphi(x))$. $\hat{\varphi}$ conveys the same information as φ but, as we will see below, integrates on $\mathcal{M}$ with known area elements. The definition of $\hat{\varphi}$ leads to the following equations $s = (s_1, s_2) = (x, \varphi(x))$, $\hat{\varphi}(M) = \Gamma_\varphi$ and therefore $f_N(s_2) = f_N \circ \varphi(x)$. We apply these in the substitution rule with $\hat{\varphi}$ which results in the following energy function:

$$E(\hat{\varphi}) = \int_M \operatorname{dist}(f_M(x), f_N \circ \varphi(x)) \cdot \|\hat{\varphi}'(x)\| \, \mathrm{d}x \tag{7.2}$$

Since $\mathcal{M}$ is a one-dimensional manifold, the norm can be calculated as $\|\hat{\varphi}'(x)\| = \sqrt{\mathrm{d}\hat{\varphi}^\top \mathrm{d}\hat{\varphi}}$. Fortunately, $\mathrm{d}\hat{\varphi}$ only depends on $\mathrm{d}\varphi(x)$ with an additional constant entry from where x was mapped to itself:

$$\mathrm{d}\hat{\varphi}(x) : T_x M \to T_x M \times T_y N \tag{7.3}$$

$$v \mapsto \begin{pmatrix} v \\ \mathrm{d}\varphi(x) v \end{pmatrix} = \begin{pmatrix} 1 \\ \mathrm{d}\varphi(x) \end{pmatrix} v \tag{7.4}$$

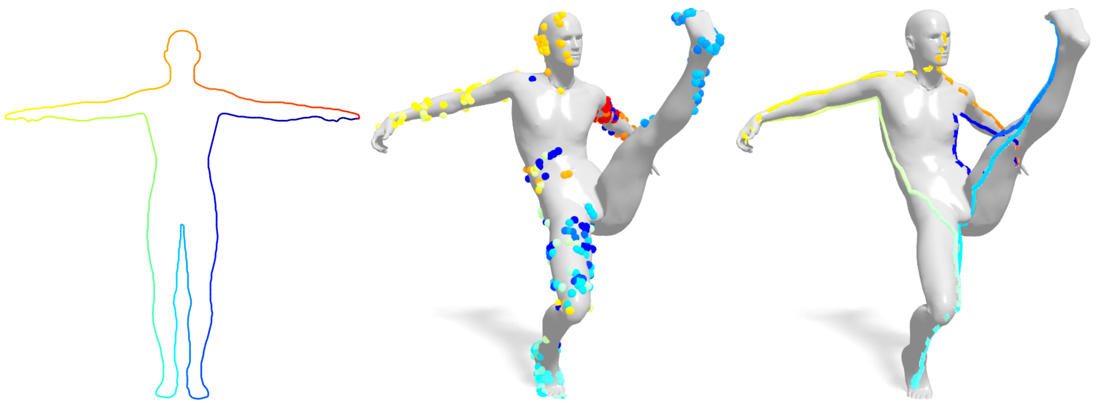


Figure 7.2: Matching between a 2D query shape (left) and a 3D target shape achieved by solving a LAP between the same point-wise features our method uses (middle) and our shortest-path method (right).

Including this in Eq. (7.2) leads to the following energy function which depends only on φ again:

$$E(\varphi) = \int_M \operatorname{dist}(f_M(x), f_N \circ \varphi(x)) \sqrt{1 + d\varphi_x^\top \mathrm{d}\varphi_x}\, \mathrm{d}x \tag{7.5}$$

Hence, the energy E can be broken down into the *data term* $\operatorname{dist}(f_M(\cdot), f_N \circ \varphi(\cdot))$ comparing feature values and the *regularizer* $\sqrt{1 + d\varphi^\top d\varphi}$ penalizing stretching.

Regularization. If we ignore the data term, the global minimum of E would result in a constant φ. This φ is continuous, but matches every point on M to the same point on N. It therefore ignores the similarity information stored in the data term.

Data term. If we ignore the regularizer, the global minimum of E can be computed by selecting for each $x \in M$ a $y \in N$ that minimizes the given feature distance $\operatorname{dist}(f_M(x), f_N(y))$. In this case, the minimizer of E will match similar points but φ might be neither injective nor continuous. Combining the data term with the regularization results in a smooth matching function φ that also takes similarity into account.

Alternatively to the energy described here, one could also choose to enforce injectivity of $\varphi\colon M \to N$. This would lead to a linear assignment problem (LAP), which normally results in non-continuous matchings and is rather slow. If the shapes M and N are discretized at m and n points, respectively, the overall run time of the Hungarian method (**Munkres** 1957) to solve this problem is $\mathcal{O}(n^3)$. The method that we propose does not only provide for a smooth and continuous solution, but also has a better worst case run time complexity than the LAP (cf. Theorem 2). Exploring the run time of the LAP approach for one matching instance resulted in a run time of 11 hours instead of just a few seconds for our method. The LAP matching result using the same features as in our experiments can be seen in Figure 7.2.

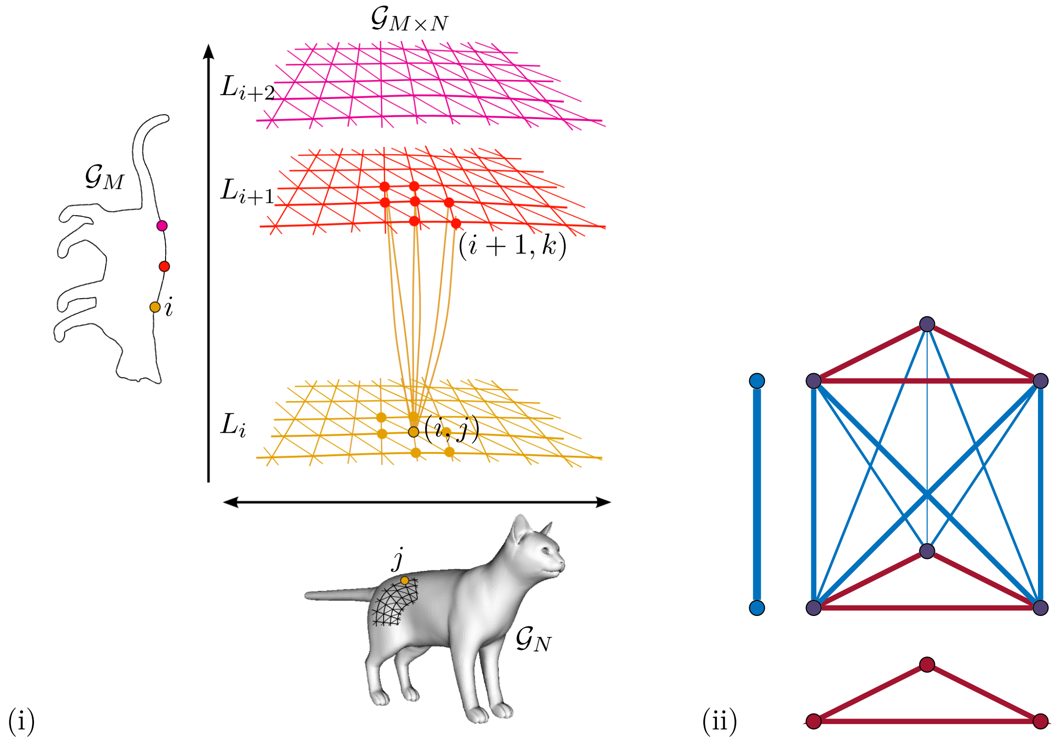


Figure 7.3: Connections in the product graph. (i) A node (i,j) in the product graph $\mathcal{G}_{M\times N}$ represents a match between the vertex $i \in \mathcal{V}_M$ of the contour M and the vertex $j \in \mathcal{V}_N$ of the surface N. All feasible matches with respect to vertex i form the layer $L_i = \{i\} \times \mathcal{V}_N$. Edges are defined between a node (i,j) and $(i+1,j)$ as well as (i,k) and $(i+1,k)$ for all surface vertices $k \in \mathcal{V}_N$ that are adjacent to j. All these edges are directed and enforce a continuous matching. (ii) Different from Section 5.3 we take the product of a line and the edges of each triangle, not the triangle itself, because this results in a graph and we want to apply a graph algorithm.

7.3 Discrete Problem Formulation

In the discrete case we represent the contour $\mathcal{M}$ by a series of line segments such that $M = (V_M, E_M)$ and $V_M = \{0, \dots, m\}$ and $E_M = \{(i_0, i_1) \in V_M \times V_M | i_0 \in V_M \text{ and } i_1 = (i_0 + 1) \bmod (m+1)\}$. The 3D shape $\mathcal{N}$ is represented by a triangle mesh $N = (V_N, F_N, E_N)$ although we will not make use of F_N here. Solving for the optimal correspondence could be formulated through a linear program, as has been done in **Windheuser et al.** (2011a) for two 3D shapes. This case has one major advantage over the setting in **Windheuser et al.** (2011a) because we know that the solution is 1-dimensional itself (see Section 6.2.1). Optimization problems where the dimension or co-dimension of the solution is 1 can be solved more efficiently, and in this case we see that our problem can be formulated as a shortest path problem.

As discussed in Section 5.3 the discrete product would include the product of each face of M and N, so the product of a each line segment and a triangle. The resulting components are

3-dimensional and have the form of cylinders with triangle bases (see Figure 5.3). In order to be able to apply a graph algorithm later on, we choose to only use the edge information of the triangle mesh, which directly translate to an undirected graph, and use the product of these two graphs. We use the strong product of both graphs which can be seen in Figure 7.3. Using the direct product would enforce the non-collapsing property of immersions but in the discrete case this is only meaningful if the sampling density of both inputs is comparable. See Definition 10 and Definition 9 for the difference between the graph products. This is not given in most data, therefore we choose to allow collapsing. However, as we will see later on, a collapsed area does not have zero energy but is instead penalized and this is the reason why this relaxation does not lead to zero solutions.

Finally, the product graph of M and N looks as follows:

$$\begin{aligned} V_{M\times N} =& \{0,\ldots,m\} \times \{0,\ldots,n-1\} && (7.6)\\ E_{M\times N} =& \big\{[(i_0,j_0),(i_1,j_1)] \in V^2_{M\times N} \big| && (7.7)\\ & (i_1 = i_0) \wedge (y_{j_1}, y_{j_0}) \in E_N \text{ or} && (7.8)\\ & (i_1 = i_0+1) \wedge (j_1 = j_0) \text{ or} && (7.9)\\ & (i_1 = i_0+1) \wedge (y_{j_1}, y_{j_0}) \in E_N\} && (7.10) \end{aligned}$$

An edge between two vertices $v_{i,j}, v_{k,l}$ exists if both v_i^M, v_k^M and v_j^N, v_l^N are either the same or neighboring on M and N respectively. Therefore, the adjacency information of M, N is preserved in the product graph.

By leveraging the product graph and the fact that the solution is a 1D path, the optimal path can be found efficiently by **Dijkstra** (1959) 's algorithm. This prevents non vertex-to-vertex solutions which might be reasonable, for example, a vertex lying in the middle of a face but improves computational efficiency drastically. It does still allow shrinking and stretching of the solution through multi matches.

7.4 Optimization

The problem has many similarities to a shortest path problem. But instead of a fixed source and target point, the path should start at an unknown, optimal vertex on the product graph, go around the product graph once visiting one match for each point on $\mathcal{M}$ and end up in the first vertex again. Assuming the starting point is given, the product graph can be "cut open" and the solution will be the shortest path of this point to a copy of itself on the other side. We will use branch-and-bound to find the optimal starting point instead of considering it given.

We propose both a method to find the global optimum in Section 7.4.1 as well as an arbitrary good approximation in Section 7.4.2. Finding the approximation is more efficient in terms of run time, and we saw in our experiments that many approximate solutions are qualitatively comparable to the global optimum.

Furthermore, we will be interested in *edge-wise* instead of point-wise costs to solve the shortest path problem with **Dijkstra** (1959) 's algorithm. We use the distances as stored in D at each endpoint of one edge $[(i_0, j_0), (i_1, j_1)] \in \mathcal{E}_{M\times N}$

$$\begin{aligned} D_{i_0,j_0} &= \operatorname{dist}\left(f_M(x_{i_0}), f_N(y_{j_0})\right) \\ D_{i_1,j_1} &= \operatorname{dist}\left(f_M(x_{i_1}), f_N(y_{j_1})\right) \end{aligned}$$

and linearly interpolate the costs along the edge. This is equal to integrating over the average of both values and results in the cost function

$$\begin{aligned} &\mathcal{C}_{M\times N}[(i_0, j_0), (i_1, j_1)] = \\ &\qquad \frac{D_{i_0,j_0} + D_{i_1,j_1}}{2} \cdot \left\|(x_{i_0}, y_{j_0}) - (x_{i_1}, y_{j_1})\right\|. \end{aligned}$$

(x, y) is a 5D-coordinate with the stacked coordinate values from $x \in M$ (2D) and $y \in N$ (3D). This is a discretization of the line integral between both vertices from equation 7.1.

7.4.1 Global Optimization

To solve the shortest path problem, a fixed source and target set is needed. We have no information about which vertices are contained in the solution but to have a circular path, we know that each $x \in M$ has to represented at least once in the solution. Therefore, the representation of the 2D shape M is cut at an arbitrary x and is extended by having two copies of x_0, namely at position $i = 0$ and at position $i = m$. As a result, any continuous matching can be represented by a path from $(0, j)$ to (m, j). Hence, an optimal matching can be cast as finding a shortest path in a graph if an initial match $(x_0, y_j) \in \Gamma_\varphi$ is given. Such a computation can easily be done **Dijkstra** (1959) 's algorithm. Using a priority heap the computation takes $\mathcal{O}(mn \cdot \log(mn))$ steps. Since there is no path from (i_1, j_1) to (i_0, j_0) if $i_1 > i_0$, we associate to each *layer* $\{i\} \times \{0, \dots, n-1\}$ a different priority heap and reduce the runtime to $\mathcal{O}(mn \log(n))$. These observations lead to the following theorem and the fact that we have to test n different initial matches.

Theorem 2. *Given a 2D query shape M and a 3D target shape N, discretized by m and n vertices, respectively, we can find a minimizer of* (7.1) *in $\mathcal{O}(mn^2 \log(n))$ steps. If $n = \mathcal{O}(m^2)$, this leads to the subcubic runtime of $\mathcal{O}(n^{2.5} \log(n))$.*

This theorem shows that we can find a globally optimal matching in polynomial time. Nonetheless, this may still lead to a high runtime since we have to find for each vertex $y \in \mathcal{V}_N$ a shortest path in $\mathcal{G}_{M\times N}$. In order to circumvent this problem we follow a branch-and-bound strategy inspired by the method of **Appleton and C. Sun** (2003).

The main idea is to follow a coarse-to-fine strategy. First we compute the shortest path between the sets $\{0\} \times R$ and $\{m\} \times R$, which connects $(0, i)$ with (m, j). In the first iteration,

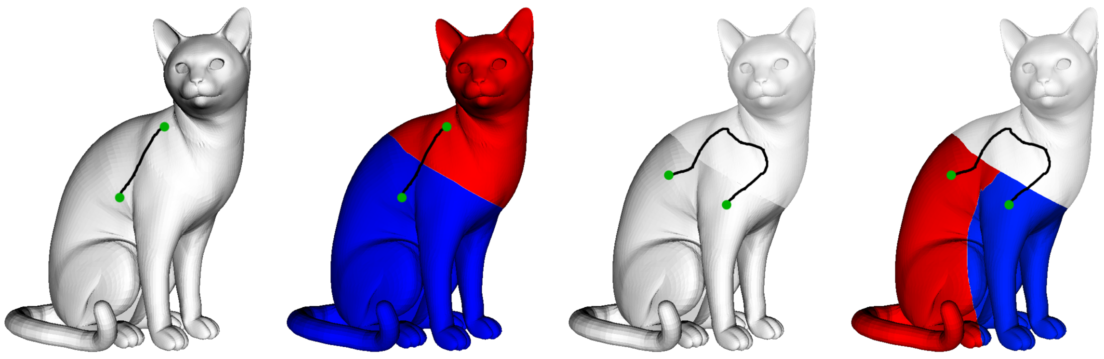


Figure 7.4: Exemplary first two iterations of the branch-and-bound algorithm. Paths are projected from the product manifold on the 3D shape. (Left) Absolute energy minimal path p_1 through the cut product manifold. (Middle Left) 3D Shape is separated in two subareas. One containing the source and the other the sink point of p_1. (Middle Right) In the next iteration one area is chosen; blue in this example but red will be processed later in the same way. The source and sink of the optimal path p_2 have to lie in the blue area. The rest of the path can go anywhere. This excludes the previous minimum p_1. (Right) If the source and sink of p_2 do not conincide, the blue area is again split in two.

we set $R = \mathcal{V}_N$. In this case i and j do not need to coincide but the energy of this path provides a lower bound on the optimal closed solution. If this path connects the corresponding points, *i.e.* $i = j$, we found a valid path. Otherwise, we separate the region R into two sub-regions, R_1 containing i and R_2 containing j, and recompute shortest paths starting and ending in $\{0\} \times R_i$ and $\{m\} \times R_i$ respectively. Since the shortest path of corresponding sub-regions will exclude the previously computed path, the optimal path in both R_1 and R_2 will have a larger energy than from the previously computed path. Thus, a natural order in which the subregions should be processed is induced. We propose to separate R with respect to the geodesic distance dist_N on the target shape N. We continue this process until we find our first matching path with $i = j$. Then, we still have to process those subdomains whose lower bound is smaller than the computed matching path. Afterwards, we are sure we found the globally optimal matching path Γ_φ. See Figure 7.4 for an illustration of one iteration.

7.4.2 Approximation

We observe that in practice already a few iterations of Algorithm 1 suffice if the features are distinctive and give a clear optimum. But in some cases, especially interclass matchings where the global optimum is not natural, branch-and-bound is converging very slowly because many equally bad solutions exist. To circumvent this we implemented an extension using both an upper and lower bound of the optimal solution. If both are within a certain range of each other, the branch-and-bound stops. This guarantees a solution that is provably close to the optimum.

Algorithm 1 2D-to-3D matching via branch-and-bound.The blue code denotes the addition needed for the approximation algorithm.

```
Input: $\mathcal{G}_{M\times N} = (\mathcal{V}_{M\times N}, \mathcal{E}_{M\times N}, \mathcal{C}_{M\times N})$, $\epsilon$
Output: Matching path $\Gamma_\varphi$
    Let $R := \{0, \dots, n-1\}$ ;
    Define $= \{R\}$ and $b\colon \to \mathbb{R}$ via $b(R) = 0$ ;
    Define isFound=false ;
    while isFound=false do

        lowerbound=$\min b$ ;
        Let $R \in \arg\min b$; $= \setminus\{R\}$ ;
        Find shortest path $\Gamma$ in $\mathcal{G}_{M\times N}$
         from $\{0\} \times R$ to $\{m\} \times R$ ;
        $\Gamma$ is a path from $(0, i)$ to $(m, j)$ ;
        if $i = j$ then
            isFound=true;$\Gamma_\varphi = \Gamma$
        else
            Find indices $l, k \in R$ with minimal Euclidean distance such that the shortest path
            in $\mathcal{G}_{M\times N}$ to $(m, k)$ originates in $(0, l)$ ;
            upperbound $= dist((0,k),(m,l)) + dist((m,l),(m,k))$ in $\mathcal{G}_{M\times N}$
            if lowerbound $\geq (1-\epsilon)\cdot$ upperbound then
                $\Gamma_\varphi = dist((0,l),(m,l))$ ; break ;
            end if
                Divide $R$ into $R = R_1 \cup R_2$ such that
                    $x \in R_1$:
                $\Leftrightarrow \mathrm{dist}_N(x,i) < \mathrm{dist}_N(x,j)$ ;
                Set $:= \cup\{R_1, R_2\}$ ;
                Set $b(R_1) = b(R_2) = \mathrm{length}(\Gamma)$ ;
                if $\mathrm{length}(\Gamma) \leq \min b$ then
                    isFound=true ;
                end if
        end if
        end while
```

The lower bound comes, as in the previous section, from paths with $i \neq j$. The upper bound is obtained by choosing the open path in the current region whose endpoints $(0, i)$ and (m, j) have the smallest Euclidean distance between i and j. We then extend the path on the product manifold via Dijkstra to be a closed path. Because each closed path is a valid solution the energy of this path is an upper bound for the global optimum. Notice that there are no guarantees on the tightness of this upper bound. If the upper bound is within some chosen ϵ of the lower bound from the previous region (see global optimization) we know we are close to the global optimum and stop. The ϵ can indicate the absolute or relative error but we choose it as the relative one to be insensitive to the scale of the energies.

To keep the solution as close to the optimum as possible, we recompute the path between the

$(0, i)$ and (m, i) that closed the distance between the bounds. As a result of approximating the closing of the path from (m, j), a different path between $(0, i)$ and (m, i) might have a lower energy. We keep the starting point as the initial match and recompute the shortest path in the product manifold. Because we can compute the optimum with respect to a given starting point, the energy of this path will be lower or equal to the energy of the upper bound.

7.5 Trans-Dimensional Features

Defining descriptors between dimensional structures with different dimensions is challenging because the geometry cannot be identical by construction. Therefore, a comparison has to be done with a clear idea of what relationship exists between the different dimensions or metric learning can be applied as in **B. Li et al.** (2015). In the following, we argue for the adoption of spectral quantities for applications including 2D contours of 3D shapes. Note that differently from existing methods for 2D-to-3D correspondence, we compute local features independently for each given pair of shapes, *i.e.* no cross-modal metric learning is carried out. Our approach is based on the observation that the contour describes the boundary of what is supposedly visible of the 3D shape. This implies that the contour is a kind of projection of the 3D geometry. We do not restrict ourselves to perspective projection though because it is not invariant to pose changes. This means the interior of the contour is actually a meaningful shape and (at least in parts) has a meaningful counterpart on the 3D surface.

7.5.1 Spectral Features

Spectral properties are often used in non-rigid shape analysis due to their isometry invariance. They have not been adapted for the 2D-3D case, in large parts because different dimensional structures are not isometric. Features like curvature can work in between dimensions but are not suitable for deformable shapes. This makes this setting extremely challenging. Learning features invariant to dimension and pose might be possible but it has not been done before.

To this end, we consider the solid U of M, *i.e.* $\partial U = M$ (cf. Definition 7). In other words, we model the 2D query as a flat 2-manifold *with* boundary. This new manifold can be regarded as a nearly isometric transformation (due to flatness and possibly a change in pose) of a *portion* of the full 3D target (see Fig. 7.5). Taking this perspective allows us to leverage some recent advances in partial 3D matching (**Rodolà et al.** 2016), namely that partiality transformations of a surface preserve the Laplacian eigenvalues and eigenfunctions, up to some bounded perturbation.

An implication of this is that we can still compute spectral descriptors on the flat solid U and expect them to be comparable with those on the full 3D target. By doing so, we make the assumption that U can be approximated as a part of nearly-isometrically deformed M for the features to be comparable. We then define the feature maps $f_M^{\text{HKS}}, f_M^{\text{WKS}} \colon M \to \mathbb{R}^d$ on M by restricting the descriptors computed on U to its boundary curve $\partial U = M$ (see Fig. 7.6, top row). In other words, we have the spectral features $f_M = f_U|_M$ for the query shape M.

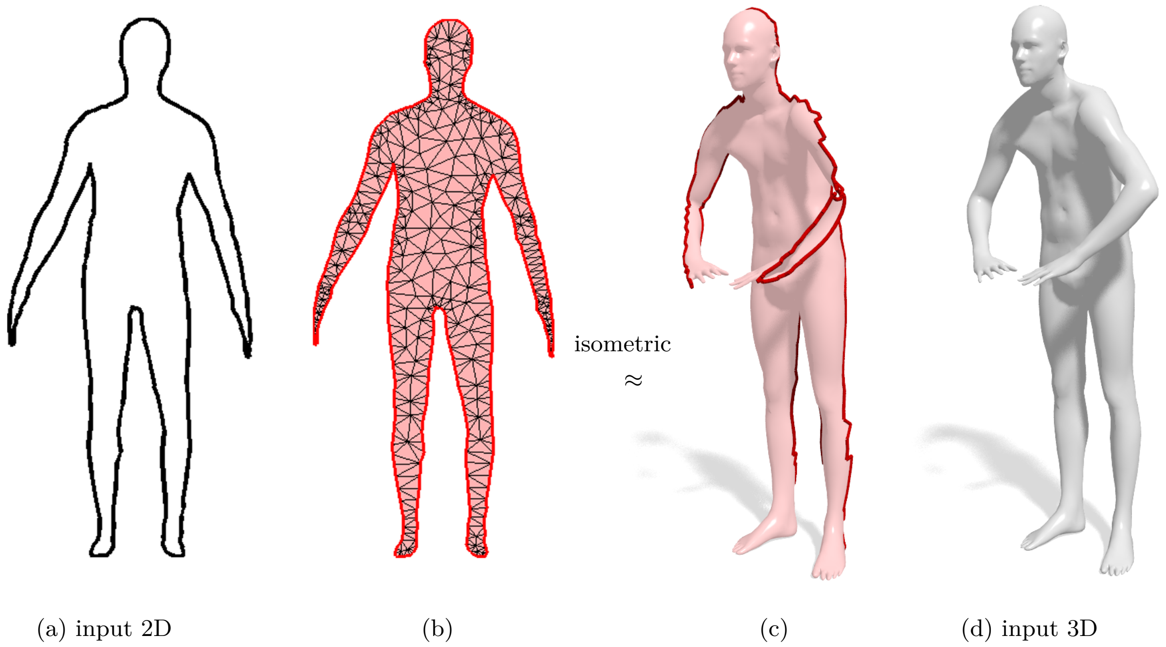


Figure 7.5: Spectral features are constructed by considering the query shape (a) as the boundary of a 2D region (b), which is assumed to be a near-isometric deformation of a sub-region (c) of the 3D target shape (d). The 2D tessellation (b) is obtained via **Shewchuk** (2002).

Popular spectral features are the scaled Heat Kernel Signature (HKS) (**J. Sun et al.** 2009) and the Wave Kernel Signature (WKS) (**Aubry et al.** 2011),

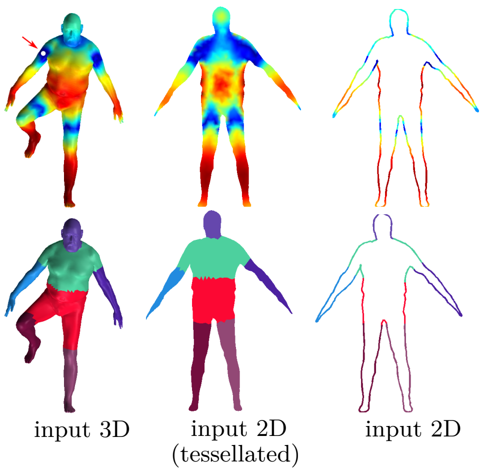
$$f_k^{\mathrm{HKS}}(x) = \sum_{j\geq 1} e^{-t_k\lambda_j}\psi_j^2(x), \tag{7.11}$$

$$f_k^{\mathrm{WKS}}(x) = \sum_{j\geq 1} e^{-\frac{(\log e_k - \log\lambda_j)^2}{2\sigma^2}}\psi_j^2(x), \tag{7.12}$$

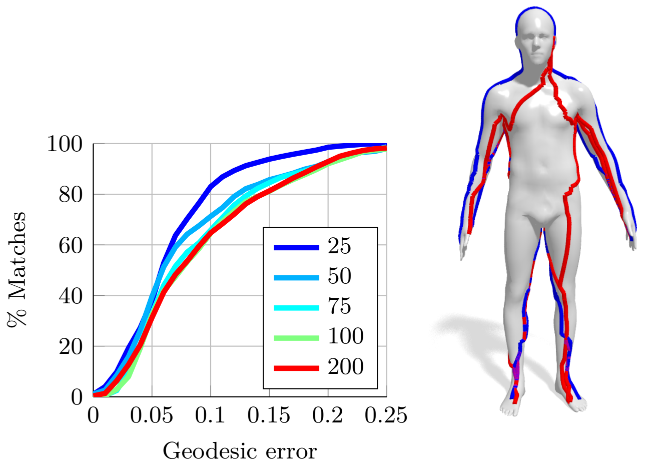
where $k = 1, \ldots, d$ are the dimensions of the descriptors. We use both without adjustments directly on N and M by using the equality $\partial U = M$. In our experiments, we used $d = 100$ and the parameters $t_1, \ldots, t_d$ of HKS and $e_1, \ldots, e_d, \sigma$ of WKS were taken as suggested by the respective authors. All descriptors were normalized to have maximum value 1 in order to improve their robustness.

7.5.2 Segment Features

Spectral properties are inheritly unaware of intrinsic symmetries and therefore any feature derived from them will also be symmetry invariant. As shown in Figure 7.9 our features suffer from the same ignorance. To improve the visual quality of our result, we use a 'coarse' feature of the corresponding regions on the 2D query and the 3D shape. This is purely



a) Spectral (top) and Segment (bottom) features in 2D and 3D.



b) Evaluation of number of eigenfunctions used in spectral feature calculation.

Figure 7.6: (a) Computation of local features for elastic matching. *Top*: L_1-distance (blue to red) between spectral 3D descriptors of a reference point (white dot) and 3D descriptors computed on the remaining shape (left) as well as 2D descriptors on the tessellated query solid (middle). *Bottom*: Consensus regions detected in 3D and 2D using **Rodolà et al.** (2014b). The restricted features (right) are used in the energy (7.1) to drive the matching process. (b) Sensitivity of the spectral features to increasing number of eigenfunctions. On the right we show typical solutions obtained when using 25 (blue) and 200 (red) eigenfunctions.

for visual reason and we show in Section 7.6.3 that the retrieval results are not affected by not using this feature. Based on the previous observations about spectral compatibility, we are able to automatically extract compatible regions on the two objects (namely on U and N) by consensus segmentation (**Rodolà et al.** 2014b), a deformation-invariant region detection technique which directly operates with the Laplace-Beltrami eigenfunctions of a given shape. The region detection step on the two shapes is performed independently; we then obtain the 2D-to-3D region mapping by solving a simple linear assignment problem via the Hungarian algorithm (**Munkres** 1957). Note that this assignment problem is typically very small. Assuming we have r regions per shape (usually in the range of 5 to 10), the final result of this procedure is a pair of corresponding labelings $f_M^{\text{SEG}} : M \to \mathbb{N}^r$ and $f_N^{\text{SEG}} : N \to \mathbb{N}^r$ (see Fig. 7.6, bottom row).

7.5.3 Distance function

The final feature maps are obtained by simple concatenation, namely $f := (f^{\text{HKS}}, f^{\text{WKS}}, f^{\text{SEG}})$. In order to compare the feature maps on M and N, we define the distance function:

$$\begin{aligned} \text{dist}(f_M(x), f_N(y)) &= \|f_M^{\text{HKS}}(x) - f_N^{\text{HKS}}(y)\|_1 \\ &+ \|f_M^{\text{WKS}}(x) - f_N^{\text{WKS}}(y)\|_1 \end{aligned} \tag{7.13}$$

if $f_M^{\text{SEG}}(x) = f_N^{\text{SEG}}(y)$, and set $\text{dist}(f_M(x), f_N(y)) = \tau$ otherwise. Here, $\tau > 0$ is a large positive value to prevent matching points belonging to different regions. In our experiments, we used $\tau = 10^3$.

7.5.4 Sensitivity analysis

In most shape analysis applications, only the first k eigenfunctions of Δ are used to define f^{WKS} and f^{HKS}. In the classical 3D-to-3D setting, for large k the resulting descriptors tend to be more accurate, but at the same time become more sensitive to the lack of isometry relating the two shapes. We performed a sensitivity analysis of our elastic matching method on a subset of our FAUST-derived dataset with annotated ground-truth to determine the optimal number of eigenfunctions.

We observed a similar trend to the 3D-to-3D case in our 2D-to-3D setting where naturally the isometry is only approximate giving better results for a low k, as reported in Fig. 7.6. From this analysis we selected $k = 25$ as the fixed number of eigenfunctions for all subsequent experiments. In the figure, we plot cumulative curves showing the percent of matches with a geodesic error smaller than a variable threshold (see Eq. (7.14)).

7.6 Experiments

In this section, we apply the proposed method to the problem of sketch-based deformable shape retrieval. We emphasize that our method is parameter-free, and the only choice is with respect to the 2D and 3D features $f_M(\cdot), f_N(\cdot)$ as well as the distance function $\text{dist}(\cdot, \cdot)$ between them. These choices depend on the specific application and help to illustrate the flexibility of our general 2D-to-3D shape matching approach.

7.6.1 Setting

We consider a particular shape retrieval setting in which the dataset is assumed to be a collection of 3D shapes and the query is a 2D silhouette (possibly drawn by a human) represented by a closed planar curve. Differently from previous techniques (**Eitz et al.** 2012; **Furuya and Ohbuchi** 2014; **B. Li et al.** 2015; **H. Su et al.** 2015), our method does not use learning to compute features and most importantly, we allow the shapes to deform in a non-rigid fashion. The 2D-to-3D shape similarity is obtained by considering the minimal value of the energy $E(\varphi^*)$ obtained by our optimization problem.

Datasets. Due to the novelty of the application, to date there is no benchmark available for evaluating deformable 2D-to-3D shape retrieval methods. We therefore construct such a benchmark using the FAUST (**Bogo et al.** 2014), TOSCA (**A. M. Bronstein et al.** 2008) and a subset of Non-rigid world (**A. M. Bronstein et al.** 2006a) (additional poses for TOSCA, gorilla, lioness and seahorse) datasets. The FAUST dataset consists of 100 human shapes, subdivided into 10 classes (different individuals), each in 10 different poses. The latter

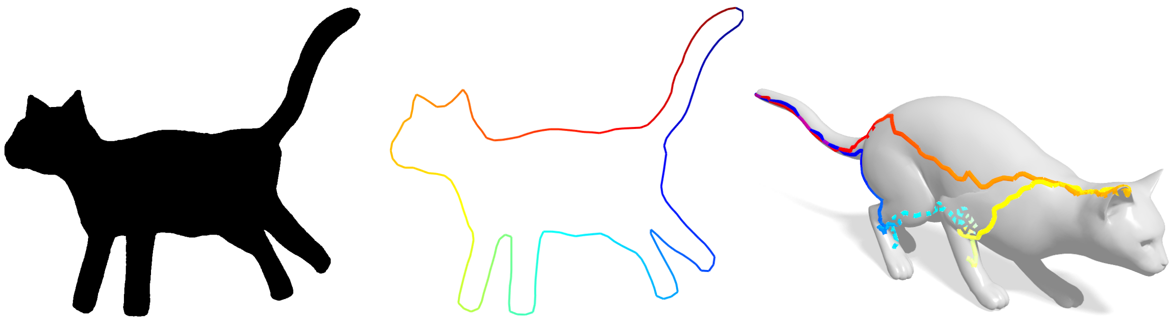


Figure 7.7: Handdrawn query for a cat shape and the extracted contour from the image. The image was simply thresholded to get a binary segmentation and then the contour extracted. Personal experience showed that drawing a solid silhouette is easier than directly drawing the contour. This also guarantees a shape as defined in Def. 7 without self-intersections or other degenerations.

two consist of over 100 shapes, subdivided into 12 classes (humans and animals in different poses). Shape sizes are fixed to around 7K (FAUST) and 10K (TOSCA) vertices.

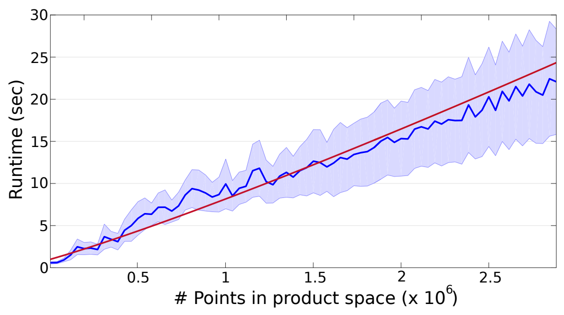
In FAUST each class comes with a ‘null’ shape in a “neutral pose”, where no deformation has been applied, which we use to define the 2D queries. To this end we cut each null shape across a plane of symmetry and project the resulting boundary onto a plane. This gives rise to 2D queries of 200-400 points on average. Note that by doing so we retain the ground truth point-to-point mapping between the resulting 2D silhouette and the originating 3D target. This allows us to define a quantitative measure on the quality of the 2D-to-3D matching between objects of the same class. In the extended dataset, silhouettes of one shape showing all important extremities of the class are produced and the 2D query is extracted from the binary image. Hence, no point-to-point but only class ground truths are available. The same method can be applied to hand-drawn silhouettes[1].

As an addition to the queries produced through the actual 3D data, we drew a human by hand and used this as an additional query for the retrieval to show that the method also works on queries not produced using the targets. See Figure 7.7 for the sketch and query.

Error measure. Let M be a 2D shape represented as a planar curve and N a 3D shape represented as a surface. Let $\varphi\colon M \to N$ be a matching between a 2D query shape and a 3D target shape, and let φ_0 be the ground-truth matching. The matching error of φ at point $x \in M$ is given by

$$\varepsilon_\varphi(x) = \frac{\mathrm{dist}_N(\varphi(x), \varphi_0(x))}{\mathrm{diam}(N)}, \tag{7.14}$$

[1]The dataset containing shapes and matchings is publicly available at https://vision.in.tum.de/~laehner/Elastic2D3D/



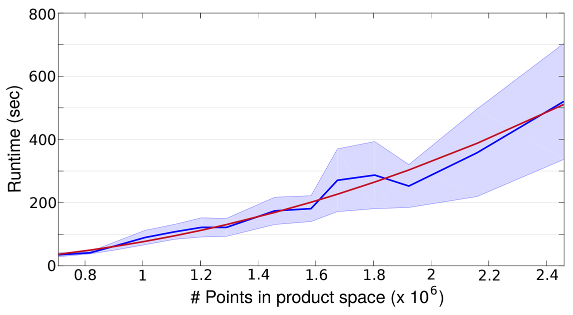


Figure 7.8: Blue: Mean runtime with standard deviation shaded. Red: Optimal curve of form $a \cdot x^b$ fitted to the data. (Left) Runtime of our matching method on 3D targets with ∼7k vertices (FAUST dataset). On the x axis we vary the size of the 2D query from 25 to 400 points. The fitted curve has $a = 0.0245$ and $b = 1.1518$ so the real runtime is nearly linear in m. (Right) Runtime with a 100 vertex query and upsampled FAUST shapes. On the x axis we vary the size of the 3D targets from 7k to 24k. The fitted curve has $a = 2.6466 \cdot 10^{-7}$ and $b = 2.1147$ making the real runtime nearly quadratic in n with a really small constant.

where $\text{dist}_N : N \times N \to \mathbb{R}_0^+$ denotes the geodesic distance on N and $\text{diam}(N) = \max_{x,y \in N} \text{dist}_N(x, y)$ the geodesic diameter of N. Note that due to the normalization, the values of the error ε are within $[0, 1]$.

7.6.2 Runtime

We implemented our method in C++ and ran it on an Intel Core i7 3.4GHz CPU. In Figure 7.8 we show the execution times of our method on the FAUST dataset (10 queries and 100 targets). The plotted results show that for 3D shapes of fixed size, in practice the runtime grows linearly with the number of 2D query points m and quadratic with the number of points on the 3D target n.

Figure 7.9 shows the decrease of the average runtime with growing parameter ϵ for the approximation algorithm. The plot shows that after decreasing rapidly over $\epsilon \approx 0.05$ the runtime stays relatively constant with no change in the MAP. Therefore, choosing a small ϵ leads to a good quality-runtime ratio.

7.6.3 Sketch-based shape retrieval

The 2D-to-3D shape similarity is obtained by ranking the matching energy $E(\varphi^*)$ obtained by our optimization problem. We evaluated the performance of our retrieval pipeline on the extended 2D-to-3D TOSCA dataset. As baselines for our comparisons we use the spectral retrieval method of **Reuter et al.** (2006) and a pure region-based retrieval technique using segments computed with **Rodolà et al.** (2014b); both are the foundation for the features we use. The rationale of these experiments is to show that these features are not sufficient to

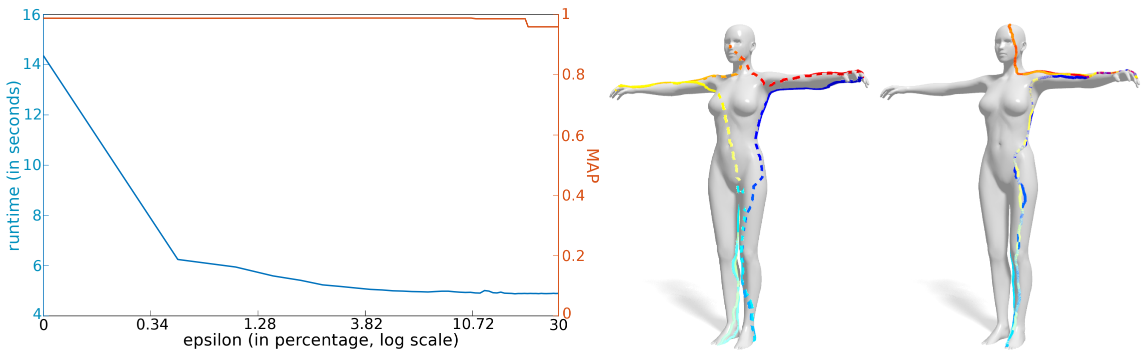


Figure 7.9: (Left) The average runtime of the TOSCA dataset with values between 0 (globally optimal) and 0.3 (within 30% of the optimum). The runtime nearly stagnates after $\epsilon = 0.05$ (only one or two iterations) but small ϵ already lead to significant improvement in the runtime and comparable results (as shown in Table 7.1). The segmentation feature was not used. (Right) Example of the same query matched to the same shape, once using the segmentation features (left) and once without (right). Dashed lines indicate a path on the backside normally not visible in this perspective. The matchings are very similar but without segments the path only considers one of the symmetric sides (normally because it is slightly shorter).

guarantee good retrieval performance. However, using these quantities in our elastic matching pipeline enables promising results even in challenging cases.

The first baseline method we compare against is Shape-DNA (**Reuter et al.** 2006) using the (truncated) spectrum of the Laplace-Beltrami operator as a global isometry-invariant shape descriptor. We apply this method to compare targets in the 3D database with flat tessellations of the 2D queries.

The second method used in the comparisons is a simple evaluation of the matching cost obtained when putting the consensus regions into correspondence via linear assignment (see Fig. 7.6). Since this step typically produces good coarse 2D-to-3D matchings, it can be used as a retrieval procedure per se.

We test the global optimization using all features we presented before. For the approximate optimization we use all features except the segment features. The reason is we noticed that the energies without the segments are comparable for same-class matching; the only difference being the matchings only take place on one symmetric half of the 3D shape (see Fig. 7.9). Although this is not the favored solution, since it does not substantially change the energy, it is suitable for retrieval. Unfortunately, without the coarse matching of the segments the globally optimal solution for interclass matchings can take longer to converge because there are many equally good solutions - still with a high energy in comparison. Therefore, we use the segment features for the globally optimal method but show that we can get similar results without this feature using the approximate method.

The results of the shape retrieval experiments are reported in Table 7.1. $\epsilon = 0.001$ was used for the approximation experiments. We used average precision (AP) and mean average precision

	Global	**Approx.**	sDNA	Consensus
cat	**1.0000**	**1.0000**	0.2852	0.2518
dog	**1.0000**	0.9615	0.3519	0.3970
horse	**0.9987**	0.9313	0.4469	0.3432
human	**1.0000**	**1.0000**	0.7447	0.9985
lioness	**0.9984**	0.9587	0.7022	0.5419
seahorse	**1.0000**	**1.0000**	0.0779	**1.0000**
wolf	**1.0000**	0.8082	0.2230	0.2470
human (hd)	**1.0000**	**1.0000**	0.7096	0.9462
cat (hd)	**0.9888**	0.9772	0.8622	-
MAP	**0.9984**	0.9597	0.4720	0.5907

Table 7.1: Retrieval results on the 2D-to-3D extended dataset. For each method we show per-class AP and, in the last row, the MAP. The global algorithm uses all features and the approximation does not use the segmentation. ϵ was chosen to be 0.001. hd refers to handdrawn sketches. sDNA refers to **Reuter et al.** (2006) and Consensus to **Rodolà et al.** (2014b).

(MAP) as measures of retrieval performance[2]. The results slightly differ from the ones reported in **Lähner et al.** (2016b) due to a small bug in the code. Additional qualitative examples of solutions obtained with our method are shown in Fig. 7.10.

7.7 Conclusion

This chapter introduced a polynomial-time solution for matching deformable planar contours to 3D shapes. Although the different dimensionality poses additional problems in terms of comparing features and defining the optimal solution, the one dimensional structure of the submanifold of the solution allows us to perform an efficient optimization using Dijkstra's algorithm. Additionally, the output is guaranteed to be a continuous correspondence. This is due to the co-dimension of the problem being 1. In the next chapters, we will see that in the higher dimensional case, where the co-dimension is 2, the problem cannot be solved efficiently to the global optimum anymore.

While the manifold structure may limit the expressiveness of the sketch in some cases, it allows to prove that the worst-case complexity of this algorithm is $\mathcal{O}(mn^2 \log(n))$, where m and n denote the number of samples on the query curve and the 3D shape respectively. But our experiments show that in practice the runtime remains linear with respect to m, even when employing a branch-and-bound strategy, making this a very efficient approach that matches queries with hundreds of vertices to 3D shapes with over $10k$ vertices in a few seconds. Additionally, the algorithm can compute the globally optimal or a ϵ-tight solution.

[2] *Precision* measures the percentage of correctly retrieved shapes.

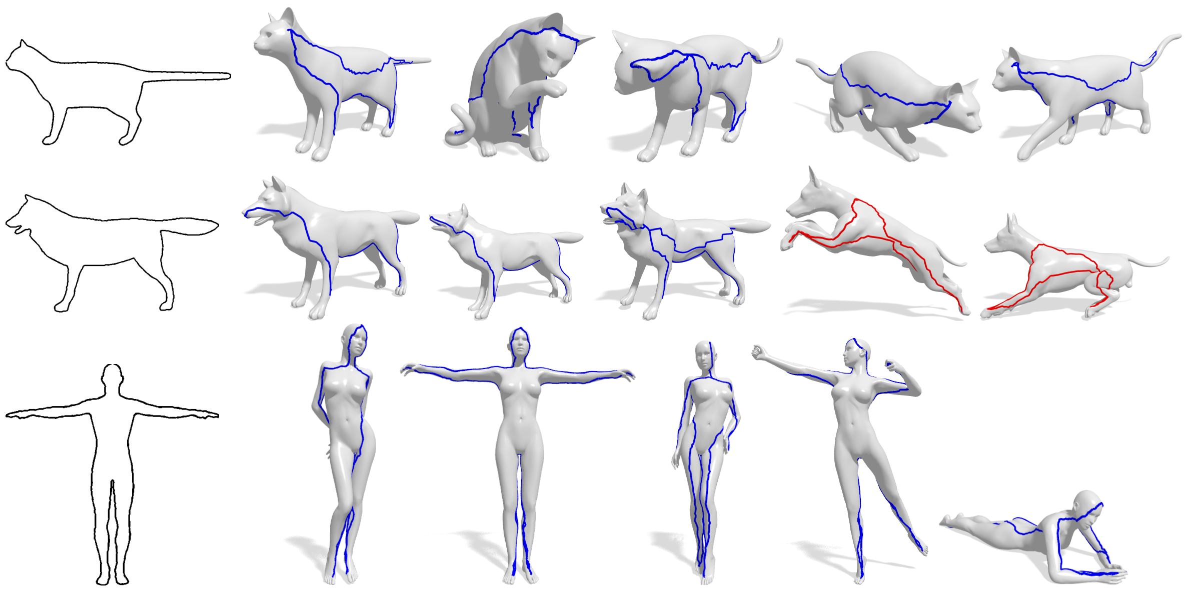


Figure 7.10: Retrieval examples on the TOSCA dataset. Two of the three 2D queries (cat and wolf) have missing parts (two legs in contrast to Fig. 7.7). Each row shows the top 5 results (ranked by matching energy) provided by our method. The corresponding matching curves are shown on top of the 3D targets. Note that the dataset only contains 3 wolf shapes, which show up as the top 3 matches. The next matches are shapes of the class “dog”, which is a semantically similar class.

The approximate method is more efficient if the features are not discriminative enough which causes many iterations in the branch-and-bound strategy. An interesting extension would be to replace the product graph with the proper discrete product manifold and use a fast marching algorithm to find the optimal path. While this would increase the complexity, it is a more faithful discretization of the geodesic in the product space. Additionally, it can return correspondences that are immersions, even under inconsistent sampling.

CHAPTER 8

Kernel Matching between 2D Manifolds

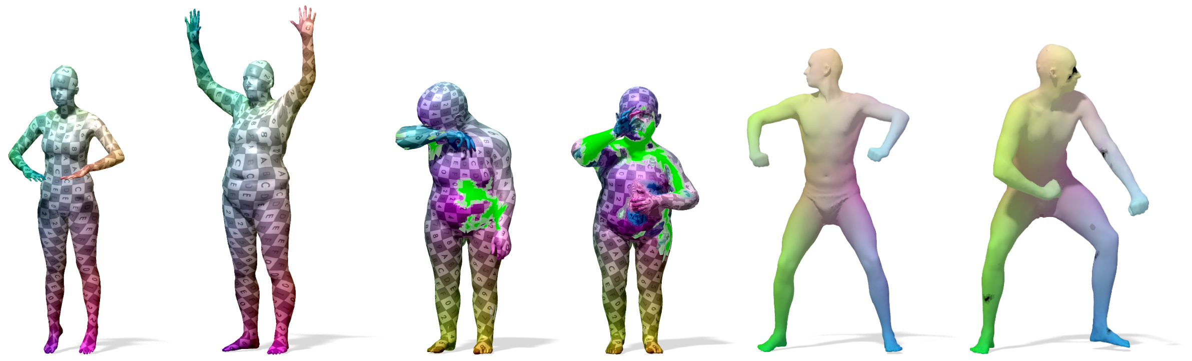


Figure 8.1: Qualitative examples on FAUST models (left), SHREC'16 (middle) and SCAPE (right). In the SHREC experiment, the green parts mark where no correspondence was found. Notice how those areas are close to the parts that are hidden in the other model. The missing matches marked in black in the SCAPE pair are an artifact due to the multiscale approach.

This chapter looks at the problem of correspondence between two two-dimensional manifolds instead of a one- and two-dimensional manifold as in the last chapter. The previous method utilized the fact that the submanifold describing an diffeomorphism (see Chapter 6) has the same dimension as the lowest dimensional input, for contours this means one-dimensional. In the case of two two-dimensional manifold, both the dimension of the solution as well as the co-dimension to the solution space are two. This leads to a more complex problem. **Windheuser et al.** (2011a) formulated the solution as an Integer Linear Program (ILP) using similar geometric consistency constraints as we did in the previous chapter, see Section 6.3.3. Unfortunately, solving ILPs is NP-complete, and the optimization takes a few hours for meshes with only $1k$ vertices. The method introduced in this chapter has a similar interpretation optimizing for a submanifold in the product space, but we proposed an efficient optimization trading guarantees on the optimality of the solution against applicability to arbitrarily high resolutions. The geometric interpretation of our method is discussed in detail in Section 8.4.

This chapter is based on **M. Vestner***, **Z. Lähner***, **A. Boyarski***, **O. Litany**, **R. Slossberg**, **T. Remez**, **E. Rodolà**, **A. M. Bronstein**, **M. M. Bronstein**, **R. Kimmel**, and **D. Cremers** (Oct. 2017). Efficient Deformable Shape Correspondence via Kernel Matching. In: *International Conference on 3D Vision (3DV)*. The major novelty is a simple method that works out-of-the-box for finding a high quality, continuous correspondence between two not necessarily isometric shapes. The method is an improved version of **Vestner et al.** (2017) and adds scalability, and superior theoretical properties. Additionally, we provide theoretical insights and several possible mathematical interpretations of the algorithm that shed light on its effectiveness. In particular, we contrast the method with other shape matching approaches and elaborate on the computational benefits of using kernels rather than distances as pairwise descriptors. The key insight is the realization that a high quality regular correspondence can be obtained from a rough irregular one by a sequence of smoothing and projection operations. We report drastic runtime and scalability improvements compared to **Vestner et al.** (2017), and present an extension to the setting of partial shape correspondence as well as an effective multi-scale approach.

8.1 Related work

This section focuses on work related directly to this chapter, see Chapter 1 for a more general survey. Finding correspondences between shapes is a well-studied problem. Traditionally, the solution involves minimization of a distortion criterion which fits into one of the two categories: pointwise descriptor similarity (**Aubry et al.** 2011; **M. M. Bronstein and Kokkinos** 2010; **J. Sun et al.** 2009), or pairwise relations (**Chen and Koltun** 2015; **Coifman et al.** 2005; **Mémoli and Sapiro** 2005; **Torresani et al.** 2008). In the former case, matches are obtained via nearest neighbor search or, when injectivity is required, by solving a linear assignment problem (LAP). Pairwise methods usually come at a high computational cost, with the most classical formulation taking the form of an NP-hard *quadratic assignment problem* (QAP) (**Pardalos and Wolkowicz** 1994). Several heuristics have been proposed to address this issue by using subsampling in **Tevs et al.** (2011) or coarse-to-fine techniques in **Sahillioglu and Yemez** (2011); **C. Wang et al.** (2011). Various relaxations have been used to make the QAP problem tractable (**Aflalo et al.** 2015b; **Chen and Koltun** 2015; **Kezurer et al.** 2015; **Leordeanu and Hebert** 2005; **Rodolà et al.** 2012), however they result in approximate solutions. In addition, pairwise geodesics are computationally expensive, and sensitive to noise. In **Hu and Guibas** (2013) the use of heat kernels was proposed as a noise-tolerant approximation of matching adjacency matrices. In **Vestner et al.** (2017) dense bijective correspondences were derived from sparse and possibly noisy input using an iterative filtering scheme, making use of geodesic Gaussian kernels.

A different family of methods look for pointwise matches in a lower-dimensional "canonical" embedding space. Such embedding can be carried out by multidimensional scaling (**A. M. Bronstein et al.** 2006a; **Elad and Kimmel** 2003) or via the eigenfunctions of the Laplace-Beltrami operator (LBO) (**Mateus et al.** 2008; **Shtern and Kimmel** 2014b). The correspondence is then calculated in the embedding space using a simple rigid alignment technique such as ICP (**Besl and McKay** 1992). Functional maps (**Kovnatsky et al.** 2015; **Ovsjanikov et al.** 2012) can be seen as a sophisticated way to initialize ICP when using this spectral embedding. Other bases can be used within the functional map framework (**Kovnatsky et al.** 2013). In particular, the eigenspaces arising from the spectral decomposition of the geodesic distance matrices have been shown to outperform the LBO basis for the case of isometric shapes (**Shamai and Kimmel** 2016). In **Windheuser et al.** (2011a) the matching problem is phrased as an integer linear program, enforcing continuity of the correspondence via a linear constraint. This additional constraint however makes the problem computationally intractable even for modestly-sized shapes, requiring the use of relaxation and post-processing heuristics.

Most recent works attempt to formulate the correspondence problem as a learning problem (**Rodolà et al.** 2014a) and design intrinsic deep learning architectures on manifolds and point clouds (**Boscaini et al.** 2016a; **Litany et al.** 2017a; **Masci et al.** 2015; **Monti et al.** 2017). As of today, these methods hold the record of performance on deformable correspondence benchmarks; however, supervised learning requires a significant annotated training set that is often hard to obtain.

8.2 Problem Formulation

The underlying assumption in this method is that the solution can be described by a one-to-one point correspondence, which means a permutation. We have both a pointwise term aligning descriptors as well as a pairwise feature term which we optimize through a series of Linear Assignment Problems using the theory of difference of convex functions. While the energy might look like an off-the-shelf Quadratic Assignment Problem, we show that our method allows for several geometric interpretations in Section 8.4. One of them includes finding a submanifold in product space that arises from an underlying diffeomorphism as we discussed in Chapter 6.

8.2.1 Linear Assignment Problem

Similarity of points is often measured with the help of pointwise descriptors $f_{\mathcal{X}} : \mathcal{X} \to \mathbb{R}^q$, $f_{\mathcal{Y}} : \mathcal{Y} \to \mathbb{R}^q$ that are constructed in a way such that similar points on the two shapes are assigned closeby (in the Euclidean sense) descriptors, while dissimilar points are assigned distant descriptors. In the discrete case, the descriptors $f_{\mathcal{X}}, f_{\mathcal{Y}}$ can be encoded as matrices $F_{\mathcal{X}}, F_{\mathcal{Y}} \in \mathbb{R}^{n\times q}$ giving rise to the optimization problem

$$\underset{\Pi\in\mathcal{P}_n}{\arg\min}\; \|\Pi F_{\mathcal{X}} - F_{\mathcal{Y}}\|_F^2 = \underset{\Pi\in\mathcal{P}_n}{\arg\max}\; \langle \Pi, F_{\mathcal{Y}} F_{\mathcal{X}}^\top \rangle \,. \tag{8.1}$$

Problem (8.1) is linear in Π and is one of the rare examples of combinatorial optimization problems that can be globally optimized in polynomial time; the best known complexity $O(n^2 \log n)$ is achieved by the auction algorithm (**Bernard et al.** 2016).

Over the last years, intrinsic features have extensively been used due to their invariance to isometry. However, they come with two main drawbacks: First, the implicit assumption that the shapes at hand are isometric is not always met in practice. Some of the best performing approaches partially tackle this problem using deep learning (**Boscaini et al.** 2015; **Boscaini et al.** 2016a; **Boscaini et al.** 2016b; **Masci et al.** 2015; **Monti et al.** 2017). Secondly, many natural shapes come with at least one intrinsic (e.g., bilateral) symmetry that is impossible to capture by even the perfect purely intrinsic features, be these handcrafted or learned. Additionally, correspondences obtained through (8.1) may suffer from severe discontinuities due to some points being mapped to the desired destination, and others to the symmetric counterpart.

8.2.2 Quadratic Assignment Problem

Another family of methods consider *pairwise descriptors* of the form $d_{\mathcal{X}} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, $d_{\mathcal{Y}} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ encoded in the discrete setting as symmetric matrices $D_{\mathcal{X}}, D_{\mathcal{Y}} \in \mathbb{R}^{n\times n}$. These methods aim at solving optimization problems of the form

$$\begin{aligned} \Pi^* &= \underset{\Pi\in\mathcal{P}_n}{\arg\min}\; \|\Pi D_{\mathcal{X}} - D_{\mathcal{Y}}\Pi\|^2 && (8.2)\\ &= \underset{\Pi\in\mathcal{P}_n}{\arg\max}\; \langle \Pi, D_{\mathcal{Y}}\Pi D_{\mathcal{X}}\rangle\,, && (8.3) \end{aligned}$$

which is a variant of the Quadratic Assignment Problem (QAP) as was discussed in Section 1.3.3.2. A typical way to circumvent the complexity issue of QAPs is to relax the integer constraint $\pi_{ij} \in \{0,1\}$ and optimize the objectives (8.2)-(8.3) over the convex set of *bistochastic matrices* $\mathcal{B}_n = \{P \geq 0 : P^\top \mathbf{1} = P\mathbf{1} = \mathbf{1}\}$. Note that when viewed as functions over this convex set, the objectives (8.2)-(8.3) are no longer equivalent. In particular, (8.2) will always be convex, while the convexity of (8.3) depends on the eigenvalues of the matrices $D_{\mathcal{X}}$ and $D_{\mathcal{Y}}$, as shown in the following theorem.

Theorem 3. *Let $D_{\mathcal{X}}, D_{\mathcal{Y}}$ be symmetric. The function $h(P) = \langle P, D_{\mathcal{Y}} P D_{\mathcal{X}}\rangle$ over the set of bistochastic matrices $\mathcal{B}_n$ is (strictly) convex iff all eigenvalues of $D_{\mathcal{X}}$ and $D_{\mathcal{Y}}$ are (strictly) positive.*

Additionally, the following holds:

Corollary 2. *If all eigenvalues of $D_{\mathcal{X}}$ and $D_{\mathcal{Y}}$ are strictly positive, the optimum of the relaxed problem coincides with that of the original combinatorial problem:*

$$\underset{P\in\mathcal{B}_n}{\arg\max}\, h(P) = \underset{\Pi\in\mathcal{P}_n}{\arg\max}\, h(\Pi)\,. \tag{8.4}$$

Notice that we can add a linear term (weighted by a scalar factor α), such as the one in (8.1), while still keeping this property: $E(P) = \alpha\langle P, F_{\mathcal{Y}} F_{\mathcal{X}}^\top\rangle + \langle P, D_{\mathcal{Y}} P D_{\mathcal{X}}\rangle\,.$

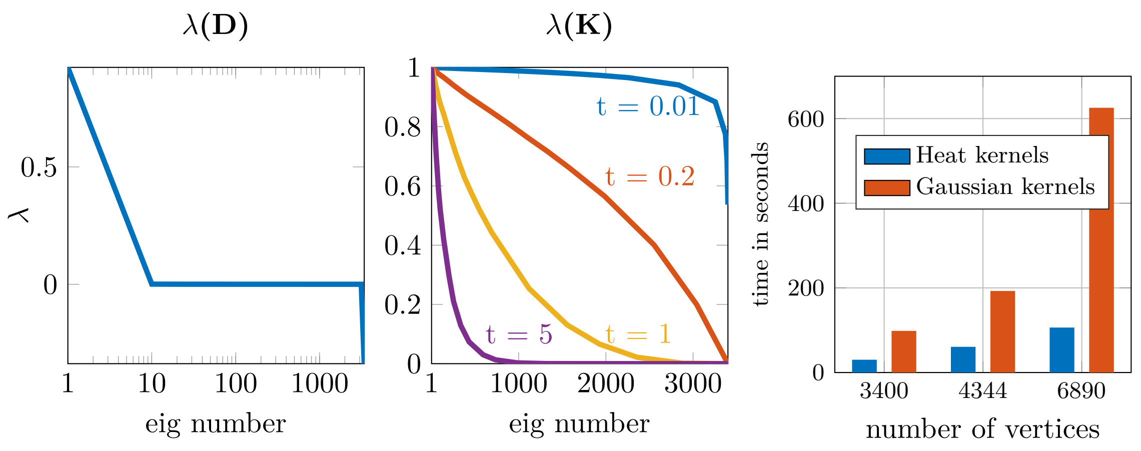


Figure 8.2: Spectrum of distance matrix (left) vs. spectrum of heat-kernel matrix (middle) for several values of $t \in [0.01, 5]$ computed on the cat shape from TOSCA. (Right) Runtime comparison of matching shapes with varying number of vertices using our algorithm with heat kernels compared to Gaussian kernels **Vestner et al.** (2017).

As discussed in Section 1.3.3.2 using geodesic distances in the QAP models isometries, and other pairwise descriptors exist for specific classes of deformations, *e.g.* equi-affine **Raviv et al.** (2011). Kernels are also a popular choice in literature (**Liu et al.** 2008; **Shtern and Kimmel** 2014a; **Vestner et al.** 2017) and in what follows, we advocate the superiority of using kernels over distances. Additionally, we show the relation of heat kernels to diffeomorphisms in Section 8.4.3.

8.2.2.1 Pairwise Distances

A common choice for pairwise descriptors are geodesic distances $d_{\mathcal{X}}(x_i, x_j)$, a choice motivated by the fact that, for isometric shapes, these are preserved by the optimal permutation Π. While accompanied with some nice theory from metric geometry (**A. M. Bronstein et al.** 2008), and providing some versatility as to the types of transformations they describe via the choice of the metric, using pairwise distances as descriptors comes with drawbacks, both from the modeling and computational point of view. To cope with these problems, different choices of distances can be used, like diffusion distances (**A. M. Bronstein et al.** 2010), which are less sensitive to topological noise, and different robust norms which are less sensitive to outliers (**Chen and Koltun** 2015). While diffusion distances relax the computational burden of computing geodesic distances, the issue of optimization still exists. This issue, as mentioned above, is usually coped with by means of relaxation.

8.2.2.2 Heat Kernel

Heat kernels are fundamental solutions to the heat diffusion equation on manifold $\mathcal{X}$,

$$\frac{\partial u(t,x)}{\partial t} = \Delta_{\mathcal{X}} u(t,x)\,, \tag{8.5}$$

with the initial condition $u(0,x) = u_0(x)$ and additional boundary conditions if applicable. Here $u : [0,\infty) \times \mathcal{X} \to \mathbb{R}$ represents the amount of heat at point x at time t. The solution is linear in the initial distribution and is given by

$$u(t,x) = \int_{\mathcal{X}} k(t,x,x')u_0(x')\mathrm{d}x'\,, \tag{8.6}$$

where $k : \mathbb{R}^+ \times \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is the *heat kernel* and its values can be interpreted as the amount of heat transported from x' to x in time t. In the Euclidean case, the heat kernel is an isotropic Gaussian kernel with the variance proportional to the diffusion time t.

For a compact manifold $\mathcal{X}$, the heat kernel can be expressed as the exponent of the Laplacian operator $\Delta_{\mathcal{X}}$,

$$k(t,x,x') = \sum_i e^{-\lambda_i t}\phi_i(x)\phi_i(x'), \tag{8.7}$$

where $\Delta_{\mathcal{X}}\phi_i(x) = \lambda_i\phi_i(x)$ is the eigendecomposition of the Laplacian with eigenvectors $\phi_1, \phi_2, \dots$ and corresponding eigenvalues $0 = \lambda_1 \le \lambda_2 \le \dots$.

In the discrete setting, the heat kernel is given by the positive-definite matrix $K_{\mathcal{X}} = e^{t\mathbf{\Delta}_{\mathcal{X}}} = \mathbf{\Phi} e^{t\mathbf{\Lambda}_{\mathcal{X}}}\mathbf{\Phi}^\top$. The constant eigenvector corresponds to the unit eigenvalue, $K_{\mathcal{X}}\mathbf{1} = \mathbf{1}$.

An issue that is often overlooked is the relation between the original and relaxed solution of (8.3), which is tightly connected to the choice of pairwise descriptors. Corollary 2 asserts the sufficient condition under which this relaxation is exact. Whereas heat kernels, being (strictly) positive definite, satisfy this condition, distance matrices never do. A distance matrix, having non-negative entries and trace zero, will always, by the Perron-Frobenius theorem, have one large positive eigenvalue and several low magnitude negative eigenvalues[1] (**Bogomolny et al.** 2003). This distribution of eigenvalues is illustrated in Figure 8.2.

8.2.3 Relaxing Bijectivity

The requirement of bijectivity is what makes a problem (8.2) computationally hard. A variety of relaxation techniques can be applied to alleviate this complexity. Amongst the most popular are relaxing the column or row sum constraints, relaxing the integer constraints, or restricting the matrix to a sphere of constant norm (**Leordeanu and Hebert** 2005). A bijective mapping

[1] In the Euclidean case, a distance matrix has exactly one positive eigenvalue and all the rest are negative with small magnitude.

can then be recovered by a post processing step, such as projection onto the set of permutation matrices

$$\Pi^* = \underset{\Pi \in \mathcal{P}_n}{\arg\min} \|\Pi - P\|^2 = \underset{\Pi \in \mathcal{P}_n}{\arg\max} \langle \Pi, P \rangle \ . \tag{8.8}$$

One popular technique in recent years replaces the combinatorially hard pointwise map recovery problem with the simpler problem of finding a linear functional map instead of a permutation **Ovsjanikov et al.** (2012). See Section 1.3.3.3 for an analysis of the relation. The fact that the map is band-limited is often erroneously referred to as smoothness in the literature; however, the bijective map recovered from such a band-limited map is not guaranteed to be continuous let alone smooth (i.e., continuously differentiable). Some incentive for smoothness exist, but the problem is now shifted to extracting a pointwise correspondence with certain properties from the functional map which is far from an easy one (**Rodolà et al.** 2015).

8.3 Optimization

We aim at maximizing $E(\Pi)$ over $\mathcal{P}_n$, which by Corollary 2 has the same optimum as the relaxed problem

$$\underset{P \in \mathcal{B}_n}{\arg\max} \quad E(P) = \underset{P \in \mathcal{B}_n}{\arg\max} \ \langle P, \alpha F_{\mathcal{Y}} F_{\mathcal{X}}^{\top} + K_{\mathcal{Y}} P K_{\mathcal{X}} \rangle \tag{8.9}$$

where $F_{\mathcal{X}}, F_{\mathcal{Y}}$ are matrices of pointwise descriptors and $K_{\mathcal{X}}, K_{\mathcal{Y}}$ are the positive-definite heat kernel matrices on $\mathcal{X}$ and $\mathcal{Y}$, respectively. This maximization problem can be seen as the minimization of the difference of convex functions:

$$\underset{P \in \mathbb{R}^{n \times n}}{\arg\min} B(P) - E(P). \tag{8.10}$$

where B is the (convex) indicator function on the set of bistochastic matrices $\mathcal{B}_n$.

A renowned way to optimize this type of energy is the difference of convex functions (DC) algorithm that starts with some initial P^0 and then iterates the following two steps until convergence:

- Select $Q^k \in \partial E(P^k)$.
- Select $P^{k+1} \in \partial B^*(Q^k)$.

Here, B^* denotes the convex conjugate of B, and $\partial E, \partial B^*$ denote the subdifferentials (set of supporting hyperplanes) of E and B^*, respectively.

For a differentiable E, the step of the DC algorithm assumes the form

$$P^{k+1} = \underset{P \in \mathcal{B}_n}{\arg\max} \quad \langle P, \nabla E(P^k) \rangle \, . \tag{8.11}$$

Moreover, the value of the objective is an increasing sequence, $E(P^{k+1}) > E(P^k)$, and each iterate P^k is a permutation matrix. We provide the proof in the supplementary material. Figure 8.3 illustrates this iterative process. Since P^k is guaranteed to be a permutation matrix, we use Π^k to denote the iterates. For our choice of E, the gradient is given by

$$\nabla E = \alpha F_{\mathcal{Y}} F_{\mathcal{X}}^\top + K_{\mathcal{Y}} \Pi K_{\mathcal{X}} \tag{8.12}$$

yielding the step

$$\Pi^{k+1} = \underset{\Pi \in \mathcal{B}_n}{\arg\max} \ \langle \Pi, \alpha F_{\mathcal{Y}} F_{\mathcal{X}}^\top + K_{\mathcal{Y}} \Pi^k K_{\mathcal{X}} \rangle \,. \tag{8.13}$$

In the experiments presented in this paper, we use the data fidelity term $\langle \Pi, F_{\mathcal{Y}} F_{\mathcal{X}}^\top \rangle$ mainly to initialize the process:

$$\Pi^0 = \underset{\Pi \in \mathcal{B}_n}{\arg\max} \ \langle \Pi, F_{\mathcal{Y}} F_{\mathcal{X}}^\top \rangle. \tag{8.14}$$

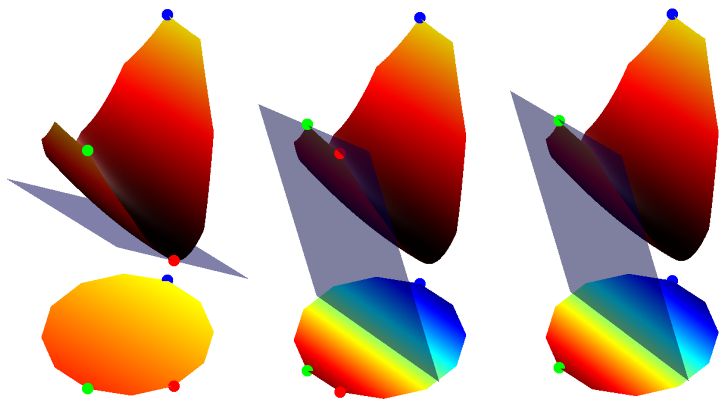


Figure 8.3: Schematic illustration of the proposed algorithm for maximizing a convex quadratic objective over a convex polytope, by successively maximizing a linear sub-estimate of it. The *hot* color map encodes the function values. The *jet* color map encodes the values of the linear sub-estimate. The point around which the objective is linearized is depicted in red. The global maximum is depicted in blue. The maximum of the linear sub-estimate is depicted in green. Notice that the algorithm travels between extreme but not necessarily adjacent points of the polytope, until it converges to a local maximum.

8.3.1 Partial Matching using Slack Variables

In a general setting, we will be dealing with shapes having different number of vertices. Let us denote by $n_{\mathcal{X}}$ the number of vertices on $\mathcal{X}$ and by $n_{\mathcal{Y}}$ the number of vertices on $\mathcal{Y}$, and assume w.l.o.g. $n_{\mathcal{X}} \geq n_{\mathcal{Y}}$. We aim at optimizing

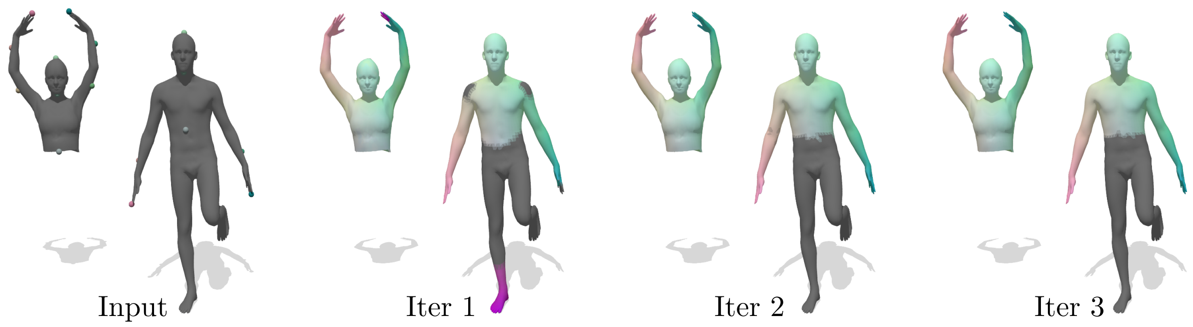


Figure 8.4: Our approach can tackle the challenging scenario of partial correspondences. As a proof of concept we initialized our method with sparse correspondences, indicated by spheres. We simulated noise by mapping a point on the left hand of the woman to the right foot of the man. At the first iteration all points spread their information, leading to a discontinuity of the mapping at the hand of the woman. After three iterations the method converged to the correct solution. This example was generated with Gaussian kernels. The proper choice of boundary conditions when using heat kernels will be discussed in future work.

$$\underset{\Pi \in \mathcal{P}_{n_\mathcal{X}}^{n_\mathcal{Y}}}{\arg\max} \left\langle \Pi, \alpha F_\mathcal{Y} F_\mathcal{X}^\top + K_\mathcal{Y} \Pi K_\mathcal{X} \right\rangle \tag{8.15}$$

where the space of *rectangular permutation matrices* $\mathcal{P}_{n_\mathcal{X}}^{n_\mathcal{Y}}$ is given by $\mathcal{P}_{n_\mathcal{X}}^{n_\mathcal{Y}} = \{\Pi \in \{0,1\}^{n_\mathcal{Y} \times n_\mathcal{X}} : \Pi \mathbf{1} \leq \mathbf{1}, \Pi^\top \mathbf{1} = \mathbf{1}\}$. Analogously to the previously discussed case in which we had $n_\mathcal{X} = n_\mathcal{Y} = n$, we iteratively solve

$$\Pi^{k+1} = \underset{\Pi \in \mathcal{P}_{n_\mathcal{X}}^{n_\mathcal{Y}}}{\arg\max} \left\langle \Pi, \alpha F_\mathcal{Y} F_\mathcal{X}^\top + K_\mathcal{Y} \Pi^k K_\mathcal{X} \right\rangle . \tag{8.16}$$

In order to solve these optimization problems we pad the rectangular matrix $\alpha F_\mathcal{Y} F_\mathcal{X}^\top + K_\mathcal{Y} \Pi^k K_\mathcal{X}$ with constant values c (slack variables) such that it becomes square. After the correspondence is computed, we discard the ones belonging to the introduced slack variables. While such a treatment does not affect the value of the maximum, the constant c has to be chosen appropriately to avoid ambiguity between the slacks and the actual vertices on $\mathcal{X}$. A drawback of this approach is that there are $(n_\mathcal{X} - n_\mathcal{Y})!$ solutions achieving the optimal score, leading to worse runtime in the presence of many slacks. See Figure 8.4 for a proof of concept of this approach.

8.3.2 Multiscale acceleration

Solving the LAP (8.16) at each iteration of the DC algorithm has a super-quadratic complexity. As a consequence, the proposed method is only feasible for small n (up to 15k on the hardware we used in our experiments). Therefore, we propose a multiscale approach that enables us to find correspondences between larger meshes.

Initilization. We start by resampling both shapes to n_0 vertices by Euclidean farthest point sampling (FPS) and solving for a bijection $\pi_0 : \mathcal{X}_0 \rightarrow \mathcal{Y}_0$, where $\mathcal{X}_0, \mathcal{Y}_0$ are the resampled shapes of iteration 0. n_0 can either be the maximum amount of vertices that can be handled (around 15k in our experiments) or smaller if runtime is crucial. This set of initial vertices is called seeds. The seeds on $\mathcal{X}$ are clustered into $n_0/(k \cdot maxP)$ voronoi cells $\mathcal{V}^0_{\mathcal{X}} = \{V^0_{\mathcal{X},i} \mid i \in \{1, \dots, k\}, V^0_{\mathcal{X},i} \subset \mathcal{X}_0\}$, and these cells are transferred to $\mathcal{Y}$ via π_0 in $\mathcal{V}^0_{\mathcal{Y}} = \{V^0_{\mathcal{Y},i} = \pi_0(V^0_{\mathcal{X},i}) \mid i \in \{1, \dots, k\}, V^0_{\mathcal{Y},i} \subset \mathcal{Y}_0\}$. The parameter $maxP$ is the maximum problem size for LAPs allowed in later iterations and normally much smaller than n_0. k determines how many new samples are added to the voronoi cells in each iteration. A small $maxP$ makes the method faster but less robust, and a small k slower but more robust. We always choose $maxP = 1500$ and $k = 3$ in our experiments. Assuming that π_0 describes a good correspondences, the underlying idea is that an unsampled point in the vicinity of $\mathcal{V}^0_{\mathcal{X},j}$ should be matched to a point in the vicinity of $\mathcal{V}^0_{\mathcal{Y},j}$ to preserve smoothness in the solution. This is also important to reduce the problem size of later iterations, because under this assumption we can optimize the neighborhood of each $V^0_{\mathcal{X},j}, V^0_{\mathcal{Y},j}$ independently.

Iterations. At the first iteration ($i = 1$) and any following iteration i, $n_i = k \times n_{i-1}$ new points are sampled in a farthest point manner on both shapes to create $\mathcal{X}_i, \mathcal{Y}_i$. Each new point is assigned to the same Voronoi cell as its nearest neighbor in $\mathcal{X}_{i-1}, \mathcal{Y}_{i-1}$ resulting in the new cells $V{i}_{\mathcal{X}, V{i}_{\mathcal{Y}}}$. If any cell has more than $maxP$ vertices, the number of cells is increased until this is not the case anymore. Next, we solve for $\pi_i : \mathcal{X}_i \rightarrow \mathcal{Y}_i$ by solving for a mapping from the m-th cell of $V^i_{\mathcal{X}}$ to the m-th of $V^i_{\mathcal{X}}$ using the proposed method and combining them into a global permutation π_i. There cannot be contradicting entries in π_i, because every sampled point belongs to exactly one voronoi cell. Notice that the m-th cells of both shapes correspond to roughly the same areas as long as the previous matching π_{i-1} is reasonable. Nevertheless, the cells could include a different amount of points due to discretization errors and inconsistencies in the Euclidean farthest point sampling. Therefore, we need to apply the partial matching scheme to each cell and some points may stay unmatched (in this iteration). Again, $\mathcal{X}$ is divided into $n_i/(k \cdot maxP)$ Voronoi cells and these are transferred to $\mathcal{Y}$ via π_i. The voronoi cells of previous iterations are discarded to allow exchange of points between cells. This is especially important on the boundary between cells because inconsistencies between $\mathcal{X}_m$ and $\mathcal{Y}_m$ are likely happening there. This proceeds until all points have been sampled. We use Euclidean FPS in all cases and build approximate Voronoi cells on remeshed versions of the shape to keep the runtime small. See Figure 8.5 for a visualization of the process.

Global Alignment. Each π_i is solved for by using descriptors and initial matches from the previous iteration in the same cell. Additionally, we add 1000 equally distributed matches from π_0 to every problem which aligns the solution along the boundaries of the cells with each other. This is important because in cells with no distinctive descriptors (mostly flat areas) the optimal solution inside one cell might be rotated and not align with the global solution anymore. Notice that, even if the shapes have the same number of vertices at the beginning,

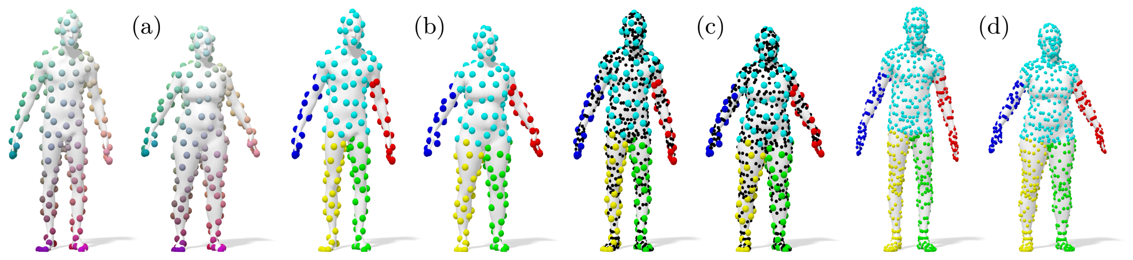


Figure 8.5: **Conceptual illustration of our multiscale approach:** (a) correspondence at a coarse scale is given; (b) vertices on the source shape are grouped into sets (left), and the known correspondence is used to group vertices on the target shape (right); (c) vertices at a finer scale are added and (d) included in the group they reside in; finally, a correspondence is calculated for each group separately.

due to the sampling and decoupling of each cell, not all vertices might be matched.

Partiality. If the matched shapes are partial versions of each other, this information needs to be propagated from the first iteration on since all later cells are solved independently and can therefore not see partiality. In this case, $n_{0,\mathcal{X}}, n_{0,\mathcal{Y}}$ can be chosen dependently on the ratio of areas or number of vertices between $\mathcal{X}$ and $\mathcal{Y}$, either assuming the scale or the discretization is comparable. Then certain points of the initial sampling will stay unmatched and be marked *forbidden*. They are handled exactly like any other seed but have their own Voronoi cell, and any point that gets a assigned to the forbidden Voronoi cell is also marked forbidden such that the information spreads into the neighborhood.

8.4 Interpretation

In what follows we provide different, yet complementary interpretations of the proposed method, shedding light on its effectiveness.

8.4.1 Alternating diffusion

To intuitively understand the efficacy of kernel alignment for the purpose of finding correspondences, consider the k-th iteration (without data term):

$$\max_{\Pi \in \mathcal{P}_n} \langle \Pi, K_{\mathcal{Y}} \Pi^k K_{\mathcal{X}} \rangle \,. \tag{8.17}$$

Let us denote by $\boldsymbol{\delta}^j$ the discrete indicator function of vertex j on shape $\mathcal{X}$, representing initial heat distribution concentrated at vertex j. This heat is propagated via the application of the heat kernel $K_{\mathcal{X}}$ to the rest of the vertices, resulting in the new heat distribution on $\mathcal{X}$ given by $k^j_{\mathcal{X}} = K_{\mathcal{X}} \boldsymbol{\delta}^j$. This heat distribution, whose spread depends on the time parameter t, is mapped via Π^k onto the shape $\mathcal{Y}$, where it is propagated via the heat kernel $K_{\mathcal{Y}}$. The ij-th element of the matrix $K_{\mathcal{Y}}\Pi^k K_{\mathcal{X}}$,

$$\begin{aligned}(K_{\mathcal{Y}}\Pi^k K_{\mathcal{X}})_{ij} &= (k_{\mathcal{Y}}^i)^\top \Pi^k k_{\mathcal{X}}^j \\ &= \sum_m (K_{\mathcal{Y}})_{i,\pi^k(m)} (K_{\mathcal{X}})_{jm}, \end{aligned} \tag{8.18}$$

represents the probability of a point i on $\mathcal{Y}$ being in correspondence with the point j on $\mathcal{X}$. This is affected by both the distance between i and $\pi^k(m)$ on $\mathcal{Y}$ for every m on $\mathcal{X}$, encoded in the entries of $(K_{\mathcal{Y}})_{i,\pi^k(m)}$, and by the distance between m and j on $\mathcal{X}$, encoded in the entries of $(K_{\mathcal{X}})_{jm}$.

This process, as illustrated in Figure 8.6, resembles the alternating diffusion process described in **Lederman and Talmon** (2015). Its success in uncovering the latent correspondence is based on the following statistical assumptions on the distribution of correspondences in the initial assignment: we tacitly assume that a sufficiently large number of (uniformly distributed) points are initially mapped correctly while the rest are mapped randomly, such that when averaging over their "votes" they do not bias towards any particular candidate.

There is an inherent trade-off between the stability of the process and its accuracy, controlled by the time parameter t. Smaller t enables more accurate correspondence, but limits the ability of far away points to compensate for local inaccuracies in the initial correspondence, while larger t allows information to propagate from farther away, but introduces ambiguity at the fine scale. Examining the extremities, when $t \to 0$ each point is discouraged to change its initial match, while as $t \to \infty$ every point becomes a likely candidate for a match. In practice, we approximately solve a series of problems parametrized by a decreasing sequence of t values, as explained in the experimental section.

8.4.2 Iterated blurring and sharpening

An alternative point of view is to recall that a diffusion process corresponds to a smoothing operation, or low-pass filtering in the spectral domain. To that end we view each iteration (8.13) as an application of a series of low-pass filters (smoothing) followed by a projection operation (deblurring/sharpening). To see that, we use the spectral decomposition of the heat kernels to rewrite the payoff matrix in (8.13)

$$\begin{aligned}K_{\mathcal{Y}}\Pi K_{\mathcal{X}} &= \Psi e^{-t\Lambda_{\mathcal{Y}}} \Psi^\top \Pi \Phi e^{-t\Lambda_{\mathcal{X}}} \Phi^\top \\ &= \Psi e^{-t\Lambda_{\mathcal{Y}}} C e^{-t\Lambda_{\mathcal{X}}} \Phi^\top. \end{aligned} \tag{8.19}$$

where the functional map C is seen as a low-pass approximation of the permutation matrix in the truncated Laplacian eigenbasis, $\mathbf{\Pi} \approx \Psi C \Phi^\top$. Equation (8.19) can thus be interpreted as applying a low-pass filter to the functional map matrix C. The second step in (8.13) can be regarded as a projection of the smoothed correspondence on the set of permutations (8.8), producing a pointwise bijection. A similar approach, but without the heat kernels, can be seen in **Melzi et al.** (2019).

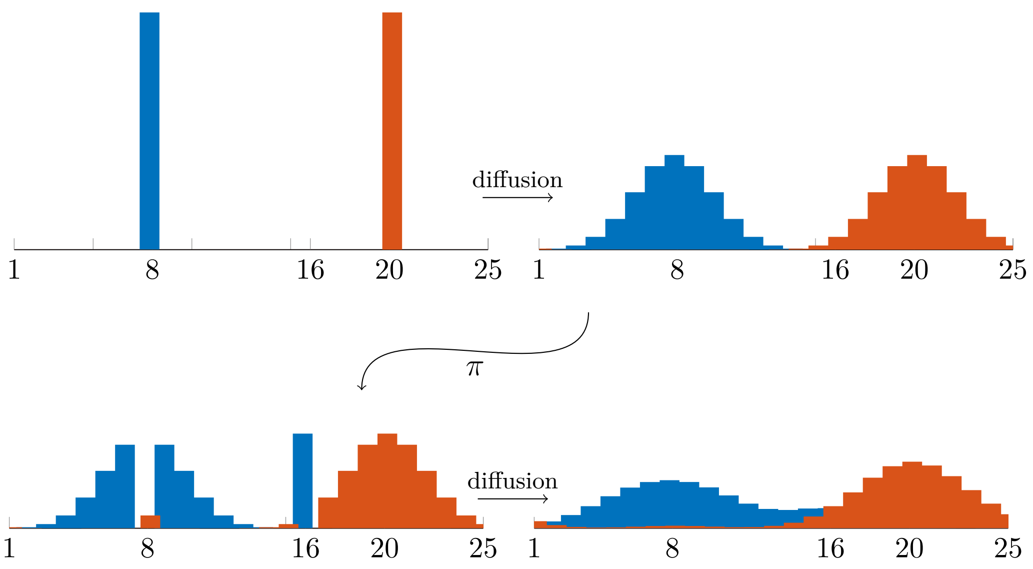


Figure 8.6: **Illustration of the alternating diffusion process** initialized with a noisy correspondence that wrongly maps $\pi(8) = 16$ and $\pi(16) = 8$ but correctly maps $\pi(x) = x$ elsewhere. Top left: Indicator functions on the source shape, one on a point with a wrong correspondence (red) and one with a correct correspondence (blue). Top right: Both indicator functions are diffused. Bottom left: The diffused functions are transported to the target shape via π. Bottom right: Diffusion on the target shape.

8.4.3 Kernel density estimation in the product space

Similar to the interpretation in **Vestner et al.** (2017), our approach can be seen as regularizing the correspondence graph $\Gamma_\pi = \{(x, \pi(x)) : x \in \mathcal{X}\}$ of the latent correspondence $\pi : \mathcal{X} \to \mathcal{Y}$ on the product manifold $\mathcal{X} \times \mathcal{Y}$. In case of a bijective, continuous π, the graph Π is a submanifold without a boundary of same dimension as $\mathcal{X}$ (2 here, see Section 6.2). In each iteration of the process a probability distribution $P : \mathcal{X} \times \mathcal{Y} \to [0, 1]$ is constructed by placing kernels (geodesic Gaussian kernels in **Vestner et al.** (2017), and heat kernels in our case) on the graph of the previous iterate and maximizing

$$\hat{\pi} = \underset{\hat{\pi}:\mathcal{X} \overset{1:1}{\to} \mathcal{Y}}{\arg\max} \int_{\mathcal{X}} P(x, \hat{\pi}(x)) \, dx \tag{8.20}$$

over the set of bijective but not necessarily continuous correspondences. Notice that the maximum of this function is achieved if Γ_π is a connected, minimal surface submanifold of $\mathcal{X} \times \mathcal{Y}$, because the overlap of kernels peaks in this case. Therefore, our approach actually aims at finding a diffeomorphism. A result of this appears when maximizing above function for two input with widely different resolutions. The result will often match all points in a connected region instead of an equal covering of the target, even if this would be semantically meaningful. The cluttered result is closer to a local diffeomorphism than the distributed solution would be.

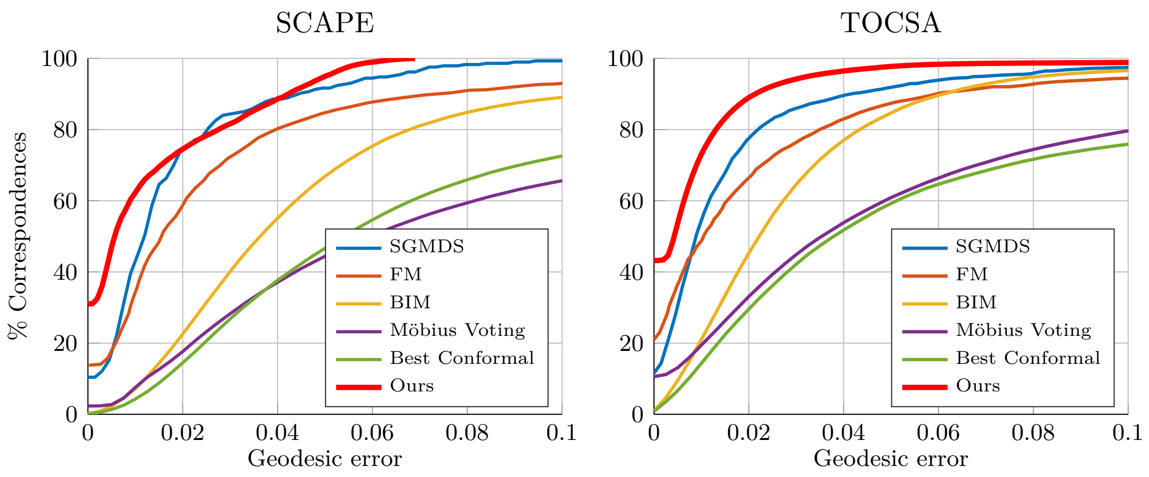


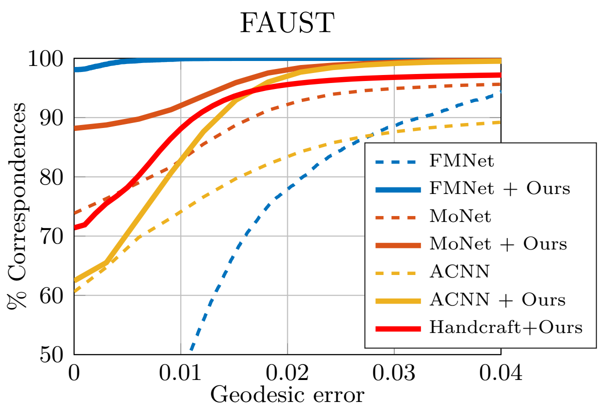
Figure 8.7: Correspondence accuracy on the SCAPE and TOSCA datasets. We compare against SGMDS (**Aflalo et al.** 2016), Functional Maps (**Ovsjanikov et al.** 2012), Blended Intrinsic Maps (**Kim et al.** 2011), Möbius Voting (**Lipman and Funkhouser** 2009) and Best Conformal Mappings (**Kim et al.** 2011).

8.5 Experiments

We performed an extensive quantitative evaluation of the proposed method on four different benchmarks. All datasets include several classes of (nearly) isometric shapes, with the last one additionally introducing strong topological noise (i.e., mesh 'gluing' in areas of contact). In our experiments we used the SHOT (**Tombari et al.** 2010) and heat kernel signature (HKS) (**J. Sun et al.** 2009) descriptors with default parameters. For the computation of heat kernels we used 500 Laplacian eigenfunctions. We provide comparisons with complete matching pipelines as well as with learning-based approaches, where we show how using our method as a post-processing step leads to a significant boost in performance. In addition Figure 8.2 provides runtime comparison against **Vestner et al.** (2017) which uses a similar method with geodesic Gaussian kernels. Code of our method is available at `https://github.com/zorah/KernelMatching`.

Error measure. We measure correspondence quality according to the Princeton benchmark protocol (**Kim et al.** 2011). Assume to be given a match $(x, y) \in \mathcal{X} \times \mathcal{Y}$, whereas the ground-truth correspondence is (x, y^*). Then, we measure the geodesic error $\epsilon(x) = d_{\mathcal{Y}}(y, y^*)/\mathrm{diam}(\mathcal{Y})$ normalized by the geodesic diameter of $\mathcal{Y}$. Ideal correspondence should produce $\epsilon = 0$. We plot cumulative curves showing the percentage of matches that have error smaller than a variable threshold.

Parameters. The optimal choice of parameters does not only depend on properties of the considered shapes (such as diameterand density of the sampling) but also on the noise of the



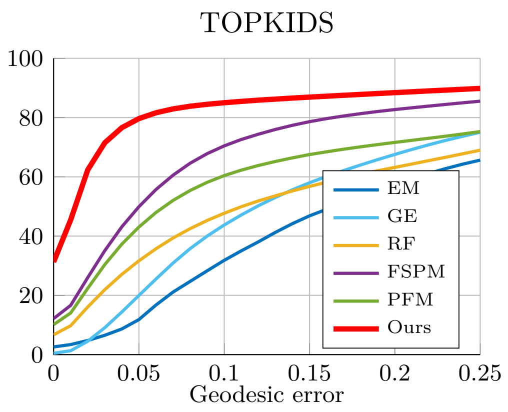


Figure 8.8: (Left) Correspondence accuracy on FAUST (**Bogo et al.** 2014). Dashed curves indicate the performance of recent deep learning methods (FMNet (**litany2017fmnet**), ACNN (**boscaini2016learning**) and MoNet (**monti2016geometric**)), solid curves are obtained using our method as post-processing. Our method based on handcrafted descriptors (SHOT) is denoted as 'Handcrafted+Ours'. (Right) Correspondence accuracy on TOPKIDS (**Lähner et al.** 2016a) comparing against EM (**Sahillioglu and Yemez** 2012), GE (**Lähner et al.** 2016a), RF (**Rodolà et al.** 2014a), FSPM (**Litany et al.** 2017b) and PFM (**Rodolà et al.** 2016).

input correspondence. The exact dependencies in particular on the latter will be investigated in follow up works.

8.5.1 Quantitative Evaluation

We evaluate our method on four common datasets for non-rigid correspondence.

TOSCA. The TOSCA dataset (**A. M. Bronstein et al.** 2008) contains 76 shapes divided into 8 classes (humans and animals) of varying resolution (3K to 50K vertices). We match each shape with one instance of the same class. For shapes having more than 10K vertices we use our multiscale acceleration with an initial problem size of 10K and a maximum problem size of 3K for all further iterations. The parameters were set to $\alpha = 10^{-10}$ and $t = [300\ 100\ 50\ 10]$, with 5 iterations per diffusion time. Figure 8.7 shows a quantitative evaluation.

SCAPE. The SCAPE dataset (**Anguelov et al.** 2005) contains 72 clean shapes of scanned humans in different poses. For this test we set $\alpha = 10^{-7}$, $t = [0.1\ 0.05\ 0.009\ 0.001\ 0.0001]$, and 5 iterations per diffusion time. We used multiscale acceleration with initial size equal to 10K vertices, and equal to 1K for subsequent iterations. Quantitative and qualitative results are given in Figure 8.7, Figure 8.9 and 8.1 (right) respectively.

FAUST. The FAUST dataset (**Bogo et al.** 2014) contains 100 human scans belonging to 10 different individuals; for these tests we used the template subset of FAUST, consisting of

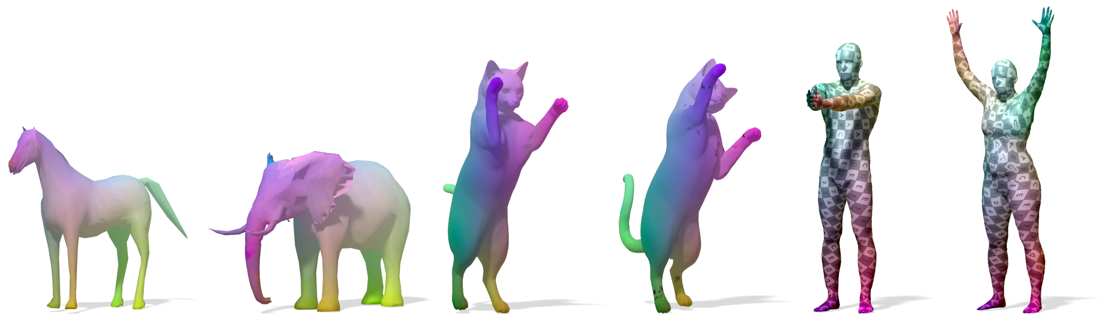


Figure 8.9: (Left) Example of an non-isometric correspondence between a horse and an elephant. Because we use local heat kernels as opposed to geodesic distances, we are not bound to isometric cases. (Middle) The multi-scale approach is not guaranteed to be consistent between cells and might create sparse non-matched points even when a bijection is possible. (Right) Example of an failure case where the upper body has a left right swap. The method got stuck in a local minimum here.

shapes with around 7K vertices each. This allowed us to run our algorithm without multiscale acceleration. We set $\alpha = 10^{-7}$ and $t = [500\ 323\ 209\ 135\ 87\ 36\ 23\ 15\ 10]$. Differently from the previous experiments, here we employ our method as a refinement step for several deep learning-based methods, demonstrating significant improvements (up to 50%) upon the 'raw' output of such approaches. The results are reported in Figure 8.8. Our results contain a few shapes in which body parts were swapped, preventing us from reaching 100%. An example is presented in the supp. material.

SHREC'16 Topology. This dataset by **Lähner et al.** (2016a) contains 25 shapes of the same class with around 12K vertices, undergoing near-isometric deformations in addition to large topological shortcuts (see Figure 8.1 middle). Here we use only SHOT as a descriptor, since HKS is not robust against topological changes. We used $\alpha = 10^{-6}$ and $t = [2.7\ 2.44\ 2.1\ 1.95\ 1.7]$, using multiscale with an initial problem of size 12k and the following problems with maximum size 1k. Quantitative results are reported in Figure 8.8.

8.5.2 Run time comparison

The run time experiments, were conducted on a MacBook pro with a 2.5 GHz Intel Core i7 processor and 16 GB RAM running Matlab 2016*b*. The experiments were conducted using 9 pairs of shapes with a varying number of vertices from the TOSCA high and low resolution meshes as well as FAUST registrations set. The complete results are presented in Table 8.1. We ran all our tests using SHOT descriptors, 10 iterations with $\alpha = 1/10^8$, 400 eigenvectors to construct the heat kernels and a logarithmic scale of time parameters between 400 and 10.

shapes in experiment	#vertices	heat kernel in sec	Gaussian kernel in sec
Tosca: cat0 to cat2	3400	29.25	97.77
Tosca: dog0 to dog2	3400	36	98.43
Tosca: centaur0 to centaur1	3400	25.31	98.91
Tosca: wolf0 to wolf1	4344	60.7	192.72
Faust models: 000 to 098	6890	109.477019	639.9
Faust models: 001 to 031	6890	104.68	609.56
Faust models: 002 to 039	6890	104.5	611.24
Faust models: 003 to 021	6890	106.41	614.23
Faust models: 004 to 033	6890	106.28	652.58

Table 8.1: Runtime comparison of matching between shapes with different number of vertices using heat kernels and Gaussian kernels.

8.6 Conclusions

In this chapter, we considered a quadratic assignment formulation for finding a continuous, possibly partial, correspondence between two non-isometric shapes using pointwise and pairwise descriptors. We showed that choosing the pairwise descriptors to be positive-definite kernel matrices (unlike the traditionally used distance matrices) makes the NP-hard QAP admit an exact relaxation over the space of bistochastic matrices, which we proposed to solve using a projected descent procedure motivated by the DC algorithm. The resulting iterations take the form of LAPs, and we solve them using a multi-scale version of the auction algorithm. The experimental evaluation on various datasets shows that the method scales well to settings with hundred thousand vertices, and that our method significantly improves the output obtained by the best existing correspondence methods.

Additionally, we provided several theoretical interpretations of our algorithm, including alternating diffusion and finding the minimal surface submanifold of the product space. As discussed before, this is a property of diffeomorphisms and our results show this in their remarkable smoothness of the correspondences. This presents a similar discretization as a QAP for diffeomorphisms as we already saw for isometries in Section 6.3.2. In the next chapter, we analyze similar properties directly on the product manifold instead of operating on each shape separately. In case of alternating diffusion, the diffusion process can be applied jointly on the product manifold instead of alternating between $\mathcal{X}$ and $\mathcal{Y}$.

CHAPTER 9

Functional Map Representation on the Product Manifold

In the last chapter we saw how our alternating diffusion algorithm could be interpreted as acting on the product manifold. In this chapter, we deepen our understanding of posing correspondence — and understanding relationships between the existing representations above — in terms of functions on the product manifold of the source and target. A motivating observation is that functional maps approximate a distribution representing the correspondence in the product space as a linear combination of *separable* tensor-product basis functions. This distribution, however, is supported on a manifold with a dimension *lower* than that of the product space: For a pair of two dimensional shapes, the distribution is supported on a two-dimensional manifold embedded in a four-dimensional space. Consequently, most of the support of the basis functions is wasted on unneeded regions of the product space.

We will see how point-to-point maps, functional maps, and soft maps all can be understood as (signed) measures on the product and how these representations might be converted to one another. More importantly, this viewpoint suggests new techniques to represent and approximate mappings directly on the product, e.g. by building a basis from eigenfunctions of the product Laplace–Beltrami operator potentially after filtering undesirable matches. After discretizing product manifolds and their Laplace–Beltrami operators, we consider map design and processing problems among two- and three-dimensional shapes. Reasoning about the product manifold leads to compact, understandable bases for map design that focus resolution in the part of the product most relevant to a correspondence task. One of such means is the construction of *inseparable* bases. To this end, we compute localized harmonics on the product manifold, and discuss a numerical scheme that keeps the complexity of such a computation comparable to that of the construction of a separable localized basis.

This chapter is based on **E. Rodolà**, **Z. Lähner**, **A. M. Bronstein**, **M. M. Bronstein**, and **J. Solomon** (2019). Functional Maps Representation on Product Manifolds. In: *Computer Graphics Forum (CGF)* 38.1.

9.1 Related Work

The first class of methods represents the correspondence on the Cartesian product of the two shapes. First methods of this type were formulated using graph matching (**Zeng et al.** 2010). Windheuser et al. optimize in a product space (**Windheuser et al.** 2011a), preserving important differential geometric properties. A similar approach was applied in (**laehner20162D3D**) for 2D-to-3D matching. In **Vestner et al.** (2017), correspondence is formulated as kernel density estimation on the product manifold, interpreted as alternating diffusion-sharpening process in (**Vestner* et al.** 2017).

Soft maps (**Solomon et al.** 2012) represent correspondence between shapes as a distribution on the product manifold with prescribed marginals reflecting area preservation. Non-convex objectives can be used to incorporate metric information into optimization for soft maps (**Mémoli** 2011; **Solomon et al.** 2016), while other objectives on soft maps can be understood as probabilistic relaxations of classical distortion measures from differential geometry (**Mandad et al.** 2017; **Solomon et al.** 2013). These methods suffer from high complexity, usually quadratic in the number of shape vertices.

Functional maps (**Ovsjanikov et al.** 2017) abandon pointwise correspondence, instead modeling correspondences as linear operators between spaces of functions. An approximation of such operators in a pair of truncated orthogonal bases dramatically reduces the problem complexity. One of the key innovations of the functional maps framework is allowing to bring a new set of algebraic methods into the domain of shape correspondence. Several follow-up works tried to improve the framework by employing sparsity-based priors (**Pokrass et al.** 2013b), manifold optimization (**Kovnatsky et al.** 2013; **Kovnatsky et al.** 2016), non-orthogonal (**Kovnatsky et al.** 2015) or localized (**Choukroun et al.** 2018; **Melzi et al.** 2017) bases, coupled optimization over the forward and inverse maps (**Eynard et al.** 2016; **Ezuz and Ben-Chen** 2017; **R. Huang and Ovsjanikov** 2017), and combination of functional maps with metric-based approaches (**Aflalo et al.** 2016; **Shamai and Kimmel** 2016). Recent works of **Nogneng and Ovsjanikov** (2017) and **Nogneng et al.** (2018) considered functional algebra (function point-wise multiplications together with addition). Generalizations addressing the settings of multiple shape (**Q.-X. Huang et al.** 2014; **Kovnatsky et al.** 2016), partial (**Litany et al.** 2016; **Rodolà et al.** 2016) or cluttered correspondence (**Cosmo et al.** 2016) have been proposed as well. Most recently, functional maps have also been used in conjunction with intrinsic deep learning methods (**Litany et al.** 2017a). For a comprehensive survey of functional maps and related techniques, we refer the reader to **Ovsjanikov et al.** (2017).

9.2 Discretization

We show how to discretize the main quantities involved in our framework on 1D and 2D manifolds, as well as their products. There are slight overlaps with the introductions in Chapter 3 and Chapter 5 but this section adds concrete formulas to all notions.

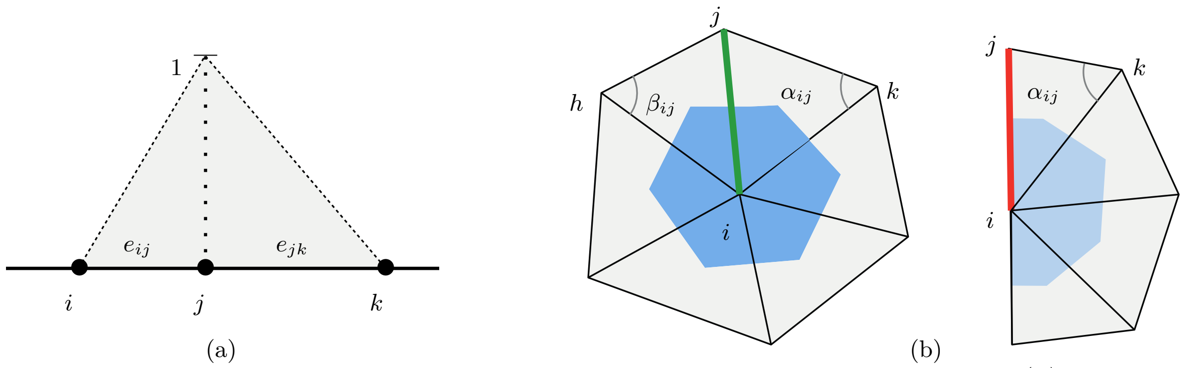


Figure 9.1: Discretization of the Laplace-Beltrami operator on a cycle graph *(a)* and on a triangular mesh *(b)* for interior (green) and boundary edges (red). We also show the hat basis function on the graph.

9.2.1 1D shapes (curves)

We model 1D manifolds equivalent to Chapter 7 as closed contours with circular topology (no boundary), discretized as 2-regular cycle graphs $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ with $n \geq 3$ nodes $\mathcal{N}$ and as many edges $\mathcal{E}$. The Laplace-Beltrami operator Δ is discretized using standard finite element method with linear hat functions (see Chapter 4); in the hat basis, scalar functions on $\mathcal{G}$ are approximated piecewise-linearly on the edges. The Laplacian takes the form of a $n \times n$ sparse matrix $\Delta = A^{-1}S$, where:

$$s_{ij} = \begin{cases} -\frac{1}{\|e_{ij}\|} & e_{ij} \in \mathcal{E} \\ -\sum_{i \neq k} w_{ik} & i = j \\ 0 & \text{otherwise} \end{cases} \tag{9.1}$$

$$a_{ij} = \begin{cases} \frac{1}{6}\|e_{ij}\| & e_{ij} \in \mathcal{E} \\ \frac{1}{3}\sum_{k \in \mathcal{N}(i)} \|e_{ik}\| & i = j \\ 0 & \text{otherwise} \end{cases} \tag{9.2}$$

and the notation is according to Figure 9.1, with $\mathcal{N}(i)$ being the set of the neighbors of node i. Note that in our tests we use non-lumped masses a_{ij}.

The product of two boundary-free 1D manifolds $\mathcal{M}, \mathcal{N}$ is a 2D manifold (a surface) $\times$ with torus topology. For the discretization of the Laplacian on $\mathcal{M} \times \mathcal{N}$, we appeal to the following:

Theorem 4. ***(Discrete product Laplacian)*** *Let $\mathcal{M}$, $\mathcal{N}$ be 1D manifolds with no boundary, discretized as 2-regular cycle graphs, and let $A_\mathcal{M}, S_\mathcal{M}$ and $A_\mathcal{N}, S_\mathcal{N}$ be the mass and stiffness matrices for $\Delta_\mathcal{M}$ and $\Delta_\mathcal{N}$ respectively, obtained via FEM with respect to piecewise linear (hat) basis functions. Then,*

$$A_{\mathcal{M}\times\mathcal{N}} = A_\mathcal{M} \otimes A_\mathcal{N} \tag{9.3}$$

$$S_{\mathcal{M}\times\mathcal{N}} = S_\mathcal{M} \otimes S_\mathcal{N} + S_\mathcal{M} \otimes S_\mathcal{N} \tag{9.4}$$

are the mass and stiffness matrices for the product manifold Laplacian $\Delta_{\mathcal{M}\times\mathcal{N}}$ with respect to piecewise bilinear basis functions, defined on a quad meshing of the toric surface $\mathcal{M}\times\mathcal{N}$. Here, $\otimes$ denotes the Kronecker product.

Proof. See Appendix B. □

Corollary 3. *The LB operator $\Delta_{\mathcal{M}\times\mathcal{N}}$ is discretized as:*

$$\Delta_{\mathcal{M}\times\mathcal{N}} = \Delta_{\mathcal{M}} \otimes I_{\mathcal{N}} + I_{\mathcal{M}} \otimes \Delta_{\mathcal{N}}\,, \tag{9.5}$$

where $I_{\mathcal{M}}, I_{\mathcal{N}}$ are $n_{\mathcal{M}}\times n_{\mathcal{M}}$ and $n_{\mathcal{N}}\times n_{\mathcal{N}}$ identity matrices.

Proof. See Appendix B. □

The discretization of $\Delta_{\mathcal{M}\times\mathcal{N}}$ does *not* require the explicit construction of a quad mesh embedded in $\mathbb{R}^4$; the toric shapes shown in these pages only serve visualization purposes. Further, the discretization (9.5) is consistent with the spectral decomposition identities (5.4); see **Fiedler** (1973) and **Hammack et al.** (2011), Proposition 33.6 for additional discussion.

9.2.2 2D shapes (surfaces)

As before, we model 2D surfaces as manifold triangle meshes $(\mathcal{V},\mathcal{E},\mathcal{F})$ with n vertices $\mathcal{V}$ connected by edges $\mathcal{E}=\mathcal{E}_\mathrm{i}\cup\mathcal{E}_\mathrm{b}$ (where $\mathcal{E}_\mathrm{i}$ and $\mathcal{E}_\mathrm{b}$ are interior and boundary edges, respectively) and triangle faces $\mathcal{F}$. In analogy to the 1D case, the discretization of the LB operator is obtained using FEM with piecewise linear basis functions on triangle elements **Duffin** (1959), taking the form of an $n\times n$ sparse matrix $\Delta = A^{-1}S$, where

$$s_{ij} = \begin{cases} (\cot\alpha_{ij} + \cot\beta_{ij})/2 & ij \in \mathcal{E}_\mathrm{i} \\ (\cot\alpha_{ij})/2 & ij \in \mathcal{E}_\mathrm{b} \\ -\sum_{k\neq i} w_{ik} & i = j \\ 0 & \text{otherwise, and} \end{cases} \tag{9.6}$$

$$a_{ij} = \begin{cases} (\mathrm{area}(T_{hij}) + area(T_{ijk}))/12 & ij \in \mathcal{E}_\mathrm{i} \\ \mathrm{area}(T_{ijk})/12 & ij \in \mathcal{E}_\mathrm{b} \\ \frac{1}{6}\sum_{k\in(i)} A(T_k) & i = j \\ 0 & \text{otherwise.} \end{cases} \tag{9.7}$$

Here, area(T) denotes the area of triangle T and $\mathcal{N}(i)$ is the set of the neighbors of vertex i; see Figure 9.1 for notation. These are equivalent to the matrices from Chapter 4, except they allow shapes with boundary.

Given two 2D manifolds $\mathcal{M}$ and $\mathcal{N}$, their product is a 4D manifold $\mathcal{M}\times\mathcal{N}$. The LB operator on $\mathcal{M}\times\mathcal{N}$ is discretized similarly the lower-dimensional case:

Corollary 4. *Let $\mathcal{M}$, $\mathcal{N}$ be surfaces discretized as triangle meshes, and let $\mathcal{S}_\mathcal{M}, \mathcal{W}_\mathcal{M}$ and $\mathcal{S}_\mathcal{N}, \mathcal{W}_\mathcal{N}$ be the mass and stiffness matrices for $\Delta_\mathcal{M}$ and $\Delta_\mathcal{N}$. Then, equations* (9.3)-(9.5) *provide a valid discretization of the LB operator $\Delta_{\mathcal{M}\times\mathcal{N}}$. This discretization is equivalent to the application of FEM on a 3-3 duoprism tessellation of the 4D product manifold $\mathcal{M}\times\mathcal{N}$ using multilinear basis functions.*

Proof. See Appendix B. □

We emphasize that the computation of the product Laplacian does not require constructing a high-dimensional embedding for $\mathcal{M}\times\mathcal{N}$, avoiding cumbersome manipulation of duoprismic product elements.

Finally, scalar functions on a manifold $\mathcal{M}$ are represented by n-dimensional vectors $f = (f(x_1), \dots, f(x_n))^\top$, where $x_1, \dots, x_n$ denote graph nodes and mesh vertices in the 1D and 2D case respectively. Inner products $\langle f, g\rangle_\mathcal{M}$ are discretized as $f^\top A g$, where A is the mass matrix. On product manifolds, scalar functions are represented as $n_\mathcal{M} \times n_\mathcal{N}$ matrices F, usually deriving from an outer product $f \otimes g$ discretized as $fg^\top$; inner products are computed as $\mathrm{vec}(F)^\top A\ \mathrm{vec}(G)$.

9.3 Map representation on the product manifold

In the following we will analyze the structure of certain type of maps from $\mathcal{M}$ to $\mathcal{N}$ on the product manifold. Our results are based on the functional map framework (**Ovsjanikov et al.** 2012). A functional map T associated to a map $\tilde{\pi} : \mathcal{M} \to \mathcal{N}$ is a linear mapping $T : \mathcal{F}(\mathcal{N}) \to \mathcal{F}(\mathcal{M})$ defined as in **Ovsjanikov et al.** (2012):

$$T(g) = g \circ \tilde{\pi}\,. \tag{9.8}$$

Note how this construction allows to move from identifying a map between manifolds to identifying a linear operator between Hilbert spaces. The functional map T admits a matrix representation wrt. orthogonal bases $\{\phi_i\}_{i\geq 1}, \{\psi_j\}_{j\geq 1}$ on $\mathcal{F}(\mathcal{M})$ and $\mathcal{F}(\mathcal{N})$ respectively, with coefficients $C = (c_{ij})$ determined as follows:

$$T(g) = \sum_{i,j\geq 1} \langle \psi_j, g\rangle_\mathcal{N} \underbrace{\langle \phi_i, T(\psi_j)\rangle_\mathcal{M}}_{c_{ij}} \phi_i\,. \tag{9.9}$$

Soft correspondences can be represented by their densities, *i.e.* nonnegative scalar functions $\mu : \mathcal{X}\times\mathcal{Y} \to [0,1]$ defined on the product manifold $\mathcal{X}\times\mathcal{Y}$ satisfying $\tilde{\mu}(x)(B) = \int_{B\subseteq\mathcal{Y}} \mu(x,y)\,\mathrm{d}y$ for all $x \in$ and all measurable subsets $B \subseteq \mathcal{Y}$. As a particular case, a bijection $\tilde{\pi} : \mathcal{X} \to \mathcal{Y}$ induces a soft map $\tilde{\mu}$ by requiring, for all $x \in \mathcal{X}$, that $\tilde{\mu}(x)(B) = 1$ if and only if $\tilde{\pi}(x) \in B \subseteq \mathcal{Y}$, *i.e.* the image $\tilde{\mu}(x)$ is a unit Dirac mass $\delta_{\tilde{\pi}(x)}$ centered at $\tilde{\pi}(x)$.

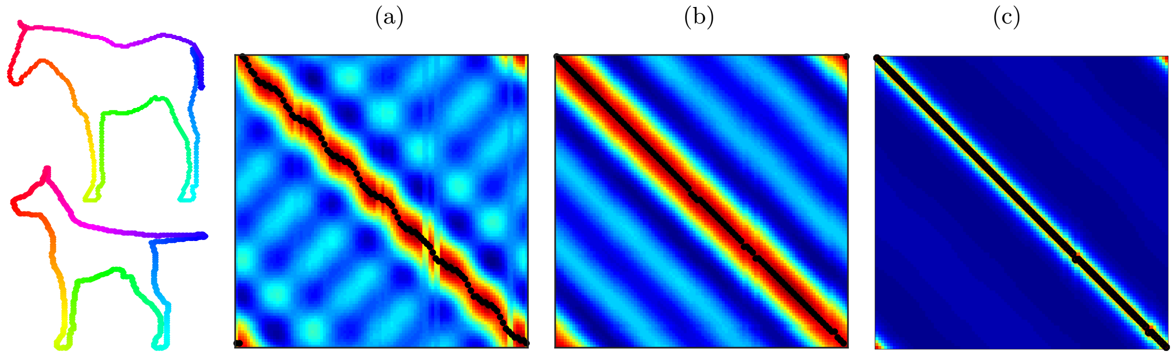


Figure 9.2: The ground truth map (here the identity) between the two shapes on the left as reconstructed using *(a)* the functional map representation with respect to LB eigenbases of the two shapes, *(b)* the separable LB eigenfunctions of the product manifold (here a flat torus, represented in the parametric domain), and *(c)* the inseparable localized harmonics on the product manifold. The black dots represent the maximum likelihood estimate for the underlying pointwise map (ideally, the identity).

9.3.1 Soft functional maps

It will be instrumental for our purposes to introduce a "soft" generalization of functional maps. For soft maps $\tilde{\mu} : \mathcal{M} \to \mathrm{Prob}(\mathcal{N})$ with associated density $\mu \in L^1(\mathcal{M} \times \mathcal{N})$, we define a *soft functional map* T as the expectation

$$T_\mu(g)(x) = \int_{\mathcal{N}} g(y)\mu(x,y)\,\mathrm{d}y\,. \tag{9.10}$$

It is easy to check that T_μ is linear in g, hence admitting a matrix representation with coefficients defined as in (9.9). If the density μ encodes a non-soft map (i.e. whenever $\mu(x,\cdot)$ is concentrated at one point), the definition (9.10) boils down to the original definition (9.8), $T(g)(x) = \int_{\mathcal{N}} g(y)\,\delta_{\tilde{\pi}(x)}(y)\,\mathrm{d}y = (g \circ \tilde{\pi})(x)$, where the last equivalence stems from the sampling property of Dirac deltas.

We begin our discussion by deriving a connection between functional map matrices and expanding soft map measures in the Laplace–Beltrami basis:

Theorem 5 (Equivalence)**.** *Let $T_\mu : \mathcal{F}(\mathcal{N}) \to \mathcal{F}(\mathcal{M})$ be a soft functional map* (9.10) *with underlying density $\mu \in L^1(\mathcal{M} \times \mathcal{N})$. Further, let $c_{ij} = \langle \phi_i, T_\mu(\psi_j) \rangle_{\mathcal{M}}$ be the matrix coefficients of T_μ in the orthogonal bases $\{\phi_i\}_{i \geq 1}, \{\psi_j\}_{j \geq 1}$, and let $p_{ij} = \langle \phi_i \otimes \psi_j, \mu \rangle_{\mathcal{M} \times \mathcal{N}}$ be the expansion coefficients of μ in the product basis $\{\phi_i \otimes \psi_j\}_{i,j}$, such that $\mu = \sum_{ij} (\phi_i \otimes \psi_j) p_{ij}$. Then, $c_{ij} = p_{ij}$ for all i, j.*

Proof. The functional map matrix coefficients are computed as:

$$
\begin{aligned}
c_{ij} = \langle \phi_i, T_\mu(\psi_j) \rangle_{\mathcal{M}} &= \int_{\mathcal{M}} \phi_i(x) T_\mu(\psi_j)(x)\, \mathrm{d}x && (9.11)\\
&= \int_{\mathcal{M}} \phi_i(x) \int_{\mathcal{N}} \psi_j(y)\mu(x,y)\, \mathrm{d}y\, \mathrm{d}x && (9.12)\\
&= \int_{\mathcal{M}\times\mathcal{N}} \phi_i(x)\psi_j(y)\mu(x,y)\, \mathrm{d}a\,, && (9.13)
\end{aligned}
$$

while the expansion coefficients of μ are given by

$$
p_{ij} = \langle \phi_i \otimes \psi_j, \mu \rangle_{\mathcal{M}\times\mathcal{N}} = \int_{\mathcal{M}\times\mathcal{N}} \phi_i(x)\psi_j(y)\mu(x,y)\, \mathrm{d}a\,. \tag{9.14}
$$

Comparing equations (9.13) and (9.14), we see that $c_{ij} = p_{ij}$ for any choice of $i, j \geq 1$. $\square$

Note that Theorem 5 applies to any choice of orthogonal bases $\{\phi_i\}_{i\geq 1} \in \mathcal{F}(\mathcal{M}), \{\psi_j\}_{j\geq 1} \in \mathcal{F}(\mathcal{N})$.

9.3.2 Spectral representation

Consider the order-k, band-limited approximation of μ:

$$
\mu \approx \sum_{\ell=1}^{k} \xi_\ell p_\ell\,, \tag{9.15}
$$

where each ξ_ℓ is an eigenfunction of $\Delta_{\mathcal{M}\times\mathcal{N}}$ which uniquely identifies, via (5.4), a pair of eigenfunctions ϕ_i, ψ_j on $\mathcal{M}$ and $\mathcal{N}$ respectively. According to Theorem 5, the expansion coefficients p_ℓ are exactly those appearing in the functional map matrix C, when this is expressed in the Laplacian eigenbases of $\mathcal{M}$ and $\mathcal{N}$ as originally proposed by **Ovsjanikov et al.** (2012). There is, however, a crucial difference in the way the two sets of coefficients are stored. We come to the following observation:

Truncation. The product eigenfunctions ξ_ℓ appearing in the summation (9.15) are associated to the product eigenvalues $\alpha_i + \beta_j$, which are ordered non-decreasingly. In contrast, in **Ovsjanikov et al.** (2012) it was proposed to truncate the two summations in (9.9) to $i = 1, \dots, k_{\mathcal{M}}$ and $j = 1, \dots, k_{\mathcal{N}}$, where indices i and j follow the non-decreasing order of the eigenvalue sequences α_i and β_j *separately*.

We see that, due to the different ordering, the eigenfunctions ϕ_i, ψ_j involved in the approximation (9.15) of μ are not necessarily all those involved in the construction of C (9.9), assuming $k = k_{\mathcal{M}} k_{\mathcal{N}}$. In the former case we operate with a reduced basis directly on $\mathcal{M} \times \mathcal{N}$, while in the latter case we consider two reduced bases on $\mathcal{M}$ and $\mathcal{N}$ *independently*. This has direct implications on the quality of the approximated maps, as illustrated in Figure 9.2.

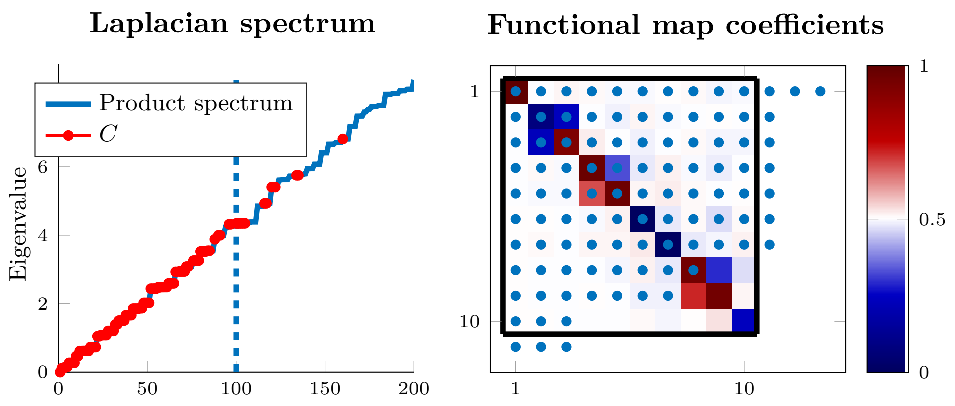


Figure 9.3: (Left) The $k = 100$ frequencies involved in the construction of a 10×10 functional map matrix C correspond to an irregular sampling of the Laplacian spectrum of the product manifold. (Right) In turn, only some of the coefficients c_{ij} of matrix C appear among the *first* k expansion coefficients p_{ij} of the map in the product eigenbasis. Here C is framed in black, while the blue dots identify the first k coefficients p_{ij}.

9.3.3 Relation to finite sections

The functional map representation was originally introduced in **Ovsjanikov et al.** (2012) as a convenient language for solving map inference problems of the type:

$$CA = B\,, \tag{9.16}$$

where matrices $B = (\langle \phi_i, f_j \rangle_{\mathcal{M}})$, $A = (\langle \psi_i, g_j \rangle_{\mathcal{N}})$ contain Fourier coefficients of a given set of corresponding "probe" functions $f_j, g_j, j = 1, \ldots, q$ on $\mathcal{M}$ and $\mathcal{N}$, respectively. Typically, descriptors are used. In the problem above, one is asked to estimate the functional map C.

By truncating the matrix C to the left upper $k_{\mathcal{M}} \times k_{\mathcal{N}}$ submatrix as proposed in **Ovsjanikov et al.** (2012), one obtains a finite-dimensional approximation of the infinite linear system (9.16). This procedure, known as the *finite section method* (**Gröchenig et al.** 2010), does not always guarantee convergence, and a series of remedies using rectangular sections ($k_{\mathcal{M}} \neq k_{\mathcal{N}}$) have been proposed in the literature for general systems (see **Glashoff and Ortlieb** (2017) for a discussion pertaining to functional maps).

Recall that, according to Theorem 5, the matrix elements c_{ij} correspond to the expansion coefficients p_{ij} appearing in (9.15). Thus, the approximation carried out in (9.15) can be regarded as a kind of "irregular" finite section (see Figure 9.3). In contrast with purely *algebraic* approaches considering general systems of linear equations, however, our approach carries a *geometric* meaning in that it directly reflects the geometry of the correspondence manifold.

9.4 Spectral Map Processing

We consider curves and surfaces as our shapes. Despite their different intrinsic dimensions, our framework applies to both without specific adjustment.

9.4.1 Localized Spectral Encoding

Theorem 5 establishes the equivalence between the soft functional map T_μ representation coefficients c_{ij} in the bases $\{\phi_i\}_{i\geq 1} \subseteq \mathcal{F}(\mathcal{M})$ and $\{\psi_j\}_{j\geq 1} \subseteq \mathcal{F}(\mathcal{N})$ and the coefficients p_ℓ of the underlying density μ Fourier series (9.15) in the eigenbasis $\{\xi_\ell\}_{\ell\geq 1} \subseteq \mathcal{F}(\mathcal{M}\times\mathcal{N})$ of the product manifold Laplacian $\Delta_{\mathcal{M}\times\mathcal{N}}$. This equivalence directly stems from ξ_ℓ's having the separable form $\phi_i \otimes \psi_j$, see Section 5.2.3. It may be advantageous, however, to consider different orthonormal bases on $\mathcal{M}\times\mathcal{N}$ that are not necessarily separable. In particular, we observe that μ tends to be localized on the product manifold $\mathcal{M}\times\mathcal{N}$ (see Figure 9.2), and thus the standard outer product basis is extremely wasteful as it is supported on the entire $\mathcal{M}\times\mathcal{N}$.

A better alternative is the use of *localized manifold harmonics* (**Choukroun et al.** 2018; **Melzi et al.** 2017). Assume that we are given a rough indication of the support of μ in the form of a step potential function

$$V(x,y) = \begin{cases} \nu & \mu(x,y) \approx 0; \\ 0 & \text{otherwise.} \end{cases} \tag{9.17}$$

where $\nu \geq 1$. Then, the variational problem

$$\begin{aligned} \min_{\xi_1,\dots,\xi_k} \quad & \sum_{\ell=1}^{k} \int_{\mathcal{M}\times\mathcal{N}} \left(\|\nabla_{\mathcal{M}\times\mathcal{N}} \xi_\ell\|^2_{g_\mathcal{M}\oplus g_\mathcal{N}} + V\xi_\ell^2 \right) \mathrm{d}a \\ \text{s.t.} \quad & \langle \xi_\ell, \xi_{\ell'} \rangle_{\mathcal{F}(\mathcal{M}\times\mathcal{N})} = \delta_{\ell,\ell'} \end{aligned} \tag{9.18}$$

produces a set of orthonormal functions denoted by $\hat{\xi}_1,\dots,\hat{\xi}_k$ that, for a sufficiently large value of ν, are also localized in the support of V. Note that this new basis $\{\hat{\xi}_\ell\}_{\ell=1}^k$ is *no more separable*, i.e., the functions $\hat{\xi}$ are not in general expressible as outer products of functions defined on the originating domains. See Figures 9.4 and 9.5 for an illustration.

The basis $\{\hat{\xi}_\ell\}_{\ell=1}^k$ turns out to be the eigenbasis of the *Hamiltonian operator* (**Choukroun et al.** 2018) $H = \Delta_{\mathcal{M}\times\mathcal{N}} + V$ and can be computed by the eigendecomposition of the product Laplacian matrix with the addition of diagonal potential. We note that the size of such problem can be huge (if the shapes are discretized with $n \sim 10^3$ points, the product Laplacian matrix has size $n^2 \times n^2 = 10^6 \times 10^6$; see Theorem 4), and despite its extreme sparsity, computationally expensive.

As an alternative, we consider a *patch* $\mathcal{P} \subset \mathcal{M}\times\mathcal{N}$ of the product manifold corresponding to $V > 0$, and define the eigenproblem

$$\begin{aligned} \Delta_\mathcal{P}\bar{\xi}_\ell(x,y) &= \gamma_\ell \bar{\xi}_\ell(x,y) & (x,y) &\in \mathrm{int}(\mathcal{P}) \\ \bar{\xi}_\ell(x,y) &= 0 & (x,y) &\in \partial\mathcal{P} \end{aligned} \tag{9.19}$$

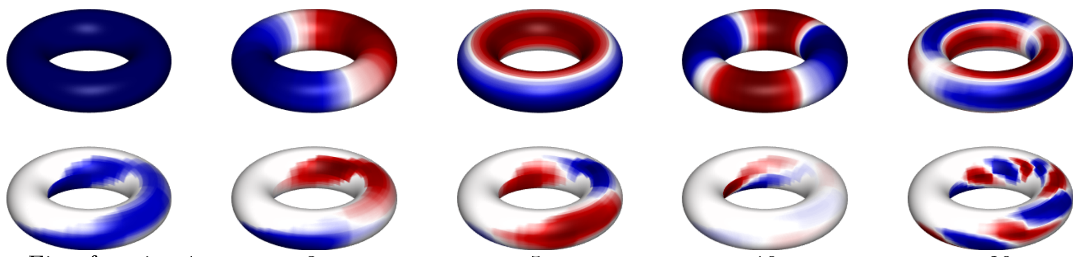


Figure 9.4: Examples of basis functions on the product manifold (here visualized as a torus embedded in $\mathbb{R}^3$) of two 1D shapes. We plot a few standard LB eigenfunctions (top row) and localized manifold harmonics (bottom row). The first basis function in the bottom row also indicates the used region. Here and in the following, we use the present color scheme (blue denotes negative values, red positive values, white is zero).

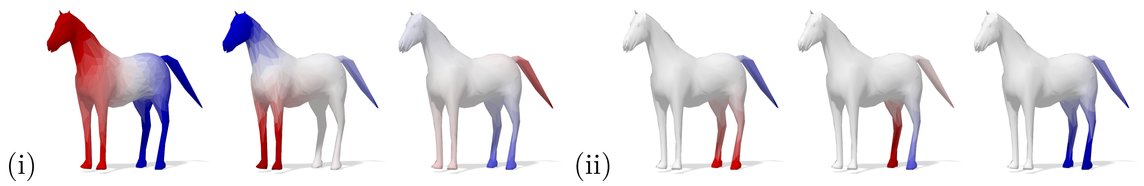


Figure 9.5: Projecting the basis functions on the product manifold of horse and elephant back onto the factor shapes (here only the horse projection is visualized). (i) Projection of three product LB eigenfunctions, which correspond exactly to three standard LB eigenfunctions on the horse shape. (ii) Projection of three localized harmonics; these projections do *not* correspond to any LB eigenfunction on the horse. Still, note how they capture the geometric features of the underlying shape.

of the *product patch Laplacian* $\Delta_{\mathcal{P}}$ with Dirichlet boundary conditions. If the patch is selected in such a way that its size scales as $\mathcal{O}(n)$ rather than $\mathcal{O}(n^2)$ in the size of the shapes (in practice, this can be achieved by taking a fixed-size band around the initial correspondence), the computation of the localized basis $\{\bar{\xi}_\ell\}_{\ell=1}^k$ has the same complexity as eigendecomposition of the individual Laplacians $\Delta_{\mathcal{M}}, \Delta_{\mathcal{N}}$.

9.4.2 Map refinement

As an illustrative application of our framework, we introduce a simple procedure for recovering dense maps from sparse inputs.

We start with an initial, possibly sparse and noisy, correspondence which will be iteratively refined. The input correspondence can be used to define a heat distribution u_0 on the product manifold that is zero everywhere except on the vertices representing the input correspondence. u_0 is diffused generously such that information about the spatial closeness is exchanged between neighboring inputs, $u = hd(u_0, t)$. This function can be approximated using the LB eigenfunctions and eigenvalues for the entire product manifold which can be easily constructed as described in Section 5.2.2. We threshold u to a connected region R spanning each of the orig-

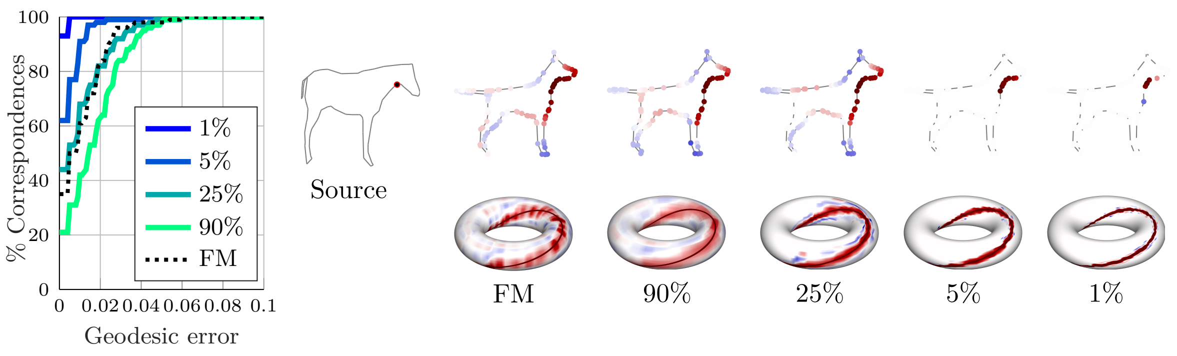


Figure 9.6: Product space approximation of the correspondence between one-dimensional shapes with $k = 100$ basis functions. Bases constructed on bands of different size (1%, 5%, 25% and 90% of the total product manifold area) around the true correspondence are shown. Separable basis (FM) is shown as a reference. Left: accuracy of the correspondence increases the product space basis becomes more localized. Right (top row): image of a delta function by the functional maps. Right (bottom row): True correspondence (curve) and its approximation in inseparable product space bases with a varying degree of localization. The product manifold is depicted as a two-dimensional torus (first row).

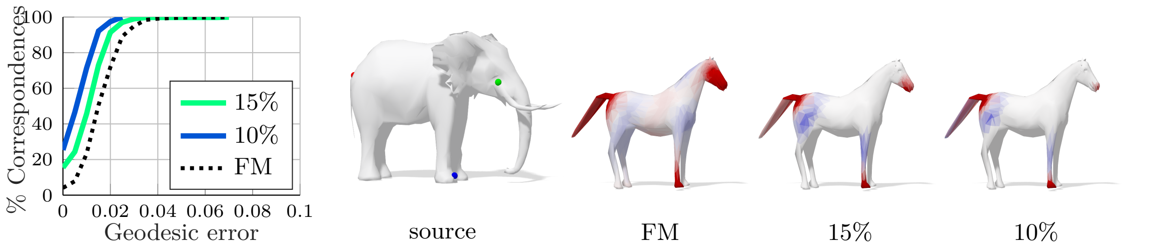
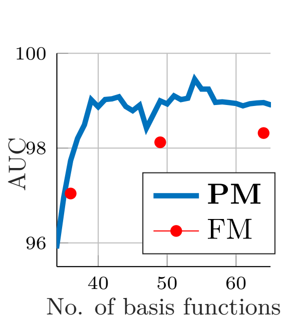


Figure 9.7: Map approximation between two-dimensional shapes (surfaces) with $k = 500$ basis functions on bands of different size (10% and 15% of the total 4D product manifold area) around the true correspondence. We also show images on the horse of delta functions supported at three points (red, green, blue) on the elephant. Here, the functional map (FM) was calculated using $30 \times 30 = 900$ basis functions.

inal shapes completely. This can be solved using a simple min-max optimum and the region can be expanded until connected if not initially the case. A dense, intermediate correspondence is found by solving a sparse LAP (using the previously extracted region) or maximum likelihood. This correspondence defines the initial heat distribution u_0 for the next iteration, but this time the heat is only diffused on R using spectral approximation with Dirichlet eigenfunctions on R and a for a shorter time. The decreased region makes it possible to increase accuracy in the approximation with the same amount of eigenfunctions. This process is repeated until R is really tight around the found correspondence. An example result on two surfaces can be seen in Figure 9.8. Notice that the heat distribution in the first iteration does not need to come



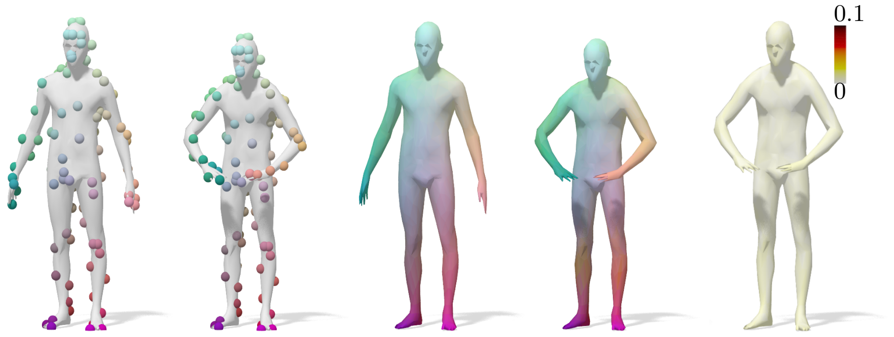


Figure 9.8: (Left) AUC at increasing number k of basis functions, respectively localized harmonics on the product manifold (PM) and standard LB eigenfunctions (FM); note that FM can only be applied for factorizable k due to the construction of matrix C (in this plot, $k = 36, 49, 64$). (Right) An example of the proposed map refinement. We show the input correspondence on top (sparse point-to-point matches, $\sim 10\%$ of all points) and the recovered dense map below. The heatmap on the bottom right encodes geodesic error of the recovered correspondence.

from a input correspondence, but can be constructed from any pointwise similarity function.

9.5 Conclusion

This chapter introduced a novel perspective on map representation and processing, where pointwise, functional, and soft maps can be understood as densities on the product of the input shapes. We saw that the Laplace-Beltrami operator can be discretized on the product manifold and has both separable default eigenfunctions to which we proposed the adoption of inseparable localized harmonics for compactly encoding correspondences while ensuring minimal energy dispersion. We applied a similar strategy as in Chapter 8 to solve for a minimal surface submanifold in the product space representing a diffeomorphism. However, this time we operated directly on the product manifold. Our theoretical contributions suggest a new perspective on properties of the correspondence manifold as well as the possibilities of more accurate representation for map inference and processing.

The main limitation of the approach lies in its scalability. While we showed that one can reduce the computational complexity to $\mathcal{O}(n)$ by appropriately selecting a localization region, considering (as a possible extension) higher-dimensional products to encode cycle-consistent maps in shape collections may soon become prohibitive. With the current approach we trade off scalability for accuracy: Maps are encoded much more precisely in the localized basis, but this requires the explicit computation of inseparable basis functions that do not admit an efficient representation in terms of outer products.

A promising direction is the introduction of product spaces within geometric deep learning pipelines, where the data is in the form of signals defined on top of a manifold. Our proposed discretization of the product Laplace-Beltrami operator, as well as its spectral decomposition, can be directly employed in such pipelines, enabling new forms of structured prediction in a range of challenging problems in vision, graphics and geometry processing.

CHAPTER 10

Continuous Deformation for Correspondence

The previous chapters looked at correspondences mostly in the form of (local) diffeo- or homeomorphisms. This is meaningful in terms of preserving neighborhood information on the surface, but most shapes are not uniquely defined through their surface. Imagine a balloon, once deflated and once inflated, and ignoring stretching of the material for a moment. Although the two surfaces are identical, it is often hard to tell what shape the balloon will assume before filling it with air. The missing information is how the volume is distributed on the inside. In this chapter we will look at how to model continuity, but not through surface alignment as before, but instead in how the inputs were deformed into each other. To that end, we consider two shapes with the same volume and try to find a volume-preserving deformation field that aligns their surfaces. The main idea is that a good correspondence will come from physically

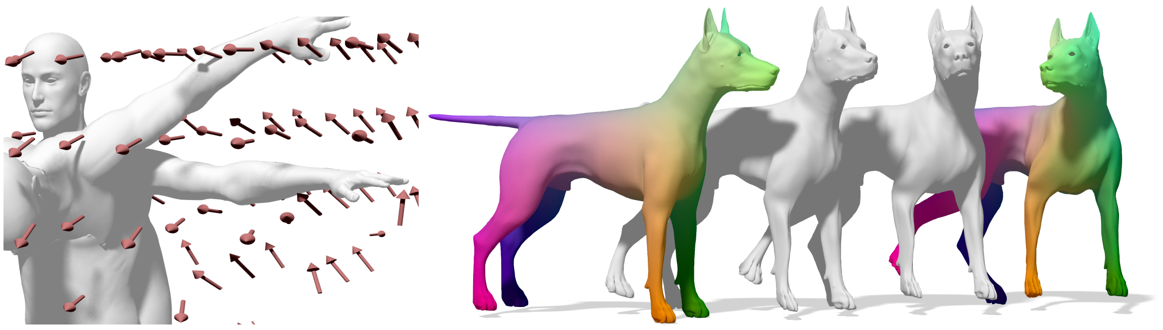


Figure 10.1: Given two input shapes, we propose to morph the source shape along a divergence-free deformation field in order align it with the target. (Left) Example of a deformation field in 3D. (Right) Example of the results of our framework. We alternate between optimizing for the deformation field and calculating correspondences. As a result, we generate highly accurate correspondences (color coded) as well a sequence of natural intermediate shapes as a by-product (white). Translation is only added for clearness in the figure.

meaningful deformation. This deformation is supposed to be low energy if it was based on a good correspondence, because no unnecessary stretching or bending is needed. We model low energy through smoothness of the deformation field. To achieve this, we do not only have to solve for the correspondence but also for the deformation field. We will show here that both benefit a joint optimization.

By definition this method cannot be purely intrinsic because volume is an extrinsic property. Therefore, it does not suffer from common problems of purely intrinsic methods like confusing left and right side, as can happen to the Kernel Matching approach from Chapter 8. On the other hand, intrinsic or very local measures are robust against extreme extrinsic changes, but we will show that the volume-preservation constraint is strong enough to not constantly fall into local optima. Directly deforming the geometry in the embedding space often yields very regular correspondences without extreme outliers (**Ma et al.** 2014). In particular, this direction allows for the creation of new, intermediate versions of the input shapes. But these methods are in general more prone to get stuck in local minima and therefore dependent on a good initial alignment of the inputs. Unfortunately, many extrinsic matching methods use linear mappings to model surface deformations (**Ma et al.** 2014; **Myronenko and Song** 2010). While this is feasible for small changes, it is often not compatible with how objects deform in the real world. On the other hand, finding a physically correct morphing between two shapes is highly complex and computationally intense, even when the perfect correspondence or prior knowledge about the input is given (**Gao et al.** 2017; **Wirth et al.** 2011).

This chapter looks at a more plausible morphing model than linear which takes into account volume-preservation during the entire deformation and can optimize for a correspondence and an interpolation at the same time. This is possible by modeling volume-preservation through zero divergence in a deformation field. This property makes our intermediate shapes more natural and our results are less likely to end up in a local minimum than with a linear mapping. In our method, the deformation field is represented in a spatially continuous, coarse-to-fine basis which allows for an efficient optimization and incentivizes low energy deformations. Moreover, the optimization can be decoupled from the resolution of the shape, and we can process shapes of arbitrary resolution with a minimal increase in complexity.

This chapter is based on **M. Eisenberger**, **Z. Lähner**, and **D. Cremers** (2019). Divergence-Free Shape Correspondence by Deformation. In: *Computer Graphics Forum (CGF)* 38.5.

10.1 Related Work

This section focuses on work directly related to this chapter. A more general overview can be found in Chapter 1.

10.1.1 Shape Registration and Matching

One recent line of work in shape matching is based on spectral decomposition of the surface Laplace-Beltrami operator (**Dubrovina and Kimmel** 2010). This is popular, because it reduces the dimensionality of the problem from the number of vertices to the number of basis functions chosen (**Ovsjanikov et al.** 2012). Nevertheless, extracting the correspondence from the low dimensional representation is still a complex problem and often the retrieved solutions are noisy or hard to compute (**Rodolà et al.** 2015). One major problem with purely spectral approaches is that intrinsic symmetries can not be distinguished, **Ren et al.** (2018) being one of few exceptions. We also use a spectral approach, but, instead of a basis for functions on the surface, we represent deformation fields in the embedding space using the eigenfunctions of the standard Laplacian. Among other things, the embedding space allows us to distinguish between intrinsincally symmetric but opposite points.

Methods based on Multi-Dimensional Scaling find correspondences by re-embedding and then aligning shapes in a (possibly smaller) embedding space with reduced complexity (**Aflalo et al.** 2016; **A. M. Bronstein et al.** 2006b). **Chen and Koltun** (2015) calculate a robust non-rigid registration based on Markov random fields but cannot retrieve a continuous deformation which we do. In **Myronenko and Song** (2010) and **Ma et al.** (2014) the authors address the non-rigid registration problem by modeling one point cloud as a Gaussian mixture model, similar to our method. Moreover, they also determine the correspondences and point mappings in an alternating manner using a expectation maximization algorithm. This work is strongly related to our framework, but no intermediate deformation is modeled. There also exist extensions of this method which additionally include descriptor values (**Ma et al.** 2017; **Ma et al.** 2016). **Q.-X. Huang et al.** (2008) achieve accurate non-rigid alignments but rely on good initial correspondence and expensive geodesic distance computation to find these.

10.1.2 Deformation Fields

Deformation fields have a long history in image registration. One of the first approaches in that direction is the Large Deformation Diffeomorphic Metric Mapping (LDDMM) framework (**Beg et al.** 2005). **Ashburner** (2007) made use of deformation fields for autonomous shape morphing. They consider temporally constant deformation fields offering limited flexibility to capture more complex deformations. Solving for a space and time dependent deformation field is a highly under-determined problem. A remedy for this issue is provided by the geodesic shooting approach advocated by **Miller et al.** (2006) which only estimates the initial velocity field for each pixel. Then the velocity propagates in the image domain in order to preserve the kinetic energy and the momentum of the whole system. Further improvements of this framework were proposed in subsequent work, including a Gauss-Newton approach (**Ashburner and Friston** 2011) and a particularly efficient adjoint calculation (**Vialard et al.** 2012).

Closely related to our work is **Funck et al.** (2006) in which the authors also model volume preserving shape deformations using divergence-free vector fields. Here, deformation fields are constructed from hand crafted templates which are meant to be used as interactive shape transformation tools, whereas our method is fully automated. As in our work, in **Adams et al.** (2008) the deformations are based on a subsampling of the input shapes and can be efficiently applied to the full resolution, but the correspondence is assumed to be given.

Probabilistic interpretations of deformation fields are a popular formulation. Such a model for image registration and 2D shape registration with a Gaussian process modeling of the correspondence mapping is proposed in **Albrecht et al.** (2008). Further work specified how one can extend this approach to Gaussian processes on the surface of a three dimensional shape (**Dölz et al.** 2017; **Lüthi et al.** 2016). **Bregler et al.** (2000), **Torresani et al.** (2008), **Albrecht et al.** (2008) and **Paladini et al.** (2009) also model non-rigid transformations using a PCA type representation of permitted motions. Analogously, **Myronenko and Song** (2010) and **Ma et al.** (2014) pursue a reproducing kernel Hilbert space approach to model the vector field interpolation. However, for all these references the respective vector fields are not defined on the whole embedding space surrounding the shapes but rather only at the elements of the considered point clouds. Hence, they do not admit an interpretation as a deformation field which makes is harder to impose global properties, e.g. volume-preservation.

Another classical approach to shape deformation is based on a rotation invariant representation of triangle meshes (**Lipman et al.** 2005). In **Zhang et al.** (2008) this deformation model is used to compute a sparse set of correspondences but this method is hard to scale to high resolutions.

10.2 Contribution

We introduce a mathematical framework which solves the correspondence problem on two shapes with approximately the same volume. For this purpose, we propose to alternate between estimating the correspondences and a smooth 3D deformation field aligning the two input shapes. Our shape morphing model solves an initial value problem to shift the first shape along this deformation field. Numerically, this differential equation is integrated using a second order Runge-Kutta scheme. Our framework allows us to incorporate physical assumptions about the deformations by directly building them into the model. We suggest to impose volume preservation by enforcing the deformation fields to have zero divergence. More specifically, we define a coarse-to-fine basis representation of these vector fields where each basis function is divergence-free. This allows us to reduce the complexity by optimizing only over the most significant coefficients. We use an expectation-maximization approach to simultaneously compute a subset of the unknown point-to-point correspondences and the optimal deformation field coefficients. A schematic diagram of the complete pipeline can be found in Figure 10.2. We demonstrate that the proposed framework can be used to solve for correspondences which are on par with state-of-the-art methods. Moreover, our method can produce a sequence of reasonable intermediate shapes between the inputs as a by-product. Both can be scaled up to arbitrary resolution without a significant increase in complexity which we demonstrate on on a dataset of real scans with over $100k$ vertices.

10.3 Problem Formulation

In the following, we define the problem we want to solve and the mathematical background we use in later sections. In general we consider two point clouds $\mathcal{X} = \{x_1, \dots, x_N\} \subset \Omega$ and $\mathcal{Y} = \{y_1, \dots, y_M\} \subset \Omega$ contained in a compact domain $\Omega \subset \mathbb{R}^D$. In practice we choose $\Omega = [0, 1]^D$. The points x_n and y_m are samples from the surface of two similar $(D-1)$-dimensional Riemannian manifolds embedded in $\mathbb{R}^D$. Our method aims at aligning the point clouds $\mathcal{X}$ and $\mathcal{Y}$ in a meaningful manner. In particular, we are looking for a mapping $f : \mathcal{X} \to \Omega$ which provides the coordinates for a new embedding of each point on $\mathcal{X}$. In the end, $f(\mathcal{X})$ should be well aligned with $\mathcal{Y}$.

10.3.1 Deformation field shape morphing

We propose to model the shape morphing $f : \mathcal{X} \to \Omega$ using the following initial value problem:

$$\begin{cases} \dot{x}(t) = v(x(t)). \\ x(0) = x_{\text{init}}. \end{cases} \tag{10.1}$$

In this context, $v : \Omega \to \mathbb{R}^D$ is some fixed deformation field shifting any point $x_{\text{init}} \in \Omega$ over time. If we solve this differential equation until some fixed time t_{eval}, we get the flow $\varphi : [0, t_{\text{eval}}] \times \Omega \to \Omega$ of Eq. (10.1). The flow φ morphs the space Ω over time, it maps any input point x_{init} to its destination $\varphi(t, x_{\text{init}})$ at time $t \in [0, 1]$. Applying Eq. (10.1) to all points $x_{\text{init}} := x_n \in \mathcal{X}$ yields a morphing model for the source shape $\mathcal{X}$:

$$f(x_n) := f_n := \varphi(t_{\text{eval}}, x_n). \tag{10.2}$$

In order to make those shape deformations more plausible, we require them to be smooth in space and in time. For this purpose, we assume that the deformation fields $v \in C^\infty(\Omega, \mathbb{R}^D)$ which, according to the Picard-Lindelöf Theorem, yields smooth point trajectories $x(\cdot) := \varphi(\cdot, x_{\text{init}}) \in C^\infty([0, 1], \Omega)$, see **Teschl** (2012), Lemma 2.3, Theorem 2.5. For convenience we choose $t_{\text{eval}} = 1$ in our experiments.

Our morphing model computes natural shape deformations which can be transformed into correspondences through nearest neighbor search (See Section 10.4.3). Due to the time dependency of the flow, we additionally get intermediate poses of the input shape at times $t \in (0, 1)$ which constitute the underlying transformation. Those are typically more meaningful than naive approaches like linear interpolation between the points. We believe that having a continuous correspondence and a natural deformation are inherently connected, and solving for both simultaneously improves the results considerably.

10.3.2 Divergence-free deformations

One advantage of our morphing model Eq. (10.1) is that it allows us to incorporate assumptions about the deformation fields v into our model. In our framework we restrict these velocity

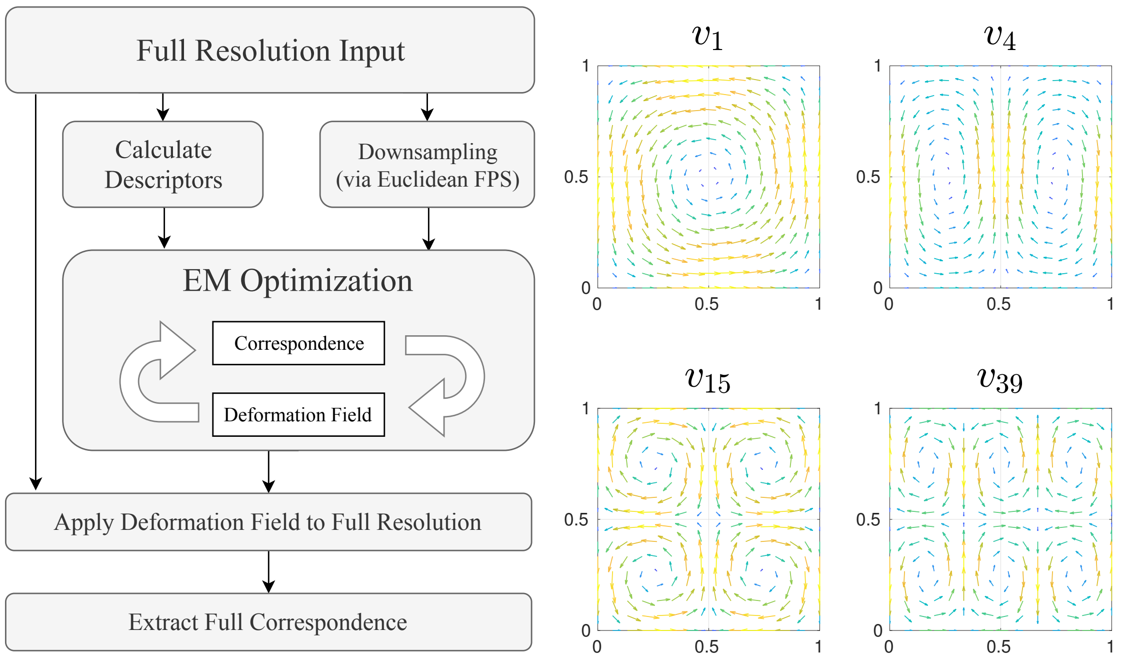


Figure 10.2: (Left) Overview over our complete pipeline. (Right) Cross section of some deformation field basis functions $v_k : \Omega \to \mathbb{R}^3$ at $x_3 = 0.5$. Notice the low frequency structures for low k and increasing frequencies with higher indices. Furthermore, one can see that our deformation fields have no flow in and out of the domain Ω at the boundary.

fields to be divergence-free, an assumption that is commonly used in mathematical modeling of in-compressible fluids (**Chorin and Marsden** 1993):

$$\nabla \cdot v = 0. \tag{10.3}$$

A well known consequence of this local property is that it yields volume preservation over time for any subset $U \subset \Omega$ of the embedding space. In particular, we can consider the set of solutions of Eq. (10.1):

$$U(t) := \left\{ \varphi(t, x_{\text{init}}) \in \Omega \middle| x_{\text{init}} \in U \right\}. \tag{10.4}$$

Then the assumption in Eq. (10.3) yields that each morphed set $U(t)$ has the same volume as U (**Teschl** 2012, Lemma 8.8). Therefore, each subvolume of the input shape $\mathcal{X}$, as well as of the embedding space, is preserved at any given time. Notice that this property is stronger than global volume preservation of the interior of $\mathcal{X}$ only. In general, two very differently shaped objects can have the same volume. However, for our method the volume of all, potentially very small, subparts is preserved. In our experiments, we found that this is a reasonable assumption for real world deformations and it provides a good regularization of our morphing model Eq. (10.1).

10.3.3 Helmholtz decomposition

Helmholtz's theorem (**Rutherford** 1962) implies that any sufficiently smooth vector field on the compact domain Ω can be decomposed into the sum of a curl-free, a divergence-free and a harmonic component. It furthermore provides us with an explicit construction of the divergence-free component that we are interested in:

$$v := \nabla \times \Phi. \tag{10.5}$$

In this context, $\Phi : \Omega \to \mathbb{R}^D$ is a potential function and $\nabla \times \cdot$ is the curl operator. Indeed, **Creusé et al.** (2015), Lemma 2.2 shows that such a Φ exists for any divergence-free, C^∞ vector field $v : \Omega \to \mathbb{R}^D$ with no outflow at the boundary:

$$\langle v, n \rangle = 0 \text{ on } \partial\Omega. \tag{10.6}$$

Furthermore, for a given Φ we always get a divergence-free vector field v as a basic property of the curl operator:

$$\nabla \cdot (\nabla \times \Phi) = 0. \tag{10.7}$$

To further restrict the space of admissible deformation fields, we additionally require the potential functions to admit Dirichlet boundary conditions $\Phi|_{\partial\Omega} = 0$. This guarantees that the potential functions are tangential to the outer normals at $\partial\Omega$, which is a necessary condition in the existence proof, see **Creusé et al.** (2015), Lemma 2.2. Moreover, we are only interested in a high expressibility in the interior of Ω, and choosing Dirchlet boundary conditions makes the representation of our deformation fields even more compact. Intuitively, it guarantees that for the resulting deformation fields v there is no flow in and out of the domain Ω (see Equation (10.6)). In the case of $D = 3$ spatial dimensions the construction of v in Eq. (10.5) admits the following form:

$$v = \begin{pmatrix} \partial_2\Phi_3 - \partial_3\Phi_2 \\ \partial_3\Phi_1 - \partial_1\Phi_3 \\ \partial_1\Phi_2 - \partial_2\Phi_1 \end{pmatrix} = \begin{pmatrix} 0 \\ \partial_3\Phi_1 \\ -\partial_2\Phi_1 \end{pmatrix} + \begin{pmatrix} -\partial_3\Phi_2 \\ 0 \\ \partial_1\Phi_2 \end{pmatrix} + \begin{pmatrix} \partial_2\Phi_3 \\ -\partial_1\Phi_3 \\ 0 \end{pmatrix}. \tag{10.8}$$

Remark. The harmonic component in the Helmholtz decomposition corresponds to global translations of the input shape $\mathcal{X}$, but we refrain from including them in our framework. For once, we would like the flow $\varphi : [0,1] \times \Omega \to \Omega$ to map all points $x_n \in \Omega$ back to the same domain. Furthermore, modeling global translations is not necessary because we shift the input shapes a priori such that their empirical mean corresponds to the center of Ω.

10.4 Method

In the following, we outline the core components of our method. First, we construct a coarse-to-fine deformation field basis with certain built-in properties like volume preservation (Section 10.4.1). Then, we show how to integrate the initial value problem of Eq. (10.1)

(Section 10.4.2). Finally, we provide details about our expectation maximization algorithm (Section 10.4.3). There, we simultaneously optimize for the unknown correspondences and an appropriate deformation field. Regarding relevant applications, we will mainly restrict ourselves to the case of 2D shapes embedded in $\mathbb{R}^3$.

10.4.1 Spatial representation

Standard discretizations of vector fields v using voxel grids have cubic complexity which makes them too costly for any reasonable resolution. To get a more compact representation, we introduce a low rank basis $\{v_1, ..., v_K\}$ of spatially dense, divergence-free deformation fields. The number of basis functions can be adjusted for either speed or expressiveness. Without loss of generality we set the domain to a D-dimensional cube $\Omega := [0, 1]^D$. In practice, we then translate and scale any shape to generously fit inside. We begin with defining a basis for the potential fields Φ. For this purpose, consider the eigenfunctions $\{\phi_1, \phi_2, ...\}$ and eigenvalues $\{\lambda_1, \lambda_2, ...\}$ of the scalar Laplacian Δ on Ω:

$$\Delta\phi_k = \lambda_k^\Delta \phi_k. \tag{10.9}$$

This basis of eigenfunctions $\{\phi_1, \phi_2, ...\}$ is ordered with descending eigenvalues $0 \geq \lambda_1^\Delta \geq \lambda_2^\Delta \geq ...$. Furthermore, we require the potential fields to admit Dirichlet boundary conditions $\Phi|_{\partial\Omega} = 0$. These ϕ_k can be computed analytically, because they are exactly the sine elements of the Fourier basis:

$$\mathcal{B}_\phi = \left\{ \phi : [0,1]^D \to \mathbb{R},\ x \mapsto \prod_{d=1}^{D} \frac{1}{2} \sin(x_d \pi j_d) \middle| j \in \mathbb{N}^D \right\}. \tag{10.10}$$

The set $\mathcal{B}_\phi = \{\phi_1, \phi_2, ...\}$ is ordered by ascending Dirichlet energy of the ϕ_k. These ϕ_k form an orthonormal basis wrt. the $\langle \cdot, \cdot \rangle_{L^2(\Omega)}$ inner product for scalar functions on Ω. We can now use $\mathcal{B}_\phi$ to construct a basis for the deformation fields $\mathcal{B}_v$ according to Eq. (10.5). Note that the basis $\mathcal{B}_\phi$ consists of scalar functions while the potential functions $\Phi : \Omega \to \mathbb{R}^D$ are vector valued. However, due to the linearity of the curl $\nabla \times \cdot$, we obtain a basis by using Eq. (10.5) one entry at a time. For $D = 3$ this can be done as follows:

$$\begin{aligned}
\mathcal{B}_v &= \bigcup_{k=1}^{\infty} \left\{ \nabla \times \begin{pmatrix} \phi_k \\ 0 \\ 0 \end{pmatrix}, \nabla \times \begin{pmatrix} 0 \\ \phi_k \\ 0 \end{pmatrix}, \nabla \times \begin{pmatrix} 0 \\ 0 \\ \phi_k \end{pmatrix} \right\} \\
&= \bigcup_{k=1}^{\infty} \left\{ \begin{pmatrix} 0 \\ \partial_3\phi_k \\ -\partial_2\phi_k \end{pmatrix}, \begin{pmatrix} -\partial_3\phi_k \\ 0 \\ \partial_1\phi_k \end{pmatrix}, \begin{pmatrix} \partial_2\phi_k \\ -\partial_1\phi_k \\ 0 \end{pmatrix} \right\}.
\end{aligned} \tag{10.11}$$

We get three deformation basis functions for each ϕ_k in Eq. (10.10). Analogously to the potential fields, the basis elements $\mathcal{B}_v = \{v_1, v_2, ...\}$ are again sorted according to the eigenvalues λ_k^Δ of the corresponding ϕ_k in descending order. Note that there are in general multiple basis functions v_k for each eigenvalue λ_k^Δ. Overall, we obtain arbitrary deformation fields v as the linear combination of the first K basis elements v_k with some coefficients a_k:

$$v(x) = \sum_{k=1}^{K} v_k(x) a_k. \tag{10.12}$$

Remark. One aspect we would like to discuss in this context is our choice of domain $\Omega = [0,1]^D$. The first basis function v_1 in Figure 10.2 is equivalent up to first order to a rotation around the x_3 axis. This especially holds near the center of the domain Ω and deteriorates at its boundary $\partial\Omega$. Those considerations raise the question whether a cubic domain Ω is the best choice for our purposes. Following the work in **Zhao and Burge** (2007); **Zhao and Burge** (2008), we could pursue our approach in a spherical domain. This would lead to more complex basis functions v_k but the first three eigenfunctions would span the space of rotations without undesirable artifacts at the boundaries of the domain. Although this would be a nice theoretical property, we refrain from using these basis functions here due their complex structure.

10.4.2 Temporal discretization

In order to evaluate the correspondence mapping f in Eq. (10.2), we have to solve the initial value problem Eq. (10.1) with a numerical integration scheme. The simplest choice in this context is the explicit Euler method. However, we decided to use a second order Runge-Kutta method (**Griffiths and Higham** 2010, Ch. 9), because it has a significantly higher accuracy and, therefore, allows for a coarser time discretization. We subdivide the time domain in an equidistant grid with $T \in \mathbb{N}$ intervals and set the step size $h = \frac{1}{T}$. This yields an explicit iteration scheme:

$$\begin{cases} x_n^{(0)} := x_n. \\ x_n^{(t+1)} := x_n^{(t)} + hv\Big(x_n^{(t)} + \frac{h}{2} v\big(x_n^{(t)}\big)\Big). \\ f_n := x_n^{(T)}. \end{cases} \tag{10.13}$$

We typically choose $T \in \{1, ..., 100\}$ in our experiments. In general, we have to make a trade off between runtime and accuracy when selecting an appropriate number of steps T. If we choose T too small, we lose some key properties of our framework like the volume preservation. This effect is illustrated in Figure 10.3 for the 2D shape of a bat transformed by a 90 degree rotation around the center. Note that the deformation field corresponding to this transformation is only approximately contained in our framework due to our choice of domain and boundary conditions, see discussion in the previous subsection. If we choose too few time steps T, the shape shifts outward and the area expands. On the other hand, this effect becomes insignificantly small if we choose $T \geq 10$.

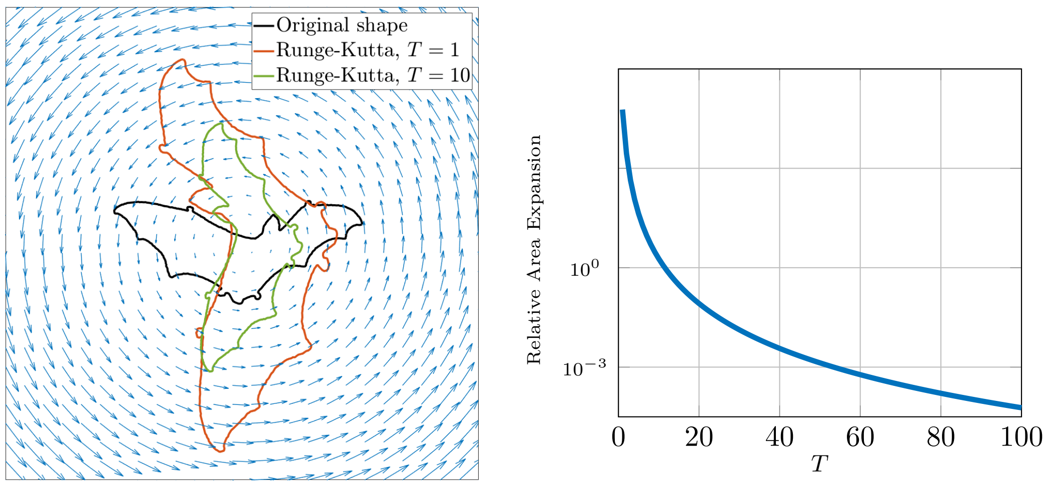


Figure 10.3: Area expansion with different step sizes using the Runge-Kutta integration. Left: Rotation around 90 degrees on a bat shape of the MPEG-7 dataset (**Ralph** n.d.) (black). If executed in one step ($T = 1$), the shape expands (red), whereas for ten steps $T = 10$ the area of the interior stays nearly the same (green). Right: Relative area expansion when performing the same deformation with an increasing number of steps T.

10.4.3 Optimization

In the previous sections we derived a coherent description of shape morphing using volume preserving deformation fields. We can now use this framework to construct an algorithm that matches two given point clouds $\mathcal{X}$ and $\mathcal{Y}$ by calculating a volume preserving deformation field between them. In order to do that, we simultaneously optimize for the deformation field coefficients a and the unknown correspondences.

Similar to **Myronenko and Song** (2010) and **Ma et al.** (2014), we approach shape registration in a probabilistic manner. We interpret the point cloud $\mathcal{X}$ as a Gaussian mixture model with the means located at the shifted points $f_n = x_n^{(T)}$ and the covariance $\sigma^2 I_D \in \mathbb{R}^{D\times D}$ for some $\sigma > 0$. This enables us to simultaneously determine the deformation field coefficients $a \in \mathbb{R}^K$ and the correspondences $W \in [0,1]^{N\times M}$ by applying an expectation maximization approach. Expectation maximization alternates between optimizing the deformation field coefficients and the correspondence while assuming the other to be fixed.

Expectation step. The expectation step calculates correspondences for a fixed deformation. We represent the correspondences between the morphed $f(\mathcal{X}) = \{f_1, \dots, f_N\}$ and the reference point cloud $\mathcal{Y} = \{y_1, \dots, y_M\}$ as soft correspondence matrices $W \in [0,1]^{N\times M}$ which arise naturally from the Gaussian mixture model assumption. High values of $W_{nm} \approx 1$ indicate a high correspondence probability for the point pair (x_n, y_m), while values close to zero indicate low probability. The expectation maximization framework yields an explicit update rule for

W given the deformation coefficients a:

$$W_{nm} := \frac{\exp\left(-\frac{1}{2\sigma^2}\mathrm{d}_{nm}^2\right)}{(2\pi\sigma^2)^{\frac{D}{2}} + \sum_{\tilde{n}=1}^{N}\exp\left(-\frac{1}{2\sigma^2}\mathrm{d}_{\tilde{n}m}^2\right)}. \tag{10.14}$$

Intuitively, W_{nm} describes the value of a Gaussian with center f_n and variance σ at point y_m. In the E-step soft correspondences $W \in [0,1]^{N\times M}$ are determined as a relaxed version of the latent variables Z:

$$\begin{aligned} W_{nm} &= \mathbb{E}_{Z|\mathcal{Y},a}(Z_{nm}) = p(Z_{nm}=1|y_m,a) \\ &= \frac{p(y_m|Z_{nm}=1,a)}{p(y_m|a)} = \frac{p(y_m|Z_{nm}=1,a)}{\sum_{\tilde{n}=1}^{N+1} p(y_m|Z_{\tilde{n}m}=1,a)}. \end{aligned} \tag{10.15}$$

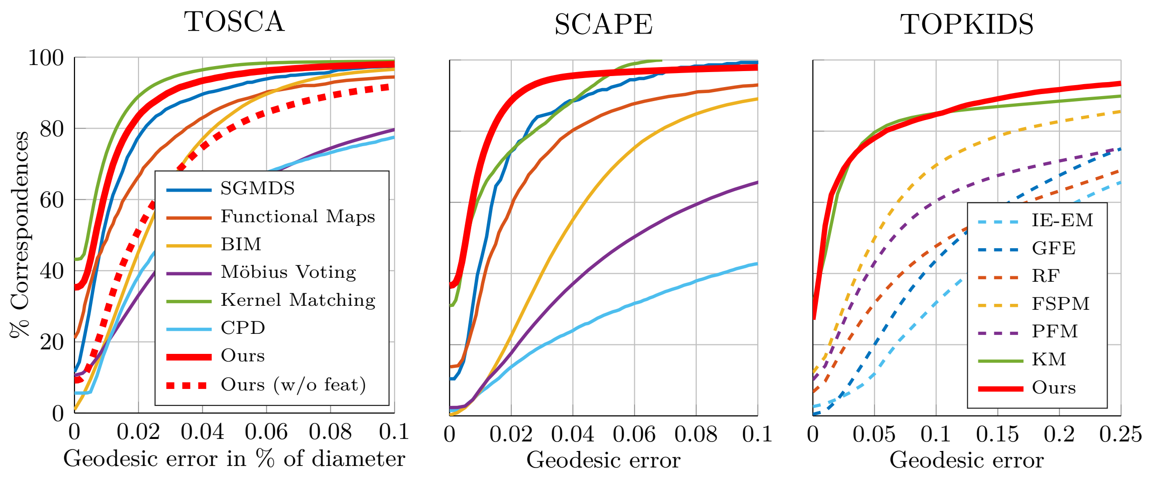


Figure 10.4: Quantitative evaluation using the Princeton benchmark protocol on the TOSCA data set (**A. M. Bronstein et al.** 2008) (left), the SCAPE data set (**Anguelov et al.** 2005) (middle), and the high-resolution TOPKIDS (**Lähner et al.** 2016a) (right). On TOSCA and SCAPE we compare against SGMDS (**Aflalo et al.** 2016), Functional Maps (**Ovsjanikov et al.** 2012), BIM (**Kim et al.** 2011), Möbius Voting (**Lipman and Funkhouser** 2009), CPD (**Myronenko and Song** 2010) and Kernel Matching (**Vestner* et al.** 2017). On TOPKIDS we compare against the competitors of the original paper IE-EM (**Sahillioglu and Yemez** 2012), GE (**Burghard et al.** 2017), RF (**Rodolà et al.** 2014a)), FSPM (**Litany et al.** 2017b), PFM (**Rodolà et al.** 2016) and Kernel Matching (KM) (**Vestner* et al.** 2017). Both the TOSCA as well as the TOPKIDS dataset contain cases which are critical for our method but our results are still on a par with state-of-the-art. See Section 10.5.1 for details. We additionally evaluate our method without using features on the TOSCA data set. The drop in performance shows that features are crucial to avoid unwanted optima.

The data likelihood terms $p(y_m|Z_{nm}=1,a)$ are defined in (C.3) and (C.4). This leads to the expression proposed in (10.14).

The E-step consists of minimizing the following energy with respect to a:

$$\begin{aligned}\mathbb{E}_{Z|\mathcal{Y},a}\big(-\log p(a|Z,\mathcal{Y})\big) \\ &= -\log p(a) - \sum_{m=1}^{M}\sum_{n=1}^{N+1} W_{nm}\log p(y_m|Z_{nm}=1,a) \\ &\propto \frac{1}{2}a^T L^{-1}a + \frac{1}{2\sigma^2}\sum_{m=1}^{M}\sum_{n=1}^{N} W_{nm}\|y_m - f_n\|_2^2. \end{aligned} \quad (10.16)$$

In this context the shifted points f_n depend on the unknown coefficients a. However, the descriptor distances $\mathrm{d}^{\mathrm{SHOT}}$ are independent of a, therefore, they vanish in the last step in (C.7) (see Appendix).

Similar to **Myronenko and Song** (2010), the normalization factor in the denominator comes from the mixture model assumption combined with an explicit modeling of outliers. In order to prevent our method from getting stuck in incorrect local optima, we include SHOT descriptors (**Tombari et al.** 2010) with standard parameters from the authors' implementation. We combine them with Euclidean distances to define a metric for pairs of points x_n and y_m:

$$\mathrm{d}_{nm}^2 := \big\|y_m - f_n\big\|_2^2 + \overline{\mathrm{d}}\big\|\mathrm{SHOT}(x_n) - \mathrm{SHOT}(y_m)\big\|_2^2. \quad (10.17)$$

We introduce the factor $\overline{\mathrm{d}} \geq 0$ to ensure that both metrics have a comparable scaling. In particular, we require both summands to have the same mean value for all point pairs $\mathcal{X}$ and $\mathcal{Y}$. Note that we use descriptor values $\mathrm{SHOT}(x_n)$ on the original shape $\mathcal{X}$ instead of the morphed shape $f(\mathcal{X})$ in order to not recompute them at every iteration.

Maximization step. The maximization step updates the deformation field for given soft correspondences W. Intuitively, we are looking for the deformation field coefficients a that best align points with high correspondence probability W_{nm}. For this purpose, we interpret the coefficients $a = (a_1, .., a_K)^\top$ as random variables with a normal distribution $a \sim \mathcal{N}(0, L)$, where $L := \mathrm{diag}(\lambda_1, ..., \lambda_K)$. If we compute the pushforward of this Gaussian according to Eq. (10.12), we get a prior distribution of deformation fields v. The weights λ_k are constructed from the eigenvalues λ_k^Δ as follows:

$$\lambda_k := \big(-\lambda_k^\Delta\big)^{-\frac{D}{2}} = \left(\pi^2 \sum_{d=1}^{D} j_d^2\right)^{-\frac{D}{2}}. \quad (10.18)$$

The mathematical background of this choice for the weights λ_k is provided by the Karhunen-Loève expansion (**Sullivan** 2015, Ch. 11) which is an extension of the principal component analysis (PCA) for function spaces. See Appendix C.3 for details on this choice. Intuitively, this kind of weighting promotes a damping of the high frequency components and smoothness of the deformation field v. The maximization step optimizes the coefficients a for their posterior

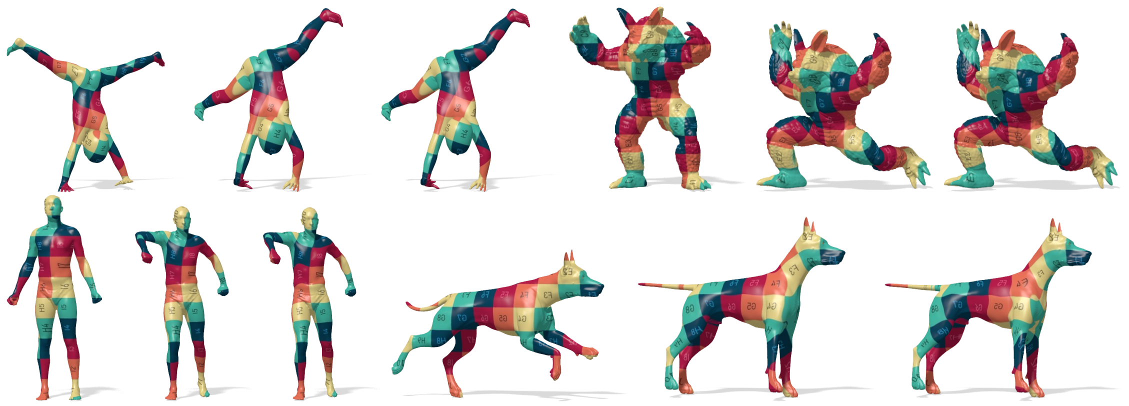


Figure 10.5: Examples of texture transfer done with our method. For each object the first image shows the source shape and texture, the second image the texture transferred with the ground-truth map and the third image the texture transferred with our correspondences. Our results are nearly identical to the ground-truth except for the dog which shows some artifacts on tail and chest.

distribution given the current correspondences which describes how well the deformation field of a explains W. This results in the following energy for a:

$$E(a) := \frac{\sigma^2}{2} a^\top L^{-1} a + \sum_{m=1}^{M} \sum_{n=1}^{N} W_{nm} \rho(\|y_m - f_n\|_2). \tag{10.19}$$

This energy E is the sum of the negative log prior including the weights λ_k (left term) and the negative log likelihood (right term) of a. The function $\rho : \mathbb{R} \to [0, \infty)$ is the Huber loss **Huber** (1964) which helps to account for outliers and makes the deformation field estimation more robust:

$$\rho(r) = \begin{cases} \frac{1}{2} r^2 & |r| \leq r_0. \\ r_0 |r| - \frac{1}{2} r_0^2 & \text{otherwise.} \end{cases} \tag{10.20}$$

In our experiments we choose the outer slope as $r_0 := 0.01$. Furthermore, we apply a Gauss-Newton type approach to minimize the energy in (10.19). This results in an iterative method similar to the Levenberg-Marquardt algorithm (**Levenberg** 1944). For this purpose, the residual term $\|y_m - f_n\|_2$ is linearized in each iteration. This requires a differentiation of the Runge-Kutta scheme (10.13) wrt. the weights a, see Appendix C.2 for an explicit formulation of the derivative $\frac{\mathrm{d}}{\mathrm{d}a} f_n$ and the Gauss-Newton update step for the energy in Equation (10.19). To summarize, our method alternates between computing the weights $W^{(i)}$ according to (10.14) and performing one Gauss-Newton update step for (10.19) to obtain $a^{(i)}$. To initialize the algorithm, we set the deformation field to zero $a^{(0)} := 0$.

10.5 Experiments

We evaluate our method for several applications to show that it is general and flexible. Although we handle shapes with up to $200k$ and more vertices, the computation of the deformation field is always done on a downsampled version of the inputs with 3000 vertices and then applied to the full resolution. We use Euclidean farthest point sampling. The downsampled shape should include points of all relevant large and fine scale structures in order for the deformation field to move these correctly, but we found 3000 sufficient for our applications. As a preprocessing step we shift both inputs such that the mean of their vertex positions is in the center of the domain and align them using principle component analysis (PCA). To avoid wrong alignments along the principle component axes, we choose the orientation that minimizes Eq. (10.17). When averaging over all experiments presented here, our algorithm takes about 370 seconds to compute the deformation and correspondences for one pair of shapes. Due to our a priori downsampling the runtime is only linearly dependent on the number of vertices, see Section 10.5.4 for a discussion of this property. All experiments were performed with MATLAB on a system with an Intel Core i7-3770 CPU clocked at 3.40GHz, 32 GB RAM and a GeForce GTX TITAN X graphics card running a recent Linux distribution. In all our experiments we only use the raw shape data, and, in particular, do not need any ground truth information or user input.

10.5.1 Matching

We verify our method using the TOSCA (**A. M. Bronstein et al.** 2008), SCAPE (**Anguelov et al.** 2005) and high-resolution TOPKIDS (**Lähner et al.** 2016a) data sets. All these shapes are synthetic and the exact intra-class correspondences are known. TOSCA contains 76 triangular meshes with 8 classes of humans and animals, SCAPE consists of 72 poses of the same person and TOPKIDS contains 26 poses of the same person in which topological merging, as it might appear in real scanning, is imitated.

We set the hyperparameters $\sigma^2 := 0.01$, $T := 20$ and choose $K = 3000$ basis functions for the deformation field. Because $W^{(i)}$ only contains 3000 correspondences, we perform a nearest-neighbor search with respect to the metric in Eq. (10.17) to obtain a dense mapping. The evaluation is done with the Princeton benchmark protocol as introduced in **Kim et al.** (2011). Given the ground-truth match $(x, y^*) \in \mathcal{X} \times \mathcal{Y}$, the error of the calculated match (x, y) is given by the geodesic distance between y and y^* normalized by the diameter of $\mathcal{Y}$:

$$\epsilon(x) = \frac{\mathrm{d}_{\mathcal{Y}}^{\mathrm{Geo}}(y, y^*)}{\sqrt{\mathrm{area}(\mathcal{Y})}}$$

We plot cumulative curves showing the percentages of matches that are below an increasing threshold. As zero is the value for ground-truth matches, the ideal curve would be constant at 100. See Figure 10.4 for our results and Figure 10.5 for example matching results showing texture transfer. On SCAPE we are able to reach state-of-the-art results whereas

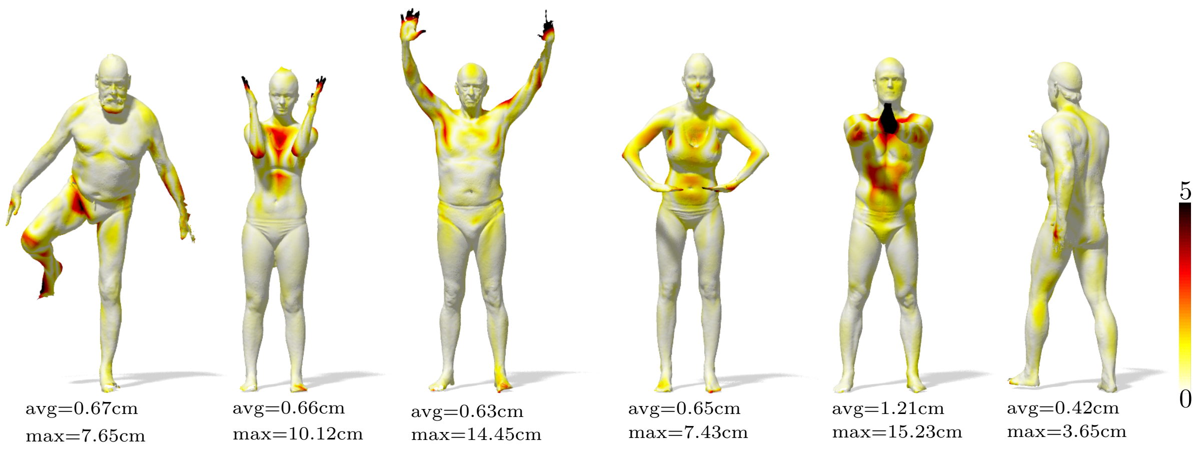


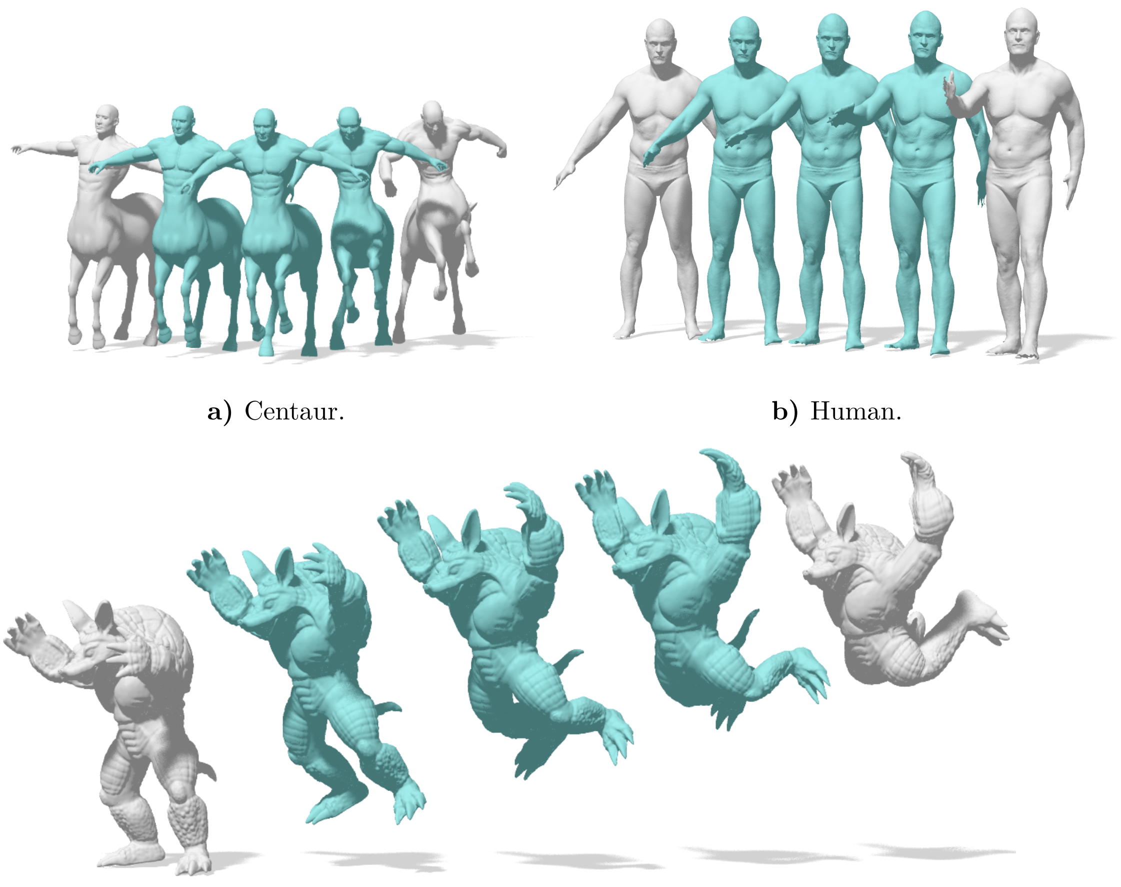
Figure 10.6: Example registrations from the FAUST scan data set. The surface color corresponds to the Euclidean surface distance between scan and registration. The scale of the scans is in real cm values and the same on all plots. We report the average and maximum error under each image. Many errors occur due to the SHOT descriptors being corrupted at holes and in noisy areas (e.g. the hands). Furthermore, in some case the assumption of exact volume preservation is too restrictive for real scans with noise and topological changes (see especially second to the right).

on TOSCA the intrinsic Kernel Matching method is slightly better. Our extrinsic approach makes self-touching poses more challenging and these cases occur fairly often in TOSCA. Although TOPKIDS is synthetic, the self-touching poses are actually merged in the geometry which makes it more challenging. On this dataset we are slightly better than Kernel Matching (see Figure 10.4).

To show the influence of features on the results, we do an evaluation of our method without using features at any point during the optimization. Instead the distance of Equation (10.17) is replaced with the pure Euclidean distance between the coordinates. The result can be seen in Figure 10.4. The performance without features decreases substantially because the Euclidean distance is a weak indicator when large deformations take place. Therefore, our method gets stuck in local optima more often without feature guidance.

10.5.2 Registration

We apply our framework to the FAUST Scan dataset (**Bogo et al.** 2014) which contains data from scans of real humans in different poses. Each of these shapes has approximately $200k$ vertices, they are sampled inconsistently and some of them are severely affected by scanning noise, holes and topological changes. We match the null shape of every person to its other poses. In Figure 10.6 we display the surface distance of the morphed shapes to the goal shape for some examples. We reach very tight alignments except in very challenging cases, like topological changes. Furthermore, the scanned volume varies slightly even between different poses of the same humans which induces small errors in our method.



a) Centaur. **b)** Human. **c)** Armadillo.

Figure 10.7: Three examples of shapes that are morphed into one another according to the initial value problem of Eq. Eq. (10.1). The centaur (a) and the human (b) are from the TOSCA **A. M. Bronstein et al.** (2008) and FAUST **Bogo et al.** (2014) dataset respectively. The armadillo (c) is from the AIM@SHAPE shape repository **AIM@SHAPE repository** (n.d.). (b) is a scan of a real person and very high resolution ($214k$ vertices). The source and target shape are shown in white and the interpolations at times $t = 0.25, 0.5, 0.75$ in blue. The translation is not part of our deformation and was only introduced for clarity in the figures/divfree.

10.5.3 Effect of the basis size

In our evaluations we consistently use $K = 3000$ deformation field basis functions. To justify this choice empirically, we compute the mean geodesic errors of each TOSCA pair for several basis sizes $K \in \{1, \dots, 3000\}$, see Figure 10.8. We observe that while the accuracy increases significantly for small $K \leq 1000$, after some point it starts plateauing. In our evaluations, we choose $K = 3000$ because we aim for a high accuracy. However, for some applications where runtime is more important than accuracy a smaller basis size $K < 3000$ might be sufficient.

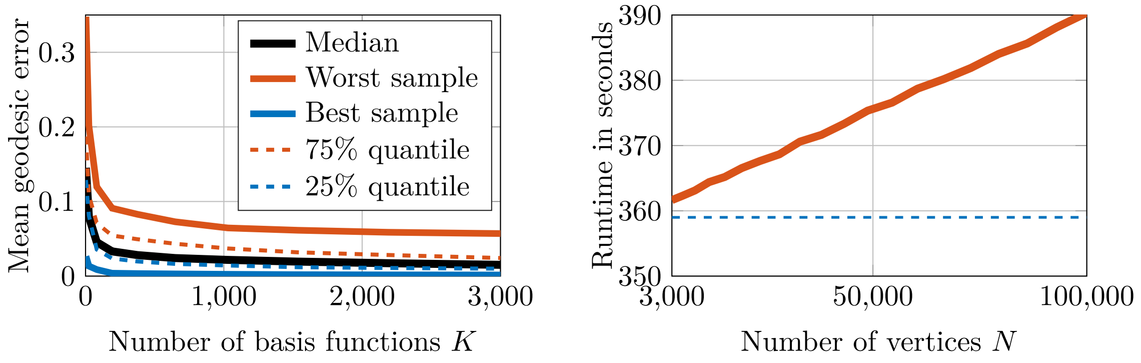


Figure 10.8: (Left) Dependency of the mean geodesic errors on TOSCA on different basis sizes $K \in \{1, \dots, 3000\}$. In particular, we show the elements at the 0%, 25%, 50%, 75%, and 100% quantile. (Right) Runtime of our method for the full resolution shape deformation for different number of vertices $N \in \{3000, \dots, 100000\}$. The full pipeline has two steps: (1) a fixed size optimization over 3000 vertices which takes around 360 seconds on average (blue dashed line), (2) applying the deformation field to the full resolution shape and extracting the correspondence for the full shape. The plot shows that our method scales linearly in the number of vertices and is therefore still feasible for very detailed shapes with over 100k vertices.

10.5.4 Runtime for high resolution

One major advantage of our method is that it is scalable to high resolution input shapes like those from FAUST because we optimize for the deformation field on downsampled shapes (3000 vertices). One point that we want to stress in this context is that this is not the same as computing matchings only on low resolution shapes. For many matching methods this scaling to the full resolution is challenging, most methods need to come up with a custom coarse-to-fine strategy. In general, it is not straightforward to extend a shape matching or deformation from a downsampled shape to the rest of the vertices. However, for our method this upscaling is trivial because the deformation field basis functions Eq. (10.11) are defined densely on the whole embedding space, therefore they can be evaluated anywhere in Ω. This upsampling scales linearly in N because the Runge-Kutta method (10.13) is computed independently for all vertices x_n. See Figure 10.8 for an empirical verification of this property. Here, the runtime for the shape deformation is computed for various downsampled versions of one high resolution shape. To sum it up, the runtime for computing shape morphings is relatively low and increases only linearly in the number of vertices N which makes our method scalable for high resolution input shapes.

10.5.5 Shape Interpolation

Interpolation. Our method morphs the input shape $\mathcal{X}$ by solving the ODE Eq. (10.1) up to time $t_{\text{eval}} = 1$. If we now instead evaluate it at an intermediate time $t \in (0, 1)$, we get interpolated shapes as a byproduct of our matching pipeline. Just like the morphed shapes

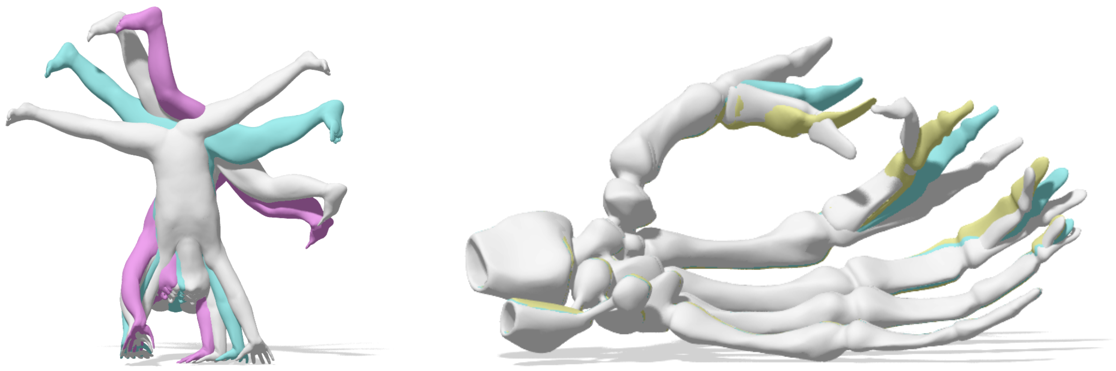


Figure 10.9: (Left) Example of an extrapolated shape from the KIDS dataset (**Rodolà et al. 2014a**). The extrapolation was achieved by using the temporally fixed deformation field v for simulating the initial value problem from Eq. (10.1) up to the time $t = 1.3$. Source and target shape are white, one interpolated shape is shown in blue and the extrapolation is pink. (Right) Example of a failure case. The source and target shapes are white, the interpolated shape at $t = 0.5$ is blue and the resulting shape at $t = 1$ is yellow. The yellow shape is supposed to be as close to the target as possible but the fingers are pushed away from each other instead. Here, the thumb and index finger are supposed to move in spatially very close areas. We are only calculating one deformation field for all time steps, therefore it is not clear for our method what motion to assign for contradicting motions. In order to resolve this problem, we need to either assign different motions to different subparts of the objects or make the deformation fields time dependent.

$f(\mathcal{X})$ those intermediate shape morphings are smooth and volume preserving which makes them look natural. Three examples with interpolated shapes are displayed in Figure 10.7.

Extrapolation. Similarly to the idea of interpolating shapes as a byproduct of our method we can also use the computed deformation field v to solve the initial value problem Eq. (10.1) up to times $t > 1$. This results in extrapolated shapes, see Figure 10.9. In contrast to shape interpolation, extrapolation is a severely underdetermined task and it is hard to evaluate quantitatively. Nevertheless, we observed that for moderate time spans $t \in [1, 1.5]$ our method produces reasonable results. In general, the morphing speed slows down at some point, especially when the shape is moving in previously unoccupied space. Intuitively, for the optimization there is no incentive to impose any particular movement on these parts of the domain Ω, if it is not relevant for the surface alignment. Still, our extrapolated shapes are visually appealing and not severely affected by distortions.

10.6 Conclusion

This chapter introduced a extrinsic shape correspondence method optimizing for both a dense surface correspondence as well as a physically meaningful deformation field aligning the inputs. Our morphing model shifts the source shape $\mathcal{X}$ along a smooth, volume preserving deformation field using a second order Runge-Kutta integration scheme in order to align it with the reference shape $\mathcal{Y}$. Additionally to aligning the inputs, this model can also be used to efficiently calculate plausible interpolated, and even extrapolated, shapes.

Our method addresses the coupled problem of finding an unknown deformation and unknown correspondence with an expectation maximization approach. Furthermore, we represent our morphing model with a low rank deformation field basis which reduces the degrees of freedom and show that this makes the optimization problem well constrained. As a result, this allows subsampling the inputs to make the problem computationally feasible, even for high resolution meshes, with only a linear increase in runtime. Quantitative evaluations for shape correspondence partly prove state-of-the-art performance of our method. Moreover, we show convincing examples of shape interpolation and extrapolation that arise naturally from our pipeline.

10.6.1 Limitations

This method works really well when some assumptions hold but is by construction not suited for some cases. The first is that the volume-preservation cannot be relaxed and therefore artifacts are bound to appear when the inputs do not have the same volume. Second, the volume preservation property applies to every subregion of the domain Ω, including the intermediate space between parts of the shape. Therefore, separating two touching parts (for example two hands) is in theory possible, but it requires many high frequency deformation basis elements which would make the optimization costly. Additionally, the assumption of Eq. (10.1) being autonomous can be problematic if different parts of the shape move through the same region of the embedding space in a contradictory manner. One example for this is a hand closing to a fist. At first the index and middle finger occupy parts of the embedding space before the thumb moves in the same area but in a different direction. See Figure 10.9.

10.6.2 Continuity

Because we make use of the global, extrinsic volume property, we can constrain our optimization with a small, subsampled version of the input shapes. Smoothness and volume-preservation guarantee that applying the deformation to the full resolution will yield meaningful results. However, smoothness in the deformation does not imply continuity in the correspondence as we defined it in Chapter 6. First, the soft correspondence W needs to be converted into a pointwise correspondence in order for the theory to apply. This is normally done by taking the column-wise maximum or some weighted nearest neighbor computation but this thresholding may lead to broken continuity in areas where the alignment is not very tight. Non-tight alignment happens regularly because high frequency details need many degrees of freedom to be properly aligned. If the deformation field would guarantee a perfect alignment, it could be used to directly extract the diffeomorphism. However, there is no guarantee on continuity in a pointwise solution of an approximate alignment. But in our experiments we saw that our method produces only very few discontinuities. Furthermore, the solution is smooth by construction in terms of the deformation field which means that without projecting onto a pointwise correspondence the solution is continuous and can serve well in applications that do not require the mesh information of $\mathcal{Y}$. Second, the divergence-free vector field does not have an equivalent on the product manifold or the product embedding space. This is because volume-preservation assumes that $\mathcal{X}$ and $\mathcal{Y}$ are embedded in the same embedding space. But the product manifold is embedded in the product of the original embedding spaces, therefore, there is no straightforward equivalent of the divergence-free vector field in the product embedding space. As a result, this approach could not be used for different dimensional shapes, but, if volume-preservation holds, adds meaningful constraint that lead to an efficient optimization.

PART **IV**

Closure

"Poirot," I said. "I have been thinking."
"An admirable exercise my friend.
Continue it."

– Agatha Christie, Peril at End House

CHAPTER 11

Conclusion

11.1 Summary

We saw several algorithms solving for continuous, non-rigid correspondences between 2D and 3D shapes and looked at their interpretation on the product manifold.

Transdimensional Correspondences. In the case of a 2D contour and a 3D shape, we can guarantee a continuous solution due to the one-dimensional structure of the solution. This is possible by explicitly modeling the product graph and, then, applying a combination of Dijkstra's algorithm and branch-and-bound to find a shortest path on it. Additionally, the oriented and multi-level structure of the product graph allows us to reduce the computational complexity by processing the graph level by level. This led to a polynomial time algorithm for finding the global optimum as well as an approximation scheme with considerable speed-up. The second hurdle in this setting is finding descriptors that are comparable between different dimensional objects but robust against pose changes at the same time. We proposed a way to use spectral features, which are commonly used on deformable 3D shapes, on 2D contours by filling their interior. This solid can be seen as an approximately isometric part of the 3D shape. Our experiments show that this combination is well-suited for correspondence and retrieval between 2D and 3D shapes with isometric deformations, and, to the best of our knowledge, it is the first algorithm to combine different dimensionality with deformable shapes.

Minimal Surface Submanifolds. The second setting includes two shapes of dimension three; a case in which the solution is two-dimensional and cannot be found through a shortest path problem anymore. Instead the solution is a two-dimensional submanifold of the four-dimensional product manifold. Our first approach optimized a QAP using heat kernels and we showed that its maximum corresponds to a minimal surface in the product. QAPs in general are NP-hard and our proposed solution is only approximate. But we used the positive-definiteness of heat kernels to show that the relaxation preserves the optimum of the original problem. We solve the QAP as a sequence of LAPs, but these also become inefficient on high resolutions. To counteract this, we proposed a multi-scale approach that propagates global information into each subproblem. Using this framework we showed state-of-the-art results on several isometric correspondence datasets. Additionally, our method can handle non-isometric cases, including topology changes and partiality, without expensive adjustments.

Many other methods also have an interpretation on the product manifold, although this is rarely explored in literature. We explored this relation for the functional map framework and showed that the coefficients used in functional maps are not optimal in terms of representation. This result arises from the special structure of the product manifold and the separability of all elements into their counterparts on the original manifolds. We applied this to the LB eigenfunctions and -values used in functional maps to show suboptimality, but the same holds for other properties and is independent of dimensions. Furthermore, we used the same observation to construct a new non-separable basis that is well suited for the correspondence problem on the product manifold and showed its applicability on a framework for refinement of correspondences.

Correspondence through Deformation. In the last chapter we looked at continuity from a different viewpoint. Instead of only considering the structure of the map $\mathcal{M} \to \mathcal{N}$, our approach imposed properties in the sequence of intermediate shapes depicting a deformation between $\mathcal{M}$ and $\mathcal{N}$. This was done by jointly optimizing for the point-wise correspondence and the volume-preserving deformation field in an expectation-maximization scheme. The underlying idea is that the knowledge about one improves the result of the second. To that end, we constructed a basis for volume-preserving deformation fields that is both frequency-ordered and has a closed-form solution. The frequency ordering performs a sort of low-pass filtering on the solution that is both efficient in terms of optimization and incentivizes a smooth deformation in the end. The closed-form solution enables optimization on a subset of vertices but allows to apply the result without discretization errors to any mesh resolution. Finally, we saw that by restricting the solution space to volume-preserving deformations scattered outliers are nearly impossible, and the low-frequency deformations get seldom stuck in local optima. Apart from state-of-the-art correspondence results, we get a realistic interpolation sequence between the input shapes as a by-product without any computation overhead.

11.2 Discussion

The methods above show great results on certain type of data, namely isometric and non-isometric cases that are enable a nearly bijective solution. Non-bijectivity and two-sided partiality is still a problem for many state-of-the-art methods. Both might require that a certain amount of vertices stays unmatched. If the exact amount is not known beforehand, it is hard to find a balance between not matching at all and matching a not-perfect pair. If not matching a vertex is not penalized at all, the optimal solution will nearly always be the empty one, because it has zero energy. Forcing every vertex to be matched, results in parts that have no ground-truth correspondence to be matched anyway, and similarity based approaches might suffer from weird optima because of their large influence on the energy function. However, choosing the right penalization depends heavily on the amount of noise in the data, prior assumptions about overlap and the method itself. This makes finding a general approach really hard, and most strategies unstable. As a result, there exist no methods that can handle two-sided partiality for non-rigid applications without having prior knowledge about the classes of shapes. This is a challenge that future methods still have to tackle.

The majority of methods in this thesis have a clean interpretation on the product manifold, even if they were not modeled like this explicitly. This is not surprising, because the product manifold itself does not have its own geometry and properties but just concatenates properties of its factors. In fact, any notion on the base manifolds has a counterpart on the product. For most local and intrinsic properties these counterparts are meaningful, and the product space can be used to obtain a clearer view of the relationship between two shapes. This can be informative even though no new information is added, because the construction is complete. Whereas comparing properties of two manifolds might lead to omitting implicit relations, the geometry of the product manifold will make these shortcomings clear. On the other hand, certain extrinsic relations are not represented on the product manifold. In Chapter 10 we considered volume-preserving deformation fields. These assume that the source and target shape are embedded in the same embedding space, and this makes volume-preservation meaningful. However, volume-preservation does not directly translate to the product manifold, because the embedding space, that was assumed to be the same, is also multiplied into its product. It means that the volume of one slice of the product manifold is still equal to the volume of the original shape, but the 3D vector field deforming one shape into the other does not have a straight-forward 6D equivalent. This is beneficial, because it means the product manifold does not assume anything about where the shapes were embedded. As an example, any product manifold method still works two two-dimensional manifolds were embedded in three and four dimensions, respectively. It is not necessarily a good assumption for every application though.

Directly working on the product manifold can be expensive and constructing it explicitly will probably exceed memory availability for most input shapes. But we saw in Chapter 7 and Chapter 9 that these issues can be overcome by only constructing locally or using closed-form solutions that exist for basically all properties. Even when operating directly on the product is not feasible, doing a theoretical analysis there is always free (in terms of computation time) and can lead to more in-depth understanding of the energy function, the optimization problem or the properties of the solution, as we saw in Chapter 8. Especially diffeomorphism, which are hard to characterize in discretized settings, have an equivalent minimal surface formulation in the product space which can be discretized and validated. Therefore, I advocate to revisit the interpretation of known algorithms on the product space to understand their theoretical properties, and keep these relationships in mind when designing new ones.

PART **V**

Appendix

Little details have special talents in creating big problems.

– Mehmet Murat Ildan

APPENDIX **A**

Appendix for Chapter 8

In Section 8.3 we propose to use the DC algorithm to optimize

$$\arg\min_{P\in^{n\times n}} B(P) - E(P). \tag{A.1}$$

where B is the convex indicator function of the set of bistochastic matrices and E is strictly convex and differentiable. We will now prove that the two steps

- Select $Q^k \in \partial E(P^k)$
- Select $P^{k+1} \in \partial B^*(Q^k)$.

of the DC algorithm are equivalent to

$$P^{k+1} = \arg\max_{P\in\mathcal{B}_n} \left\langle P, \nabla E(P^k)\right\rangle, \tag{A.2}$$

that each iterate P^k can be chosen to be a permutation matrix, and that $E(P^k)$ is a strictly increasing.

A.1 Details about the DC Algorithm

We assume that the reader is familiar with the concepts of convex conjugates and sub-gradients and just recall the following Lemma

Lemma 1. *Let X be a Banach space and $f : X \to (-\infty, \infty]$ with $\partial f \neq \emptyset$. Then $f^{**}(x) = f(x)$ and*

$$x^* \in \partial f(x) \Leftrightarrow x \in \partial f^*(x^*) \tag{A.3}$$

Moreover for convex functions f, $0 \in \partial f(x)$ is equivalent to

$$x^* = \arg\min_x f(x) \tag{A.4}$$

Let now E be convex differentiable and B the (convex) indicator function of a convex set C. We will derive equivalent expressions for the two steps in the DC algorithm for solving (A.1). Since E is differentiable, its subdifferential at any point has one element, namely the gradient at that point:

$$Q^k \in \partial E(P^k) \Leftrightarrow Q^k = \nabla E(P^k) \tag{A.5}$$

The second step $P^{k+1} \in \partial B^*(Q^k)$ can be rewritten using Lemma 1:

$$\begin{aligned} P^{k+1} \in \partial B^*(Q^k) &\Leftrightarrow Q^k \in \partial B(P^{k+1}) \\ &\Leftrightarrow 0 \in -Q^k + \partial B(P^{k+1}) \\ &\Leftrightarrow P^{k+1} = \arg\min_P \; -\langle Q^k, P\rangle + B(P) \\ &\Leftrightarrow P^{k+1} = \arg\max_{P\in C} \langle Q^k, P\rangle \end{aligned} \tag{A.6}$$

Thus the DC algorithm in this special case reads

$$P^{k+1} = \arg\max_{P\in C} \langle P, \nabla E(P^k)\rangle \,. \tag{A.7}$$

In our case the convex set C is the polyhedron of bi-stochastic matrices. Since linear functions defined on a polyhedron attain their extrema at the vertices of the polyhedron, we can choose the maximizer to be a permutation matrix.

Due to the strict convexity of E we further see:

$$\begin{aligned} E(P^{k+1}) &> E(P^k) + \langle P^{k+1} - P^k, \nabla E(P^k)\rangle \\ &\geq E(P^k) + \langle P^k - P^k, \nabla E(P^k)\rangle \\ &= E(P^k) \end{aligned} \tag{A.8}$$

where the strong inequality holds until convergence and the weak inequality follows directly from (A.7).

APPENDIX B

Appendix for Chapter 9

We provide proofs for the main propositions of the paper.

B.1 Proof of Theorem 4

Following standard FEM, we discretize the Poisson equation $\Delta_{\mathcal{M}\times\mathcal{N}} f = g$ via the weak formulation

$$\langle \Delta_{\mathcal{M}\times\mathcal{N}} f, H_j \rangle = \langle g, H_j \rangle \,, \tag{B.1}$$

where functions are expressed in the hat basis $\{H_j : \mathcal{M}\times\mathcal{N} \to \mathbb{R}\}$, and are thus approximated piecewise-linearly via the expansion $f(x) \approx \sum_{i=1}^{n} f(v_i) h_i(x)$. The left-hand side of (B.1) can be written as

$$\langle \Delta f, H_j \rangle = -\langle \nabla f, \nabla H_j \rangle = -\sum_i f(v_i) \underbrace{\langle \nabla H_i, \nabla H_j \rangle}_{s_{ij}} \,, \tag{B.2}$$

where s_{ij} are elements of the *stiffness matrix* S. The right-hand side of (B.1) can be written as

$$\langle g, H_j \rangle = \langle \sum_i g(v_i) H_i(x), H_j \rangle = \sum_i g(v_i) \underbrace{\langle H_i, H_j \rangle}_{a_{ij}} \,, \tag{B.3}$$

where a_{ij} are elements of the *mass matrix* A.

The Cartesian product of the two graphs discretizing $\mathcal{M}$ and $\mathcal{N}$ has grid topology, as illustrated in Figure B.1, and the resulting bilinear hat basis functions are expressed via the outer product $H_e = h_j \otimes h_q$. We can then compute the mass values (refer to the Figure for the color notation):

$$\begin{aligned} a_{ee} &= \langle H_e, H_e \rangle = \langle h_j \otimes h_q, h_j \otimes h_q \rangle \\ &= \int_{Q_{abde} \cup Q_{bcef} \cup Q_{degh} \cup Q_{efhi}} h_j(x) h_q(y) h_j(x) h_q(y) dx dy \\ &= \int_{E_{ijk}} h_j(x) h_j(x) dx \int_{E_{pqr}} h_q(y) h_q(y) dy \\ &= a_{jj} a_{qq} \end{aligned} \tag{B.4}$$

$$\begin{aligned} a_{ae} &= \langle H_a, H_e \rangle = \langle h_i \otimes h_r, h_j \otimes h_q \rangle \\ &= \int_{Q_{abde}} h_i(x) h_r(y) h_j(x) h_q(y) dx dy \\ &= \int_{E_{ij}} h_i(x) h_j(x) dx \int_{E_{qr}} h_r(y) h_q(y) dy \\ &= a_{ij} a_{qr} \end{aligned} \tag{B.5}$$

$$\begin{aligned} a_{de} &= \langle H_d, H_e \rangle = \langle h_i \otimes h_q, h_j \otimes h_q \rangle \\ &= \int_{Q_{abde} \cup Q_{degh}} h_i(x) h_q(y) h_j(x) h_q(y) dx dy \\ &= \int_{E_{ij}} h_i(x) h_j(x) dx \int_{E_{pqr}} h_q(y) h_q(y) dy \\ &= a_{ij} a_{qq} \end{aligned} \tag{B.6}$$

Similarly, the stiffness integrals read:

$$\begin{aligned} s_{ee} &= \langle \nabla H_e, \nabla H_e \rangle = \langle \nabla h_j \otimes h_q, \nabla h_j \otimes h_q \rangle \\ &= \langle \nabla h_j h_q, \nabla h_j h_q \rangle + 2 \langle h_j \nabla h_q, \nabla h_j h_q \rangle + \langle h_j \nabla h_q, h_j \nabla h_q \rangle \\ &= \int_{Q_{abde} \cup Q_{bcef} \cup Q_{degh} \cup Q_{efhi}} \langle \nabla h_j(x) h_q(y), \nabla h_j(x) h_q(y) \rangle dx dy + \cdots \\ &= \int_{Q_{abde} \cup Q_{bcef} \cup Q_{degh} \cup Q_{efhi}} h_q(y) h_q(y) \langle \nabla h_j(x), \nabla h_j(x) \rangle dx dy + \cdots \\ &= \int_{E_{ijk}} \langle \nabla h_j(x), \nabla h_j(x) \rangle dx \int_{E_{pqr}} h_q(y) h_q(y) dy + \cdots + \cdots \\ &= s_{jj} a_{qq} + a_{jj} s_{qq} \end{aligned} \tag{B.7}$$

$$\begin{aligned} s_{ae} &= \langle \nabla H_a, \nabla H_e \rangle = \langle \nabla h_i \otimes h_r, \nabla h_j \otimes h_q \rangle \\ &= \langle \nabla h_i h_r, \nabla h_j h_q \rangle + \langle h_i \nabla h_r, h_j \nabla h_q \rangle \\ &= s_{ij} a_{qr} + a_{ij} s_{qr} \end{aligned} \tag{B.8}$$

$$\begin{aligned} s_{de} &= \langle \nabla H_d, \nabla H_e \rangle = \langle \nabla h_i \otimes h_q, \nabla h_j \otimes h_q \rangle \\ &= \langle \nabla h_i h_q, \nabla h_j h_q \rangle + \langle h_i \nabla h_q, h_j \nabla h_q \rangle \\ &= s_{ij} a_{qq} + a_{ij} s_{qq} \end{aligned} \tag{B.9}$$

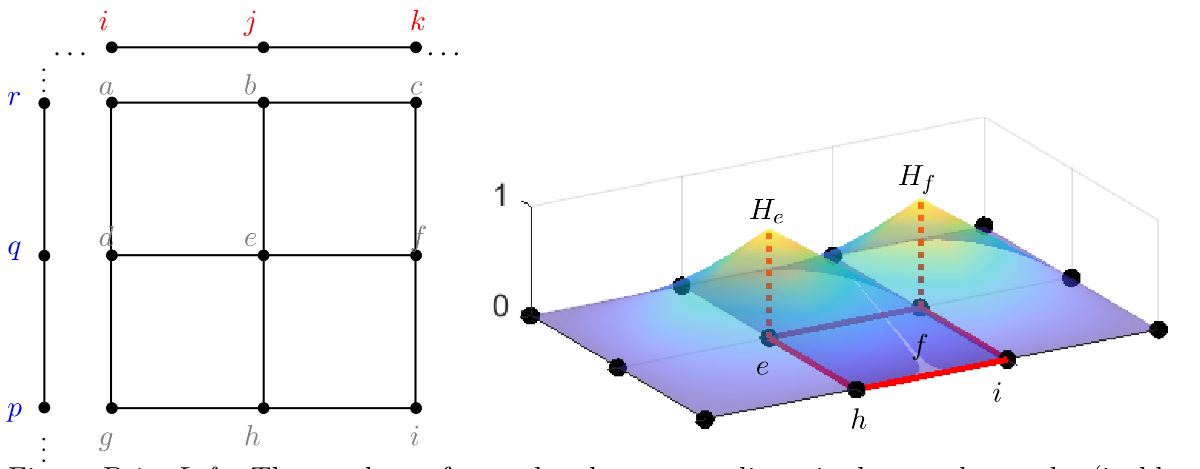


Figure B.1: *Left*: The product of two closed contours discretized as cycle graphs (in blue and red) is a quad mesh with toric topology (in grey). Uniform edge lengths are used for illustration purposes. *Right*: Two overlapping bilinear hats H_e and H_f. On the quad element Q_{efhi} (marked in red) there is non-zero overlap, hence it contributes to the computation of mass and stiffness values.

where we applied the outer product rule for the gradient operator, and used the fact that $\langle \nabla f, \nabla g \rangle = 0$ for any pair of functions on the two cycle graphs. Note the integrals a_{ae} and s_{ae} are non-zero even if nodes a and e are not connected in the product graph.

In matrix notation, formulas (B.4)-(B.9) can be succinctly written as:

$$\begin{aligned} A &= A \otimes A \\ S &= S \otimes A + A \otimes S\,, \end{aligned}$$

completing the proof. □

B.2 Proof of Corollary 3

The proof is straightforward and follows from substituting the expressions (9.3), (9.4) into the general formula $\Delta = A^{-1}S$:

$$\begin{aligned} \Delta_{\mathcal{M}\times\mathcal{N}} &= A^{-1}_{\mathcal{M}\times\mathcal{N}} W_{\mathcal{M}\times\mathcal{N}} \\ &= (A_{\mathcal{M}} \otimes S_{\mathcal{N}})^{-1}(S_{\mathcal{M}} \otimes A_{\mathcal{N}} + A_{\mathcal{M}} \otimes S_{\mathcal{N}}) \\ &= (A^{-1}_{\mathcal{M}} \otimes A^{-1}_{\mathcal{N}})(S_{\mathcal{M}} \otimes A_{\mathcal{N}}) + (A^{-1}_{\mathcal{M}} \otimes A^{-1}_{\mathcal{N}})(A_{\mathcal{M}} \otimes S_{\mathcal{N}}) \\ &= (A^{-1}_{\mathcal{M}} S_{\mathcal{M}}) \otimes (A^{-1}_{\mathcal{N}} A_{\mathcal{N}}) + (A^{-1}_{\mathcal{M}} {}_{\mathcal{M}}) \otimes (A^{-1}_{\mathcal{N}} S_{\mathcal{N}}) \\ &= \Delta_{\mathcal{M}} \otimes I_{\mathcal{N}} + I_{\mathcal{M}} \otimes \Delta_{\mathcal{N}}\,. \qquad \square \end{aligned}$$

B.3 Proof of Corollary 4

Since triangular (3-3) duoprisms are, by definition, the Cartesian product of two triangles, we can define a multilinear basis function on the product complex as the outer product of two standard hats defined on triangle meshes. We are now in the same setting as the lower dimensional case, and in particular Equations (B.4)-(B.9) remain valid. □

APPENDIX C

Appendix for Chapter 10

In section 10.4.3 of the paper, we outlined our expectation maximization framework. Here, we want to provide a more detailed description of the method. At the same time, we try to keep it as brief as possible, because most parts of this summary are standard techniques when dealing with Gaussian mixture models.

C.1 Gaussian Mixture Model

As outlined before, we interpret the shifted points $f_n = x_n^{(T)}$ as the centers of Gaussian distributions with the covariance matrix $\sigma^2 I_D \in \mathbb{R}^{D \times D}$ which in the end should describe $\mathcal{Y}$ well. Furthermore, each point y_m is assumed to correspond to some point x_n. This relationship is encoded by the correspondence matrix $Z \in \{0,1\}^{(N+1)\times M}$, where $\sum_{n=1}^{N+1} Z_{nm} = 1$. If $Z_{(N+1)m} = 1$ for some m, the point y_m does not correspond to any point x_n and it is assumed to be uniformly sampled from Ω instead. This way we acknowledge the presence of outliers and counteract them by explicitly modeling them.

According to Bayes' theorem the posterior probability distribution of the desired parameters a_k in (10.12) given the latent correspondences Z_{nm} and the observed points $\mathcal{Y}$ is defined as follows:

$$p(a|Z,\mathcal{Y}) \propto p(a)p(\mathcal{Y}|Z,a) = p(a) \prod_{m=1}^{M} p(y_m|Z,a) \tag{C.1}$$

$$= p(a) \prod_{m=1}^{M} \prod_{n=1}^{N+1} p(y_m|Z_{nm}=1,a)^{Z_{nm}}. \tag{C.2}$$

In Section 10.4.3 it was mentioned that the prior of the parameters a_k is a Gaussian distribution $a_k \sim \mathcal{N}(0,\lambda_k)$. As a shorthand notation we define the diagonal matrix $L := \operatorname{diag}(\lambda_1, ..., \lambda_K)$ and set the prior $a \sim \mathcal{N}(0,L)$ for the coefficient vector a. In order to explicitly evaluate the posterior density of a in (C.1), we have to investigate the data likelihood $p(y_m|Z_{nm}=1,a)$ in detail. For $n < N+1$ it is a Gaussian distribution in the product space of the embedding space Ω and the space of descriptor values:

$$p(y_m|Z_{nm}=1,a)=\frac{1}{(2\pi\sigma^2)^{\frac{D}{2}}}\exp\biggl(-\frac{1}{2\sigma^2}\mathrm{d}_{nm}^2\biggr). \tag{C.3}$$

For the case $n=N+1$ it is simply a uniform distribution, because y_m is considered to be an outlier:

$$p(y_m|Z_{(N+1)m}=1,a)=1. \tag{C.4}$$

Note that the GMM is not only defined on the D dimensional embedding space but rather on the product space of Ω and the (possibly high dimensional) feature space. However, this only affects how the correspondences are assigned and can be considered a theoretical nuance.

C.2 Expectation Maximization

We want to determine the coefficients a by applying an expectation maximization approach similar to **Myronenko and Song** (2010). In the E-step, soft correspondences $W\in[0,1]^{(N+1)\times M}$ are determined as a relaxed version of the latent variables Z:

$$W_{nm}=\mathbb{E}_{Z|\mathcal{Y},a}(Z_{nm})=p(Z_{nm}=1|y_m,a) \tag{C.5}$$

$$=\frac{p(y_m|Z_{nm}=1,a)}{p(y_m|a)}=\frac{p(y_m|Z_{nm}=1,a)}{\sum_{\tilde{n}=1}^{N+1}p(y_m|Z_{\tilde{n}m}=1,a)}. \tag{C.6}$$

The data likelihood terms $p(y_m|Z_{nm}=1,a)$ are defined in (C.3) and (C.4). This leads to the expression proposed in (10.14).

The M-step now consists of minimizing the following energy with respect to a:

$$\mathbb{E}_{Z|\mathcal{Y},a}\bigl(-\log p(a|Z,\mathcal{Y})\bigr)=-\log p(a)-\sum_{m=1}^{M}\sum_{n=1}^{N+1}W_{nm}\log p(y_m|Z_{nm}=1,a) \tag{C.7}$$

$$\propto\frac{1}{2}a^TL^{-1}a+\frac{1}{2\sigma^2}\sum_{m=1}^{M}\sum_{n=1}^{N}W_{nm}\|y_m-f_n\|_2^2. \tag{C.8}$$

In this context the shifted points f_n depend on the unknown coefficients a. However, the descriptor distances $\mathrm{d}^{\mathrm{SHOT}}$ are independent of a, therefore, they vanish in the last step in (C.7).

C.2.1 Robust Correspondences

As proposed in (10.19) we will reformulate the correspondence penalization term $\frac{1}{2}\|y_m-f_n\|_2^2$ in (C.7) in order to make our method more robust:

$$\rho(\|y_m-f_n\|_2). \tag{C.9}$$

Note that this has a probabilistic interpretation as a mixture of Huber densities. We choose the outer slope as $r_0 := 0.01$. For values $\|y_m - f_n\|_2 \leq r_0$ the term in (C.9) remains the same but for bigger residua the penalization by the Huber loss grows linearly. Due to this property the Huber norm does not penalize outliers exorbitantly high and is therefore more robust than the standard least squares loss.

C.2.2 Gauss Newton

In order to compute the optimal deformation parameters a, we have to minimize the energy $E(a)$ defined in (10.19). For this purpose, we first derive how to optimize the following simplified energy and later derive how to handle the Huber loss distance penalization:

$$E^{\mathrm{LS}}(a) := \frac{1}{2} a^T L^{-1} a + \frac{1}{2\sigma^2} \sum_{m=1}^{M} \sum_{n=1}^{N} W_{nm} \|y_m - f_n\|_2^2. \tag{C.10}$$

We assume in this context that the correspondences W are fixed. For the optimization we use a Gauss-Newton type method which yields an iteration scheme minimizing E^{LS}. Like in the Levenberg-Marquardt algorithm (**Levenberg** 1944) the iteration contains an additional damping term L^{-1} which is added to the Hessian of the non-linear least squares term.

Applying the standard Gauss-Newton methodology, we get an iterative method to determine the weights a. The general idea of this approach is that the shifted points $f_n(a)$ are linearized around the current iterate $a^{(i)}$ in the energy (C.10):

$$E^{\mathrm{LS}}(a) \approx \frac{1}{2} a^T L^{-1} a + \frac{1}{2\sigma^2} \sum_{m=1}^{M} \sum_{n=1}^{N} W_{nm} \|y_m - \underbrace{\big(f_n(a^{(i)}) + D_a f_n(a^{(i)})(a - a^{(i)})\big)}_{\approx f_n(a)} \|_2^2. \tag{C.11}$$

This approximate energy is linear in the current unknown a, so the remaining task is a simple linear least squares problem. The recursion formula to compute the approximate deformation parameters $a^{(i)}$ admits the following explicit form:

$$a^{(i+1)} := a^{(i)} - \big(J^T \tilde{W} J + \sigma^2 L^{-1}\big)^{-1} \big(J^T r - \sigma^2 L^{-1} a^{(i)}\big). \tag{C.12}$$

In this context J consists of the Jacobians of f_n, r of the (weighted) distance residuals and $\tilde{W}$ is a diagonal matrix containing the column sums of W. Let $e_D = (1, ..., 1)^T \in \mathbb{R}^D, e_M = (1, ..., 1)^T \in \mathbb{R}^M$, then these quantities are explicitly defined as:

$$J = \begin{pmatrix} \mathrm{D}_a f_1 \\ \vdots \\ \mathrm{D}_a f_N \end{pmatrix} \in \mathbb{R}^{ND \times K}. \tag{C.13a}$$

$$r = \begin{pmatrix} \sum_{m=1}^{M} W_{1m}(f_1 - y_m) \\ \vdots \\ \sum_{m=1}^{M} W_{Nm}(f_N - y_m) \end{pmatrix} \in \mathbb{R}^{ND}. \tag{C.13b}$$

$$\tilde{W} = \operatorname{diag}\big((We_M) \otimes e_D\big) = \operatorname{diag} \begin{pmatrix} e_D \sum_{m=1}^{M} W_{1m} \\ \vdots \\ e_D \sum_{m=1}^{M} W_{Nm} \end{pmatrix} \in \mathbb{R}^{ND \times ND}. \tag{C.13c}$$

What is left to specify is how to compute the derivatives $\mathrm{D}_a f_n \in \mathbb{R}^{D \times K}$ in (C.13a). Note that $f_n = x_n^{(T)}$ is recursively defined in (10.13), therefore we need to apply the chain rule. As a result the derivative $\mathrm{D}_a x_n^{(t)}$ is passed from the first time step $t = 0$ to the last $t = T$ and gradually modified in each step. Inserting the Karhunen-Loève representation (10.12) in the definition (10.13) yields the following recursive formula:

$$x_n^{(t+1)}(a) := x_n^{(t)} + h \sum_{k=1}^{K} v_k \Bigg(x_n^{(t)} + \frac{h}{2} \sum_{k=1}^{K} v_k\big(x_n^{(t)}\big) a_k \Bigg) a_k. \tag{C.14}$$

The dependencies on a are denoted explicitly in order to make it more comprehensible. The quantities $x_n^{(t)}(a)$ can now be differentiated wrt. a:

$$\mathrm{D}_a x_n^{(0)} = 0. \tag{C.15a}$$

$$\mathrm{D}_a x_n^{(t+1)} = \mathrm{D}_a x_n^{(t)} + h \sum_{k=1}^{K} \mathrm{D}_x v_k\big(\cdot\big) \Bigg(\Bigg(I_D + \frac{h}{2} \sum_{k=1}^{K} \mathrm{D}_x v_k\big(x_n^{(t)}\big) a_k \Bigg) \mathrm{D}_a x_n^{(t)} + \begin{pmatrix} | & & | \\ v_1\big(x_n^{(t)}\big) & \dots & v_K\big(x_n^{(t)}\big) \\ | & & | \end{pmatrix} \Bigg) a_k + h \begin{pmatrix} | & & | \\ v_1\big(\cdot\big) & \dots & v_K\big(\cdot\big) \\ | & & | \end{pmatrix}. \tag{C.15b}$$

Note that the Jacobian $\mathrm{D}_x v_k \in \mathbb{R}^{D \times D}$ can be computed analytically for any basis element v_k. In this context $I_D \in \mathbb{R}^{D \times D}$ is the identity matrix.

The only thing left to discuss is how to extend this approach for the Huber loss penalization of the energy E in (10.19). For point distances $\|y_m - f_n\|_2 \leq r_0$ the Huber loss and the least squares loss are the same. For residual values $\|y_m - f_n\|_2 > r_0$ the derivate wrt. the deformation parameters a is the following:

$$\mathrm{D}_a \rho(\|y_m - f_n\|_2) = r_0 \frac{(f_n - y_m)^T}{\|f_n - y_m\|_2} \mathrm{D}_a f_n. \tag{C.16}$$

This eliminates the possibility of a direct Gauss-Newton type optimization which requires non linear least squares terms. We can, however, incorporate this in our algorithm using a simple heuristic. For this purpose we multiply the respective weights W_{nm} with the factor $r_0 \frac{1}{\|f_n - y_m\|_2}$ for $\|y_m - f_n\|_2 > r_0$ in each iteration.

C.3 Karhunen-Loève expansion of the deformation field

Here, we provide some theoretical justification for the particular choice of basis in (10.11) and the construction of the weights (10.18). These λ_k can be interpreted to be the eigenvalues of the linear operator $\mathcal{C} := (-\Delta)^{-\frac{D}{2}}$ corresponding to the eigenfunctions ϕ_k. We can then apply the so-called Karhunen-Loève expansion (**Sullivan** 2015, Ch. 11) to our setup. This framework provides us with an alternative representation of the potential field Φ which can in turn be used to define the deformation field v. For further reference concerning the mathematical foundation of this approach the interested reader is referred to **Stuart** (2010), **Cotter et al.** (2013) and **Dashti and Stuart** (2017). Following this approach, one can now derive a construction which enables us to sample arbitrary square integrable scalar fields $\hat{\Phi} : \Omega \to \mathbb{R}$:

$$\hat{\Phi}(x) = \sum_{k=1}^{\infty} \phi_k(x) \sqrt{\lambda_k} \xi_k. \tag{C.17}$$

According to the Karhunen-Loève expansion the coefficients are samples of the Gaussian distributions $\xi_k \sim \mathcal{N}(0, 1)$. This approach can be applied to get an alternative description of each entry of the potential vector field Φ. Inserting it in (10.5), we obtain an alternative representation of the deformation field v. In particular, we get the summation (10.12) for the basis elements (10.11) in the 3D case. Indeed we can derive the following Gaussian prior distribution for the weights a_k:

$$a_k = \sqrt{\lambda_k} \xi_k \sim \mathcal{N}(0, \lambda_k). \tag{C.18}$$

Remark The choice of the exponent $\frac{D}{2}$ in the definition of the weights λ_k (10.18) is not arbitrary. In general it is supposed to be chosen strictly larger than $\frac{D}{2}$ for our resulting basis to fulfill certain approximation properties in the limit of infinitely many basis functions, see **Dashti and Stuart** (2017), Ch. 2.4. However, we achieved good results in our experiments by choosing it as small as possible in order to not suppress the high frequencies more severely than necessary. In particular, the expressiveness of or method seems to deteriorate when a large exponent is chosen because the weights (10.18) decay too rapidly. Therefore, we typically set it to $\frac{D}{2}$ which works fine, although this is not theoretically justified when the number of basis functions approaches infinity. On the other hand, choosing it smaller than $\frac{D}{2}$ causes the expected value of the velocity series to diverge for $K \to \infty$.

To conclude this section, we want to motivate our choice of the Karhunen-Loève framework and the particular linear operator $\mathcal{C}$ to model the deformation fields v. In the context of the Karhunen-Loève expansion the operator $\mathcal{C}$ is called covariance operator. It is typically chosen to incorporate some assumptions about the regularity of the produced sample functions. A natural assumption about the deformation fields v is that they are as uniform as possible. This yields that the resulting correspondence mappings are to some degree spatially continuous.

Therefore, we require the Dirichlet energy to be small:

$$\|\nabla v\|_{L_2}^2 = \sum_{d=1}^{D} \int_{\Omega} \|\nabla v_d(x)\|_2^2 \mathrm{d}x. \tag{C.19}$$

We can achieve this by penalizing the high frequency components of v. These frequencies are strongly related to those of the potential field Φ, because according to (10.5) the basis elements are simply mapped onto the velocity basis elements. This mapping does not change the frequencies:

$$\|\nabla v\|_{L_2} = \|\nabla(\nabla \times \Phi)\|_{L_2} = \|\nabla \Phi\|_{L_2}. \tag{C.20}$$

If we choose, *e.g.* $D = 2$, one can prove that the Dirichlet energy $\|\nabla v\|_{L_2}^2$ is equivalent to the squared ℓ_2 norm of the weights ξ:

$$\|\xi\|_{\ell_2}^2 = \|\nabla v\|_{L_2}^2, \text{ for } D = 2. \tag{C.21}$$

A derivation of this property can be found in **Dashti and Stuart** (2017), Ch. 7.1.3. In the case of finitely many parameters $\xi_1, ..., \xi_K$ the norm $\|\xi\|_{\ell_2}$ is equivalent to the Euclidean norm $\|\xi\|_2$ of the vector $\xi = (\xi_1, ..., \xi_K)^T$. The term $\|\xi\|_2^2$ is in turn proportional to the negative log likelihood of the standard normal distributed parameter $\xi \sim \mathcal{N}(0, I_K)$:

$$-\log(p(\xi)) = \frac{K}{2}\log(2\pi) + \frac{1}{2}\|\xi\|_2^2 \propto \frac{1}{2}\|\xi\|_2^2. \tag{C.22}$$

This indicates that a maximum likelihood approach involving ξ leads to an enforcement of uniformity of the vector field v. This can be extended to the case $D = 3$ in a similar manner but we refrain from providing more details here for the sake of brevity.

Own Publications

Eisenberger, M., Z. Lähner, and D. Cremers (2019). Divergence-Free Shape Correspondence by Deformation. In: *Computer Graphics Forum (CGF)* 38.5 (pages 14, 17, 114).

— (2020). Smooth Shells: Multi-Scale Shape Registration with Functional Maps. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pages 18, 19).

Lähner, Z., D. Cremers, and T. Tung (Sept. 2018). DeepWrinkles: Accurate and Realistic Clothing Modeling. In: *European Conference on Computer Vision (ECCV)* (page 18).

Lähner, Z., E. Rodolà, M. M. Bronstein, D. Cremers, O. Burghard, L. Cosmo, A. Dieckmann, R. Klein, and Y. Sahillioglu (May 2016a). SHREC16: Matching of Deformable Shapes with Topological Noise. In: *Eurographics Workshop on 3D Object Retrieval (3DOR)* (pages 18, 53, 95, 96, 123, 126).

Lähner, Z., E. Rodolà, F. R. Schmidt, M. M. Bronstein, and D. Cremers (May 2016b). Efficient Globally Optimal 2D-to-3D Deformable Shape Matching. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pages 17, 62, 79).

Rodolà, E., Z. Lähner, A. M. Bronstein, M. M. Bronstein, and J. Solomon (2019). Functional Maps Representation on Product Manifolds. In: *Computer Graphics Forum (CGF)* 38.1 (pages 17, 99).

Vestner*, M., Z. Lähner*, A. Boyarski*, O. Litany, R. Slossberg, T. Remez, E. Rodolà, A. M. Bronstein, M. M. Bronstein, R. Kimmel, and D. Cremers (Oct. 2017). Efficient Deformable Shape Correspondence via Kernel Matching. In: *International Conference on 3D Vision (3DV)* (pages 12, 17, 82, 100, 123).

Bibliography

Adams, **B.**, **M. Ovsjanikov**, **M. Wand**, **H.-P. Seidel**, **and L. J. Guibas** (2008). Meshless Modeling of Deformable Shapes and their Motion. In: *Eurographics/SIGGRAPH Symposium on Computer Animation* (page 115).

Aflalo, **Y.**, **H. Brezis**, **and R. Kimmel** (2015a). On the Optimality of Shape and Data Representation in the Spectral Domain. In: *(SIAM) Journal on Imaging Sciences* (page 38).

Aflalo, **Y.**, **A. M. Bronstein**, **and R. Kimmel** (2015b). On convex relaxation of graph isomorphism. In: *Proceedings of the National Academy of Sciences (PNAS)* 112.10 (page 82).

Aflalo, **Y.**, **A. Dubrovina**, **and R. Kimmel** (2016). Spectral generalized multi-dimensional scaling. In: *International Journal on Computer Vision (IJCV)* 118.3 (pages 94, 100, 115, 123).

AIM@SHAPE repository (n.d.). *AIM@SHAPE repository.* `http://visionair.ge.imati.cnr.it:8080/ontologies/shapes/viewgroup.jsp?id=657-Armadillo_1` (page 128).

Albrecht, **T.**, **M. Lüthi**, **and T. Vetter** (2008). A statistical deformation prior for non-rigid image and shape registration. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (page 116).

Anguelov, **D.**, **P. Srinivasan**, **D. Koller**, **S. Thrun**, **J. Rodgers**, **and J. Davis** (2005). SCAPE: shape completion and animation of people. In: *Proceedings of ACM SIGGRAPH* (pages 95, 123, 126).

Appleton, **B. and C. Sun** (2003). Circular shortest paths by branch and bound. In: *Pattern Recognition* 36.11 (pages 64, 69).

Ashburner, **J.** (2007). A fast diffeomorphic image registration algorithm. In: *Neuroimage* 38.1 (page 115).

Ashburner, **J. and K. J. Friston** (2011). Diffeomorphic registration using geodesic shooting and Gauss–Newton optimisation. In: *NeuroImage* 55.3 (page 115).

Aubry, **M.**, **U. Schlickewei**, **and D. Cremers** (2011). The wave kernel signature: A quantum mechanical approach to shape analysis. In: *International Conference on Computer Vision (ICCV)* (pages 10, 73, 82).

Bay, **H.**, **T. Tuytelaars**, **and L. Van Gool** (2006). SURF: Speeded Up Robust Features. In: *European Conference on Computer Vision (ECCV)* (page 8).

Beg, **M. F.**, **M. I. Miller**, **A. Trouvé**, **and L. Younes** (2005). Computing large deformation metric mappings via geodesic flows of diffeomorphisms. In: *International Journal of Computer Vision (IJCV)* 61.2 (page 115).

Bender, **J.**, **M. Müller**, **and M. Macklin** (2015). *Position-Based Simulation Methods in Computer Graphics.* Eurographics Tutorial (page 13).

Berg, **A.**, **T. Berg**, **and J. Malik** (2005). Shape Matching and Object Recognition using Low Distortion Correspondences. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (page 11).

Berger, **M.**, **P. Gauduchon**, **and E. Mazet** (1971). *Le spectre d'une variété Riemannienne.* Lecture notes in mathematics. Springer-Verlag (page 44).

Bernard, **F.**, **Z. K. Suri**, **and C. Theobalt** (2020). MINA: Convex Mixed-Integer Programming for Non-Rigid Shape Alignment. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (page 13).

Bernard, **F.**, **C. Theobalt**, **and M. Moeller** (2018). DS*: Tighter Lifting-Free Convex Relaxations for Quadratic Matching Problems. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (page 12).

Bernard, **F.**, **N. Vlassis**, **P. Gemmar**, **A. Husch**, **J. Thunberg**, **J. Goncalves**, **and F. Hertel** (2016). Fast correspondences for statistical shape models of brain structures. In: *SPIE Medical Imaging* (page 84).

Besl, **P. J. and N. D. McKay** (Feb. 1992). A method for registration of 3-D shapes. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 14.2 (pages 9, 83).

Bogo, **F.**, **J. Romero**, **M. Loper**, **and M. J. Black** (2014). FAUST: Dataset and evaluation for 3D mesh registration. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pages 75, 95, 127, 128).

Bogomolny, **E.**, **O. Bohigas**, **and C. Schmit** (2003). Spectral properties of distance matrices. In: *Journal of Physics A* 36.12 (page 86).

Boscaini, **D.**, **J. Masci**, **S. Melzi**, **M. M. Bronstein**, **U. Castellani**, **and P. Vandergheynst** (2015). Learning class-specific descriptors for deformable shapes using localized spectral convolutional networks. In: *Computer Graphics Forum (CGF)* 34.5 (pages 10, 84).

Boscaini, **D.**, **J. Masci**, **E. Rodolà**, **and M. M. Bronstein** (2016a). Learning shape correspondence with anisotropic convolutional neural networks. In: *Neural Information Processing Systemns (NIPS)* (pages 10, 83, 84).

Boscaini, **D.**, **J. Masci**, **E. Rodolà**, **M. M. Bronstein**, **and D. Cremers** (2016b). Anisotropic diffusion descriptors. In: *Computer Graphics Forum (CGF)*. Vol. 35. 2 (page 84).

Bregler, **C.**, **A. Hertzmann**, **and H. Biermann** (2000). Recovering non-rigid 3D shape from image streams. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* 2 (page 116).

Bronstein, **A. M.**, **M. M. Bronstein**, **and R. Kimmel** (2008). *Numerical Geometry of Non-Rigid Shapes.* Springer (pages 75, 85, 95, 123, 126, 128).

Bronstein, **A. M.**, **M. M. Bronstein**, **and R. Kimmel** (2006a). Efficient computation of isometry-invariant distances between surfaces. In: *SIAM Journal on Scientific Computing* 28.5 (pages 75, 82).

— (2006b). Generalized multidimensional scaling: a framework for isometry-invariant partial surface matching. In: *Proceedings of the National Academy of Sciences (PNAS)* 103.5 (pages 63, 115).

Bronstein, A. M., M. M. Bronstein, R. Kimmel, M. Mahmoudi, and G. Sapiro (2010). A Gromov-Hausdorff framework with diffusion geometry for topologically-robust non-rigid shape matching. In: *International Journal of Computer Vision (IJCV)* 89.2 (page 85).

Bronstein, M. M. and I. Kokkinos (2010). Scale-invariant heat kernel signatures for non-rigid shape recognition. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (page 82).

Burghard, O., A. Dieckmann, and R. Klein (2017). Embedding Shapes with Green's functions for Global Shape Matching. In: *Computers & Graphics* 68C (page 123).

Burghard, O. and R. Klein (2017). Efficient Lifted Relaxations of the Quadratic Assignment Problem. In: *Vision, Modeling & Visualization (VMV)* (page 11).

Canzani, Y. (2013). *Analysis of manifolds via the Laplacian.* Lecture notes from a Harvard course (pages 7, 11).

Chavel, I. (1984). *Eigenvalues in Riemannian geometry.* Second. Academic Press (pages 7, 43).

Chen, Q. and V. Koltun (2015). Robust nonrigid registration by convex optimization. In: *International Conference on Computer Vision (ICCV)* (pages 82, 85, 115).

Chorin, A. J. and J. E. Marsden (1993). *A Mathematical Introduction to Fluid Mechanics.* Springer (page 118).

Choukroun, Y., A. Shtern, A. Bronstein, and R. Kimmel (2018). Hamiltonian operator for spectral shape analysis. In: *Transactions on Visualization and Computer Graphics* 26.2 (pages 100, 107).

Coifman, R. R., S. Lafon, A. B. Lee, M. Maggioni, B. Nadler, F. Warner, and S. W. Zucker (2005). Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. In: *Proceedings of the National Academy of Sciences (PNAS)* 102.21 (page 82).

Corman, E., M. Ovsjanikov, and A. Chambolle (2014). Supervised Descriptor Learning for Non-Rigid Shape Matching. In: *European Conference on Computer Vision (ECCV) Workshop* (page 10).

Cosmo, L., A. Norelli, O. Halimi, R. Kimmel, and E. Rodolà (2020). LIMP: Learning Latent Shape Representations with Metric Preservation Priors. In: *European Conference on Computer Vision (ECCV)* (page 8).

Cosmo, L., M. Panine, A. Rampini, M. Ovsjanikov, M. Bronstein, and E. Rodolà (2019). Isospectralization, or how to hear shape, style, and correspondence. In: *Conference on Computer Vision and Pattern Recognition (CVPR)* (page 11).

Cosmo, L., E. Rodolà, J. Masci, A. Torsello, and M. M. Bronstein (2016). Matching deformable objects in clutter. In: *International Conference on 3D Vision (3DV)* (page 100).

Cotter, S. L., G. O. Roberts, A. M. Stuart, and D. White (2013). MCMC methods for functions: modifying old algorithms to make them faster. In: *Statistical Science* 28.3 (page 151).

Crane, K., C. Weischedel, and M. Wardetzky (2017). The Heat Method for Distance Computation. In: *Communication ACM* 60.11 (pages 7, 11).

Creusé, **E.**, **S. Nicaise**, **and Z. Tang** (2015). Helmholtz decomposition of vector fields with mixed boundary conditions and an application to a posteriori finite element error analysis of the Maxwell system. In: *Mathematical Methods in the Applied Sciences* 38.4 (page 119).

Dashti, **M. and A. M. Stuart** (2017). The Bayesian approach to inverse problems. In: *Handbook of Uncertainty Quantification* (pages 151, 152).

Dibra, **E.**, **H. Jain**, **C. Oztireli**, **R. Ziegler**, **and M. Gross** (July 2017). Human Shape From Silhouettes Using Generative HKS Descriptors and Cross-Modal Neural Networks. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (page 9).

Dijkstra, **E. W.** (1959). A note on two problems in connexion with graphs. In: *Numerische Mathematik* 1 (pages 68, 69).

do Carmo, **M. P.** (1976). *Differential Geometry of Curves and Surfaces.* Second Edition 2016. Prentice-Hall (page 23).

— (1992). *Riemannian Geometry.* Mathematics (Boston, Mass.) Birkhäuser (pages 23, 25, 30).

Dölz, **J.**, **T. Gerig**, **M. Lüthi**, **H. Harbrecht**, **and T. Vetter** (2017). Efficient computation of low-rank Gaussian process models for surface and image registration. In: (page 116).

Donati, **N.**, **A. Sharma**, **and M. Ovsjanikov** (2020). Deep Geometric Functional Maps: Robust Feature Learning for Shape Correspondence. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (page 10).

Dosovitskiy, **A.**, **P. Fischer**, **E. Ilg**, **P. Häusser**, **C. Hazrba**, **V. Golkov**, **P. v.d. Smagt**, **D. Cremers**, **and T. Brox** (2015). FlowNet: Learning Optical Flow with Convolutional Networks. In: *International Conference on Computer Vision (ICCV)* (page 8).

Dubrovina, **A. and R. Kimmel** (2010). Matching shapes by eigendecomposition of the Laplace-Beltrami operator. In: (page 114).

Duffin, **R. J.** (1959). Distributed and lumped networks. In: *Journal of Mathematics and Mechanics* 8.5 (page 102).

Dym, **N.**, **H. Maron**, **and Y. Lipman** (2017). DS++: a Flexible, Scalable and Provably Tight Relaxation for Matching Problems. In: *ACM Transactions on Graphics (ToG)* 36.6 (page 12).

Edelstein, **M.**, **D. Ezuz**, **and M. Ben-Chen** (2020). ENIGMA: Evolutionary Non-Isometric Geometry MAtching. In: *ACM Transactions on Graphics (ToG)* (page 12).

Eitz, **M.**, **R. Richter**, **T. Boubekeur**, **K. Hildebrand**, **and M. Alexa** (2012). Sketch-Based Shape Retrieval. In: *ACM Transactions on Graphics (ToG)* 31.4 (pages 64, 75).

Elad, **A. and R. Kimmel** (2003). On bending invariant signatures for surfaces. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 25.10 (page 82).

Estellers, **V.**, **F. R. Schmidt**, **and D. Cremers** (2018). Robust Fitting of Subdivision Surfaces for Smooth Shape Analysis. In: *International Conference on 3D Vision (3DV)* (page 28).

Euclid (300BC). *Elements* (page 3).

Eynard, **D.**, **E. Rodolà**, **K. Glashoff**, **and M. M. Bronstein** (2016). Coupled functional maps. In: *International Conference on 3D Vision (3DV)* (page 100).

Ezuz, **D. and M. Ben-Chen** (2017). *Deblurring and Denoising of Maps between Shapes.* Wiley Online Library (page 100).

Ezuz, **D.**, **B. Heeren**, **O. Azencot**, **M. Rumpf**, **and M. Ben-Chen** (2019). Elastic Correspondence between Triangle Meshes. In: *Computer Graphics Forum (CGF)* (pages 13, 50).

Fiedler, **M.** (1973). Algebraic connectivity of graphs. In: *Czechoslovak Mathematical Journal* 23.98 (page 102).

Funck, **W. von**, **H. Theisel**, **and H.-P. Seidel** (2006). Vector field based shape deformations. In: *ACM Transactions on Graphics (ToG)*. Vol. 25. 3. ACM (page 115).

Furuya, **T. and R. Ohbuchi** (2014). Hashing Cross-Modal Manifold for Scalable Sketch-Based 3D Model Retrieval. In: *International Conference on 3D Vision (3DV)* (pages 64, 75).

Gao, **L.**, **S.-Y. Chen**, **Y.-K. Lai**, **and S. Xia** (2017). Data-Driven Shape Interpolation and Morphing Editing. In: *Computer Graphics Forum (CGF)* 36 (page 114).

Ginzburg, **D. and D. Raviv** (2019). *Cyclic Functional Mapping: Self-Supervised Correspondence between non-isometric deformable shapes.* arXiv:1912.01249 (page 14).

Glashoff, **K. and C. P. Ortlieb** (2017). Composition Operators, Matrix Representation, and the Finite Section Method: A Theoretical Framework for Maps between Shapes. In: *arXiv preprint arXiv:1705.00325* (page 106).

Gold, **S. and A. Rangarajan** (1996). A graduated assignment algorithm for graph matching. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 18.4 (page 12).

Griffiths, **D. and D. J. Higham** (2010). *Numerical Methods for Ordinary Differential Equations. Initial Value Problems.* Springer (page 121).

Gröchenig, **K.**, **Z. Rzeszotnik**, **and T. Strohmer** (2010). Convergence analysis of the finite section method and Banach algebras of matrices. In: *Integral Equations and Operator Theory* 67.2 (page 106).

Groueix, **T.**, **M. Fisher**, **V. G. Kim**, **B. C. Russell**, **and M. Aubry** (2018). 3D-CODED: 3D Correspondences by Deep Deformation. In: *European Conference on Computer Vision (ECCV)* (page 14).

Guillemin, **V. and A. Pollack** (2010). *Differential Topology.* AMS Chelsea Publishing Series. American Mathematical Society (page 43).

Guo, **Y.**, **M. Bennamoun**, **F. Sohel**, **M. Lu**, **J. Wan**, **and N. M. Kwok** (2016). A Comprehensive Performance Evaluation of 3D Local Feature Descriptors. In: *International Journal of Computer Vision (IJCV)* (page 9).

Halimi, **O.**, **O. Litany**, **E. Rodolà**, **A. M. Bronstein**, **and R. Kimmel** (2019). Unsupervised Learning of Dense Shape Correspondence. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (page 14).

Hammack, **R.**, **W. Imrich**, **and S. Klavar** (2011). *Handbook of Product Graphs.* Discrete Mathematics and its Applications. Second Edition 2016. CRC Press (pages 39, 102).

Hartley, **R. I. and A. Zisserman** (2004). *Multiple View Geometry in Computer Vision.* Second. Cambridge University Press (page 8).

Heeren, **B.**, **M. Rumpf**, **M. Wardetzky**, **and B. Wirth** (2012). Time-Discrete Geodesics in the Space of Shells. In: *Computer Graphics Forum (CGF)* 31.5 (page 13).

Herzog, R., D. Mewes, M. Wand, L. Guibas, and H.-P. Seidel (2015). LeSSS: Learned Shared Semantic Spaces for Relating Multi-Modal Representations of 3D Shapes. In: *Computer Graphics Forum (CGF)* 34.5 (page 64).

Hochreiter, S. and J. Schmidhuber (1997). Long Short-Term Memory. In: *Neural Computing* 9.8 (page 19).

Hu, N. and L. Guibas (2013). Spectral descriptors for graph matching. In: *arXiv preprint arXiv:1304.1572* (page 82).

Huang, R. and M. Ovsjanikov (2017). Adjoint Map Representation for Shape Analysis and Matching. In: *Computer Graphics Forum (CGF)* 36.5 (page 100).

Huang, Q.-X., B. Adams, M. Wicke, and L. Guibas (2008). Non-Rigid Registration Under Isometric Deformations. In: *Computer Graphics Forum (CGF)* 27.5 (page 115).

Huang, Q.-X., F. Wang, and L. Guibas (2014). Functional map networks for analyzing and exploring large shape collections. In: *ACM Transactions on Graphics (ToG)* 33.4 (pages 13, 100).

Huber, P. J. (1964). Robust Estimation of a Location Parameter. In: *Annals of Statistics* 53.1 (page 125).

Hueting, M., M. Ovsjanikov, and N. J. Mitra (2015). CrossLink: joint understanding of image and 3D model collections through shape and camera pose variations. In: *ACM Transactions on Graphics (ToG)* 34.6 (page 64).

Isola, P., J.-Y. Zhu, T. Zhou, and A. A. Efros (2017). Image-to-Image Translation with Conditional Adversarial Networks. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (page 19).

Johnson, A. E. and M. Hebert (1999). Using spin images for efficient object recognition in cluttered 3D scenes. In: *Transcations on Pattern Analysis and Machine Intelligence (TPAMI)* 21.5 (page 10).

Kaick, O. van, K. Xu, H. Zhang, Y. Wang, S. Sun, A. Shamir, and D. Cohen-Or (2013). Co-Hierarchical Analysis of Shape Structures. In: 32.4 (page 8).

Kezurer, I., S. Z. Kovalsky, R. Basri, and Y. Lipman (2015). Tight relaxation of quadratic matching. In: *Computer Graphics Forum (CGF)*. Vol. 34. 5 (pages 11, 82).

Kim, V. G., Y. Lipman, and T. A. Funkhouser (2011). Blended intrinsic maps. In: *ACM Transactions on Graphics (ToG)* 30.4 (pages 94, 123, 126).

Kimmel, R. and J. A. Sethian (1998). Computing Geodesic Paths on Manifolds. In: *Proceedings of the National Academy of Sciences (PNAS)* 95 (page 7).

Knoverek, C. R., G. K. Amarasinghe, and G. R. Bowman (2019). Advanced Methods for Accessing Protein Shape-Shifting Present New Therapeutic Opportunities. In: *Trends in Biochemical Sciences* 44.4 (page 3).

Kovnatsky, A., M. M. Bronstein, A. M. Bronstein, K. Glashoff, and R. Kimmel (2013). Coupled quasi-harmonic bases. In: *Computer Graphics Forum (CGF)* 32.2 (pages 83, 100).

Kovnatsky, A., M. M. Bronstein, X. Bresson, and P. Vandergheynst (2015). Functional Correspondence by Matrix Completion. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pages 83, 100).

Kovnatsky, **A.**, **K. Glashoff**, **and M. M. Bronstein** (2016). MADMM: a generic algorithm for non-smooth optimization on manifolds. In: *European Conference on Computer Vision (ECCV)* (page 100).

Leal-Taixé, **L.**, **A. Milan**, **K. Schindler**, **D. Cremers**, **I. Reid**, **and S. Roth** (2017). *Tracking the trackers: an analysis of the state of the art in multiple object tracking.* arXiv preprint arXiv:1704.02781 (page 4).

Lederman, **R. R. and R. Talmon** (2015). Learning the geometry of common latent variables using alternating-diffusion. In: *Applied and Computational Harmonic Analysis* (page 92).

Lee, **S. C. and M. Kazhdan** (2019). Dense Point-to-Point Correspondences Between Genus-Zero Shapes. In: *Computer Graphics Forum (CGF)* 38 (5) (page 6).

Leordeanu, **M. and M. Hebert** (2005). A spectral technique for correspondence problems using pairwise constraints. In: *International Conference on Computer Vision (ICCV)* (pages 12, 63, 82, 86).

Levenberg, **K.** (1944). A method for the solution of certain non-linear problems in least squares. In: *Quarterly of Applied Mathematics* 2.2 (pages 125, 149).

Li, **B.**, **Y. Lu**, **C. Li**, et al. (2015). A comparison of 3D shape retrieval methods based on a large-scale benchmark supporting multimodal queries. In: *Computer Vision and Image Understanding (CVIU)* 131 (pages 64, 72, 75).

Lipman, **Y. and T. Funkhouser** (2009). Möbius voting for surface correspondence. In: *ACM Transactions on Graphics (ToG).* Vol. 28. 3 (pages 94, 123).

Lipman, **Y.**, **O. Sorkine**, **D. Levin**, **and D. Cohen-Or** (2005). Linear rotation-invariant coordinates for meshes. In: *ACM Transactions on Graphics (ToG).* Vol. 24. 3. ACM (page 116).

Litany, **O.**, **T. Remez**, **E. Rodolà**, **A. M. Bronstein**, **and M. M. Bronstein** (2017a). Deep functional maps: Structured prediction for dense shape correspondence. In: *International Conference on Computer Vision (ICCV)* (pages 10, 14, 83, 100).

Litany, **O.**, **E. Rodolà**, **A. Bronstein**, **and M. Bronstein** (2017b). Fully Spectral Partial Shape Matching. In: *Computer Graphics Forum (CGF)* 36.2 (pages 95, 123).

Litany, **O.**, **E. Rodolà**, **A. M. Bronstein**, **M. M. Bronstein**, **and D. Cremers** (2016). Non-Rigid Puzzles. In: *Computer Graphics Forum (CGF)* 35.5 (pages 13, 100).

Litman, **R. and A. M. Bronstein** (2014). Learning spectral descriptors for deformable shape correspondence. In: *Transcations on Pattern Analysis and Machine Intelligence (TPAMI)* 36.1 (page 10).

Liu, **X.**, **A. Donate**, **M. Jemison**, **and W. Mio** (2008). Kernel functions for robust 3D surface registration with spectral embeddings. In: *International Conference on Pattern Recognition (ICPR)* (page 85).

Loncaric, **S.** (1998). A survey on Shape Analysis Techniques. In: *Pattern Recognition* 31.8 (page 6).

Lowe, **D. G.** (1999). Object Recognition from Local Scale-Invariant Features. In: *International Conference on Computer Vision (ICCV)* (page 8).

Lucas, **B. D. and T. Kanade** (1981). An iterative image registration technique with an application to stereo vision. In: *Image Understanding Workshop* (page 8).

Lüthi, M., C. Jud, T. Gerig, and T. Vetter (2016). Gaussian Process Morphable Models. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 99 (page 116).

Ma, J., J. Jiang, C. Liu, and Y. Li (2017). Feature guided Gaussian mixture model with semi-supervised EM and local geometric constraint for retinal image registration. In: *Information Sciences* 417 (page 115).

Ma, J., J. Zhao, J. Tian, A. L. Yuille, and Z. Tu (2014). Robust point matching via vector field consensus. In: *IEEE Transactions on Image Processing* 23.4 (pages 114–116, 122).

Ma, J., J. Zhao, and A. L. Yuille (2016). Non-rigid point set registration by preserving global and local structures. In: *IEEE Transactions on Image Processing* 25.1 (page 115).

Maes, M. (1991). Polygonal shape recognition using string-matching techniques. In: *Pattern Recognition* 24.5 (page 64).

Mandad, M., D. Cohen-Steiner, L. Kobbelt, P. Alliez, and M. Desbrun (2017). Variance-Minimizing Transport Plans for Inter-surface Mapping. In: *ACM Transactions on Graphics (ToG)* 36 (page 100).

Masci, J., D. Boscaini, M. M. Bronstein, and P. Vandergheynst (2015). Geodesic convolutional neural networks on Riemannian manifolds. In: *International IEEE Workshop on 3D Representation and Recognition (3DRR)* (pages 83, 84).

Masci, J., M. M. Bronstein, A. M. Bronstein, and J. Schmidhuber (2013). Multimodal similarity-preserving hashing. In: *Transcations on Pattern Analysis and Machine Intelligence (TPAMI)* 36.4 (page 61).

Mateus, D., R. Horaud, D. Knossow, F. Cuzzolin, and E. Boyer (2008). Articulated shape matching using Laplacian eigenfunctions and unsupervised point registration. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (page 82).

Melzi, S., J. Ren, E. Rodolà, A. Sharma, P. Wonka, and M. Ovsjanikov (2019). ZoomOut: Spectral Upsampling for Efficient Shape Correspondence. In: *Computer Graphics Forum (CGF)* (page 92).

Melzi, S., E. Rodolà, U. Castellani, and M. M. Bronstein (2017). Localized Manifold Harmonics for Spectral Shape Analysis. In: *Computer Graphics Forum (CGF)* (pages 100, 107).

Mémoli, F. (2011). Gromov–Wasserstein distances and the metric approach to object matching. In: *Foundations of Computational Mathematics* 11.4 (page 100).

Mémoli, F. and G. Sapiro (2005). A theoretical and computational framework for isometry invariant recognition of point cloud data. In: *Foundations of Computational Mathematics* 5.3 (pages 63, 82).

Miller, M. I., A. Trouvé, and L. Younes (2006). Geodesic shooting for computational anatomy. In: *Journal of Mathematical Imaging and Vision* 24.2 (page 115).

Milnor, J. (1969). A Problem in Cartography. In: *The American Mathematical Monthly* 76 (page 25).

Monti, F., D. Boscaini, J. Masci, E. Rodolà, J. Svoboda, and M. M. Bronstein (2017). Geometric deep learning on graphs and manifolds using mixture model CNNs. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pages 83, 84).

Munkres, J. (1957). Algorithms for the Assignment and Transportation Problems. In: *Journal of the Society for Industrial and Applied Mathematics* 5.1 (pages 66, 74).

Myronenko, A. and X. Song (2010). Point Set Registration: Coherent Point Drift. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 32.12 (pages 114–116, 122–124, 148).

Natsume, R., S. Saito, Z. Huang, W. Chen, C. Ma, H. Li, and S. Morishima (June 2019). SiCloPe: Silhouette-Based Clothed People. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (page 9).

Nogneng, D. and M. Ovsjanikov (2017). Informative Descriptor preservation via commutativity for shape matching. In: *Computer Graphics Forum (CGF)* 36.2 (page 100).

Nogneng, D., S. Melzi, E. Rodolà, U. Castellani, M. Bronstein, and M. Ovsjanikov (2018). Improved Functional Mappings via Product Preservation. In: *Computer Graphics Forum (CGF)* (page 100).

Ovsjanikov, M., M. Ben-Chen, J. Solomon, A. Butscher, and L. Guibas (2012). Functional Maps: A Flexible Representation of Maps Between Shapes. In: *ACM Transactions on Graphics (ToG)* 31.4 (pages 12, 13, 63, 83, 87, 94, 103, 105, 106, 115, 123).

Ovsjanikov, M., E. Corman, M. Bronstein, E. Rodolà, M. Ben-Chen, L. Guibas, F. Chazal, and A. Bronstein (2017). Computing and Processing Correspondences with Functional Maps. In: *ACM SIGGRAPH 2017 Courses* (page 100).

Ovsjanikov, M., Q.-X. Huang, and L. Guibas (2011). A Condition Number for Non-Rigid Shape Matching. In: *Computer Graphics Forum (CGF)* 30.5 (page 11).

Paladini, M., A. Del Bue, M. Stosic, M. Dodig, J. Xavier, and L. Agapito (2009). Factorization for non-rigid and articulated structure using metric projections. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (page 116).

Pardalos, P. M. and H. Wolkowicz (1994). *Quadratic Assignment and Related Problems.* Vol. 16. American Mathematical Society (page 82).

Pinkall, U. and K. Pothier (1993). Computing Discrete Minimal Surfaces and their Conjugates. In: *Experimental Mathematics* (page 34).

Pokrass, J., A. M. Bronstein, and M. M. Bronstein (2013a). Partial shape matching without point-wise correspondence. In: *Numerical Mathematics: Theory, Methods and Applications* 6 (page 63).

Pokrass, J., A. M. Bronstein, M. M. Bronstein, P. Sprechmann, and G. Sapiro (2013b). Sparse modeling of intrinsic correspondences. In: *Computer Graphics Forum (CGF)* 32.2 (page 100).

Pons-Moll, G., S. Pujades, S. Hu, and M. Black (2017). ClothCap: Seamless 4D Clothing Capture and Retargeting. In: *ACM Transactions on Graphics (ToG)* 36.4 (page 18).

Ralph, R. (n.d.). *MPEG-7 Core Experiment CE-Shape-1 Test Set.* `http://www.dabi.temple.edu/~shape/MPEG7/dataset.html` (page 122).

Rampini, A., I. Tallina, M. Ovsjanikov, M. M. Bronstein, and E. Rodolà (2019). Correspondence-Free Region Localization for Partial Shape Similarity via Hamiltonian Spectrum Alignment. In: *International Conference on 3D Vision (3DV)* (page 11).

Raviv, D., A. M. Bronstein, M. M. Bronstein, R. Kimmel, and N. Sochen (2011). Affine-invariant geodesic geometry of deformable 3D shapes. In: *Computers & Graphics* 35.3 (page 85).

Ren, J., A. Poulenard, P. Wonka, and M. Ovsjanikov (2018). Continuous and Orientation-preserving Correspondences via Functional Maps. In: *ACM Transactions on Graphics (ToG)* 37.6 (page 115).

Reuter, M., F.-E. Wolter, and N. Peinecke (2006). Laplace-Beltrami Spectra As 'Shape-DNA' of Surfaces and Solids. In: *Computer Aided Design* 38.4 (pages 11, 77–79).

Rodolà, E., A. Bronstein, A. Albarelli, F. Bergamasco, and A. Torsello (2012). A game-theoretic approach to deformable shape matching. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pages 63, 82).

Rodolà, E., S. R. Bulò, T. Windheuser, M. Vestner, and D. Cremers (2014a). Dense Non-Rigid Shape Correspondence Using Random Forests. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pages 14, 83, 95, 123, 130).

Rodolà, E., L. Cosmo, M. M. Bronstein, A. Torsello, and D. Cremers (2016). Partial Functional Correspondence. In: *Computer Graphics Forum (CGF)* (pages 13, 63, 72, 95, 100, 123).

Rodolà, E., M. Moeller, and D. Cremers (2015). Point-wise Map Recovery and Refinement from Functional Correspondence. In: *Vision, Modeling and Visualization (VMV)*. Eurographics Association (pages 13, 63, 87, 115).

Rodolà, E., S. Rota Bulò, and D. Cremers (2014b). Robust Region Detection via Consensus Segmentation of Deformable Shapes. In: *Computer Graphics Forum (CGF)* 33.5 (pages 74, 77, 79).

Rodolà, E., A. Torsello, T. Harada, Y. Kuniyoshi, and D. Cremers (2013). Elastic Net Constraints for Shape Matching. In: *International Conference on Computer Vision (ICCV)* (pages 12, 63).

Roufosse, J.-M., A. Sharma, and M. Ovsjanikov (2019). Unsupervised Deep Learning for Structured Shape Matching. In: *International Conference on Computer Vision (ICCV)* (page 14).

Rustamov, R. (2007). Laplace-Beltrami eigenfunctions for deformation invariant shape representation. In: *Symposium on Geometry Processing (SGP)* (page 10).

Rustamov, R., M. Ovsjanikov, O. Azencot, M. Ben-Chen, F. Chazal, and L. Guibas (2013). Map-Based Exploration of Intrinsic Shape Differences and Variability. In: *ACM Transactions of Graphics (ToG)* (page 13).

Rusu, R. B., N. Blodow, and M. Beetz (2009). Fast Point Feature Histograms (FPFH) for 3D Registration. In: *International Conference on Robotics and Automation (ICRA)* (page 10).

Rutherford, A. (1962). *Vectors, tensors, and the basic equations of fluid mechanics.* Englewood Cliffs, N.J., Prentice-Hall (page 119).

Sahillioglu, Y. (2018). A Genetic Isometric Shape Correspondence Algorithm with Adaptive Sampling. In: *ACM Transactions on Graphics (ToG)* 37.5 (page 12).

Sahillioglu, Y. and Y. Yemez (2011). Coarse-to-Fine Combinatorial Matching for Dense Isometric Shape Correspondence. In: *Computer Graphics Forum (CGF)*. Vol. 30. 5 (page 82).

— (2012). Minimum-Distortion Isometric Shape Correspondence Using EM Algorithm. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 34.11 (pages 95, 123).

Sahni, S. and T. Gonzalez (1976). P-complete approximation problems. In: *Journal of the ACM* 23.3 (page 12).

Schlegel, V. (1883). Theorie der homogen zusammengesetzten Raumgebilde. In: *Leop.-Carol. Deutsche Akademie der Naturforscher* XLIV.4 (page 45).

Schmidt, F. R., D. Farin, and D. Cremers (2007). Fast Matching of Planar Shapes in Sub-cubic Runtime. In: *International Conference on Computer Vision (ICCV)* (page 8).

Schmidt, F. R., E. Töppe, and D. Cremers (June 2009). Efficient Planar Graph Cuts with Applications in Computer Vision. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (page 64).

Schmidt, F., E. Töppe, D. Cremers, and Y. Boykov (2007). Efficient Shape Matching via Graph Cuts. In: *Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR)* (page 64).

Schoenemann, T. and D. Cremers (2010). A Combinatorial Solution for Model-based Image Segmentation and Real-time Tracking. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 32.7 (page 63).

Schoenemann, T. and D. Cremers (2007). Globally Optimal Image Segmentation with an Elastic Shape Prior. In: *International Conference on Computer Vision (ICCV)* (page 8).

Shamai, G. and R. Kimmel (2016). Geodesic Distance Descriptors. In: *CoRR* abs/1611.07360 (pages 83, 100).

Sharp, N. and K. Crane (2020). A Laplacian for Non-Manifold Triangle Meshes. In: *Proc. of Symposium on Geometry Processing (SGP)* (page 36).

Shewchuk, J. R. (2002). Delaunay Refinement Algorithms for Triangular Mesh Generation. In: *Computational Geometry: Theory and Applications* 22.1-3 (page 73).

Shtern, A. and R. Kimmel (2014a). Iterative closest spectral kernel maps. In: *International Conference on 3D Vision (3DV)*. Vol. 1. IEEE (page 85).

— (2014b). Matching the LBO eigenspace of non-rigid shapes via high order statistics. In: *Axioms* 3.3 (page 82).

Solomon, J., K. Crane, and E. Vouga (2014a). *Laplace-Beltrami: The Swiss Army Knife of Geometry Processing.* SGP Tutorial (page 33).

Solomon, J., F. de Goes, G. Peyré, M. Cuturi, A. Butscher, A. Nguyen, T. Du, and L. Guibas (2015). Convolutional Wasserstein Distances: Efficient Optimal Transportation on Geometric Domains. In: *ACM Transactions on Graphics (ToG)* (page 8).

Solomon, J., L. Guibas, and A. Butscher (2013). Dirichlet energy for analysis and synthesis of soft maps. In: *Computer Graphics Forum (CGF)*. Vol. 32. 5 (page 100).

Solomon, J., A. Nguyen, A. Butscher, M. Ben-Chen, and L. Guibas (2012). Soft maps between surfaces. In: *Computer Graphics Forum (CGF)*. Vol. 31. 5 (pages 50, 100).

Solomon, J., G. Peyré, V. G. Kim, and S. Sra (2016). Entropic metric alignment for correspondence problems. In: *ACM Transactions on Graphics (ToG)* 35.4 (page 100).

Solomon, J., R. Rustamov, L. Guibas, and A. Butscher (2014b). Earth mover's distances on discrete surfaces. In: *ACM Transactions on Graphics (ToG)* (page 8).

Sorkine, **O. and M. Alexa** (2007). As-Rigid-As-Possible Surface Modeling. In: *Symposium on Geometry Processing (SGP)* (page 7).

Strebe, **D. R.** (2011). *Images are Mercator Projection, Mollweide Projection, Azimuthal Equidistant Projection.* Licenced under CC BY-SA (https://creativecommons.org/licenses/by-sa/3.0), https://commons.wikimedia.org/wiki/File:Mercator_projection_Square.JPG, https://commons.wikimedia.org/wiki/File:Mollweide_projection_SW.jpg, https://commons.wikimedia.org/wiki/File:Azimuthal_equidistant_projection_SW.jpg (page 23).

Stuart, **A. M.** (2010). Inverse problems: a Bayesian perspective. In: *Acta Numerica* 19 (page 151).

Su, **H.**, **S. Maji**, **E. Kalogerakis**, **and E. G. Learned-Miller** (2015). Multi-view convolutional neural networks for 3D shape recognition. In: *International Conference on Computer Vision (ICCV)* (pages 64, 75).

Su, **Y.**, **Y. Liu**, **B. Cuan**, **and N. Zheng** (2015). Contour Guided Hierarchical Model for Shape Matching. In: *International Conference on Computer Vision (ICCV)* (page 8).

Sullivan, **T. J.** (2015). *Introduction to Uncertainty Quantification.* Springer (pages 124, 151).

Sun, **J.**, **M. Ovsjanikov**, **and L. Guibas** (2009). A Concise and Provably Informative Multi-scale Signature Based on Heat Diffusion. In: *Symposium on Geometry Processing (SGP)* (pages 10, 48, 73, 82, 94).

Surazhsky, **V.**, **T. Surazhsky**, **D. Kirsanov**, **S. J. Gortler**, **and H. Hoppe** (2005). Fast Exact and Approximate Geodesics on Meshes. In: *ACM Transactions on Graphics (ToG)* 24.3 (pages 7, 11).

Tam, **G. K. L.**, **Z.-Q. Cheng**, **Y.-K. Lai**, **F. C. Langbein**, **Y. Liu**, **D. Marshall**, **R. R. Martin**, **X.-F. Sun**, **and P. L. Rosin** (2013). Registration of 3D Point Clouds and Meshes: A Survey from Rigid to Nonrigid. In: *Transactions on Visualization and Computer Graphics* 19.7 (pages 4, 9).

Teschl, **G.** (2012). *Ordinary Differential Equations and Dynamical Systems.* AMS (pages 117, 118).

Tevs, **A.**, **A. Berner**, **M. Wand**, **I. Ihrke**, **and H.-P. Seidel** (2011). Intrinsic shape matching by planned landmark sampling. In: *Computer Graphics Forum (CGF).* Vol. 30. 2 (page 82).

Tombari, **F.**, **S. Salti**, **and L. Di Stefano** (2010). Unique signatures of histograms for local surface description. In: *European Conference on Computer Vision (ECCV)* (pages 7, 10, 94, 124).

Torresani, **L.**, **V. Kolmogorov**, **and C. Rother** (2008). Feature correspondence via graph matching: Models and global optimization. In: *European Conference on Computer Vision (ECCV)* (pages 82, 116).

Vestner, **M.**, **R. Litman**, **E. Rodolà**, **A. Bronstein**, **and D. Cremers** (2017). 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 12, 82, 85, 93, 94, 100).

Vialard, F.-X., L. Risser, D. Rueckert, and C. J. Cotter (2012). Diffeomorphic 3D image registration via geodesic shooting using an efficient adjoint calculation. In: *International Journal of Computer Vision* 97.2 (page 115).

Wang, C., M. M. Bronstein, A. M. Bronstein, and N. Paragios (2011). Discrete minimum distortion correspondence problems for non-rigid shape matching. In: *International Conference on Scale Space and Variational Methods in Computer Vision (SSVM)* (page 82).

Wang, F., Q. Huang, and L. J. Guibas (2013). Image Co-segmentation via Consistent Functional Maps. In: *International Conference on Computer Vision (ICCV)* (page 13).

Wardetzky, M., S. Mathur, F. Kälberer, and E. Grinspun (2007). Discrete Laplace Operators: No Free Lunch. In: *Symposium on Geometry Processing (SGP)* (page 34).

Weiss, S., R. Maier, D. Cremers, R. Westermann, and N. Thuerey (2020). Correspondence-Free Material Reconstruction using Sparse Surface Constraints. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (page 6).

Weyl, H. (1911). Über die asymptotische Verteilung der Eigenwerte. In: *Nachrichten von der Gesellschaft der Wissenschaften zu Göttingen, Mathematisch-Physikalische Klasse* (page 36).

Windheuser, T., U. Schlickewei, F. R. Schmidt, and D. Cremers (2011a). Geometrically Consistent Elastic Matching of 3D Shapes: A Linear Programming Solution. In: *International Conference on Computer Vision (ICCV)* (pages 13, 51, 56, 63, 67, 81, 83, 100).

— (2011b). Large-Scale Integer Linear Programming for Orientation-Preserving 3D Shape Matching. In: *Computer Graphics Forum (CGF)* 30.5 (page 13).

Windheuser, T., M. Vestner, E. Rodolà, R. Triebel, and D. Cremers (2014). Optimal Intrinsic Descriptors for Non-Rigid Shape Analysis. In: *British Machine Vision Conference (BMVC)* (page 10).

Wirth, B., L. Bar, M. Rumpf, and G. Sapiro (2011). A Continuum Mechanical Approach to Geodesics in Shape Space. In: *International Journal of Computer Vision (IJCV)* 93.3 (page 114).

Yoshiyasu, Y., W.-C. Ma, E. Yoshida, and F. Kanehiro (2014). As-Conformal-As-Possible Surface Registration. In: *Computer Graphics Forum (CGF)* 33.5 (page 9).

Zeng, Y., C. Wang, Y. Wang, X. Gu, D. Samaras, and N. Paragios (2010). Dense non-rigid surface registration using high-order graph matching. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (page 100).

Zhang, H., A. Sheffer, D. Cohen-Or, Q. Zhou, O. Van Kaick, and A. Tagliasacchi (2008). Deformation-driven shape correspondence. In: *Computer Graphics Forum.* Vol. 27. 5. Wiley Online Library (page 116).

Zhao, C. and J. H. Burge (2007). Orthonormal vector polynomials in a unit circle, Part I: basis set derived from gradients of Zernike polynomials. In: *Optics Express* 15.26 (page 121).

— (2008). Orthonormal vector polynomials in a unit circle, Part II: completing the basis set. In: *Optics Express* 16.9 (page 121).

Zhou, Q.-Y., J. Park, and V. Koltun (2016). Fast Global Registration. In: *European Conference on Computer Vision (ECCV)* (page 9).

Zhu, C., R. Yi, W. Lira, I. Alhashim, K. Xu, and H. Zhang (2017). Deformation-Driven Shape Correspondence via Shape Recognition. In: *ACM Transactions on Graphics (ToG)* 36.4 (page 14).