



Functional representation of deformable surfaces for geometry processing

Etienne Corman

► To cite this version:

Etienne Corman. Functional representation of deformable surfaces for geometry processing. Computer Vision and Pattern Recognition [cs.CV]. Université Paris Saclay (COMUE), 2016. English. NNT : 2016SACLX075 . tel-01493112

HAL Id: tel-01493112

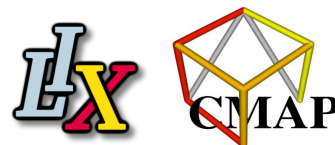
<https://pastel.hal.science/tel-01493112>

Submitted on 21 Mar 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

NNT : 2016SACLX075



THÈSE DE DOCTORAT
DE
L'UNIVERSITÉ PARIS-SACLAY
PRÉPARÉE À
L'ÉCOLE POLYTECHNIQUE

ÉCOLE DOCTORALE N°573
INTERFACES – Approches interdisciplinaires :
fondements, applications et innovations

Spécialité de doctorat : Mathématiques appliquées

par

M. Etienne CORMAN

Représentation fonctionnelle des surfaces déformables
pour l'analyse et la synthèse géométrique

Thèse soutenue à Palaiseau, le 18 novembre 2016

Composition du jury :

M. Tamy BOUBEKEUR	Président	Télécom ParisTech
M. Quentin MERIGOT	Rapporteur	Université Paris-Sud
M. Michael BRONSTEIN	Rapporteur	Università della Svizzera Italiana
M. Leonidas GUIBAS	Examineur	Stanford University
M. Boris THIBERT	Examineur	Université Grenoble Alpes
M. Antonin CHAMBOLLE	Directeur de thèse	École Polytechnique
M. Maks OVSJANIKOV	Directeur de thèse	École Polytechnique



Remerciements

Je remercie chaleureusement toutes les personnes qui m'ont aidé durant mes travaux de recherche, notamment mes directeurs de thèse Maks Ovsjanikov et Antonin Chambolle, pour leurs intérêts et leurs soutiens durant ces trois années. J'aimerais remercier mon jury : Quentin Mérigot et Michael Bronstein qui ont rapporté cette thèse, Leonidas Guibas, Tamy Boubekou et Boris Thibert qui ont accepté de faire partie du jury. J'aimerais aussi remercier la DGA pour avoir financé ces trois années de doctorat. Merci aux chercheurs rencontrés au cours de cette thèse et avec qui j'ai eu le plaisir collaborer : Justin Solomon, Omri Azencot et Mirela Ben-Chen.

Je tiens à remercier le personnel administratif du CMAP en particulier Nasséra Naar et Alexandra Noiret ainsi que Evelyne Rayssac au LIX sans qui cette soutenance n'aurait pas eu lieu.

Durant cette période sur plateau de Saclay il m'a été permis d'effectuer de nombreuses rencontres au CMAP, au LIX ainsi que parmi mes voisins de l'équipe Geometrica (ou DataShape ou tagada je ne sais plus). Je remercie donc chaleureusement toutes les personnes qui ont contribué à la bonne humeur et à l'atmosphère sympathique (mais studieuse) de ces laboratoires. Il serait malheureusement impossible de nommer toutes ces personnes ici sans faire d'impardonnables oublis. Fort de ce constat, j'espère que chacun pourra se reconnaître dans les lignes qui suivent.

Merci à vous polytechniciens, polytechniciennes, centraliens, centraliennes, normaliens, normaliennes, ENStaiens, physiciens, physiciennes, statisticiens, statisticiennes, mathémagiciens, mathémagiciennes, adorateur de la mécanique kantique, propagandiste du transport optimal, annonceur des méthodes de Stein, admirateur des distances de Wasserstein, renonciatrice des problèmes inverses, ordonnateur du permutahédre, approximateur d'espace tangents, explicateur de la persistance homologique, promoteur en TDA, scrutateur de graphes de Reeb, théoricien de l'action utilitariste, mapper-évangéliste, polygon mesh processeur, géomètre différentiel, flotteur par courbure moyenne, profiteuse de quiver, zigzagueuse en persistance, distance à la mesuriste, optimiste de forme, projeteur en simplex, peuple premier du bureau 2033, habitants de la deuxième ère du 2033, représentante de la minorité génoise, migrante portoricaine, festivaliers du film indépendant de Porquerolles, grand commandeur de la guerre des poireaux, décorateur en déchets animaliers, dompteur de moustiques, survivaliste primitiviste, lecteur de la Jaune et la Rouge, fan de Stupeflip, dessinateur de lapin compulsif, breton indépendantiste, touristes venus d'Ulm, grimpeur-bloqueur, gréviste CNTiste, militant de l'Humanuté.

(_/)
(0.o)
(> <)
(")_(")

Contents

1	Introduction	1
2	Introduction en français	7
3	Context in Differential Geometry and its Discrete Equivalents	13
3.1	Differential Geometry	13
3.1.1	Manifold	13
3.1.2	Tangent Vectors and Local Coordinates	15
3.1.3	Mappings	15
3.1.4	Riemannian Manifold	16
3.1.5	Integral on Manifold	19
3.1.6	Gradient, Divergence and Laplacian	20
3.2	Discretization	24
3.2.1	Discrete Manifold	24
3.2.2	Finite Element Method and Cotangent Weight Formula	25
3.2.3	Discrete Local Coordinates	26
4	Operator Representation	31
4.1	Functional Map	31
4.1.1	Mathematical Properties	32
4.1.2	Discrete Functional Maps	34
4.2	Shape Differences	37
4.2.1	Definition	38
4.2.2	Fundamental Properties	39
4.2.3	Algebraic Properties	40
4.2.4	Discrete Shape Differences	43
4.3	Organization of the Thesis	44
I	Shape to Deformation	47
5	Supervised Descriptor Learning for Non-Rigid Shape Matching	49
5.1	Introduction	49
5.1.1	Related Work	50
5.2	Consistent Functional Maps	52
5.2.1	Functional Map Representation	52
5.2.2	Main Challenge	53
5.2.3	Algorithm Outline	54
5.3	Selection of the Best Functional Correspondences	54
5.3.1	Weighting the probe functions	55

5.3.2	Finding the best weights	56
5.4	Basis function extraction	57
5.4.1	Identifying stable subspaces	57
5.4.2	Functional map to a test shape using a reduced basis	58
5.5	Experimental Results	58
5.5.1	Functional correspondences	58
5.5.2	Isometric Shape Matching	58
5.5.3	Non-Isometric Shape Matching	60
5.6	Conclusion	62
6	Continuous Matching via Vector Field Flow	65
6.1	Introduction	65
6.2	Related Work	66
6.3	Functional Maps Conversion	68
6.3.1	Main Challenges	68
6.3.2	Algorithm Overview	69
6.4	Family of Diffeomorphisms	70
6.5	Optimal vector field	71
6.5.1	Optimization Problem	71
6.5.2	Properties	72
6.5.3	Practical Choice of the Norm	73
6.6	Vector Field Flow on Manifold	74
6.7	Results	74
6.7.1	Symmetry Blending	75
6.7.2	Error using a computed functional map	76
6.7.3	Parameters Dependence	77
6.7.4	Performance	78
6.8	Conclusion, Limitations and Future Work	79
II	Deformation to Shape	81
7	Functional Characterization of Intrinsic and Extrinsic Geometry	83
7.1	Introduction	83
7.2	Related Work	85
7.3	Overview	86
7.4	Structure of Discrete Inner Products	87
7.4.1	Discrete Inner Products	87
7.5	Encoding Extrinsic Structure	91
7.5.1	Extrinsic Alternatives	91
7.5.2	Offset Surfaces	93
7.5.3	Recovery of Embedding	95
7.5.4	Discussion	96
7.6	From Inner Products to Shape Differences	97

7.6.1	Discrete Shape Differences	97
7.6.2	Source-Truncated Correspondence	98
7.6.3	Source- and Target-Truncated Correspondence	98
7.7	Recovery of Intrinsic and Extrinsic Structure	99
7.7.1	Triangle Area Computation	99
7.7.2	Edge Length Computation	100
7.7.3	Global Extrinsic Reconstruction	102
7.8	Experiments	102
7.8.1	Shape Space	102
7.8.2	Effects of Truncation	104
7.8.3	Intrinsic Recovery	105
7.8.4	Reconstruction	107
7.8.5	Timings	108
7.9	Discussion & Conclusion	109
8	Functional Characterization of Deformation Fields	113
8.1	Introduction	113
8.2	Related Work	115
8.3	Overview	116
8.4	Extrinsic Vector Fields as Operators	117
8.4.1	Isometric Shape Difference Operator	117
8.4.2	Infinitesimal Deformations as Operators	120
8.4.3	Main Properties of Infinitesimal Shape Differences	122
8.5	Discrete Setting	127
8.5.1	Discrete unified shape difference	127
8.5.2	Infinitesimal shape difference	130
8.5.3	Properties	133
8.6	Representation in basis	135
8.6.1	Basis for the Function Space	135
8.6.2	Vector field basis	135
8.7	Experiments	135
8.7.1	Deformation transfer	135
8.7.2	Functional map inference	138
8.8	Conclusion and Future Work	144
9	Conclusion	145
	Bibliography	147

Introduction

Operator Representation in Geometry Processing

A challenging issue in geometry processing is the wide range of data structures. Discrete geometry is encoded in various ways mostly depending on the acquisition procedure and the application domain. Raw data from a scanner are stored as noisy point-clouds; physically-based simulation is typically done on simplicial complexes; animators and architectural geometry practitioners traditionally work with quad meshes. However, standard problems in geometry processing, such as shape matching, deformation transfer and deformation analysis, should be solved regardless of the representation. Most of the time each community provides their own solutions with respect to their native data which often do not allow cross-data analysis. This leads to the necessity of defining a framework agnostic to the underlying structure.

The basic idea of the functional representation framework, first introduced in [88], is to reach for the most common denominator of geometry analysis, namely real-valued functions. Functions have been historically used and well-studied in many contexts providing a large literature to rely on. For example differential operators have been defined on triangle and tetrahedral meshes (Finite Element Method [19], Discrete Exterior Calculus [52]), polygonal surfaces [3] and point-clouds (meshfree methods based on Radial Basis Functions or Moving Least Squares [46]) to name just a few. Moreover many of these methods come with convergence and stability analysis. Consequently functional operators can be implemented in many situations while studying a single continuous theory. As an example in [34] the authors propose an algorithm to compute geodesic distances based on the Laplace-Beltrami operator allowing them to exhibit results on radically different data structures. This flexibility is also one key of the success of spectral methods and global descriptors such as *Global Point Signature* [105], *Heat Kernel Signature* [119] and *Wave Kernel Signature* [5].

The baseline idea developed in this thesis is to represent standard tools for deformable surfaces (diffeomorphisms, intrinsic distortions, tangent and extrinsic vector fields) as operators acting on functions, following the intuition that their discretizations have been well-studied and are easily comparable across data structures. The starting point of the thesis is the functional map representation for diffeomorphisms, and shape differences, representing intrinsic distortions. We propose in Chapter 5 a novel method, robust to noise and large deformations, to solve shape matching problems with functional maps. Chapters 6, 7 and 8 are providing different answers to an open problem in this framework: the conversion of operator-based representation to point-based representation. Namely, given a functional map we propose an algorithm to convert it to a continuous vertex-to-

vertex assignment or given shape difference operators we suggest solutions to find the underlying deformed surfaces.

Functional Maps

The space of diffeomorphisms is challenging to handle in both continuous and discrete settings. In the simple case of two triangle meshes, the simplest representation consists of assigning a vertex on the first shape to a vertex on the second shape. However when dealing with different number of vertices this makes, for example, the computation of its inverse difficult. A more complete description is therefore needed. The most straightforward answer is to consider a piecewise linear map with the possibility of mapping vertices inside triangles. This way it is possible to compute the map and its inverse consistently. Such a description of discrete diffeomorphism has been used in practice [2, 1] but suffers some drawbacks as it cannot be stored compactly and often leads to challenging optimization problems.

In this context representing the mapping φ as the linear composition operators $f \mapsto f \circ \varphi$ provides a flexible alternative globally agnostic to change of connectivity and easily defined on various types of data. Functional maps are a simplified and compact representation of correspondences as they are often computed between reduced function bases. They can be stored as matrices and analyzed using standard linear algebra tools. Follow-up works (e.g. [100, 101, 64, 96]) focus on computations of functional maps between shapes with unknown correspondences by minimizing various kinds of distortion with guiding functional constraints. However, functional constraints are often unreliable in case of large or non-isometric deformation and do not fully describe the mapping. We propose a supervised learning method to identify the most trustworthy set of constraints and the part of the surface on which the functional map is accurate. Another challenge of the operator representation is the problem of obtaining a point-to-point map from a functional map. The current methods [88, 102] do not ensure that resulting map is continuous and may lead to flaws in the map. To overcome this issue we restrict the conversion to continuous maps by using an adequate representation of vector fields as operators.

Shape Differences

Representing metric distortion can be challenging when dealing with heterogeneous data. The *shape difference* operators [106] take advantage of the functional map representation to provide a coordinate free encoding of the metric distortion up to isometric deformation. Those operators act linearly on functions and separate local area change and angle modification, even if those changes are noisy and not easy to define directly across different data structures, function correspondences induce a notion of smoothness up to a certain level-of-details.

Shape difference operators were originally presented as analysis tools since they can be used for comparison and composition of deformations across shapes. Interestingly the space of shape differences seems “flat” and well behaved. For instance in Figures 7.7 and 7.9 (see also Figure 3 in [106]) a collection of surfaces, represented as a collection

of operators, is mapped to a low dimensional space with a simple Principal Component Analysis. Yet the embedding accurately depicts the degrees of freedom of the deformation. This remark along with interesting algebraic properties lead to the intuition that operators are more convenient to describe deformations than Euclidean embeddings.

In this thesis we attempt to use the shape difference operators for deformation synthesis and for exploring the space of deformable surfaces. This only prior work by Boscaini and al. in [15] reconstructs an embedded surface from an operator representing the intrinsic structure of the shape. However, this approach is limited as it does not take into accounts extrinsic information. We explore different ways of encoding extrinsic curvature information as operators with theoretical guarantees of completeness.

Thesis Summary

This thesis focuses on processing deformable surfaces in the operator representation framework. The study is split in two parts coinciding with the analysis of two types of operators: functional maps and shape differences.

Shape to Deformation

The goal of this part is to find a mapping between two given shapes. We are in particular interested in finding correspondences between intrinsic structures. To make the problem more tractable we will assume a restrictive deformation model of nearly-isometric deformations. This question is closely related to the *shape matching* problem using only intrinsic features.

The shape matching problem for nearly-isometric deformation has been presented as the first application of the functional representation in [88] and is still the most active research direction in this field. One reason of this success is the description of a challenging problem as a least squares system. The initial formulation takes into account the isometric distortion and some given functional correspondences often obtained from descriptors (HKS, WKS) stable under mild deformation. However, the resulting map can suffer artifacts due to false or noisy correspondences and misleading spectral interpolation. Many variations have been proposed to improve the quality of the results [100, 64, 96], to allow partial matching [101] and to exploit the cycle consistency in map networks [54]

In Chapter 5 we take a slightly different approach: instead of modifying the original formulation we aspire to obtain the best results with respect to the input. The possibly unreliable functional constraints imply that the correspondences need to be sorted and weighted and that the resulting functional map is reliable only on a subspace of functions. Our main contribution is a supervised learning algorithm able to compute a set of weights jointly assessing the utility of each descriptor and identify the most accurately mapped functions.

Once a reliable functional map is computed, another challenge is to convert it to a classical point-to-point map used in most applications. The original article suggested a procedure that can be described as an *Iterative Closest Point* algorithm in the spectral domain. Although it has been improved upon by some follow-up work [102], major issues

remain: the existence of internal symmetries and the absence of global constraints linking nearby points lead to discontinuities in the resulting map. In Chapter 6 we provide a method to restrict the conversion process to continuous maps. While this problem is challenging with standard analysis tools, it becomes much more tractable with the operator representation. The outcome is a provably smooth conversion of a functional map using vector field flow.

Overall, the main contribution of this first part is a complete pipeline for shape matching problems in the functional map framework: from the computation of reliable maps to a final point-to-point correspondence.

While the first part is aimed at finding similarities across near-isometric shapes, in the second part, our goal is to quantify and manipulate differences in the form of non-isometric distortion. In particular we are interested in a complete representation of surfaces as operators.

Deformation to Shape

Shape differences enjoy many interesting algebraic properties. In particular they allow the transfer, composition and addition (in special cases) of deformations. Thus, exploring and creating new intrinsic deformation is relatively easy in this representation as it only requires matrix manipulations. The only missing step to produce embedded surfaces is a conversion from the functional characterization to a surface embedding. Namely, given a base shape and shape differences, find the deformed surface. This problem is closely related to recovering the shape from its Laplacian operator which is known to be possible theoretically but for which very few computational methods are available. This thesis attempts to fill this hole with two different approaches.

The first development in Chapter 7 starts with the remark that the discrete metric on triangle meshes (edge length) can be recovered from shapes differences by solving a set of linear equations. It follows that in the case of noisy and limited information the metric can be recovered through two convex optimization problems. However, intrinsic information alone is not sufficient to recover an embedded surface as multiple solutions exist. We investigate the possibility of using offset surfaces to encode curvatures. The outcome is a complete characterization of triangle meshes through functional operators allowing us to deform surfaces according to shape difference operators and a corresponding continuous theory.

The second development in Chapter 8 introduces a functional characterization of extrinsic vector fields. To do so we introduce a unique shape difference accountable for all intrinsic changes. Then we consider the operator associated to an infinitesimal displacement. This way deformation fields are characterized by the distortion they induce on the metric. Interestingly they enjoy similar properties as shape differences (composition, transfer) enabling significantly simpler deformation reconstruction compared to the previous method. We provide theoretical proofs of informativeness in both continuous and discrete cases.

This thesis constitutes a first step toward using functional representation for characterization of deformable surfaces and deformation synthesis.

Publications linked to this thesis

This thesis is mainly based on the publications listed below:

- E. CORMAN, M. OVSJANIKOV AND A. CHAMBOLLE, *Supervised descriptor learning for non-rigid shape matching*, in ECCV 2014 Workshops, Part IV, Springer International Publishing, 2014. [\[31\]](#)
- E. CORMAN, M. OVSJANIKOV AND A. CHAMBOLLE, *Continuous matching via vector field flow*, in Proceedings of the Eurographics Symposium on Geometry Processing, vol. 34, Eurographics Association, 2015, pp. 129–139. [\[32\]](#)
- E. CORMAN, S. SOLOMON, M. BEN-CHEN, L. GUIBAS AND M. OVSJANIKOV, *Functional Characterization of Intrinsic and Extrinsic Geometry*, currently under minor revision for Transaction On Graphics
- E. CORMAN AND M. OVSJANIKOV, *Functional Characterization of Deformation Fields*, to be submitted (2016)

Introduction en français

L’une des difficultés rencontrées dans le traitement de forme 3D est la large gamme de structures de données. Les formes 3D et autres géométries discrètes sont codées de diverses manières dépendant principalement de la méthode d’acquisition et du domaine d’application. Les données brutes provenant d’un scanner sont stockées sous forme de nuages de points souvent bruités ; les simulations physiques s’effectuent habituellement sur des complexes simpliciaux ; les modeleurs 3D ainsi que la communauté de la géométrie architecturale travaillent traditionnellement avec des maillages quadrangulaires. Cependant, de nombreux problèmes standards liés au traitement de la géométrie, tels que la mise en correspondance de formes, le transfert de déformations et l’analyse de déformations, doivent être résolus indépendamment de la représentation utilisée. La plupart du temps chaque communauté fournit ses propres solutions vis-à-vis de ses données de base qui, souvent, ne sont pas adaptables aux autres types de représentation. Cela nous conduit à la nécessité de définir un cadre d’analyse indépendant des structures sous-jacentes.

L’idée à l’origine de la représentation fonctionnelle, d’abord introduite dans [88], est de revenir au dénominateur commun dans l’analyse de la géométrie, à savoir les fonctions à valeurs réelles. Les fonctions définies sur la géométrie ont été historiquement utilisées et étudiées de manière approfondie dans de nombreux contextes fournissant une abondante littérature sur laquelle se reposer. Par exemple, les opérateurs différentiels ont été définis sur des maillages triangulaires et tétraédriques (Méthode des Eléments Finis [19], Discrete Exterior Calculus [52]), sur des maillages polygonaux [3] et des nuages de points (méthodes “sans maillage” basées sur des fonctions de base radiale ou Moving Least Squares [46]) pour n’en citer que quelques-uns. De plus, beaucoup de ces méthodes vont de paire avec une analyse de leur convergence et de leur stabilité. Par conséquent les opérateurs fonctionnels peuvent être mis en œuvre dans de nombreuses situations tout en se référant à une théorie unique. A titre d’exemple, dans [34] les auteurs proposent un algorithme pour calculer les distances géodésiques grâce à l’opérateur de Laplace-Beltrami. Ils obtiennent ainsi des résultats sur des structures de données radicalement différentes. Cette flexibilité est également l’une des clés du succès des méthodes spectrales et des descripteurs globaux tels que *Global Point Signature* [105], *Heat Kernel Signature* [119] et *Wave Kernel Signature* [5].

L’idée de base développée dans cette thèse est de représenter des outils standards pour les surfaces déformables (difféomorphismes, distorsions intrinsèques, champs de vecteurs tangents et extrinsèques) par des opérateurs agissant sur des fonctions, suivant l’intuition que leurs discrétisations ont été préalablement étudiées et sont facilement comparables quelles que soient les structures de données considérées. Le point de départ de cette thèse est la représentation fonctionnelle des difféomorphismes appelée *Application*

Fonctionnelle (ou “Functional Map” en anglais), et des distorsions intrinsèques nommées *Opérateurs de Déformation* (ou “Shape Differences” en anglais). Nous présentons dans le Chapitre 5 une nouvelle méthode, robuste aux bruits et aux grandes déformations, pour trouver des correspondances entre surfaces en utilisant une approche fonctionnelle. Les Chapitres 6, 7 et 8 fournissent des réponses différentes à un problème ouvert dans le cadre fonctionnel : la conversion de la représentation à base d’opérateurs vers une représentation ponctuelle. Plus précisément, étant donné une application fonctionnelle, nous proposons un algorithme pour la convertir en une affectation continue de sommet à sommet ou à partir d’un opérateur de différence de forme, nous proposons des solutions pour retrouver la surface déformée sous-jacente.

Applications fonctionnelles

L’espace des difféomorphismes est difficile à appréhender que ce soit dans le cadre continu ou discret. Considérons deux maillages triangulaires, la représentation la plus simple consisterait à attribuer à chaque sommet de la première forme un sommet de la seconde. Toutefois, lorsque l’on considère deux maillages avec un nombre différent de sommets le calcul de l’application inverse devient difficile. Une description plus complète est donc nécessaire. Une possibilité serait de considérer une application linéaire par morceaux permettant d’assigner des sommets à l’intérieur des triangles. De cette façon, il est possible de calculer le difféomorphisme et son inverse de façon cohérente. Une telle description discrète a été mise en œuvre notamment dans [2, 1] mais possède quelques inconvénients car elle ne peut-être stockée de manière compacte et conduit souvent à des problèmes d’optimisations non-convexes difficiles à appréhender.

Dans ce contexte, représenter l’application φ qui lie deux formes géométriques par l’opérateur de composition $f \mapsto f \circ \varphi$ fournit une alternative globalement indifférente au changement de connectivité et facilement définie sur différents types de données. Les applications fonctionnelles sont des représentations simplifiées et compactes des correspondances car elles sont souvent calculées entre des bases de fonctions de tailles réduites. Elles peuvent être stockées sous forme de matrices et analysées à l’aide d’outils d’algèbre linéaire standards. Les travaux qui ont suivis (par exemple [100, 101, 64, 96]) se concentrent sur des calculs d’applications fonctionnelles entre surfaces dont la mise en correspondance est inconnue. Pour ce faire ils minimisent différents types de distorsions intrinsèques auxquelles s’ajoutent des contraintes fonctionnelles. Cependant, ces contraintes fonctionnelles sont souvent peu fiables en cas de déformations importantes ou non-isométriques et ne décrivent les correspondances entre les surfaces que de façon incomplète. Nous proposons donc une méthode d’apprentissage supervisée pour identifier l’ensemble des contraintes le plus digne de confiance et la partie de la surface sur laquelle l’application fonctionnelle est la plus précise. Un autre défi de cette représentation fonctionnelle est le problème de l’obtention d’une application point-à-point à partir d’un opérateur. Les méthodes actuelles [88, 102] ne garantissent pas que l’application résultante soit continue et peuvent conduire à des défauts de continuité. Pour surmonter ces problèmes, nous limitons la conversion à des applications continues en utilisant une représentation adéquate des champs de vecteurs par des opérateurs.

Opérateurs de déformation

Représenter les distorsions de la métrique peut être difficile lorsqu'il s'agit de traiter des données de provenances variées. Les *opérateurs de déformation* [106] tirent parti de la représentation fonctionnelle pour fournir un encodage des distorsions intrinsèques faisant abstraction du plongement de la surface dans l'espace ambiant. Ces opérateurs agissent linéairement sur les fonctions et font la distinction entre les modifications de volumes et l'évolution des angles. Ces changements sont locaux, facilement perturbables et difficiles à définir correctement selon les différentes structures de données. Toutefois les opérateurs de déformation parviennent à les rendre stables et facilement analysables.

Les opérateurs de déformation ont été initialement présentés comme des outils d'analyse, car ils peuvent être utilisés pour comparer et composer des déformations. Il est intéressant de remarquer que l'espace généré par ces opérateurs semble "plat" et stable. Par exemple les Figures 7.7 et 7.9 (voir aussi Figure 3 dans [106]) représentent une collection de surfaces, interprétée comme une collection d'opérateurs, est envoyée dans un espace euclidien de petite dimension grâce à une simple analyse en composantes principales. Ce plongement décrit précisément les degrés de liberté des déformations. Cette remarque ainsi que des propriétés algébriques intéressantes conduisent à l'intuition que les opérateurs sont plus commodes pour décrire des déformations intrinsèques que l'analyse directe d'un plongement dans l'espace euclidien.

Dans cette thèse, nous tentons d'utiliser les opérateurs de déformation pour synthétiser et explorer l'espace des surfaces déformables. L'unique travail préalable par Boscaïni et al. dans [15] retrouve le plongement d'une surface à partir d'un opérateur représentant la structure intrinsèque de la forme. Cependant, cette approche est limitée car elle ne prend pas en compte les informations extrinsèques. Nous explorons différentes façons de stocker les informations données par la courbure en utilisant des opérateurs fonctionnels.

Résumé de la thèse

Cette thèse porte sur le traitement de surfaces déformables dans le cadre d'une représentation fonctionnelle. L'étude est divisée en deux parties qui coïncident avec l'analyse de deux types d'opérateurs : les applications fonctionnelles et les opérateurs de déformation.

Des surfaces aux déformations

Le but de cette partie est de trouver des correspondances entre deux formes données. Nous nous intéressons en particulier aux informations délivrées par la structure intrinsèque de ces formes. Pour rendre le problème plus abordable, nous nous plaçons dans un modèle restrictif en ne considérant que des déformations quasi-isométriques.

Trouver des correspondances entre formes 3D pour des déformations quasi-isométriques fut présentée comme la première application de la représentation fonctionnelle par [88] et est toujours la direction de recherche privilégiée dans ce domaine. L'une des raisons de ce succès étant que ce problème, difficile de prime abord, devient un système des moindres carrés dans cette nouvelle représentation. La formulation la plus basique prend

en compte la distance à l'isométrie et des correspondances fonctionnelles, souvent obtenues à partir de descripteurs globaux (HKS, WKS) restant stables même après de légères déformations. Cependant, l'application résultante peut avoir des artefacts dus à de mauvaises correspondances fonctionnelles, à des perturbations ou à une interpolation spectrale trompeuse. De nombreuses variantes ont été proposées pour améliorer la qualité des résultats [100, 64, 96]) afin de permettre des correspondances partielles [101] ou de tirer parti de cohérences cycliques dans des réseaux de difféomorphismes [54].

Dans le Chapitre 5 nous prenons une approche légèrement différente : au lieu de modifier la formulation initiale, nous tentons d'obtenir les résultats optimaux à partir des données initiales. Les contraintes fonctionnelles, potentiellement peu fiables, nous obligent à trier et à pondérer ces correspondances. Par ailleurs, l'application fonctionnelle résultante n'est fiable que sur un sous-espace de fonctions qu'il nous appartient d'identifier. Notre principale contribution est un algorithme d'apprentissage supervisé capable de calculer un ensemble de pondérations évaluant conjointement l'utilité de chaque descripteur et capable de déterminer les fonctions les plus fidèlement transférées par l'application.

Une fois calculées les applications fonctionnelles offrent un autre défi qui consiste à les convertir en une représentation ponctuelle, utilisée dans la plupart des applications. L'article original propose une méthode qui peut être comprise comme l'algorithme *Iterative Closest Point* appliqué au domaine spectral. Bien qu'il ait été amélioré par des articles postérieurs [102], un problème majeur demeure : l'existence de symétries internes et l'absence de contraintes globales reliant les points proches les uns des autres conduisent à des discontinuités dans l'application résultante. Dans le Chapitre 6 nous fournissons une méthode pour restreindre ce processus de conversion aux seules applications continues. Ce problème, difficile en utilisant des outils d'analyse standards, devient beaucoup plus abordable en utilisant une représentation fonctionnelle. Ce problème, difficile en utilisant des outils d'analyse standards, devient beaucoup plus abordable en utilisant une représentation fonctionnelle. Le résultat est la conversion d'une application fonctionnelle vers une application continue en utilisant le flot d'un champ de vecteurs.

Ainsi, la contribution principale de cette première partie est une procédure complète pour résoudre les problèmes de mise en correspondance de formes dans le cadre d'une représentation fonctionnelle : du calcul d'une application fonctionnelle fiable à une correspondance ponctuelle entre surfaces.

Alors que la première partie vise à trouver des similarités entre des formes quasi-isométriques, dans la deuxième partie notre objectif est de quantifier et de manipuler des distorsions non-isométriques de la métrique. En particulier, nous nous intéressons à une représentation complète des surfaces par des opérateurs fonctionnels.

Synthèse de formes et représentation fonctionnelle

Les opérateurs de déformation jouissent de nombreuses propriétés algébriques intéressantes. En particulier, ils permettent le transfert, la composition et l'addition (dans des cas particuliers) de déformations. Ainsi, l'exploration et la création de nouvelles déformations intrinsèques sont aisées dans cette représentation car elles ne nécessitent que des manipulations de matrices. L'unique étape manquante est la conversion d'une

caractérisation fonctionnelle des déformations en un plongement dans l'espace euclidien. Plus précisément, étant donné une forme de base et un opérateur de déformations, il nous faut retrouver la surface déformée. Ce problème est étroitement lié à la reconstruction d'une forme à partir de son laplacien, connu pour être théoriquement possible bien que peu d'algorithmes soient actuellement disponibles pour le résoudre. Cette thèse tente de combler ce vide par deux approches différentes.

Le premier développement, présent dans le Chapitre 7, commence avec la remarque que la métrique discrète d'un maillage triangulaire (longueur des arêtes) peut être récupérée à partir des opérateurs de déformations, et ce, en résolvant un système d'équations linéaires. Il en résulte que dans le cas d'informations bruitées et limitées la métrique peut être récupérée grâce à deux problèmes d'optimisations convexes. Cependant, l'information intrinsèque n'est pas suffisante pour obtenir le plongement d'une surface car il existe plusieurs solutions à ce problème. Dans ce Chapitre nous étudions donc la possibilité d'utiliser des surfaces parallèles pour encoder la courbure. Il en ressort une caractérisation complète des maillages triangulaires par des opérateurs fonctionnels ainsi que des garanties théoriques de reconstruction.

Le deuxième développement inclut dans le Chapitre 8 introduit une caractérisation fonctionnelle des champs de vecteurs extrinsèques. Pour cela, nous introduisons un opérateur de déformation unique, responsable de tous les changements de la métrique. Ensuite, nous considérons l'opérateur associé à un déplacement infinitésimal. De cette façon, les champs de déformation sont caractérisés par la distorsion qu'ils induisent sur la métrique. Ils jouissent de propriétés similaires aux opérateurs de déformations (composition, transfert), tout en permettant une reconstruction des déformations nettement plus simple que la méthode précédente.

Cette thèse constitue une première étape vers l'utilisation de la représentation fonctionnelle pour la caractérisation des surfaces déformables et la synthèse de déformation.

Publications liées à cette thèse

Cette thèse est en partie basée sur les publications suivantes :

- E. CORMAN, M. OVSJANIKOV AND A. CHAMBOLLE, *Supervised descriptor learning for non-rigid shape matching*, in ECCV 2014 Workshops, Part IV, Springer International Publishing, 2014. [31]
- E. CORMAN, M. OVSJANIKOV AND A. CHAMBOLLE, *Continuous matching via vector field flow*, in Proceedings of the Eurographics Symposium on Geometry Processing, vol. 34, Eurographics Association, 2015, pp. 129–139. [32]
- E. CORMAN, S. SOLOMON, M. BEN-CHEN, L. GUIBAS AND M. OVSJANIKOV, *Functional Characterization of Intrinsic and Extrinsic Geometry*, currently under minor revision for Transaction On Graphics
- E. CORMAN AND M. OVSJANIKOV, *Functional Characterization of Deformation Fields*, to be submitted (2016)

Context in Differential Geometry and its Discrete Equivalents

This chapter introduces the main mathematical tools that will be used throughout the thesis. A brief overview of differential geometry focused on intrinsic geometry is provided. We put emphasis on the role of the metric tensor and the pullback metric in the definitions of differential operators. This chapter also provides a brief remainder of the Finite Element Method applied to the discretization of common differential operators. This chapter can be skipped by readers familiar with the continuous theory.

3.1 Differential Geometry

The study of surface deformations is central in this thesis. Therefore some notions and terminology from differential geometry are recalled here. The aim of this section is to introduce the notions of metric tensor and more importantly of pullback metric by a diffeomorphism. The pullback metric accounts for the intrinsic distortion underlying a mapping between shapes. This will be a key tool for our analysis of surface deformations as it allows a classification of mappings according to their impact on the local structure.

To represent diffeomorphisms and distortions as operators on functions, we will need some basic differential operators such as gradients and Laplacians. We will focus on the change of variables which is equivalent to a change of metric in differential operators. This remark allows us to construct operators provably informative of the intrinsic structure and explain the leading role of the Laplace-Beltrami operator in geometry processing.

This basic introduction to differential geometry is greatly inspired by [78, 103]. The Einstein summation convention is sometimes used to lighten the notations. Repeated indices imply a summation, i.e. $\mathbf{g}_{ij}f_j$ means $\sum_j \mathbf{g}_{ij}f_j$. Moreover when dealing with tensors upper indices denote an element of the inverse matrix $\mathbf{A}^{ij} = (\mathbf{A}^{-1})_{ij}$.

3.1.1 Manifold

In practice we will most of the time deal with surfaces in \mathbb{R}^3 . However, operations on intrinsic informations (e.g. transport, composition of distortion) may lead to metric tensors which do not correspond to embedded surfaces and are only meaningful for general manifolds. Thus, it is important to start with abstract definitions.

A topological manifold M is a space such that each point has a neighborhood diffeomorphic to \mathbb{R}^n . To introduce differential calculus, it is necessary to always refer to the Euclidean space where derivatives are well-defined. To do so we introduce a

local parametrization of M called an *atlas*. An atlas on M is a countable collection of pairs (U_α, ψ_α) named *charts* where U_α are sets covering M and $\psi_\alpha : U_\alpha \rightarrow \mathbb{R}^n$ are homeomorphisms acting as local parametrizations.

Since the sets U_α cover the entire manifold some points maybe found in two different charts. If U_α and U_β overlap then one can change the coordinate system by means of the *transition map*:

$$\psi_{\beta\alpha} = \psi_\beta \circ \psi_\alpha^{-1} : \psi_\alpha(U_\alpha \cap U_\beta) \rightarrow \psi_\beta(U_\alpha \cap U_\beta).$$

Note that $\psi_{\beta\alpha}$ maps an open set of \mathbb{R}^n to an open set of \mathbb{R}^n , so at a point p in $U_\alpha \cap U_\beta$ one can decompose this mapping in terms of coordinates $(x_1(p), \dots, x_n(p))$ with respect to U_α and $(y_1(p), \dots, y_n(p))$ with respect to U_β . Two systems of coordinates are linked by:

$$y_i(p) = \psi_{\beta\alpha}(x_1(p), \dots, x_n(p))_i.$$

A transition map and two coordinate systems are represented in Figure 3.1.

This formulation is convenient as it allows us to use standard differential calculus in \mathbb{R}^n to study the properties of a curved surface. In particular the Jacobian matrix of $\psi_{\beta\alpha}$ will appear in coordinate change for vector fields (Section 3.1.3).

The charts introduce a local structure on the manifold. However, to have a differential manifold we need to ensure that the local properties are consistent among themselves.

Definition 3.1.1 A topological manifold M equipped with an atlas $\mathcal{S} = (U_\alpha, \psi_\alpha)$ is called *differential manifold* if all its coordinate changes $\psi_{\beta\alpha}$ are \mathcal{C}^∞ maps.

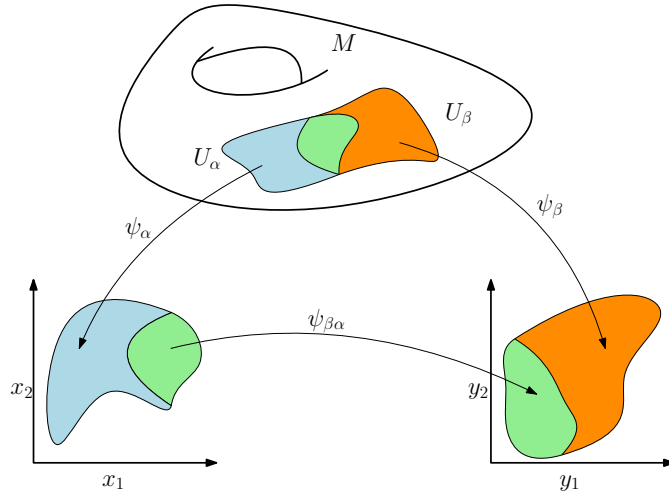


Figure 3.1 – Two charts and a transition map on a manifold M .

3.1.2 Tangent Vectors and Local Coordinates

Tangent vectors are a first step to introduce local intrinsic structure on a manifold. Let $f : M \rightarrow \mathbb{R}$ be a real valued function on a differential manifold. If, for all coordinate systems (U, ψ) in an atlas, $f \circ \psi^{-1} : \psi(U) \subset \mathbb{R}^n \rightarrow \mathbb{R}$ is a \mathcal{C}^∞ function on $\psi(U)$ then f is by definition a \mathcal{C}^∞ function on M .

Now let (U, ψ) be a coordinate system around a point $p \in M$ and (x_1, \dots, x_n) the coordinate functions with respect to U . The derivative of f in the direction x_i denoted $\partial_i f$ is naturally defined as:

$$\partial_i f = \frac{\partial(f \circ \psi^{-1})}{\partial x_i} \circ \psi \in \mathcal{C}^\infty(M).$$

In case of a surface in \mathbb{R}^3 , the gradient of a function is tangent to this surface. Moreover, the set of all possible directions for function derivatives defines the set of all tangent vectors. By analogy, the application $V_p : \mathcal{C}^\infty(M) \rightarrow \mathbb{R}$ acting on differentiable functions at point $p \in M$:

$$V_p(f) = \sum_{i=1}^n V_i(p) \partial_i f|_p, \quad (3.1)$$

is understood as a vector tangent to the manifold. The *local coordinates* of the vector $V_i(p) \in \mathbb{R}$ depend on the chart (U, ψ) and correspond to the derivative in the canonical directions $\left. \frac{\partial}{\partial x_i} \right|_p$. We denote $T_p M$, the set of all tangent vectors at point p , defined as the set of all linear combinations of the basis $\left\{ \left. \frac{\partial}{\partial x_i} \right|_p \right\}_i$. The canonical basis satisfies the property $\frac{\partial x_j}{\partial x_i} = \delta_{ij}$.

A vector field on M is a smooth assignment of a tangent vectors $V_p \in T_p M$ at each point p . Namely V is a \mathcal{C}^∞ vector field if:

$$V(f) \in \mathcal{C}^\infty(M), \quad \forall f \in \mathcal{C}^\infty(M).$$

We denote the set of smooth vector fields $\mathcal{V}(M)$.

3.1.3 Mappings

Along this thesis we will study relations between manifolds through mappings. Let M and N be two differential manifolds, $\varphi : N \rightarrow M$ is a map between them. Since the map is not real-valued we have to take into account the parametrizations from M and N to introduce a differential structure.

Definition 3.1.2 *The map $\varphi : N \rightarrow M$ is a \mathcal{C}^∞ map if for all charts (U, ψ) on N and (V, ϑ) on M , the functions $\vartheta \circ \varphi \circ \psi^{-1} : \mathbb{R}^n \rightarrow \mathbb{R}^n$ are of class $\mathcal{C}^\infty(\mathbb{R}^n)$.*

Local coordinate systems $(x_1(p), \dots, x_n(p))$ around $p \in N$ and $(y_1(\varphi(p)), \dots, y_n(\varphi(p)))$ around $q = \varphi(p) \in M$ are linked by the mapping φ :

$$y_i(\varphi(p)) = \vartheta \circ \varphi \circ \psi^{-1}(x_1(p), \dots, x_n(p))_i. \quad (3.2)$$

Differential map

A \mathcal{C}^∞ map not only describes pointwise correspondences but also map tangent vectors from one surface to another. The distortion of a tangent space is determined by the Jacobian of the mapping $d\varphi_p$.

Using the chain rules for derivatives, Equation (3.2) yields a change of coordinates formula or *pushforward* of tangent vector fields:

$$d\varphi_p \left(\frac{\partial}{\partial x_i} \right) = \sum_{j=1}^n \frac{\partial y_j}{\partial x_i} \frac{\partial}{\partial y_j}, \quad (3.3)$$

where $d\varphi$ denote the differential map or Jacobian of the transformation. Therefore, a map not only relates points but also tangent spaces. The differential map defines a linear mapping between the tangent space on N at point p and the tangent space on M at point $\varphi(p)$:

$$d\varphi_p : T_p N \rightarrow T_{\varphi(p)} M.$$

Categories of mappings

Mappings are categorized according to the properties of the differential map:

- *Immersion*: At each point $p \in N$ the Jacobian is injective. It implies that the image of the manifold $\varphi(N)$ can intersect itself so it may not be a submanifold of M .
- *Embedding*: φ is immersion and also a homeomorphism from N onto $\varphi(N)$. In this case no self-intersection can occur and the image $\varphi(N)$ is a submanifold of M .
- *Diffeomorphism*: $\varphi : N \rightarrow M$ is a bijection and its inverse $\varphi^{-1} : M \rightarrow N$ is a \mathcal{C}^∞ map. These requirements are stronger than for embeddings, in particular it implies that the Jacobian is bijective and that N, M have the same intrinsic dimension.

3.1.4 Riemannian Manifold

Metric tensor

The metric tensor extends the notion of scalar product in \mathbb{R}^n to tangent vectors. It is important to recall that general metric tensors can be arbitrarily chosen and are not necessarily induced by the embedding of a surface. The choice of the metric determines the specifying of the tangent plane, thus, specifying distances on the manifold and many differential operators.

Definition 3.1.3 A Riemannian metric is a family of bilinear forms defined at each point p of the differential manifold M by:

$$\mathbf{g}_p : T_p M \times T_p M \rightarrow \mathbb{R}.$$

As Euclidean scalar products the metric should be symmetric, positive definite at each point. Moreover, a Riemannian metric varies smoothly on the manifold: for all differentiable vector fields V, U on M , $p \mapsto \mathbf{g}_p(V_p, U_p)$ is a smooth function from M to \mathbb{R} .

Given a system of local coordinates (x_1, \dots, x_n) around p , the set $\left(\frac{\partial}{\partial x_1}, \dots, \frac{\partial}{\partial x_n}\right)$ is a basis of the tangent space $T_p M$. So, the metric can be written as a matrix of $\mathbb{R}^{n \times n}$:

$$\mathbf{g}_{ij}(p) := \mathbf{g}_p \left(\frac{\partial}{\partial x_i} \Big|_p, \frac{\partial}{\partial x_j} \Big|_p \right).$$

Thus, the metric tensor can be interpreted as a Gramian matrix of tangent vectors. Geometrically the square root of the determinant is the volume of the parallelotope defined by the basis $\frac{\partial}{\partial x_i}$. Therefore, the quantity $\sqrt{\det(\mathbf{g})}$ can be understood as a local volume or area on the manifold.

Given two tangent vectors $V, U \in T_p M$ their scalar product is computed as a usual bilinear form within the coordinate system:

$$\mathbf{g}_p(V, U) = \sum_{ij} V_i \mathbf{g}_{ij} U_j.$$

Surface in \mathbb{R}^3

An important example that will be frequently used in this thesis is the case where M is an embedded surface of \mathbb{R}^3 . In this configuration, the ambient Euclidean scalar product $\langle \cdot, \cdot \rangle$ naturally induces a metric on the surface. Given a chart (U, ψ) in the neighborhood of $p \in M$, the mapping $F = \psi^{-1} : \psi(U) \subset \mathbb{R}^2 \rightarrow U \subset M \subset \mathbb{R}^3$ maps an open subset of \mathbb{R}^2 to an open subset of \mathbb{R}^3 . Therefore $\frac{\partial F}{\partial x_i}(p)$ is a vector of \mathbb{R}^3 tangent to the surface at p and can be identified with a basis of $T_p M$. The induced metric in this basis is:

$$\mathbf{g}_{ij}(p) = \left\langle \frac{\partial F}{\partial x_i}(p), \frac{\partial F}{\partial x_j}(p) \right\rangle. \quad (3.4)$$

Many local coordinates expressions can be replaced by vectors in the ambient space. For instance the scalar product $\mathbf{g}(V, U)$ can be replaced by $\langle \bar{V}, \bar{U} \rangle$ where \bar{V} is the tangent vector V expressed in the global coordinate system. To avoid heavy notations we will consider the change of coordinates implicit when using the ambient scalar product.

Geodesic distance

The metric is a local quantity determining lengths of vectors. It induces a global structure on the manifold enabling us to measure distances between points. The geodesic distance is defined as the shortest path between two points on a manifold.

The application $c : [0, 1] \rightarrow M$ is a parametrized curve on M . Its velocity c' is a tangent vector whose length can be computed with the metric \mathbf{g} . The arc length $L(c)$ of the curve c is defined by similarity with the Euclidean case:

$$L(c) := \int_0^1 \sqrt{\mathbf{g}_{c(t)}(c'(t), c'(t))} dt.$$

The geodesic distance $d : M \times M \rightarrow [0, +\infty)$ is the minimum over all curves on M :

$$d(p, q) := \min_{c: [0,1] \rightarrow M} L(c) \quad \text{subject to } c(0) = p, c(1) = q.$$

Figure 3.2 shows an example of geodesic distance field on a surface.



Figure 3.2 – Isolines of the geodesic distance, computed with the spectral method [34].

The pullback metric

The pullback metric is a key tool for analyzing metric distortion. Namely, given an immersion $\varphi : N \rightarrow M$ between the differential manifolds M and N , we are able to express \mathbf{g}^M , the metric tensor on M , as a distorted metric on N . This new tensor encodes the distortion of the tangent space by the underlying deformation φ .

Definition 3.1.4 *The pullback of a metric on M , denoted $\varphi^* \mathbf{g}^M$, is defined as the action of the tensor on the pushforward tangent vectors:*

$$(\varphi^* \mathbf{g}^M)_p(V, U) = \mathbf{g}_{\varphi(p)}^M(d\varphi(V), d\varphi(U)), \quad \forall V, U \in T_p N.$$

Given a coordinate system, Equation (3.3) leads to a local expression of the pullback metric:

$$(\varphi^* \mathbf{g}^M)_{ij}(p) = \frac{\partial y_k}{\partial x_i} \mathbf{g}_{kl}^M(\varphi(p)) \frac{\partial y_l}{\partial x_j}.$$

The Jacobian of the immersion distorts the bilinear form in a symmetric way. If φ is an immersion, the tensor $\varphi^* \mathbf{g}^M$ is a metric on N .

Three types of mappings are defined according to the intrinsic properties they preserve (local areas, angles, geodesic distances). Figure 3.3 shows examples of deformation of the plane.

- **Area preserving map.** A diffeomorphism is said to be area preserving if local areas are preserved:

$$\det(\varphi^* \mathbf{g}^M)_p = \det(\mathbf{g}^N)_p, \quad \forall p \in N.$$

- **Conformal map.** A diffeomorphism $\varphi : N \rightarrow M$ is said to be conformal if there exists a positive function $h \in \mathcal{C}^\infty(M)$, sometimes called the conformal factor, such that:

$$h(q)(\varphi^* \mathbf{g}^M)_p = \mathbf{g}_p^N \quad \forall p \in N.$$

Such deformations preserve angles between curves but affect local areas. A classical example of such deformation are the Möbius transformations which span all the bijective conformal maps from the sphere to itself.

- **Isometry.** A conformal and area preserving diffeomorphism is an isometry. In this case, the pullback metric agrees with the metric on N :

$$(\varphi^* \mathbf{g}^M)_p = \mathbf{g}_p^N \quad \forall p \in N.$$

A map is an isometry if and only if geodesic distances are preserved, namely $d^M(\varphi(p), \varphi(q)) = d^N(p, q)$, as proven by the Myers–Steenrod theorem [83].

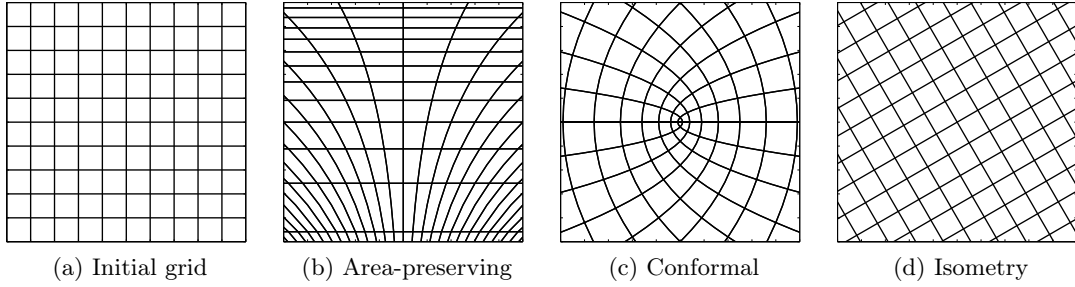


Figure 3.3 – Three categories of mapping according to the intrinsic property they preserve.

3.1.5 Integral on Manifold

Area preserving maps are naturally related to integral on manifold. In fact, a change of variables in integral is equivalent to using the measure induced by the pullback metric [78].

Volume form

First, we introduce the volume form $d\mu_{\mathbf{g}}$ defined locally by: $d\mu_{\mathbf{g}} = \sqrt{\det(\mathbf{g})} dx_1 \wedge \cdots \wedge dx_n$. The volume form induces a Borel measure $\mu_{\mathbf{g}}$:

$$\mu_{\mathbf{g}}(U) = \int_U d\mu_{\mathbf{g}}, \quad \forall U \subset M.$$

Note that μ will abusively stand for the local area $\sqrt{\det(\mathbf{g})}$ instead of the measure.

Given a chart (U_α, ψ_α) and a coordinate system (x_1, \dots, x_n) the integral is then defined through its mapping to the Euclidean space:

$$\int_{U_\alpha} f(p) \, d\mu_{\mathbf{g}}(p) = \int_{\psi_\alpha(U_\alpha)} (f \sqrt{\det(\mathbf{g})}) \circ \psi_\alpha^{-1}(x_1, \dots, x_n) \, dx_1 \dots dx_n.$$

Pushforward measure

Let $\varphi : N \rightarrow M$ be a diffeomorphism between two manifolds and \mathbf{g}^N a metric on N . The pushforward measure $\varphi_*\mu_{\mathbf{g}^N}$ is a measure on M standing for the change of variables:

$$\int_M f \, d(\varphi_*\mu_{\mathbf{g}^N}) := \int_N f \circ \varphi \, d\mu_{\mathbf{g}^N}.$$

Applying the change of coordinate formula for volume form [78] leads to Proposition 3.1.5.

Proposition 3.1.5 *The volume form induced by the pushforward measure is the volume form corresponding to the pullback metric:*

$$d(\varphi_*\mu_{\mathbf{g}^N}) = d\mu_{(\varphi^{-1})^*\mathbf{g}^N}.$$

When φ is area preserving, integrals are left untouched by a change of variables:

$$\int_M f \, d\mu_{\mathbf{g}^M} = \int_N f \circ \varphi \, d\mu_{\mathbf{g}^N}, \quad \forall f \text{ integrable.}$$

Inner product

Let $f, g : M \rightarrow \mathbb{R}$ be smooth functions on M . The space of square integrable functions with respect to the volume form $d\mu_{\mathbf{g}}$ is defined as:

$$L^2(M, \mathbf{g}) = \left\{ f : M \rightarrow \mathbb{R} \text{ integrable} : \int_M f^2 \, d\mu_{\mathbf{g}} < \infty \right\}.$$

This space is equipped with the inner product $\langle f, g \rangle_{L^2}^{(M, \mathbf{g})} := \int_M fg \, d\mu_{\mathbf{g}}$ inducing the norm $\|f\|_{L^2}^{(M, \mathbf{g})} := \left(\langle f, f \rangle_{L^2}^{(M, \mathbf{g})} \right)^{\frac{1}{2}}$. Mention of the metric and the manifold is sometimes omitted to avoid heavy notation

3.1.6 Gradient, Divergence and Laplacian

Gradient, divergence and Laplacian are three fundamental differential operators to study intrinsic geometry. Their definitions entirely rely on the underlying metric tensor. Moreover, as we will see later, they also fully specify the metric structure of the manifold. For this reason, the Laplacian is a key tool in geometry processing and computer graphics.

In particular it has been used for: fairing and compression (e.g. [59]), geodesic computation (e.g. [34]) and global descriptors (e.g. [99, 119, 5]) to name just few applications. Two properties explain this success: its invariance under isometric deformation and its eigenfunctions comparable to a Fourier basis on manifolds.

For a more comprehensive study of the Laplace-Beltrami operator in differential geometry we refer the interested reader to [103].

Laplace-Beltrami operator

The directional derivative $V(f)$ is the action of a tangent vector on a function. It can also be thought of as the scalar product between a fixed vector and the function gradient. Thus, the gradient can be defined as the tangent vector satisfying:

$$V(f) = \mathbf{g}(V, \nabla f), \quad \forall V \in \mathcal{V}(M).$$

Given a local coordinate system we can write $V_i \partial_i f = V_i \mathbf{g}_{ij} \mathbf{g}^{jk} \partial_k f$, so the components of the gradient are given by: $(\nabla f)_i = \mathbf{g}^{ij} \partial_j f$.

The gradient depends on the inverse metric tensor. As our main topic is intrinsic geometry, it is interesting to relate the pushforward gradient with the pullback metric.

Lemma 3.1.6 *Let $\varphi : N \rightarrow M$ be a diffeomorphism between two differential manifolds and \mathbf{g}^N a metric on N . The pushforward of the gradient is equal to the gradient defined by the pullback metric:*

$$d\varphi_p (\nabla_{\mathbf{g}^N} (f \circ \varphi)) = \nabla_{(\varphi^{-1})^* \mathbf{g}^N} f(\varphi(p)).$$

Proof Given local coordinates (x_1, \dots, x_n) around $p \in N$ and (y_1, \dots, y_n) around $q = \varphi(p) \in M$, a direct computation yields:

$$\begin{aligned} d\varphi_p (\nabla_{\mathbf{g}^N} (f \circ \varphi)) &= d\varphi_p \left(\mathbf{g}_N^{ij} \frac{\partial}{\partial x_j} (f \circ \varphi) \frac{\partial}{\partial x_i} \right) \\ &= \mathbf{g}_N^{ij} \frac{\partial}{\partial x_j} (f \circ \varphi) \frac{\partial y_k}{\partial x_i} \frac{\partial}{\partial y_k} \\ &= \mathbf{g}_N^{ij} (\varphi^{-1}(q)) \frac{\partial y_l}{\partial x_j} \frac{\partial f}{\partial y_l} (q) \frac{\partial y_k}{\partial x_i} \frac{\partial}{\partial y_k} \\ &= ((\varphi^{-1})^* \mathbf{g}^N)^{ij} (q) \frac{\partial f}{\partial y_j} (q) \frac{\partial}{\partial y_i} \\ &= \nabla_{(\varphi^{-1})^* \mathbf{g}^N} f(q). \end{aligned}$$

□

This simple lemma expresses the idea that instead of using the differential map to transfer gradients of functions from N to M , one can transfer intrinsic information

and then compute gradients locally with respect to this metric. Computing a the differential map may be challenging when considering meshes with different triangulation but transferring metrics, as we will see later, is more tractable since it is can be done by pullback of the Laplacian.

For manifolds without boundaries the Laplacian can be defined through the integration by parts identity:

$$\int_M \mathbf{g}(\nabla g, \nabla f) d\mu_{\mathbf{g}} = - \int_M g \Delta f d\mu_{\mathbf{g}}. \quad (3.5)$$

The Laplacian is therefore a symmetric and negative definite operator on smooth functions. The expression in local coordinates relies heavily on the metric:

$$\Delta f = \frac{1}{\sqrt{\det(\mathbf{g})}} \partial_i \left(\sqrt{\det(\mathbf{g})} \mathbf{g}^{ij} \partial_j f \right).$$

Eigen-decomposition

The Laplace-Beltrami operator has a countable set of eigenfunctions:

$$\Delta_{\mathbf{g}} \phi_i = \lambda_i \phi_i,$$

where the eigenvalues are negative and decreasing $0 = \lambda_0 \geq \lambda_1 \geq \dots \geq \lambda_i$. Moreover, if M is a smooth manifold the eigenfunctions are infinitely differentiable functions. Moreover they provide an orthonormal basis of $L^2(M, \mathbf{g})$ ordered from low to high frequency, as the total integral of the gradient equals the absolute value of the eigenvalue. Meaning that if $\int_M f^2 d\mu_{\mathbf{g}} < \infty$ then f can be written has a linear combination of the functions $\{\phi_i\}_{i \geq 0}$:

$$f = \lim_{k \rightarrow +\infty} \sum_{i=0}^k \langle f, \phi_i \rangle_{L^2}^{(M, \mathbf{g})} \phi_i.$$

In the computer graphics literature, the basis $\{\phi_i\}_{i \geq 0}$ is often understood as a generalization of Fourier basis on Manifolds. An example is shown in Figure 3.4. They are sometimes referred to as *manifolds harmonics* [121].

Commutativity with isometries

The Laplacian heavily depends on the metric. Like gradients, change of variables in Laplace-Beltrami operators is equivalent to changing the metric tensor.

Lemma 3.1.7 *Let $\varphi : N \rightarrow M$ be a diffeomorphism between manifolds and \mathbf{g} a metric on N , then:*

$$\Delta_{\mathbf{g}}(f \circ \varphi) = (\Delta_{(\varphi^{-1})^* \mathbf{g}} f) \circ \varphi, \quad \forall f \in \mathcal{C}^\infty(M).$$

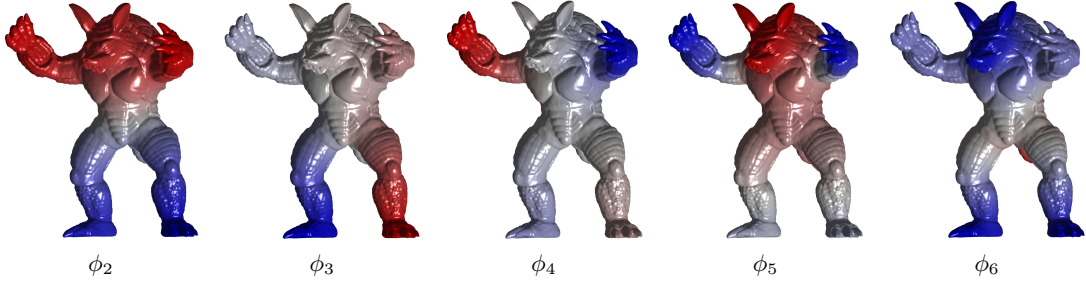


Figure 3.4 – Example of Laplace-Beltrami eigenfunctions computed on a surface. The first eigenfunction, not represented here, is constant.

Proof To lighten the notations we denote \mathbf{g}^* the pullback metric $(\varphi^{-1})^*\mathbf{g}$. Given the local coordinates (x_1, \dots, x_n) around $p \in N$ and (y_1, \dots, y_n) around $q = \varphi(p) \in M$, the Laplace-Beltrami operator can be transported from N to M using Lemma 3.1.6:

$$\begin{aligned}
 \int_N g \circ \varphi \Delta_{\mathbf{g}}(f \circ \varphi) d\mu_{\mathbf{g}} &= - \int_N \mathbf{g}_p(\nabla_{\mathbf{g}}(g \circ \varphi), \nabla_{\mathbf{g}}(f \circ \varphi)) d\mu_{\mathbf{g}}(p) \\
 &= - \int_N \mathbf{g}_p(d\varphi_{\varphi(p)}^{-1}(\nabla_{\mathbf{g}^*}g), d\varphi_{\varphi(p)}^{-1}(\nabla_{\mathbf{g}^*}f)) d\mu_{\mathbf{g}}(p) \\
 &= - \int_M \mathbf{g}_{\varphi^{-1}(q)}(d\varphi_q^{-1}(\nabla_{\mathbf{g}^*}g), d\varphi_q^{-1}(\nabla_{\mathbf{g}^*}f)) d\mu_{\mathbf{g}^*}(q) \\
 &= - \int_M \mathbf{g}^*(\nabla_{\mathbf{g}^*}g, \nabla_{\mathbf{g}^*}f) d\mu_{\mathbf{g}^*} \\
 &= \int_M g \Delta_{\mathbf{g}^*}f d\mu_{\mathbf{g}^*} \\
 &= \int_N g \circ \varphi (\Delta_{\mathbf{g}^*}f) \circ \varphi d\mu_{\mathbf{g}}.
 \end{aligned}$$

The fundamental lemma of calculus of variations leads to the equality between the functions. \square

As a consequence of Lemma 3.1.7, the Laplace-Beltrami operator commutes with isometries. Moreover, the converse also holds: the Laplacian determines the metric up to isometries. This property is well-known in the computer graphics community and makes this operator very attractive for shape analysis. For instance, Proposition 3.1.8 ensures that the Heat Kernel Signature [119] and Wave Kernel Signature [5] fully encode intrinsic information.

Proposition 3.1.8 *Let $\varphi : N \rightarrow M$ be a bijection between manifolds then it is an isometry if and only if:*

$$\Delta_{\mathbf{g}^N}(f \circ \varphi) = (\Delta_{\mathbf{g}^M}f) \circ \varphi, \quad \forall f \in \mathcal{C}^\infty(N).$$

Proof We prove the necessary and sufficient conditions separately:

- Assuming that φ is an isometry and using Lemma 3.1.7 lead to the equalities $\Delta_{\mathbf{g}^M} f = \Delta_{(\varphi^{-1})^* \mathbf{g}^N} f = (\Delta_{\mathbf{g}^N} (f \circ \varphi)) \circ \varphi^{-1}$.
- Now suppose that the Laplacians commute with the map φ . The operators on both manifolds are therefore isospectral and the eigenfunctions of one operator are eigenfunctions of the other after composition with φ . It follows that the heat kernel [103], defined as $h_t(p, q) = \sum_{i \geq 0} \exp(t\lambda_i) \phi_i(p) \phi_i(q)$, is preserved by the map:

$$h_t^N(p, q) = h_t^M(\varphi(p), \varphi(q)), \quad \forall p, q \in N, \forall t > 0.$$

This equality holds true if and only if φ is an isometry, as detailed in [119].

□

3.2 Discretization

To be defined correctly, differential operators require smooth surfaces. Obviously, in a discrete setting smoothness has to be redefined since most of the data (scanner, MRI) are point clouds or meshes. We will limit our theoretical study to triangular meshes. Although in practice the *functional map* framework is stable across data representations, all of the results presented in this thesis assume that the data comes in the form of triangle meshes. Therefore, in this section we show how differential operators are approximated using the standard Finite Element Method.

3.2.1 Discrete Manifold

Triangle meshes are composed of three sets $(\mathcal{X}, \mathcal{E}, \mathcal{F})$ creating a *discrete manifold*. The set of *vertices* \mathcal{X} whose elements $x_i \in \mathbb{R}^n$ are points in space. The set of *edges* \mathcal{E} linking two vertices together forming an undirected graph. If a pair (i, j) belongs to \mathcal{E} then the edge vector is denoted $e_{ij} = x_j - x_i$ where the order of the indices define the orientation. We require an additional structural element that is the graph should be composed only by triangles such that an edge is shared by only two triangles. A *triangle* is defined by the indices of three vertices (i, j, k) belonging to the set \mathcal{F} . Each side consists of an edge, so $\{(i, j), (j, k), (k, i)\}$ belongs to \mathcal{E} . The notations are summarized in Figure 3.5a.

Meshes are thought of a piecewise linear approximations of a smooth manifold. When refined by adding sample points, the approximation error should decrease and eventually go to zero. On such discrete structure it seems natural to consider only piecewise linear functions. So, a discrete function f is defined as an assignment of a scalar value per vertex $f : \mathcal{X} \rightarrow \mathbb{R}$ and extended linearly inside triangles. For a point p inside the triangle (x_i, x_j, x_k) the interpolation is expressed using the *hat basis* composed of the functions B_i defined in Figure 3.5b:

$$f(p) = f_i B_i(p) + f_j B_j(p) + f_k B_k(p).$$

The measure $\mu(T)$ denotes the area of triangle T and $\ell_{ij} = \|x_i - x_j\|$ the edge length. These two quantities will be used to describe the intrinsic structure of a discrete manifold.

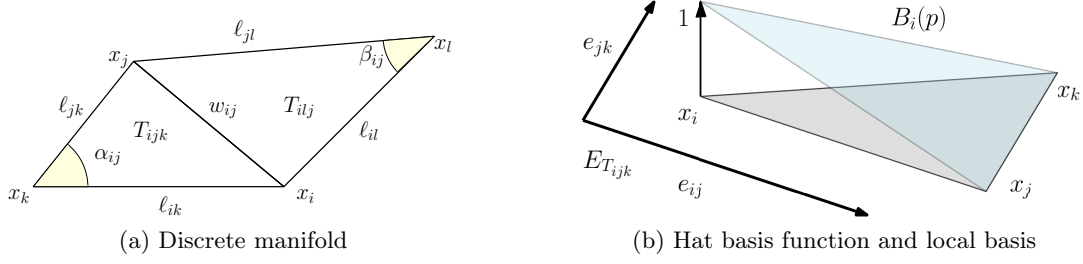


Figure 3.5 – Notations used for the Finite Element Method.

3.2.2 Finite Element Method and Cotangent Weight Formula

For a short but efficient introduction to finite elements for computer graphics the reader may enjoy [16] Chapter 3.

Inner product

Once the interpolation is written with the hat basis we can define the discrete inner product $\langle \cdot, \cdot \rangle_{L^2}$ by computing the integration of piecewise linear functions. Functions on vertices are elements of $\mathbb{R}^{\mathcal{X}}$ so the inner product is a symmetric definite matrix A , called *mass matrix*, of size $|\mathcal{X}| \times |\mathcal{X}|$. The element A_{ij} is then the inner product between B_i and B_j :

$$A_{ij} = \sum_{T \in \mathcal{F}} \int_T B_i(p) B_j(p) dp.$$

For computational efficiency it is sometimes preferable to use the *lumped* mass matrix. This matrix is diagonal whose elements are $\sum_j A_{ij}$.

Gradient

Piecewise-linear functions are only differentiable inside triangles. Therefore, the discrete gradient takes as input a function defined at vertices and outputs a vector per triangle. By linearity, it is enough to compute the gradient of the hat functions to find the gradient of any function [16]:

$$\nabla f(x) = \frac{1}{2\mu(T_{ijk})} \mathcal{R}^{90^\circ} E_{T_{ijk}} \begin{pmatrix} f_k - f_j \\ f_i - f_j \end{pmatrix}, \quad (3.6)$$

where \mathcal{R}^{90° is the counterclockwise rotation by 90° around the normal and $E_{T_{ijk}} = (x_j - x_i, x_k - x_j)$ contains a local basis of edges.

Laplacian operator

The discretization of the Laplace-Beltrami operator L is obtained by discretizing the inner product $\int_M \langle \nabla f, \nabla g \rangle d\mu$ represented by the *stiffness matrix* $W \in \mathbb{R}^{|\mathcal{X}| \times |\mathcal{X}|}$ and is

deduced from W by integration by parts (Equation (3.5)) so $L = A^{-1}W$. The discrete inner product follows the discrete gradient in Equation 3.6:

$$g^\top W f = \sum_{T \in \mathcal{F}} \frac{1}{4\mu(T_{ijk})} \begin{pmatrix} g_k - g_j \\ g_i - g_j \end{pmatrix}^\top E_T^\top E_T \begin{pmatrix} f_k - f_j \\ f_i - f_j \end{pmatrix}. \quad (3.7)$$

After rearranging the terms the matrix becomes a graph Laplacian:

$$(Wf)_i = \sum_{j \sim i} w_{ij}(f_i - f_j),$$

$$w_{ij} = \begin{cases} \frac{1}{8\mu(T_{ijk})}(\ell_{ij}^2 - \ell_{jk}^2 - \ell_{ki}^2) \\ + \frac{1}{8\mu(T_{ijl})}(\ell_{ij}^2 - \ell_{jl}^2 - \ell_{li}^2), & (i, j) \in \mathcal{E} \\ -\sum_{k \sim i} w_{ki}, & i = j \\ 0, & \text{else.} \end{cases} \quad (3.8)$$

The *cotangent weight formula*, a more popular expression of the Laplacian in the computer graphics community, is obtained by remarking that $\cot \alpha_{ij} = \frac{1}{4\mu(T_{ijk})}(-\ell_{ij}^2 + \ell_{jk}^2 + \ell_{ki}^2)$. So the weight at an edge $(i, j) \in \mathcal{E}$ reads:

$$w_{ij} = -\frac{1}{2}(\cot \alpha_{ij} + \cot \beta_{ij}). \quad (3.9)$$

where α_{ij} and β_{ij} are the angles opposite of the edge.

Eigenfunctions

Given the discrete Laplacian, finding the eigen-decomposition is done by solving the generalized eigenvalue problem:

$$W\Phi = A\Phi\Lambda, \quad \text{subject to } \Phi^\top A\Phi = I,$$

where Λ is the matrix of eigenvalues. Since W is symmetric and A is positive definite, there exists a solution with non-negative eigenvalues. The matrices A and W are sparse so this problem is solved efficiently by standard sparse solvers.

3.2.3 Discrete Local Coordinates

Although the Finite Element Method has a nice interpretation as an approximation of a function space by piecewise linear functions, it does not relate immediately to the continuous concept of metric and local charts. In this section we make one link between those two worlds.

The metric tensor expressed as a matrix of edge lengths was first introduced in 1961 in [98] by Regge. It was originally intended for the study of discrete general relativity. This field is often referred as *Regge calculus*. For an accessible introduction see [60].

Discrete gradient

Triangle faces provide a straightforward analog of tangent spaces as each triangle can be easily mapped to a 2D plane. Of course, unlike the continuous case, the transitions between coordinates are not smooth. In this section, we are going to construct the gradient operator from directional derivatives.

The directional derivative of a piecewise linear function f in the direction of an edge is, by definition, given by the classical finite difference formula:

$$\left\langle \nabla f, \frac{e_{ij}}{\|e_{ij}\|} \right\rangle = \frac{f_j - f_i}{\|e_{ij}\|}.$$

We denote $E_T = (e_{ij}, e_{jk})$ a particular coordinate system at a given triangle. Inside a triangle the derivative of f can only be computed along the edges, therefore we store them in the matrix $\partial f = \begin{pmatrix} f_j - f_i \\ f_k - f_j \end{pmatrix}$. Given a vector V in the discrete tangent space and its decomposition in local coordinate $V = E_T \alpha$, the directional derivative is expressed through a matrix multiplication:

$$V(f) = \alpha^\top \partial f = \langle \nabla f, V \rangle.$$

This formulation is the discrete counterpart of Equation (3.1). In the direction of the local basis we have the equality: $E_T^\top \nabla f = \partial f$. To obtain a gradient we need to invert E_T knowing that gradients are tangent vectors. To do so we are going to use the metric induced by the ambient space defined by:

$$\mathbf{g}_T := E_T^\top E_T,$$

as suggested by Equation (3.4). This leads to the following defining Proposition.

Proposition 3.2.1 *Given a local coordinate system $E_T = (e_{ij}, e_{jk})$ in triangle T , the gradient in local coordinate reads $\mathbf{g}_T^{-1} \partial f$ and in the global coordinate system:*

$$\nabla f = E_T \mathbf{g}_T^{-1} \partial f.$$

In this discussion we merely computed the derivative of a piecewise linear function therefore it should correspond to the Finite Element Method.

Discrete metric

The metric tensor possesses several properties that will prove useful in Chapters 7 and 8 when converting functional representation of deformation into embedded surfaces.

Proposition 3.2.2 *Given a local frame $E_T = (e_{ij}, e_{jk})$ the metric tensor $\mathbf{g}_T = E_T^\top E_T$ is symmetric positive definite and its determinant is such that:*

$$\det(\mathbf{g}_T) = 4\mu(T)^2.$$

Moreover the metric tensor can be rewritten exclusively in term of edge lengths:

$$\mathbf{g}_T = \frac{1}{2} \begin{pmatrix} 2\ell_{ij}^2 & \ell_{ki}^2 - \ell_{ij}^2 - \ell_{jk}^2 \\ \ell_{ki}^2 - \ell_{ij}^2 - \ell_{jk}^2 & 2\ell_{jk}^2 \end{pmatrix}.$$

Proof The discrete metric is by definition symmetric positive definite for non-degenerate triangles. The computation of the determinant follows Heron's formula.

The second statement comes from the fact that the edges sum to zero $e_{ij} + e_{jk} + e_{ki} = 0$ so:

$$\ell_{ki}^2 = \langle e_{ij} + e_{jk}, e_{ij} + e_{jk} \rangle = \ell_{ij}^2 + \ell_{jk}^2 + 2\langle e_{ij}, e_{jk} \rangle.$$

□

Equivalence with Finite Element Method

The discretization using the Finite Element Method, Equation (3.6), and using the local coordinates, Proposition 3.2.1, are the same. First let's recall this simple property satisfied by every invertible symmetric matrix in $\mathbb{R}^{2 \times 2}$:

$$\mathbf{g}_T^{-1} = \frac{1}{4\mu(T)^2} D^\top \mathbf{g}_T D, \quad \text{where } D = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}. \quad (3.10)$$

Starting from Proposition 3.2.1 and using Equation (3.10) lead directly to the conclusion:

$$\begin{aligned} \nabla f &= E_T \mathbf{g}_T^{-1} \partial f \\ &= \frac{1}{2\mu(T)} \left(\frac{1}{2\mu(T)} E_T D^\top E_T^\top \right) E_T D \partial f \\ &= -\frac{1}{2\mu(T)} \mathcal{R}^{90^\circ} E_T D \partial f. \end{aligned}$$

The inner product matrix W also enjoys a formulation in terms of local coordinate equivalent to Equation (3.7):

$$g^\top W f = \sum_{T \in \mathcal{F}} \begin{pmatrix} g_j - g_i \\ g_k - g_j \end{pmatrix}^\top \mathbf{g}_T^{-1} \begin{pmatrix} f_j - f_i \\ f_k - f_j \end{pmatrix} \mu(T).$$

Extension to simplicial complexes

One advantage of Regge calculus is that metric tensors and gradients can be easily defined for higher dimensional simplicial complexes. Moreover, they are still computable (but not necessarily meaningful) for non-manifold triangulations while being equivalent to the Finite Element Method in nicer scenarios.

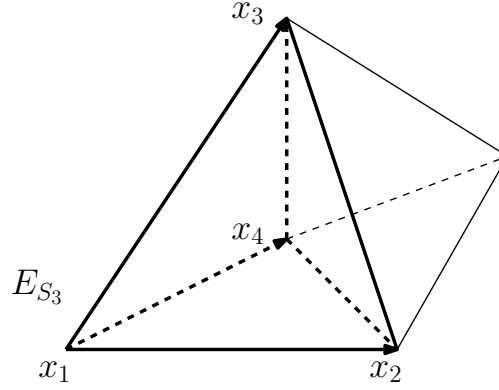


Figure 3.6 – Local basis in a tetrahedra.

For example in Figure 3.6 a k -simplex S_k is defined by $k + 1$ vertices (x_1, \dots, x_{k+1}) . The first k edges form the local basis: $E_{S_k} = (x_2 - x_1, \dots, x_{k+1} - x_1)$. The discrete metric tensor $\mathbf{g}_{S_k} = E_{S_k}^\top E_{S_k}$ is expressed locally using the edge lengths:

$$\mathbf{g}_{ij} = \begin{cases} \ell_{1i}^2, & i = j \\ \frac{1}{2}(\ell_{ij}^2 - \ell_{1i}^2 - \ell_{1j}^2), & i \neq j. \end{cases}, \quad 1 \leq i, j \leq k.$$

The volume of the simplex is accessible through the determinant of the metric by $\mu(S_k) = \sqrt{\det(\mathbf{g}_{S_k})}/k!$. The gradient inside the simplex is now computed with the formula:

$$\nabla f = E_{S_k} \mathbf{g}_{S_k}^{-1} \begin{pmatrix} f_2 - f_1 \\ \dots \\ f_k - f_1 \end{pmatrix}.$$

Given a pure simplicial k -complex \mathcal{S} , the inner product matrix W can be assembled using the following rule:

$$g^\top W f = \sum_{S_k \in \mathcal{S}} \begin{pmatrix} g_2 - g_1 \\ \dots \\ g_k - g_1 \end{pmatrix}^\top \mathbf{g}_{S_k}^{-1} \begin{pmatrix} f_2 - f_1 \\ \dots \\ f_k - f_1 \end{pmatrix} \mu(S_k).$$

Operator Representation

This chapter reviews two tools of the operator based representation framework: *functional maps* [88], representing diffeomorphisms, and *shape differences* [106], encoding the intrinsic deformation. The next chapters of this thesis explore the possibilities offered by these representations therefore this introduction attempts to give a complementary background to the reader and highlight open problems in this framework. We focus the presentation on the theoretical properties of these operators and how they are preserved after discretization. Although most of the results discussed here were already provided in the original papers, some are novel and not mentioned in follow-up works. In particular, Theorem 4.1.1 gives a characterization of composition operators never used in computer graphics and Propositions 4.2.3 and 4.2.4 describe the precise conditions in which compositions and summations are allowed for shape differences.

4.1 Functional Map

Real-valued functions are easily generated on many types of shapes (coordinate functions, descriptors) plus they enjoy interesting algebraic structure as they can be added, multiplied and composed with maps. To take advantage of this structure one can represent a diffeomorphism $\varphi : N \rightarrow M$ as a composition operator C_φ acting on real-valued functions:

$$\begin{aligned} C_\varphi : L^2(M) &\longrightarrow L^2(N) \\ f &\longmapsto f \circ \varphi. \end{aligned}$$

Instead of studying directly the deformation induced by φ , composition operators assess the changes in the space of square-integrable functions L^2 defined on the manifolds M and N . These operators have been studied and used in many fields including dynamical systems (Koopman operator theory [51]) and functional analysis [112]. The immediate property of this point of view is that a composition acts linearly with respect to functions. In the discrete settings, composition are therefore represented by matrices, meaning that the space of diffeomorphisms can be analyzed with standard linear algebra tools. Of course, all linear operators do not represent a composition (for example $f \mapsto 0$ is not a composition) but it suggests a convexification of the space of diffeomorphisms as linear mappings. In computer graphics composition operators were first introduced in [88] under the name of *functional maps*. They were originally used in intrinsic shape matching problems and led to many follow-up works. Two directions are now prevalent: either to make better use of the structure of the space (e.g. [101, 100, 90]) or to extend the functional representation to other quantities (e.g. metric distortion [106], tangent vector fields [6], fluid mechanics [8]).

In this section the term *composition operator* refers to the composition with an underlying map and should not be mistaken for *functional map* designating a linear operator acting on real-valued functions.

4.1.1 Mathematical Properties

All results presented here were first stated in [88] except for Theorem 4.1.1. Although this fundamental property is well-known in operator theory [112], it has not been used in a computer graphics context. A more comprehensive study and careful computations can be found in [112].

Characterization

A functional map is a linear operator between function spaces. A first question is what characteristic properties a linear operator should satisfy to be a composition operator. The following theorem provides an answer through a single property of the composition: the composition of a multiplication is the multiplication of the compositions.

Theorem 4.1.1 *Let $C : L^2(M) \rightarrow L^2(N)$ be a non-zero linear operator. Then, there exists a map $\varphi : N \rightarrow M$ such that $C = C_\varphi$ if and only if:*

$$C(fg) = C(f)C(g), \quad \forall f, g \in L^2(M).$$

A sketch of the proof is formally reproduced here as it helps building an intuition (for a complete version see [112] Theorem 2.1.13). The necessary condition is obviously satisfied by a composition operator. The sufficient condition is proven in three steps. The first part is to show that the image of an indicator function on M is an indicator function on N . Second, C must define a homomorphism ϕ between algebras of measurable sets and third ϕ induces a diffeomorphism φ .

Let χ_U be the indicator function of a measurable set $U \subset M$. The necessary condition of Theorem 4.1.1 yields:

$$C(\chi_U) = C(\chi_U^2) = C(\chi_U)C(\chi_U) = (C(\chi_U))^2.$$

Therefore $C(\chi_U)$ is also an indicator function. Any measurable set $U \subset M$ is mapped to a set $V \subset N$ through C , defining a map between sets by: $\phi(U) = V$.

Let U_1 and U_2 be two disjoint subsets of M , then:

$$C(\chi_{U_1 \cup U_2}) = C(\chi_{U_1} + \chi_{U_2}) = C(\chi_{U_1}) + C(\chi_{U_2}) = \chi_{V_1} + \chi_{V_2}.$$

Thus the image of a union of sets by ϕ is the union of the images. Similarly ϕ preserves intersections and differences between sets. So it can be shown that ϕ is a homomorphism between σ -algebras.

Finally, it follows, using a technical property of Borel sets (Theorem 2.1.12 in [112]), that there exists a measurable transformation φ such that $\phi(U) = \varphi^{-1}(U)$.

Algebraic properties

Many algebraic properties of the space of diffeomorphisms are naturally transferred to the space of composition operators. In a slightly simpler form, however, since we are dealing with linear operators.

The most basic property we can hope for is composition of mappings. Obviously, a composition of two mappings is also a composition operator since $C_\varphi(C_\psi(f)) = f \circ \psi \circ \varphi = C_{\psi \circ \varphi}(f)$. Thus, a composition of mappings becomes a composition of linear operators and we have proved the following:

Proposition 4.1.2 *Let $\varphi : N \rightarrow M$ and $\psi : M \rightarrow O$ be mappings between manifolds then their composition is the composition of the composition operators:*

$$C_\varphi \circ C_\psi = C_{\psi \circ \varphi}.$$

As a consequence of Proposition 4.1.2, an invertible mapping φ becomes an invertible composition operator by noting that $C_\varphi \circ C_{\varphi^{-1}}(f) = f$.

Corollary 4.1.3 *Let $C_\varphi : L^2(M) \rightarrow L^2(N)$ be a composition operator then C_φ is invertible if and only if φ is invertible. Moreover, we have:*

$$C_\varphi^{-1}(f) = C_{\varphi^{-1}}(f), \quad \forall f \in L^2(N).$$

Intrinsic information

Composition operators contain valuable information about the intrinsic structure of the manifold. For example Proposition 3.1.5 states that the pushforward measure describes the area distortion of the underlying deformation. Introducing the composition operator in the change of variables formula leads to the equality:

$$\langle C_\varphi(f), C_\varphi(g) \rangle_{L^2}^M = \int_M fg \sqrt{\frac{\det((\varphi^{-1})^* \mathbf{g}^N)}{\det(\mathbf{g}^M)}} d\mu_{\mathbf{g}^M}. \quad (4.1)$$

Thus the linear operator $(C_\varphi^* \circ C_\varphi)(f) : L^2(M) \rightarrow L^2(M)$, where the adjoint operator C_φ^* is defined by $\langle C_\varphi^*(f), g \rangle_{L^2(M)} = \langle f, C_\varphi(g) \rangle_{L^2(N)}$, accounts for the multiplication of a function by the ratio of local areas. Thus, $C_\varphi^* \circ C_\varphi$ is an operator representation of the area distortion which will reappear in Section 4.2.

Proposition 3.1.8 completes the intrinsic information given by composition operators: composition operators representing isometries should commute with the Laplacian.

Theorem 4.1.4 *Let $C_\varphi : L^2(M) \rightarrow L^2(N)$ be a composition operator then,*

- φ is area preserving if and only if:

$$C_\varphi^*(C_\varphi(f)) = f, \quad \forall f \in L^2(M).$$

- φ is an isometry if and only if:

$$C_\varphi(\Delta_N f) = \Delta_M C_\varphi(f), \quad \forall f \in L^2(M), \Delta_N f \in L^2(N).$$

Theorem 4.1.4 shows that composition operators fully characterize intrinsic geometry. However, as such, we cannot use them to compare metrics of different manifolds since by definition compositions start and end at different function spaces. For intrinsic distortion analysis we will prefer the *shape difference* operators introduced in Section 4.2.

4.1.2 Discrete Functional Maps

In the discrete setting the space L^2 is replaced by an approximation with piecewise linear functions – a standard choice in FEM. A function is then a vector in $\mathbb{R}^{|\mathcal{X}|}$ assigning a real value per vertex. Therefore, functional maps will be understood as matrices. Let $(\phi_1^M, \dots, \phi_{|\mathcal{X}|}^M)$ be an orthonormal basis of such a space (i.e. $\phi_i^\top A \phi_j^M = \delta_{ij}$ where A is the mass matrix), then any function f is uniquely defined by the coefficients $a_i^M = f^\top A_M \phi_i^M$ in the decomposition $f = \sum_{i=1}^{|\mathcal{X}|} a_i^M \phi_i^M$. The functional map is represented by a matrix transporting coefficients on M to coefficients on N :

$$C_\varphi(f) = \sum_{j=1}^{|\mathcal{X}|} a_j^M C_\varphi(\phi_j^M) := \sum_{i,j=1}^{|\mathcal{X}|} C_{ij} a_j^M \phi_i^N.$$

The elements of C_φ are by definition the coefficients of the functions $C_\varphi(\phi_j^M)$ when decomposed in the orthonormal basis ϕ^N . This leads to the scalar products:

$$C_{ij} := (\phi_i^N)^\top A_N \mathbf{P}_{MN} \phi_j^M, \quad 1 \leq i, j \leq |\mathcal{X}|.$$

where A_N is the mass matrix on N and \mathbf{P}_{MN} assigns vertices on M to vertices on N . Let Φ_M be the matrix containing all the basis functions ϕ_i^M , the discrete functional map reads:

$$C_{MN} = \Phi_N^\top A_N \mathbf{P}_{MN} \Phi_M. \quad (4.2)$$

Representing a given vertex-to-vertex assignment by a functional map is usually done in two steps:

1. Compute a basis (or an orthonormal family) of piecewise linear functions on M and N ,
2. Use Equation (4.2) to obtain C_{MN} .

This representation is flexible as the choice of basis can be adapted to each application. However, this is still an ongoing research topic. Some authors for example suggest a compactly supported basis [87]. In this thesis, we will mostly focus on the popular choice of the basis of eigenfunctions of the Laplace-Beltrami operator.

Discrete properties

Every matrix does not represent a valid functional map since it has to satisfy Proposition 4.1.1, namely the composition of a product should be a product of the compositions. In fact, the product of two piecewise linear functions is not piecewise linear. So, this property cannot hold exactly when considering classic Finite Elements. This problem will not be discussed further in this thesis even though it could improve the operator representation of mappings.

Nevertheless many of the other properties are still satisfied exactly at the condition that the basis span the entire space of discrete functions, namely $\Phi^\top A \Phi = I$ and $\Phi \Phi^\top A = I$. Using Equation (4.2) the following operations hold:

- A composition of mappings becomes a multiplication of matrices as suggested by Proposition 4.1.2:

$$\begin{aligned} C_{PN}C_{MP} &= (\Phi_N^\top A_N \mathbf{P}_{PN} \Phi_P)(\Phi_P^\top A_P \mathbf{P}_{MP} \Phi_M) \\ &= \Phi_N^\top A_N \mathbf{P}_{PN} \mathbf{P}_{MP} \Phi_M \\ &= C_{MN}. \end{aligned}$$

- Proposition 4.1.3 remains true since $C_{NM} = \Phi_M^\top A_M \mathbf{P}_{MN}^{-1} \Phi_N$ is the inverse of the functional map $C_{MN} = \Phi_N^\top A_N \mathbf{P}_{MN} \Phi_M$.
- Theorem 4.1.4 is still valid in a discrete sense if the meshes have same connectivity. Proposition 7.4.1 in Chapter 7 states that the mass matrix uniquely determines the triangle areas. So $C_{MN}^\top C_{MN}$ is the identity matrix if and only if M and N have same triangle areas.

The main theorem in [137] proves that the stiffness matrix uniquely defines the edge length up to global scaling. This strong result is mostly due to the rigidity of triangle meshes. In particular, $C_{MN} (\Phi_M^\top W_M \Phi_M) = (\Phi_N^\top W_N \Phi_N) C_{MN}$ if and only if M and N have identical triangle inner angles.

In practice a basis spanning the full space is not desirable as it requires $|\mathcal{X}|$ functions when shapes can contain several dozen thousand vertices. Besides, if the discrete manifolds have different connectivity, an accuracy up to a triangle is not relevant in many applications. It is often a better choice to use a small (compared to the number of vertices) family of smooth functions so the functional map is resilient to discretization noise.

Reduced basis

The eigenfunctions of the Laplace-Beltrami operator form a basis of the space of square-integrable functions. Moreover, they are naturally ordered according to their smoothness since they are minimizers of the Dirichlet's energy [125].

Therefore, we set our reduced basis to be the first k eigenfunctions of the Laplace-Beltrami operator. We are not considering the full function space but only a reduced space

of smooth functions. This way we are compressing the information into a $k_N \times k_M$ matrix by keeping only the low frequency structure of the diffeomorphism. Nevertheless, this basis has several interesting properties regarding the representation of nearly isometric deformations. Since the Laplacian commutes with isometries, nearly isometric maps are represented by nearly diagonal matrices. Furthermore, a recent paper by Rodolà and al. [101] remarks that even when large parts of a shape are missing the functional map has a slanted-diagonal structure, making this representation useful for challenging problems such as partial matching. Figure 4.1 shows examples of functional maps in three cases.

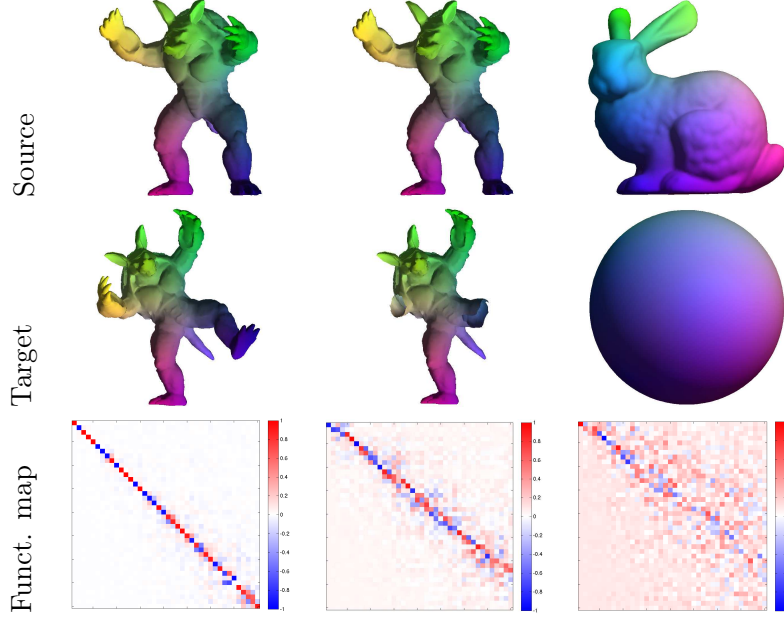


Figure 4.1 – The first two rows represent three maps via color correspondence and the third their corresponding functional maps computed with 40 eigenfunctions. Three test cases are organized in three columns: a nearly isometric map is represented by an almost diagonal functional map, a partial matching becomes a slanted-diagonal matrix and a non-isometric (but almost conformal) map which do not exhibit a very sparse structure.

Regarding algebraic operations on functional maps, all the discrete properties mentioned above still hold but only in a reduced space of functions. In the reduced basis of the LB eigenfunctions the functional map acts as a low-pass filter on M and on N . Two failure cases can be identified depending on k_N and k_M the size of the bases. First scenario, the function do not lie in the span of Φ_M so the projection onto the basis distorts the function. The projection, however, is well transferred to N . Second scenario, f is represented by the family Φ_M but the transfer induces high frequency distortions lying outside of the span of Φ_N . Increasing k_N solves this issue. Figure 4.2 illustrates the effect of varying k_M and k_N on function transfer. In conclusion, a function f is accurately transferred if f and $\mathbf{P}_{MN}f$ are well represented in their respective basis. We

can characterize this subspace by:

$$\left\{ f \in \mathbb{R}^{|\mathcal{X}|} : f \in \text{span}(\Phi_M), \mathbf{P}_{MN} f \in \text{span}(\Phi_N) \right\}.$$

The problem of identifying the subspace of stable functions is tackled in Chapter 5.

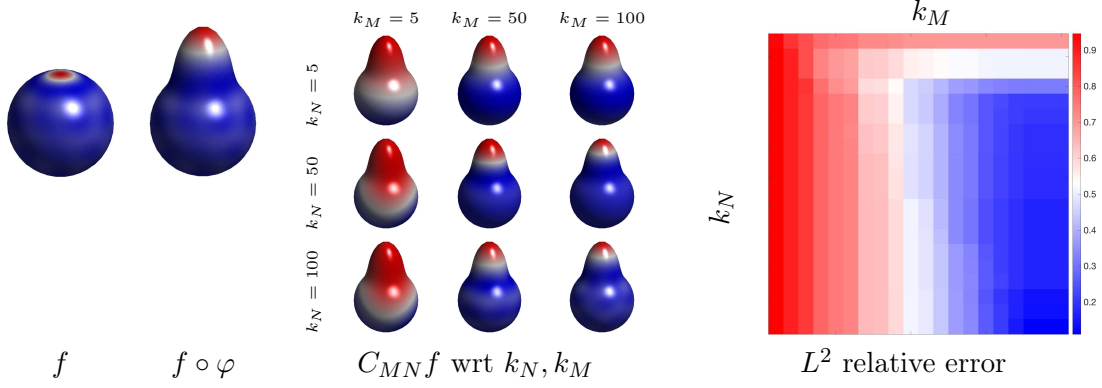


Figure 4.2 – Left: The function f on M and its exact transfer on N . Middle: Representation of the function $C_{MN}f$ on the shape N when the functional map C_{MN} is computed for varying k_N and k_M . Right: The relative error $\|f \circ \varphi - C_{MN}f\|_{L^2}^N / \|f \circ \varphi\|_{L^2}^N$ for various values of k_N and k_M . This experiment highlights the effect of basis sizes k_N and k_M on the transfer of a function by a functional map. When $k_M < k_N$ the function is badly represented on M but the projection is well transferred to N . When $k_N < k_M$ the function is well represented on M but the projection is badly transferred to N .

Other problems in the functional map framework

In this section we have only considered the problem of representing a given mapping between surfaces. The study of approximating an unknown functional map with minimal distortion using functional correspondences is covered in Chapter 5.

One of the challenge in this framework is to convert a functional map into a point-to-point map used in many applications. In [88] the authors propose a way to solve this problem by transferring highly localized functions. However, those functions are not well represented in a reduced basis introducing noise and destroying the smoothness of the recovered map. We propose a way to address this issue in Chapter 6.

4.2 Shape Differences

Characterization and operations on deformations are a fundamental tools in geometry processing with many applications, including deformation design, shape search and the organization of shape collections. The composition operator introduced in the previous section provides a lot of information on the intrinsic structure of a diffeomorphism as shown in Theorem 4.1.4. However, composition is not a convenient tool to compare

mappings or metrics. For example, if M and N_1, N_2 are shapes, by definition the functional maps ending points are respectively $L^2(N_1)$ and $L^2(N_2)$. As such it seems difficult to compare the deformation between M and N_1 with the deformation from M to N_2 without a map linking N_1 to N_2 . Moreover, an isometric deformation does not change the metric but is represented by a functional map different from identity. Thus, we need a representation of intrinsic deformation invariant under composition with isometries.

The shape differences address some of these issues. They are operators acting on functions and representing the intrinsic distortion induced by a given functional map. They are also symmetric allowing comparison between deformations and invariant under isometric composition.

4.2.1 Definition

Introduced by [106], the shape difference operators describe a shape deformation by considering the change of two inner products between functions $\langle f, g \rangle_{L^2}^M := \int_M fg \, d\mu$ and $\langle f, g \rangle_{H_0^1}^M := \int_M \mathbf{g}(\nabla f, \nabla g) \, d\mu$. Namely, given a pair of shapes M, N and a functional map $C_\varphi : L^2(M) \rightarrow L^2(N)$ the authors introduce the area-based and conformal shape difference operators $D_A : L^2(M) \rightarrow L^2(M)$ and $D_C : H_0^1(M) \rightarrow H_0^1(M)$ respectively, as linear operators acting on (and producing) real-valued functions on M implicitly via the following equations:

$$\begin{aligned} \langle f, D_A(g) \rangle_{L^2}^M &:= \langle C_\varphi(f), C_\varphi(g) \rangle_{L^2}^N & \forall f, g \in L^2(M) \\ \langle f, D_C(g) \rangle_{H_0^1}^M &:= \langle C_\varphi(f), C_\varphi(g) \rangle_{H_0^1}^N & \forall f, g \in H_0^1(M). \end{aligned} \quad (4.3)$$

By $H_0^1(M)$ we denote the function space of square integrable functions with $L^2(M)$ gradients and zero integrals:

$$H_0^1(M) = \left\{ f \in L^2(M) : \int_M \|\nabla f\|^2 \, d\mu < +\infty, \int_M f \, d\mu = 0 \right\}.$$

This space seems natural when studying the conformal shape difference since D_C maps any constant function to zero. When equipped of the scalar product $\langle \cdot, \cdot \rangle_{L^2} + \langle \cdot, \cdot \rangle_{H_0^1}$, H_0^1 is a Hilbert space.

The bilinear form $(f, g) \mapsto \langle f, g \rangle_{H_0^1}^M$ is continuous and coercive thanks to the Wirtinger's inequality [20]. Moreover, for a given g in $H_0^1(M)$ the linear form $f \mapsto \langle C_\varphi(f), C_\varphi(g) \rangle_{H_0^1}^N$ is continuous assuming that C_φ represents a composition with a diffeomorphism. All conditions of the Lax-Milgram theorem [20] are satisfied therefore there exists a unique $D_C(g)$ satisfying Definition 4.3. The existence and uniqueness of the area-based shape difference are guaranteed by the same argument.

It is important to note that the definition of shape differences does not require a composition operator but merely a functional map (i.e. a linear map between function spaces). The properties derived in the following, however, assumes there exists an underlying diffeomorphism $\varphi : N \rightarrow M$.

4.2.2 Fundamental Properties

As shown by [106] those operators fully characterize intrinsic deformations. More precisely, the following equivalences link properties of the mapping to properties of the operators.

Theorem 4.2.1 *Given a pair of surfaces M, N and a diffeomorphism $\varphi : N \rightarrow M$ with the functional representation C_φ , the followings hold for all functions f in $H_0^1(M)$:*

- φ is area-preserving if and only if $D_A(f) = f$,
- φ is conformal if and only if $D_C(f) = f$,
- φ is an isometry if and only if $D_A(f) = f$ and $D_C(f) = f$.

The last item is a direct consequence of the first two. The first item has already been proven by Theorem 4.1.4. Besides Equation (4.1) provides a direct representation of the operator D_A :

$$D_A(f) = \sqrt{\frac{\det((\varphi^{-1})^* \mathbf{g}^N)}{\det(\mathbf{g}^M)}} f, \quad \forall f \in L^2(M). \quad (4.4)$$

The proof of the second item requires a technical lemma proven in [109] and reproduced here as it will be useful later on (in particular in Chapter 8). Lemma 4.2.2 generalizes the fundamental lemma of calculus of variations to symmetric tensors.

Lemma 4.2.2 *Let (M, \mathbf{g}) be a Riemannian manifold (possibly with Lipschitz boundary) and let \mathbf{A} be a differentiable symmetric 2-tensor field on M . Then \mathbf{A} is the null tensor if and only if:*

$$\int_M \mathbf{g}(\nabla f, \mathbf{A} \nabla g) \, d\mu_{\mathbf{g}} = 0, \quad \forall f, g \in C^\infty(M).$$

The proof of the last point follows after a few computations.

Proof of Theorem 4.2.1

Let \mathbf{g}^N be the metric on N and let \mathbf{g}^* denote $(\varphi^{-1})^* \mathbf{g}^N$ the pullback metric on M . Using Lemma 3.1.6 and the definition of the pullback metric, we have:

$$\begin{aligned} \langle C_\varphi(f), C_\varphi(g) \rangle_{H_0^1}^N &= \int_N \mathbf{g}_p^N \left(d\varphi_{\varphi(p)}^{-1} (\nabla_{\mathbf{g}^*} f), d\varphi_{\varphi(p)}^{-1} (\nabla_{\mathbf{g}^*} g) \right) d\mu_{\mathbf{g}}(p) \\ &= \int_M \mathbf{g}_{\varphi^{-1}(q)}^N \left(d\varphi_q^{-1} (\nabla_{\mathbf{g}^*} f), d\varphi_q^{-1} (\nabla_{\mathbf{g}^*} g) \right) d\mu_{\mathbf{g}^*}(q) \\ &= \int_M \mathbf{g}^* (\nabla_{\mathbf{g}^*} f, \nabla_{\mathbf{g}^*} g) \, d\mu_{\mathbf{g}^*}. \end{aligned}$$

The scalar product $\mathbf{g}^\star(\nabla_{\mathbf{g}^\star} f, \nabla_{\mathbf{g}^\star} g)$ is rearranged to $\mathbf{g}^M(\nabla_{\mathbf{g}^M} f, \mathbf{A} \nabla_{\mathbf{g}^M} g)$ where $\mathbf{A}_{ij} = (\mathbf{g}^\star)^{ik} \mathbf{g}_{kj}^M$ and \mathbf{g}^M is the metric on M . The definition of the pushforward measure leads to the equation:

$$\langle C_\varphi(f), C_\varphi(g) \rangle_{H_0^1}^N = \int_M \mathbf{g}^M(\nabla_{\mathbf{g}^M} f, \lambda \mathbf{A} \nabla_{\mathbf{g}^M} g) \, d\mu_{\mathbf{g}^M}, \quad \lambda = \sqrt{\frac{\det(\mathbf{g}^\star)}{\det(\mathbf{g}^M)}}. \quad (4.5)$$

We are now ready to prove the equivalence.

- Suppose that φ is conformal then $\mathbf{g}^M = \lambda \mathbf{g}^\star$. In Equation (4.5) it implies that $\lambda \mathbf{A}_{ij} = \delta_{ij}$ so D_C is the identity operator.
- Suppose that D_C is identity then using Definition (4.3) and Equation (4.5) lead to:

$$\int_M \mathbf{g}^M(\nabla_{\mathbf{g}^M} f, (\text{Id} - \lambda \mathbf{A}) \nabla_{\mathbf{g}^M} g) \, d\mu_{\mathbf{g}^M} = 0, \quad \forall f, g.$$

Thanks to Lemma 4.2.2 we conclude that $\mathbf{g}^M = \lambda \mathbf{g}^\star$.

□

Shape differences are interpreted as ratio of intrinsic quantities. The area-based operator is the ratio between the local area of the pullback metric with the local area of the reference metric. A similar explanation applies to the conformal operator, Equation (4.5) allows us to rewrite Definition (4.3):

$$\begin{aligned} \int_M \langle \nabla f, \nabla D_C(g) \rangle \, d\mu &= \int_M \langle \nabla f, \mathbf{B} \nabla g \rangle \, d\mu, \\ \mathbf{B}_{ij} &= \sqrt{\frac{\det((\varphi^{-1})^\star \mathbf{g}^N)}{\det(\mathbf{g}^M)}} ((\varphi^{-1})^\star \mathbf{g}^N)^{ik} \mathbf{g}_{kj}^M. \end{aligned} \quad (4.6)$$

Thus, D_C describes the ratio of the pullback metric and the reference metric weighted by the local area. The tensor \mathbf{B} is immune to multiplication of the metric by a positive function, so this measurement of distortion cannot be used to assess area changes. In that sense D_A and D_C are orthogonal and complementary when describing intrinsic deformations.

The shape difference operators are defined by the mean of a diffeomorphism but Equations (4.5) and (4.6) rely only weakly on it. So these operators can be extended to (and should be thought as) comparison of an arbitrary metric, not necessarily induced by an embedded surface, with the reference metric.

4.2.3 Algebraic Properties

The results of this section, namely Propositions 4.2.3 and 4.2.4, are novel and do not appear in the original paper [106] or any follow-up papers.

Linear combination

Unlike composition operator there is no known defining property equivalent to Theorem 4.1.1. However, Equations (4.4) and (4.6) provide a sense of the structure of the shape difference space.

Equation (4.4) tells us that the area-based shape difference is a multiplication operator associated to a positive function. Thus multiplying D_A by a positive scalar represents a global scaling of the target metric. Moreover adding two area-based shape differences results in a linear operator representing a shape difference. However, the underlying metric may not correspond to an embedded surface.

For the conformal operator the situation is even simpler. From Equation (4.6) we take out that this operator is characteristic of a 2-tensor with determinant one. In general linear combinations do not produce a valid conformal shape difference.

Besides those algebraic considerations, the space of shape differences is vast as it represents any metric distortion. At the moment, constraining shape differences to represent the metric of an embedded surface is an open question.

Deformation support

The addition between operators is only possible in case of non-overlapping deformations. As a consequence of Theorem 4.2.1, the set $\Omega \subset M$ supports a deformation represented by the operator D (area or conformal-based) if and only if:

$$D(f) = f, \quad \forall f \in H(M) : \text{Supp} f \cap \Omega = \emptyset,$$

where $\text{Supp} f = \{p \in M : f(p) \neq 0\}$ denotes the support of a real-valued function on M . The following proposition specifies necessary conditions for summation of deformations.

Proposition 4.2.3 *Let $D_1, D_2 : H(M) \rightarrow H(M)$ be two shape difference operators with respective supports Ω_1 and Ω_2 such that $\Omega_1 \cap \Omega_2 = \emptyset$, then the linear operator:*

$$D_+(f) = D_1(f) + D_2(f) - f, \quad \forall f \in H(M),$$

is the shape difference operator corresponding to the deformation D_i on Ω_i and to an isometry elsewhere.

Proof The proof is restricted to the case of conformal shape difference as it is very similar for the area-based one. Using Equation (4.6) the operator D_+ is separated into a sum of three integrals:

$$\langle f, D_+(g) \rangle_{H_0^1}^M = \int_{\Omega_1} \langle \nabla f, \mathbf{B}_1 \nabla g \rangle \, d\mu + \int_{\Omega_2} \langle \nabla f, \mathbf{B}_2 \nabla g \rangle \, d\mu + \int_{\Omega_1^c \cup \Omega_2^c} \langle \nabla f, \nabla g \rangle \, d\mu.$$

Since the support of these integrals are disjoint they can be written as a single integral:

$$\langle f, D_+(g) \rangle_{H_0^1}^M = \int_M \langle \nabla f, \mathbf{B} \nabla g \rangle \, d\mu,$$

where \mathbf{B} is the smooth tensor on M defined by:

$$\mathbf{B}(p) = \begin{cases} \mathbf{B}_1(p), & p \in \Omega_1 \\ \mathbf{B}_2(p), & p \in \Omega_2 \\ \text{Id}(p), & p \in \Omega_1^c \cup \Omega_2^c. \end{cases}$$

□

Composition of diffeomorphisms

Another possible operation on shape difference operators is the composition of two (possibly overlapping) deformations. As the property introduced here applied to both types of operators we will denote D the operator corresponding to either D_A or D_C and H the function space L^2 or H_0^1 .

The following proposition was proved in a restricted discrete setting for one operator in the original article [106].

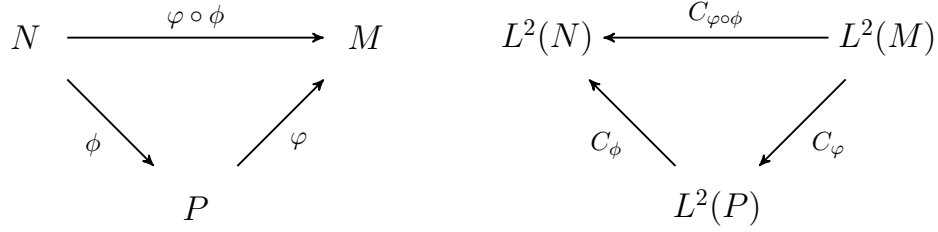


Figure 4.3 – Left: the point-to-point maps ϕ, φ linking manifolds M, N, P . Right: dual representation as functional maps.

Proposition 4.2.4 *Assuming that $D^\varphi : H(M) \rightarrow H(M)$ represents the distortion of the metric between the surfaces M and P induced by the diffeomorphism $\varphi : P \rightarrow M$ and $D^\phi : H(P) \rightarrow H(P)$ the distortion between the surfaces P and N linked through $\phi : N \rightarrow P$ (see Figure 4.3).*

The distortion $D^{\varphi \circ \phi} : H(M) \rightarrow H(M)$ associated to $\varphi \circ \phi : N \rightarrow M$ is given by:

$$D^{\varphi \circ \phi} = D^\varphi \circ C_\varphi^{-1} \circ D^\phi \circ C_\phi.$$

Proof The proof relies only on Definition (4.3). Let f, g be a pair functions in $H(M)$:

$$\begin{aligned} \left\langle f, D^{\varphi \circ \phi}(g) \right\rangle_H^M &= \left\langle C_{\varphi \circ \phi}(f), C_{\varphi \circ \phi}(g) \right\rangle_H^N \\ &= \left\langle C_\varphi(f), D^\phi(C_\phi(g)) \right\rangle_H^P \\ &= \left\langle f, D^\varphi \left(C_\varphi^{-1} \left(D^\phi(C_\phi(g)) \right) \right) \right\rangle_H^M. \end{aligned}$$

□

Proposition 4.2.4 confirms the intuition that the shape differences are invariant under composition with isometries. Suppose that the diffeomorphism ϕ is an isometry then Theorem 4.2.1 tells us that D^ϕ is the identity operator on $H(P)$. Using the composition formula we conclude that the distortion of the composition is equal to the distortion induced by φ alone:

$$D^{\varphi \circ \phi} = D^\varphi \circ C_\varphi^{-1} \circ D^\phi \circ C_\varphi = D^\varphi.$$

The shape difference operators associated to the inverse mapping φ^{-1} can be computed from D^φ using the composition formula: $D^{\varphi^{-1} \circ \varphi} = D^{\varphi^{-1}} \circ C_\varphi \circ D^\varphi \circ C_\varphi^{-1}$. Formally, we have:

$$D^{\varphi^{-1}} = C_\varphi \circ (D^\varphi)^{-1} \circ C_\varphi^{-1}.$$

Distortion pullback

An attractive property of this framework is the possibility to transport a distortion from a shape to another. Given a shape difference on shape P and a functional map between M and P , we would like to describe an equivalent distortion of the metric on M . Intuitively, the operator $D^\star = C_\varphi^{-1} \circ D^\phi \circ C_\varphi$ is a good candidate for the pullback of the shape difference D^ϕ encoding the metric deformation from \mathbf{g}^P to $(\phi^{-1})^\star \mathbf{g}^N$. It follows from Proposition 4.2.4 that D^\star is characteristic of the same distortion but after a pullback on M , namely it compares $(\varphi^{-1})^\star \mathbf{g}^P$ with $((\varphi \circ \phi)^{-1})^\star \mathbf{g}^N$.

This operator pullback suggested various experiments in the original shape difference article [106]. For example the authors find similar deformations across different shapes collections. Given a collection of cats in different positions the pullback shape differences are used to identify similar poses in a collection of lions.

4.2.4 Discrete Shape Differences

In the discrete setting, a functional map is defined by Equation (4.2). Thus, given orthonormal bases Φ_M and Φ_N , Definition (4.3) yields the matrices $D_A, D_C \in \mathbb{R}^{k_M}$:

$$\begin{aligned} D_A &= C_{MN}^\top C_{MN}, \\ D_C &= \left(\Phi_M^\top W_M \Phi_M \right)^{-1} C_{MN}^\top \left(\Phi_N^\top W_N \Phi_N \right) C_{MN}. \end{aligned}$$

We recover the interpretation of shape differences as ratios. The area-based shape difference is the ratio between mass matrices and the conformal operator compares stiffness matrices.

When the basis is restricted to the first eigenfunctions of the Laplacian, the conformal shape difference has a slightly simpler expression ([106] Section 5 Option 2):

$$D_C = \Lambda_M^{-1} C_{MN}^\top \Lambda_N C_{MN},$$

where Λ is the diagonal matrix storing the eigenvalues of the Laplacian.

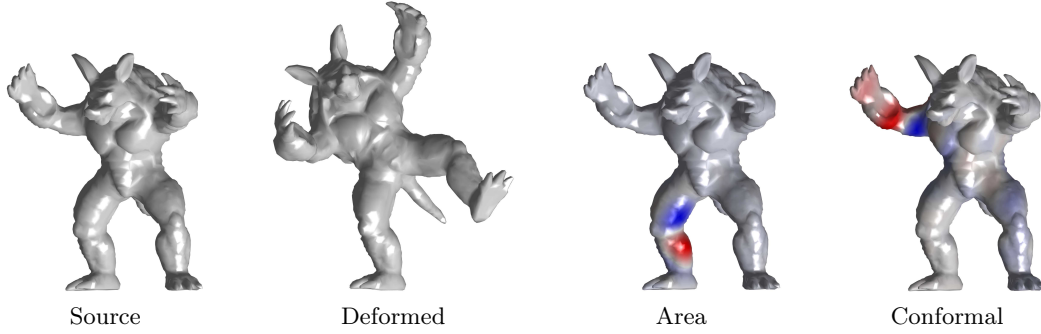


Figure 4.4 – Most distorted functions corresponding to the area and conformal-based shape differences for a nearly isometric deformation of the armadillo.

When represented by matrices the shape differences can be analyzed by standard linear algebra tools. For example, the eigenvalues offer a quantification of the distortion. According to Theorem 4.2.1, an eigenvalue close to one is associated to an eigenfunction highlighting isometric areas. Figure 4.4 shows the most distorted function in the simple deformation scenario.

Discrete properties

Theorem 4.2.1 is satisfied by the discretization. More precisely, in Chapter 7, we show that one can recover the discrete metric (edge lengths) from both shape differences by solving two linear systems of equations. In fact, this chapter proposes a solution to the broader problem of finding embedded surfaces from operators.

The algebraic operations of addition and composition described by Propositions 4.2.3 and 4.2.4 hold true for a smooth subspace of functions. This is a direct consequence of Section 4.1.2 showing that the algebraic properties of functional maps are preserved after discretization.

As for functional maps, basis reduction can generate artifacts depending on the basis sizes k_M on M and k_N on N . A low k_N leads to a bad representation of high frequency distortions since some functions in Φ_M are not represented in Φ_N after transfer. So the shape differences unjustly map to zero some functions. It follows that eigenvalues (and eigenfunctions) lower the one are less reliable than eigenvalues greater than one. A low k_M implies that Φ_M contains only low frequency functions so the shape differences tend to smooth high frequency deformation. Overall Figure 4.5 shows that a tradeoff has to be found between k_M and k_N to represent correctly the distortion at a certain level of detail.

4.3 Organization of the Thesis

The rest of the thesis is organized in two parts. The first part is dedicated to the computation and analysis of functional maps, in particular applied to shape matching

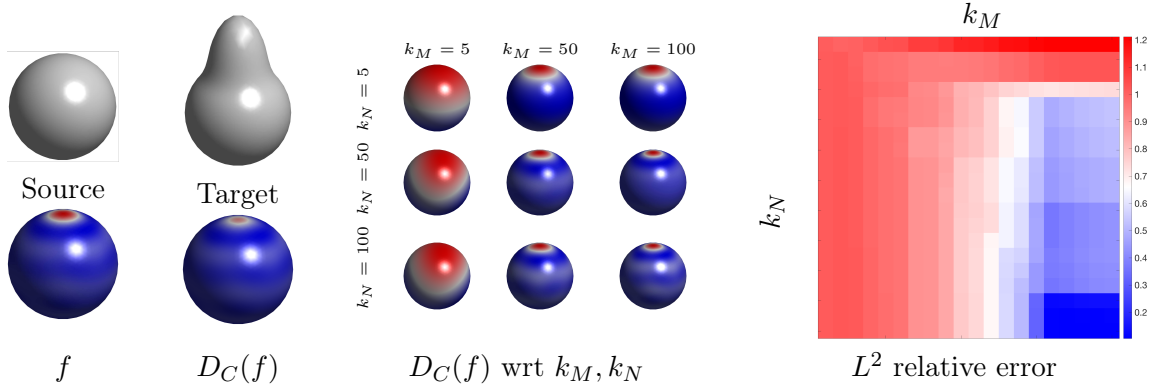


Figure 4.5 – Left: The function f on M and its image by the conformal shape difference defined between the source and target meshes. Middle: Representation of the function $D_C(f)$ when the functional map is computed for varying k_N and k_M . Right: The relative error between $D_C(f)$ computed with a functional map for various values of k_N and k_M and the exact conformal shape difference. This experiment highlights the effect of basis sizes k_N and k_M on the conformal shape difference operator. When $k_M < k_N$ the shape difference cannot represent high frequency distortion. When $k_N < k_M$ the shape difference may represent parasite distortion due to the truncation of the basis.

problems. In Chapter 4 we have assumed that the underlying diffeomorphism was given as a point-to-point correspondence so the functional map is an approximation of a composition operator. Chapter 5 tackles the converse problem: given two shapes we try to compute the functional map that fits a particular deformation model, namely the one that minimizes the intrinsic distortion. For this purpose we provide a supervised learning algorithm able to select features and identify functions that jointly produce the best functional map. To complete our tour on shape matching problems we move on to converting a functional map to a point-to-point representation. Although this problem was already considered in the original article, it gave no guarantee of recovering a continuous map. Chapter 6 suggests a solution based on repairing a given diffeomorphism by a vector field flow.

The second part explore intrinsic deformation through the prism of shape differences. As proven in the previous section shape differences completely characterize the metric in the continuous setting. Chapter 7 expends the analysis to triangular meshes by proposing an algorithm to recover edge lengths from operators. Furthermore, shape differences are blind to some curvature deformations hence giving an incomplete description of embedded surfaces. A possible solution explored in Chapter 7 is to encode the metric of an offset surface leading to a coordinate free description of triangular meshes by four operators.

Shape difference operators were introduced for analysis purposes. However, their interesting algebraic structure makes them suitable for exploration and synthesis of new deformations. In Chapter 7 a first step is made in the direction of deforming shapes using an operator-based representation with one drawback: to obtain embedded surfaces we need first to extract the metric information and then reconstruct the mesh by solving an

Multidimensional scaling problem which is computationally expensive and error prone. Chapter 8 looks at this problem from another angle and studies a characterization of deformation fields up to isometric deformation. For this, we first introduce a unified shape difference that fully characterizes all intrinsic changes. Then we consider the operator associated to an infinitesimal displacement, and characterize deformation fields by the distortion they induce on the metric. Interestingly these operators enjoy similar properties as shape differences (composition, transfer), which enable significantly simpler deformation reconstruction compared to the previous method, and enable novel deformation synthesis applications. We provide theoretical proofs of informativeness in both continuous and discrete settings.

Part I

Shape to Deformation

This first part is dedicated to the computation and analysis of functional maps. Namely, we use feature selection to improve solutions of shape matching problems in Chapter 5 and then propose an algorithm to convert functional maps to continuous point-to-point maps in Chapter 6.

Supervised Descriptor Learning for Non-Rigid Shape Matching

In this chapter, we present a novel method for computing correspondences between pairs of non-rigid shapes. Unlike the majority of existing techniques that assume a deformation model, such as intrinsic isometries, *a priori* and use a pre-defined set of point or part descriptors, we consider the problem of *learning* a correspondence model given a collection of reference pairs with known mappings between them. Our formulation is purely intrinsic and does not rely on a consistent parametrization or spatial positions of vertices on the shapes. Instead, we consider the problem of finding the optimal set of descriptors that can be jointly used to reproduce the given reference maps. We show how this problem can be formalized and solved for efficiently by using the recently proposed functional maps framework. Moreover, we demonstrate how to extract the functional subspaces that can be mapped reliably across shapes. This gives us a way to not only obtain better functional correspondences, but also to associate a confidence value to the different parts of the mappings. We demonstrate the efficiency and usefulness of the proposed approach on a variety of challenging shape matching tasks.

5.1 Introduction

Finding high quality correspondences is a key component in many tasks including statistical shape analysis [49], deformation transfer [118] and interpolation (morphing) [61] among others. While a number of efficient techniques have been proposed to address the problem of rigid alignment [120], the problem of general non-rigid shape matching remains difficult.

Most existing methods for finding correspondences between non-rigid shapes rely on an *a priori* deformation model, which specifies the space of “reasonable” maps between shapes. Perhaps the most popular and widely used such model is that of approximate intrinsic isometries [22, 74], where the mapping is assumed to preserve geodesic distances between all pairs of points on the shapes. A more general possibility is to consider conformal deformations, which are only assumed to preserve angles [69, 62] or to parameterize the space of possible maps using a fixed deformation model [138]. Although these techniques can produce good results when the deformation satisfies the *a priori* model, they can fail badly as soon as even moderate deviations from the model are introduced. This is especially critical since many natural deformations, such as articulated motion of humans or animals are known to induce potentially significant geodesic distortion [106]. Incorporating the possibility for such distortion into a deformation model is challenging especially using a purely axiomatic (theoretical) approach.

Rather than trying to devise a theoretical deformation model capable of adapting to known deformations, several communities have tackled this challenge by using a data-driven approach, where the space of “reasonable” maps or deformations is learned from a set of examples, e.g. [29]. Since obtaining example deformations is often significantly easier than devising a unified theoretical deformation model, such an approach allows the resulting techniques to remain flexible yet efficient in the particular settings where they are applied.

Most data-driven approaches for devising a deformation model, however, rely heavily on a consistent parametrization of the deformation domain (e.g. on a fixed grid in Euclidean space), and perform statistical analysis on the positions of vertices of the shapes [30, 14, 37, 43]. When computing correspondences between pairs of surfaces in 3D, such parametrization is often unavailable and moreover, shapes can undergo severe deformations which are difficult to capture using purely extrinsic approaches.

In this chapter, we propose a purely intrinsic method for exploiting prior correspondence information between pairs of shapes to find better correspondences between a reference shape and a new previously unseen instance. Rather than doing the learning over, e.g., the positions of the vertices on the shapes, we propose to find the optimal set of *point descriptors* that can be jointly used to reproduce the given reference maps. While such an optimization is, in general, very complicated, since even to evaluate how well the descriptors can reproduce a given map would require a full solution of the shape matching problem, we show how this problem can be formalized and solved for efficiently by using the recently proposed functional maps framework [88]. Moreover, we demonstrate how to extract the functional subspaces that can be mapped reliably across shapes. This gives us a way to not only obtain better functional correspondences, but also to associate a confidence value to the different parts of the mappings. Our approach is also quite general since it can be used as a preprocessing step of other methods using functional maps [90, 54, 6] in order to improve the quality of the results and help to handle difficult deformation. Note that in this chapter we focus on the shape matching problem which is the most developed application of the functional maps.

5.1.1 Related Work

Non-rigid shape matching is a very-well developed area and its complete overview is beyond the scope of this chapter (see, e.g., [21, 127] for recent surveys of this field). We therefore concentrate on the work directly related to ours, namely near-isometric shape matching with special emphasis on approaches that utilise prior knowledge for establishing correspondences between pairs of shapes.

The vast majority of techniques for non-rigid shape matching implicitly make use of a deformation model for finding correspondences between geometric shapes. Perhaps the most common model in the context of intrinsic (i.e., not relying on vertex positions and not assuming approximate alignment) approaches is approximate isometries, introduced by Bronstein et al. [22] and Mémoli [74]. This model has been used by a large number of methods, (e.g., [55, 123, 89, 107, 88] among many others) that all assume that the sought correspondences must approximately preserve pairwise geodesic distances. Another set of

approaches is based on a more relaxed model, conformal mappings, used by, e.g., [69, 62] where only angles are assumed to be preserved. Other techniques, such as the one used by Zhang et al. [138] explicitly deform a shape using a fixed deformation model to find correspondences between non-rigid shapes.

All of these approaches use a model given *a priori* to find correspondences, which can be problematic if the real deformations do not agree with the given model. Interestingly, it has recently been observed [106] that even articulated motion of humans can induce noticeable isometric distortion, which could explain some of the difficulties encountered by previously proposed techniques.

In contrast, other works have proposed to *learn* an appropriate deformation model from a set of examples, and then use this model for shape matching. Perhaps the best-known example of this approach are Active Shape and Active Appearance Models [29, 30] and their variants (see, e.g., [43]) used widely in Computer Vision. In a similar vein, techniques in Statistical Shape Analysis [37] use the distribution of positions of pre-specified landmark points in 3D to learn a statistical deformation model over which inference can be made. Related techniques are commonly used in medical imaging and Morphometrics [14] and in Geometry Processing communities, e.g. [4, 49] among many others. However, all of these methods assume the existence of a common domain over which learning can be made, and which most often is done using vertex coordinates of either landmark points or all points on a fixed reference shape. In the context of intrinsic shape matching, where shapes lack labeled landmark points and can undergo severe deformations, vertex coordinates are often not relevant, limiting the applicability of such techniques.

Rather than relying on vertex positions, recent methods have considered using derived properties such as point or triangle *descriptors* for learning. Thus, Kalogerakis et al. [58] and Van Kaick et al. [126] have proposed using a set of example shapes to train classifiers for part segmentation and labeling, which can then be used to establish part-level correspondences. Similarly, Chen et al. [26] explore the predictive power of various descriptors for detecting distinctive landmark (schelling) points identified by users. These methods, while similar to ours in learning on the level of descriptors do not, however, specifically address the shape matching problem.

Perhaps most closely related to ours are recent works by Litman et al. [71] and Rodolà et al. [100], where the authors use a set of examples to learn the most informative descriptors that are used directly in the context of shape matching. Our approach is fundamentally different, however, since rather than trying to identify descriptors that can distinguish different points, we propose to find the optimal *descriptor set* that can be used to *jointly produce the entire map across shapes*. We thus avoid the problem of obtaining *consistent* correspondences present in these approaches (and obtained during post-processing), since consistency is incorporated directly in the learning stage. Crucially, we use the recently proposed functional map representation [88] that allows us to formulate the learning problem purely intrinsically, while permitting to directly control and optimize for the influence of descriptors on the quality of the final map.

Goals Given a collection of (training) shapes with known correspondences our goal is to identify the most informative descriptor set that can be used to solve the non-rigid

shape matching problem on new (test) instances. Besides we want to learn where are the most stable correspondences.

5.2 Consistent Functional Maps

Our method is based on the functional map representation introduced in [88]. In this section, we give a brief overview of the representation and the method used in [88] to construct a functional map for a given pair of shapes.

While our method is general, throughout the chapter we assume that all shapes are represented as triangle meshes, and all functions are expressed as vectors in the basis of the eigenfunctions of the Laplace-Beltrami operator. This basis needs to be computed beforehand on each shape. The objective is to output a uniquely defined functional map.

5.2.1 Functional Map Representation

The functional map representation is based on the observation that given two surfaces M_0 and M_i , a point-to-point map $\varphi_i : M_i \rightarrow M_0$ induces a map between function spaces $C_i : L^2(M_0) \rightarrow L^2(M_i)$, where $L^2(M)$ is the set of square integrable functions defined on the surface M . The functional map C_i is defined by composition with φ_i as $C_i f = f \circ \varphi_i$. The operator C_i is a linear transformation and given a basis it can be represented as a matrix in the discrete setting. This matrix can be easily computed if the map φ is known. The basic method described in [88] approximates the functional map C_i using a set of linear constraints. The first type of constraints is given by a set of pairs of functions, which we refer to below as “probe functions”, that are expected to be preserved by the deformation. The second is a regularization term coming from the deformation model. This leads to the least squares problem:

$$X_i = \arg \min_C \|CG_0 - G_i\|_F^2 + \alpha \|C \odot W\|_F^2, \quad (5.1)$$

where $\|\cdot\|_F$ denotes the Frobenius norm. The use and meaning of each term will be detailed in the following paragraphs.

Probe Functions

The probe functions can be represented by two matrices G_0 and G_i , where each pair of corresponding columns represents a pair of functions g_0, g_i such that $C_i g_0 \approx g_i$ is expected to hold for the unknown C_i . In practice we normalize the corresponding column so that each column has the same L^2 norm and g_0, g_i contain the coefficients of the probe functions in a given basis (e.g. LB eigenfunctions). Thus, the functional map should verify $CG_0 \approx G_i$. In the context of isometric matching the probe functions are given by classical descriptors, such as the HKS [119], or WKS [5].

Regularization

In addition to the probe function constraints, the authors of [88] have proposed a regularization using the assumption that the deformation is nearly isometric. This assumption is equivalent to the commutativity of C_i with the Laplace-Beltrami operator, namely $C_i \Delta_0 = \Delta_i C_i$. In the discrete setting the eigenfunctions of the Laplace-Beltrami operator are used as function basis. Thus, this equation can be written as $C_i \odot W = 0$ where “ \odot ” denotes the component-wise multiplication and the matrix W is defined by $W_{kl} = \lambda_k^i - \lambda_l^0$ with λ_k^i the k^{th} eigenvalue of the Laplace-Beltrami operator on the surface M_i .

Other assumptions on the deformation model can be used. For example in [65] the authors regularize the shape matching problem by imposing that the map should be area-preserving implying that the matrices C_i are a-orthonormal as explained in Chapter 4. However, this assumption is weaker and leads to a non-convex problem more challenging to incorporate in our supervised learning algorithm.

Uniqueness of the solution. In practice the eigenvalues of the Laplace-Beltrami operator of two different shapes are always numerically different except for the zero eigenvalues. Thus, the only zero coefficient of W is W_{11} which weights the coefficient C_{11} of the functional map. Since the corresponding eigenfunctions are constant, C_{11} maps the constant functions of $L^2(M_0)$ into the constant functions of $L^2(M_i)$. The coefficient C_{11} should always be one. Therefore, since W is non zero everywhere, the solution of (5.1) is unique without any assumptions on G_0 .

However, the probe functions G_0 and G_i are in practice composed of symmetric functions, so we cannot hope to have information about the antisymmetric functions. Nevertheless, the problem (5.1) still has a unique solution that might simply map the antisymmetric functions poorly.

5.2.2 Main Challenge

In the original article [88] the probe functions are assumed to be given, so how to choose them was not discussed. As mentioned in introduction, this choice can already be challenging. For example in Figure 5.1a the smoothed Gaussian curvature computed on two different meshes provides a decent functional correspondence. At the same time, in Figure 5.1b the logarithm of the Gaussian curvature, while intrinsic in theory, does not result in a useful correspondence.

One option to identify the best descriptors would be to simply find the most stable probe functions in the example (training) set, by learning spectral descriptors [71] for example. However, some descriptors (e.g., the constant function) can be stable without at all being informative. More importantly, however, as can be seen from Equation (5.1), the descriptors influence the resulting functional maps X_i *jointly*. As an example, if a correspondence is described by several probe functions the resulting functional map will tend to respect this constraint while other meaningful correspondences will be arbitrarily put aside due to their low redundancy. So picking the best descriptors independently will not necessarily result in high quality maps.

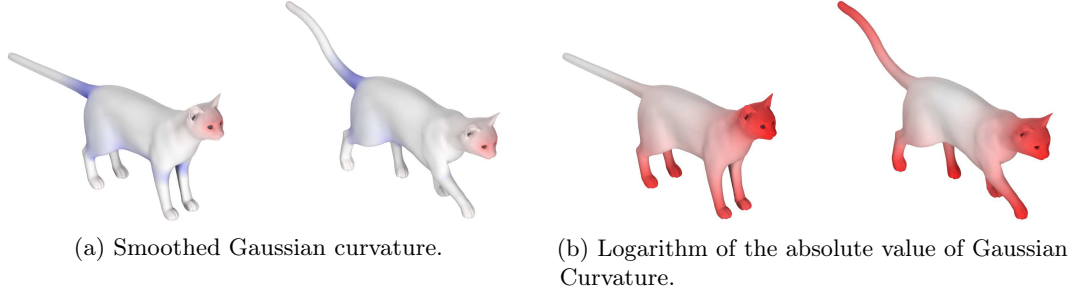


Figure 5.1 – Probe functions computed independently on two shapes. One carries meaningful information (a) and the other is misleading (b).

Thus, the key idea developed in this chapter is to introduce weights for probe functions, over which learning can be done. As explained below, the probe function constraint will be replaced by: $\|CG_0D - G_iD\|_F^2$ where D is an unknown diagonal matrix of weights. The weights D will be optimized so that the weighted descriptors are jointly as informative as possible. This will allow us to improve the quality of the functional maps and to extract the most stable functional subspaces.

5.2.3 Algorithm Outline

We propose a two-step method described in following two sections and summarized in Figure 5.2. Given collection of shapes, we learn the most informative set of weights D by solving an optimization problem. We then extract a functional basis whose components are ordered by quality of correspondence. When given a previously unseen shape, we use this information to compute a high-quality functional map using the optimal weights and to discard the badly mapped functions by reducing the functional space.

5.3 Selection of the Best Functional Correspondences

The idea developed here is to assign a weight to each pair of probe functions. These weights can then be tuned according to their consistency in the matching. Since *a priori* there is no reason to choose one probe function over another, we propose to learn the optimal weights given a training set of shapes.

As input we need a set of n triangulated meshes with known correspondences representing the same object undergoing a set of deformations. Our main assumption is that the optimal weights on the probe functions should be stable across the shapes in the collection. Thus, if we are given a new deformation of the same shape, the learned weight should also select the consistent probe functions. The output of our algorithm will be a set of weights for the probe functions, which, as we will show below, can then be used to find correspondences between new, unseen shape instances.

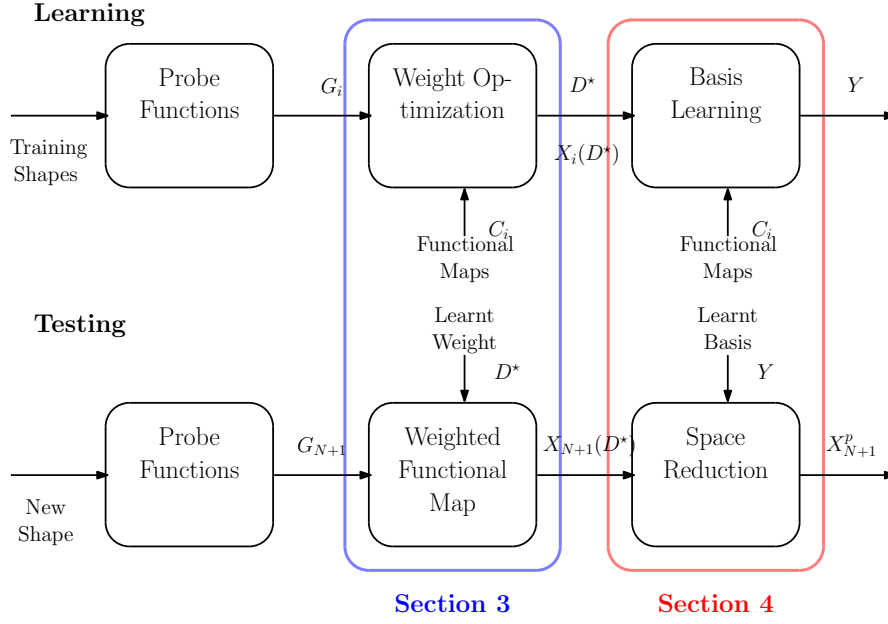


Figure 5.2 – Pipeline of the proposed algorithm with the corresponding section. Top: learning from a given collection. Bottom: processing a new shape.

5.3.1 Weighting the probe functions

As described above, our main idea is to introduce a set of weights on the functional correspondences to measure their usefulness in finding a relevant map by using a diagonal matrix D . For a given weight matrix D , the linear constraints given by the probe functions become $CG_0D = G_iD$. We can then define the function $X_i(D)$, which maps a given sets of weights to the corresponding functional map, via the solution of the optimization problem:

$$X_i(D) = \arg \min_C \|CG_0D - G_iD\|_F^2 + \alpha \|C \odot W\|_F^2 \quad (5.2)$$

We choose here to fix α and tune D . We could also try to tune all the parameters (α and D) but the coefficients would be defined up to a multiplicative constant and $C(D)$ may no longer be well-defined when α is equal to zero.

Since all the functional maps start from the reference shape M_0 , this shape obviously plays special role in our method. Ideally we would like to take as reference the most “average” shape of the collection. Following this idea, a simple procedure is presented in [111] to find the shape of the collection which minimizes the average isometric distortion. However, in our experiments we chose the symmetric standing pose as reference. Note that an interesting future work would be to use a more complex graph of correspondences between the shapes of a collection and impose cycle consistency as proposed in [54].

As discussed in the previous section $X_i(D)$ is well-defined and differentiable. Note that the weight matrix D has a global effect on $X_i(D)$, and the problem (5.2) cannot be separated in terms of the individual components of D .

5.3.2 Finding the best weights

Our goal is to find a set of weights which will provide a relevant set of functional maps. The main hypothesis is that the weights should be stable across deformations that belong to the same category. That is, the most relevant probe functions for the training shape pairs will also be well-suited to find a functional map to new unseen shapes in that category.

Learning from a given collection. We assume that we are given a collection of n nearly isometric deformations of the same object with known functional maps C_i . The optimal weights D^* are the ones that produce an approximation $X_i(D)$ that is closest to the ground truth C_i . Thus, we want to solve the following optimization problem:

$$D^* \in \arg \min_D \sum_{i=1}^n \|X_i(D) - C_i\|, \quad (5.3)$$

where the sum is over the set of given training maps C_i . Interestingly, the optimization problem 5.3 is closely related to optimal control problems as the functional maps $X_i(D)$ are themselves solutions of an optimization problem. Similar setting appears in [71] where a feedback loop is used to achieve a supervised dictionary learning.

Note that the choice of the norm is important. We would like the functional map $X_i(D)$ to match C_i over as-large-as possible functional subspace. This is equivalent to minimizing the rank of the difference $X_i(D) - C_i$. Thus, the important quantities are the singular values of the differences $X_i(D) - C_i$.

The naive choice of the squared Frobenius norm is not well suited for our problem since $\|A\|_F^2 = \|\sigma(A)\|_2^2$ where $\sigma(A)$ is the vector containing the singular values of the matrix A . Therefore this norm would give a large weight to the biggest singular values, which correspond to the worst-matched functional subspaces. Among these subspaces is the space of antisymmetric functions that we have no hope of mapping since the probe functions give us very little information about this subspace. At the same time, the small singular values have little influence on the minimization whereas they are the ones we would like to optimize.

The choice of the norm. To tackle the rank minimization problem we choose the following norm which is a regularization of the l^0 -norm:

$$\|A\|_\epsilon = \sum_{i=1}^n \frac{\sigma(A)_i^2}{\sigma(A)_i^2 + \epsilon}. \quad (5.4)$$

Note that the problem (5.3) is differentiable as long as $\|\cdot\|_\epsilon$ is differentiable. The gradient can be computed efficiently using the Jacobian matrix of the singular values as expressed in [92]. In practice, we solve this optimization problem using a standard L-BFGS algorithm. The choice of ϵ can have a big impact on the results. In fact since we are using a gradient descent method the big singular values are in the flat part of $\|\cdot\|_\epsilon$ therefore their gradient will be granted a small weight. On the contrary the singular values in the slope will have

a big influence on the minimization. So with an ϵ too small only a few singular values will be minimized but with an ϵ too large many singular values will not be well minimized. We chose the parameter ϵ such that at the initialization 80 percent of the singular values satisfy $\frac{\sigma_i^2}{\sigma_i^2 + \epsilon} \leq 0.9$.

5.4 Basis function extraction

Since the probe functions can give redundant information in some shape parts and incomplete information in others, the resulting functional map will map some subspaces of $L^2(M_0)$ with more confidence than others. Using a collection of shapes we would like to extract the most stable subspaces.

For this purpose we propose to use the learned optimal weights D and the resulting estimated functional maps $X_i(D)$ and to identify stably mapped functional subspaces by comparing $X_i(D)$ to the reference maps C_i . The input here is the same as in the previous section. We need n shapes with known ground truth functional maps C_i and a set of consistent probe functions G_i . The output will be Y an orthonormal basis of $L^2(M_0)$ ordered with decreasing confidence. As we demonstrate in Section 5.5, in most cases this order remains stable even for maps that are estimated to previously unseen shapes.

5.4.1 Identifying stable subspaces

The best mapped function $y_0 \in L^2(M_0)$ is such that $X_i y_0$ is the closest to $C_i y_0$ for all i . Such function is solution of the problem:

$$y_0 \in \arg \min_{y \in L^2(M_0), \|y\|=1} \sum_{i=1}^n \|(X_i - C_i)y\|_F^2$$

We can then iteratively define an orthonormal basis of $L^2(M_0)$ ordered by decreasing accuracy in the mapping, by solving the following problem:

$$y_{n+1} \in \arg \min_{y \in L^2(M_0), \langle y, y_j \rangle = 0 \forall j \leq n} \sum_{i=1}^n \|(X_i - C_i)y\|_F^2$$

Such a basis can be efficiently computed by considering the singular value decomposition of the matrix:

$$B = \begin{pmatrix} X_1 - C_1 \\ \dots \\ X_n - C_n \end{pmatrix} = U \Sigma V^t.$$

It is well-known that y_j must be equal to singular vectors corresponding to the j^{th} smallest singular value of B . We can, therefore, form a new orthonormal basis Y of $L^2(M_0)$ composed of the singular vectors of B by increasing singular values. This allows us to quantify the quality of the mapping of a functional subspace by looking at the singular values of B : the smaller the singular values are, the better the mapping.

5.4.2 Functional map to a test shape using a reduced basis

Now if we are given a new unseen shape M_{n+1} that does not belong to the training set, we first compute its probe functions and store them in a matrix G_{n+1} . We then compute the functional map X_{n+1} by using the previously solved for weight matrix D . Finally, since we know that X_{n+1} contains some badly mapped subspaces (for example the antisymmetric functions), by using Y_p the p first column of Y , we compute the reduced map X_{n+1}^p

$$X_{n+1}^p = X_{n+1}Y_p : L^2(M_0) \cap L^2(\text{Im}(Y_p)) \rightarrow L^2(M_{n+1}).$$

5.5 Experimental Results

5.5.1 Functional correspondences

The probe functions used to solve the problem in Eq. (5.2) are given by various descriptors computed on each shape:

- Heat Kernel Signature [119] for multiple values of t
- Wave Kernel Signature [5] at three different energies
- Gaussian and Mean Curvature
- Logarithm of the absolute value of Gaussian and Mean Curvature
- Mesh Saliency [67]

The HKS, WKS and Mesh Saliency are computed at various scales to ensure a wide variability. We process the curvature functions to obtain a family of descriptors. Since the curvatures can have very high peaks we take the logarithm of their absolute value to put more weight on the small curvatures areas. The family of functions is then created by considering the solution of the Heat Diffusion Equation at various times when each function is used as initial heat distribution over the surface.

5.5.2 Isometric Shape Matching

TOSCA Dataset. Our method is a tool which can be used in addition to other methods using the functional framework in order to improve the approximation of the functional maps. As several methods using functional maps [88, 90] have been shown to be more efficient than the state-of-the-art methods, we compare our trained maps to the baseline “original” method described in [88].

We have evaluated our method on the shape matching benchmark TOSCA [21]. For each shape class we use all the available shapes for training, except one for testing and we choose the standard undeformed pose as shape M_0 . We compare three ways of weighting the probe functions: a single weight for all the functions, a weight per category of descriptors and one weight per probe function.

For all the experiments we express all functions in the basis given by the first 50 eigenfunctions of the Laplace-Beltrami operator. We compute 50 probe functions divided in 9 categories (WKS is divided in three categories with three different energies) of descriptors. We take 5 functions per category except for the Mesh Saliency where 10 functions are computed. Since all the shapes in TOSCA have an internal symmetry, we cannot hope to recover the entire functional map, and thus Eq. (5.4) is a reasonable choice of norm.

The experiments follow the pipeline shown in Figure 5.2. First we learn the optimal weights and extract the ordered basis using the training set of shapes. Second we are given an unknown shape. We use the optimal weights to compute the functional map and the extracted basis to suppress the badly mapped function subspaces. The L-BFGS algorithm used to solve the optimization problem in Eq. (5.3) is initialized with the naive functional maps solution of (5.1) with $\alpha = 10^{-3}$. We compare all of the functional maps and subspaces computed with our method to the baseline “naive” map, obtained using the identity matrix D , which correspond to the original method described in [88].

Performance The proposed approach was implemented in MATLAB. Note that the number of vertices of each shape has no effect on the performance since all the functions are expressed in a reduced functional basis. The most time consuming task in our pipeline is the training part which requires to solve a difficult non-linear optimization problem (5.3). The processing cost is dominated by the computation of the gradient of the energy, which is done by solving two linear systems for each shape of the training set at each iteration. However, the contributions of each shape to the gradient are independent so this can be done in parallel. The learning process with a training set of 10 shapes took about 45 min on an Intel i7 processor without parallelization.

Optimal Weights Figure 5.3 (left) shows the weights obtained after solving the problem in Eq. (5.3) with a training set composed of 9 cats. To demonstrate the importance of weighting the probe functions on the quality of the functional map, we study the distribution of the singular values of the difference $X_{n+1}(D) - C_{n+1}$ for the different learned weights. In Figure 5.3 (right) each curve depicts the percentage of singular values below the threshold given on the x -axis. For the perfect map, all singular values would be zero. As can be seen, the functional maps with the optimal weights have a bigger concentration of small singular values than the naive functional map. Therefore there exists a bigger functional subspace on which these functional maps provide a good approximation of the ground truth. Note that the naive map has no small singular values and is indeed a very bad approximation.

Stable subspaces From the naive maps and functional maps with optimal weights, we extract four function bases ordered by decreasing stability. The most stable functions for each case are shown in Figure 5.4. Even the most stable functions from the naive maps are not mapped very accurately since they are very bad approximation of the groundtruth. For the other bases the functions seem consistent with the information we would expect from descriptors as HKS and WKS: a distinction between flat area (body)

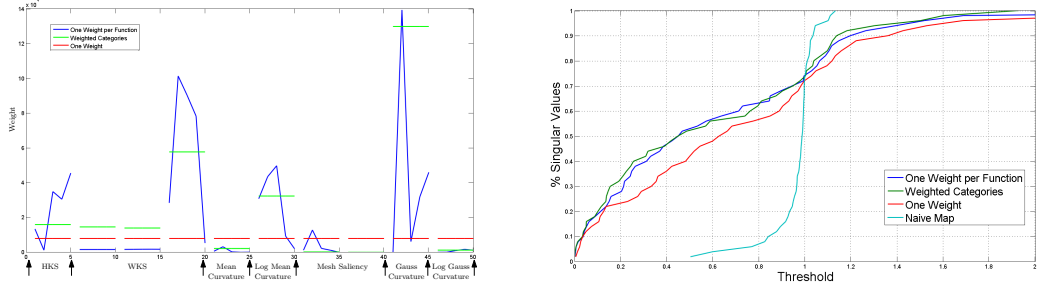


Figure 5.3 – Left: Optimal weights for different strategies after training with 9 cats. Right: Effect of the different weights on the distribution of the singular value of the difference $X_{n+1}(D) - C_{n+1}$.

and salient area (legs, tail, head). Note that even with only one weight we are able to retrieve meaningful stable areas.

We also evaluate the extracted functional basis by computing the difference between the ground truth map and our approximation on the unseen shape:

$$\epsilon_i = \|(X_{n+1}(D) - C_{n+1})y_i\|_2, \quad (5.5)$$

where y_i is the i^{th} function of the extracted basis. We compare this error for the three weighting strategies with the naive map in Figure 5.5 (left). The extracted basis was ordered by decreasing quality on the training set. Note that this order is still preserved on the unknown shape for the 25 first functions of the basis. Most of all we are able to identify the worst mapped subspaces, which can be safely removed.

Despite only estimating the functional maps on a subspace of the full functional space, we converted them to point-to-point correspondences using the method described in [88]. Figure 5.5 right compares the quality of the point-to-point correspondences before and after reducing the space dimension from 50 to 25. We obtain better results with our learned weight than with the naive map. For the weighted maps, the reductions perform better or similarly than the full maps. Thus our basis extraction manages to identify correctly the most stable subspaces. For the naive map our reduction space strategy fails as there is no well-mapped subspace.

5.5.3 Non-Isometric Shape Matching

Until now we have assumed the deformation to be nearly isometric. Our algorithm to find the optimal set of probe functions and the extraction of the most stable subspaces do not contain any explicit knowledge of the type of deformation. In fact, this assumption is only used to construct the least-squares problem (5.1). Which means that our framework can be adapted to any kind of deformation model as long as we have a consistent way of computing a functional map from probe functions and a reduced basis that is approximately stable.

To test this case, we consider a man or a gorilla and 12 women in different poses from the TOSCA dataset. The ground truth functional maps are computed using a thousand

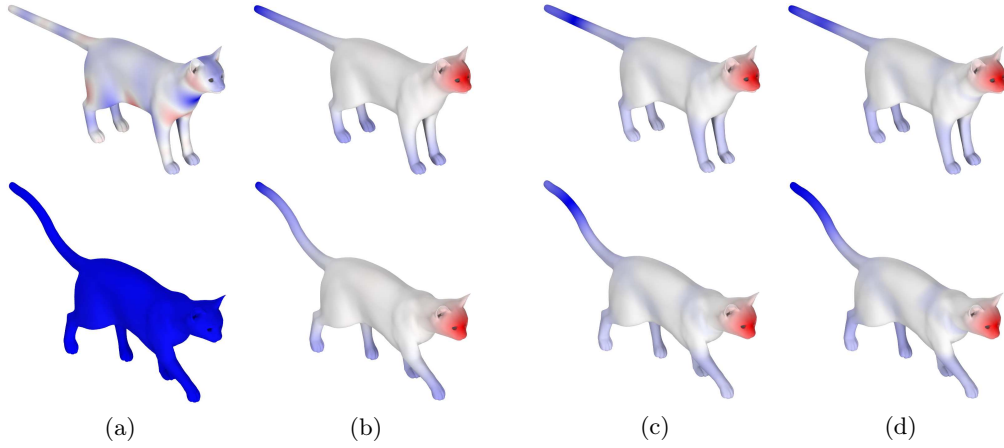


Figure 5.4 – Visualization of the first component of the extracted basis. First row: the reference shape. Second row: transfer using different functional maps: (5.4a) naive map, (5.4b) unique weight, (5.4c) one weight per category of function, (5.4d) one weight per functions.

user-picked correspondences. The meshes have different number of vertices and different connectivity. We train our method on 10 poses and use the last for testing.

The resulting problem is very noisy for two reasons. First, the ground truth functional maps are computed from sparse correspondences, and therefore can be inaccurate on some functional subspace. However, the use of a collection of shape for training allows us to remove this noise. Second, since all the probe functions used are designed for isometric deformation, few are going to contain useful information.

In order to introduce a wide variability of functions we pre-compute on each shape 50 basis functions, 310 probe functions and we put a weight on each probe function. The optimization algorithm is initialized as in the previous experiment. Figure 5.7 (left) shows the most stable functions learned from the training maps. Each of these functions is also mapped to a previously unseen pose using the “ground truth” map converted to a functional map. Note that the functions are badly transferred, due to the incompatibility of the LB basis and the noise in the input maps. Compare this with Figure 5.7 (right) where the probe functions have been weighted using our method. The stable functions indicate the head, the hands and the feet to be the most stable area. Besides, these functions are correctly mapped on a new shape using *a computed* functional map with the learned weights. Clearly, the fact that we use a collection helps removing the noise of the input data. Thus, our method is able to correctly identify the most stable functional subspace under mild assumptions on the underlying deformation.

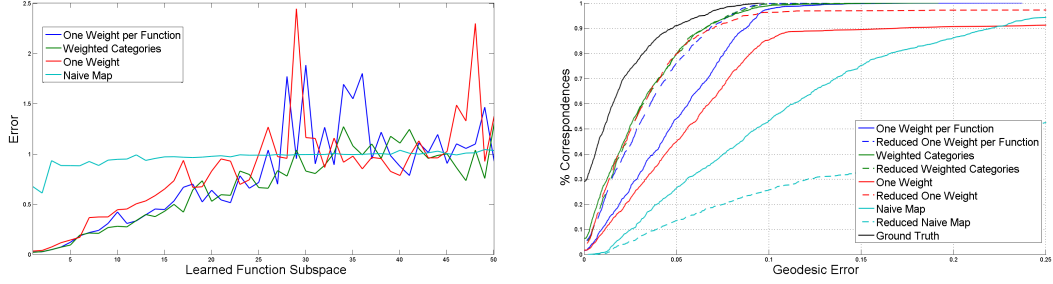
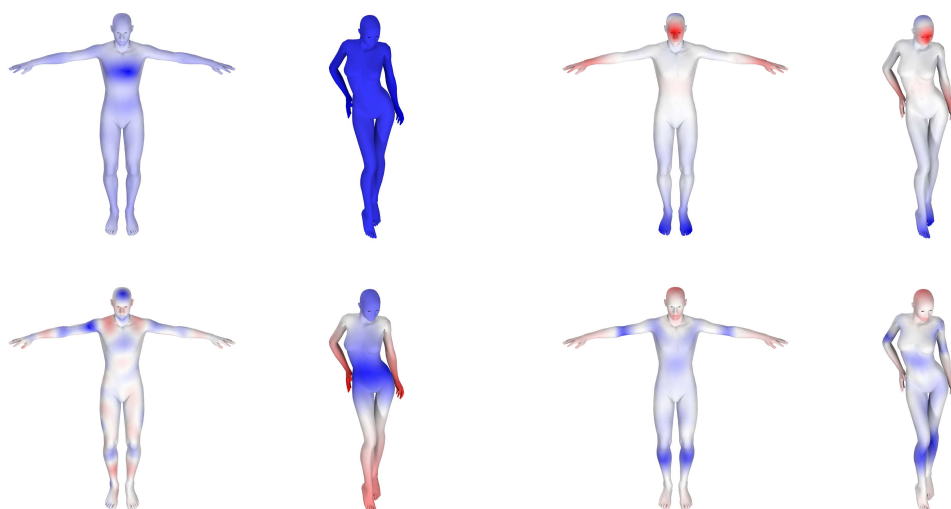


Figure 5.5 – Left: Accuracy of the extracted function basis measured with Eq. (5.5). Right: Comparison between full map (plain lines) and reduced map (dash lines). Symmetric correspondences are considered correct.

5.6 Conclusion

In this work, we presented a method to learn the most informative descriptors for non-rigid shape matching, from a given set of shape correspondences. Our method is purely intrinsic and allows us to obtain high quality *consistent* correspondences to new, unseen shapes, and to identify the most reliably mapped functional subspaces. The approach is flexible and can potentially be applied to scenarios that lack a good theoretical deformation model, as demonstrated by meaningful even non isometric deformation. One of its weaknesses is a relatively high cost for the training due to the non-convex nature of the energy. In the future, we plan to explore more efficient optimization strategies.

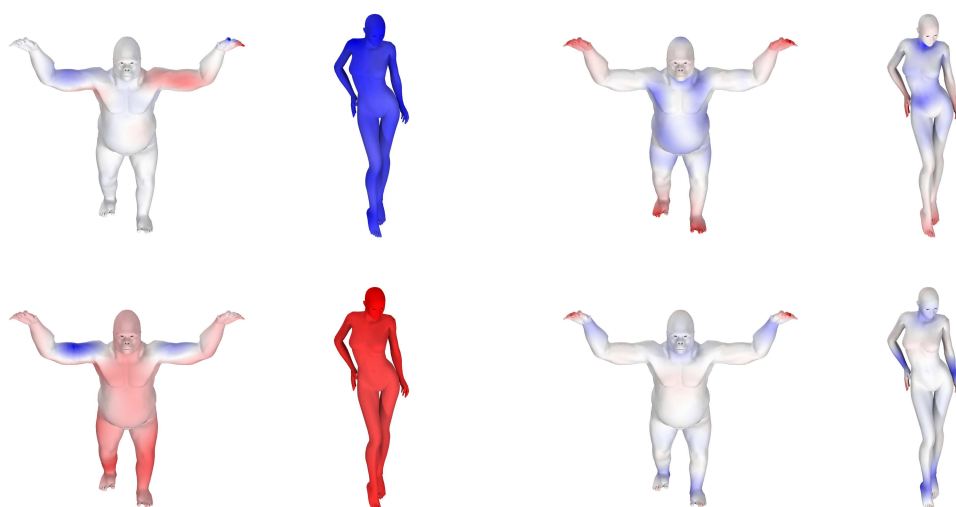
In this chapter we proposed an efficient method to compute functional maps. However in many practical scenarios, one would like to obtain pointwise correspondences between the given shapes. Thus, in the following chapter we present a method for converting functional maps to point-to-point correspondences, that enforces continuity in the resulting map.



Stable functions before training.

Stable functions after training.

Figure 5.6 – Visualization of the first two components of the extracted basis on the reference shape (man) then mapped to the unseen shape (woman) using the functional map without optimization (left) and with the learned weight (right).



Stable functions before training.

Stable functions after training.

Figure 5.7 – Visualization of the first two components of the extracted basis on the reference shape (gorilla) then mapped to the unseen shape (woman) using the functional map without optimization (left) and with the learned weight (right).

Continuous Matching via Vector Field Flow

In this chapter we present a new method for non-rigid shape matching designed to enforce continuity of the resulting correspondence. Our method is based on the functional map representation, which allows efficient manipulation and inference but often fails to provide a continuous point-to-point mapping. We address this problem by exploiting the connection between the operator representation of mappings and flows of vector fields. In particular, starting from an arbitrary continuous map between two surfaces we find an optimal flow that makes the final correspondence operator as close as possible to the initial functional map. Our method also helps to address the symmetric ambiguity problem inherent in many intrinsic correspondence methods when matching symmetric shapes. We provide practical and theoretical results showing that our method can be used to obtain an orientation preserving or reversing map starting from a functional map that represents the mixture of the two. We also show how this method can be used to improve the quality of maps produced by existing shape matching methods, and compare the resulting map’s continuity with results obtained by other operator-based techniques.

6.1 Introduction

Computing correspondences or mappings between 3D shapes is one of the key building blocks in many areas of digital geometry processing, including deformation transfer [118], shape interpolation (morphing) [61] and statistical shape analysis [49] among many others. This problem is particularly challenging in the case of shapes undergoing non-rigid deformations, where the notion of the optimal map may be difficult to define and optimize for.

As mentioned in the previous chapter, most of the successful global methods proposed to find correspondences between pairs of non-rigid shapes in the recent years have relied on a variant of the conformal [131, 69, 62] or fully isometric [22, 123, 107, 88] deformation models, which assume that either the angles or the geodesic distances between pairs of points are approximately preserved by the mapping. Although such models have very appealing theoretical properties, using them directly can often lead to difficult non-linear, non-convex optimization problems [22]. Therefore, most recent works in this direction have concentrated on finding a low-dimensional parameterization of the space of mappings, that allows for efficient optimization techniques (e.g. [69, 89, 23]).

Among such low-dimensional representations of the space of correspondences, one particularly appealing approach is based on the framework of functional maps [88], which

consider mappings as linear operators between the corresponding function spaces. One of the weaknesses of the functional map representation, however, is that by representing mappings as correspondences between functions, it requires an additional post-processing step to obtain a point-to-point map after computing the optimal functional map. The basic approach for this conversion step, proposed in [88] and used in most follow-up works, assigns points by considering the mapping between the corresponding Dirac delta-functions. Since each delta-function is mapped independently, however, this approach can (and most often does) introduce significant artifacts and discontinuities into the final point-to-point mapping (see the first two columns of Figure 6.6). This makes the resulting correspondences unusable in settings that require continuity of the mapping, such as texture transfer. Additional pair-wise terms can potentially be introduced in the conversion procedure, but this would require creating variables for points with potentially very expensive consistency constraints, which very quickly loses the appeal of the functional map framework, and reduces to direct optimization.

In this context, we propose a novel method for converting a functional map to a point-to-point map, which guarantees continuity and does not rely on any pairwise consistency constraints, making it computationally efficient. Our main idea is to represent the target point-to-point map as a composition of an arbitrary continuous map between the two surfaces and a flow associated with an unknown vector field on one of them. By relying on the recently proposed operator representation of vector fields [6], we show that the optimal vector field can be computed efficiently entirely within the functional map framework, and the computation of the final map requires a single discretization of vector field advection. We also employ the supervised learning technique presented in Chapter 5 that not only helps to obtain better functional maps but also helps to identify functional subspaces where the map is reliable, which significantly helps to improve the final point-to-point map.

Our method also helps to address the symmetric ambiguity problem inherent in many intrinsic correspondence methods when matching symmetric shapes. We provide practical and theoretical results showing that our method can be used to obtain an orientation preserving or reversing map starting from a functional map that represents the mixture of the two. Finally, we test our method on a shape collection and show that we can produce maps that are both continuous and have smaller geodesic distortion compared to the results obtained by existing techniques.

6.2 Related Work

Below we concentrate on the recent works that are directly related to ours, consisting of methods for global near-isometric shape matching with special emphasis on approaches that guarantee the continuity of the resulting maps.

As mentioned in the introduction, most of the existing techniques for non-rigid shape matching use a deformation model for finding correspondences between 3D shapes. The two most common models in this setting include approximate intrinsic isometries and conformal mappings. The former model, which was originally introduced by Bronstein

et al. [22] and Mémoli [74] assumes that pairwise geodesic distances are approximately preserved by the deformation. The first works that use this assumption lead to continuous maps by design, but result in very challenging optimization problems that are difficult to solve with more than a small number of points [22]. As a result, many follow-up techniques have used a relaxed version of the isometric mapping assumption, which result in more manageable optimization problems, but can often fail to guarantee a low distortion continuous mapping (e.g., [55, 123, 89, 107, 23]). Furthermore, an additional challenge in using the isometric model assumption is that exact intrinsic isometries are extremely rare, both in theory [44] and in practice, since most deformable shapes induce some amount of distortion.

Another set of successful techniques, which are more widely applicable than those based on the isometric mapping assumption are those that assume that the mapping is conformal, and thus only preserves angles (e.g., [48, 131, 56, 69, 62]). These techniques are appealing because a conformal mapping is known to exist between any pair of shapes with the same topology, but also because the set of such mappings can be parameterized relatively easily by using a canonical domain, such as a sphere for genus zero surfaces. Moreover, the resulting maps obtained by these approaches are typically continuous. At the same time, conformal mappings can often induce large area distortion, which can result in unrealistic correspondences between non-rigid shapes, which limits their use significantly.

A recent set of approaches that overcome the above-mentioned challenges to some extent is based on the functional map representation, introduced in [88]. As mentioned above, this framework is based on representing maps as linear operators acting on real-valued functions, and which can be encoded compactly by small-sized matrices in the discrete setting by using a multi-scale basis. Although the original approach and the follow-up works, including [65, 96], all implicitly use the isometric deformation assumption, they have been shown to be very robust to small non-isometric distortions, by extensive use of strong geometric and linear-algebraic regularization techniques. Moreover, several recent works have shown how this framework can be used in the supervised learning setting, where functional maps between unseen shapes can be obtained by exploiting information present in a small set of example maps including [100] and Chapter 5.

Despite its practical appeal, one of the limitations of the functional map framework, is that a post-processing step is necessary to convert a functional map to a point-to-point one. The method used in [88] is based on mapping Dirac delta functions. However as the points are considered independently the continuity of the resulting map is not ensured. This problem can be particularly prominent in shapes that contain intrinsic symmetries, which contain at least two equally good solutions for the optimal functional map, and the computed one is at best a linear blending of the two.

Note that, closely related to our technique, especially in the use of flows for computing continuous maps (diffeomorphisms) is the LDDMM framework [10, 76], widely used in the medical imaging community. Unlike these methods, however, our approach is purely intrinsic and operates directly on the surface of the target model, rather than deforming a template in space.

Contributions In this chapter we propose a novel method for converting a functional map into a point-to-point one, which combines the strengths of the functional map framework that allows to compute low-distortion functional maps, with those of the conformal mapping approaches, which produce continuous correspondences. Namely, starting from a map computed using the state-of-the-art conformal-based Blended Intrinsic Map approach [62], we modify it by computing the optimal vector field, whose flow, composed with the original map, would result in a functional map as close as possible to the given one. By using the recently proposed operator representation of vector fields [6] and the connection between advection and matrix exponentiation, we propose an efficient optimization approach for computing the optimal vector field entirely within the functional map framework. Moreover, we show theoretically that this approach is guaranteed to produce the correct continuous map when the input functional map represents a blending of the orientation preserving and reversing maps under certain assumptions, and demonstrate this projection step in practice.

6.3 Functional Maps Conversion

Once the functional map C is computed following the pipeline proposed in [88] or by Chapter 5, the goal of the second step is to convert it to a point-to-point map. The method proposed in [88] and reused in most of the follow-up work consists in finding the nearest neighbors of the images of Dirac-delta functions on M by C among the Dirac functions on N . Namely, for each point $p \in M$, the map: $\varphi(p)$ is computed as via $\varphi(p) = \arg \min_q \|\delta_q - C\delta_p\|$, where δ_p is an indicator function on point p , written in the appropriate basis.

6.3.1 Main Challenges

While both steps described above are very efficient in practice, the second stage has a very serious limitation, in that it processes each point independently, meaning that the final map φ may not be (and often is not) continuous. The first two columns of Figure 6.6 provide examples of discontinuous maps resulting from this conversion.

To illustrate this phenomenon, let us assume that the target shape N has an orientation-reversing (reflectional) intrinsic symmetry $S : N \rightarrow N$. In this case, there exist at least two equally good potential solutions for Eq. (5.1) and similarly, each point x may have several candidate correspondences.

In practice the functional constraints are often not sufficient to resolve symmetric ambiguities, in large part because most robust descriptors are invariant under intrinsic isometries. The best we can hope for when approximating C_φ is an exact functional map for symmetric functions (i.e. f , s.t. $f \circ S = f$) and a noisy or zero functional map for antisymmetric functions (i.e. $f \circ S = -f$). Since our approximations are obtained by solving a linear system, most likely a solution of the least squares problem will be a linear blending between the orientation preserving and reversing functional map:

$$C_\varphi^\alpha = (1 - \alpha)C_\varphi + \alpha C_{\varphi \circ S} \quad (6.1)$$

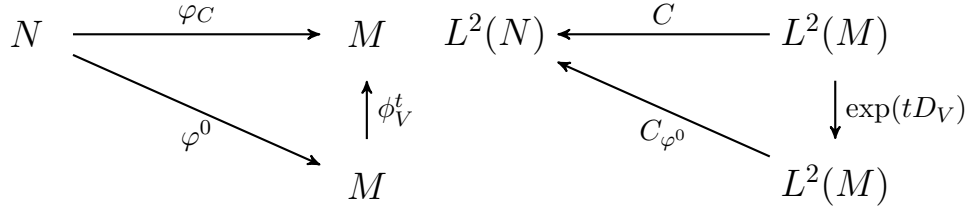


Figure 6.1 – Left: the unknown continuous map φ_C is a composition of the input φ^0 and the flow ϕ_V^t of a vector field V . Right: dual representation as functional maps.

Note that $\alpha = 0.5$ implies that all antisymmetric functions are mapped to zero.

The conversion of C to a point-to-point map in itself gives no guaranty of continuity in the resulting map. Since each Dirac function of a point x is treated independently it can be mapped indifferently to its image $\varphi^{-1}(x)$ or to its symmetric alternative $S(\varphi^{-1}(x))$. Moreover this process is not designed to be stable under the blending noise α , as in (Eq. 6.1).

In this context, the key idea developed in this chapter is to construct a point-to-point map from the functional map C by following a procedure that *guarantees* continuity, while being robust to blending noise. In particular, starting from an arbitrary continuous map between M and N , we find an optimal vector field, whose flow makes the final correspondence operator as close as possible to the given (e.g., computed) functional map. Since the flow of a vector field provides a continuous, and orientation-preserving map, the final correspondence is both continuous and has the orientation of the initial map. As we show below, this can significantly improve the quality of the resulting point-to-point map, while remaining computationally tractable and avoiding expensive second-order pairwise constraints.

6.3.2 Algorithm Overview

The algorithm proposed in this chapter takes as input a functional map $C : L^2(M) \rightarrow L^2(N)$ and an arbitrary continuous map $\varphi^0 : N \rightarrow M$. It then outputs a continuous point-to-point map $\varphi_C : N \rightarrow M$.

As mentioned above, the main idea of our algorithm is to construct the map φ_C by composing φ^0 with the flow ϕ_V^t of a well-chosen vector field V (see Figure 6.1). We will choose the vector field V such that $\phi_V^t \circ \varphi^0$ represented as a functional map is as close as possible to the input C . This can be done efficiently by representing ϕ_V^t as an operator (Section 6.4) and then solving a small-scale optimization problem as explained in Section 6.5. To find the map φ_C we solve a system of ODEs with a simple solver (Section 6.6).

The main steps of the proposed algorithm are described in Algorithm 1.

Algorithm 1: FUNCTIONAL MAP CONVERSION

Input : $C : L^2(M) \rightarrow L^2(N)$ functional map
 $\varphi^0 : N \rightarrow M$ initial continuous map

Output : φ_C : C converted into a continuous map

- 1 Find Optimal Vector field (Section 6.5);
- 2 Convert φ^0 to a functional map C_{φ^0} ;
- 3 Solve: $a^* \in \arg \min_{a \in \mathbb{R}^n} \|C_{\varphi^0} \exp(\sum_{i=1}^n a_i D_{V_i}) - C\|_\phi$;
- 4 Set: $V := \sum_{i=1}^n a_i^* D_{V_i}$;
- 5 Compute φ_C (Section 6.6);
- 6 Solve: $\frac{d}{dt} \phi_V^t(p) = V(\phi_V^t(p))$, $\phi_V^0(p) = p \in N$;
- 7 **return** $\varphi_C := \phi_V^1 \circ \varphi^0$;

6.4 Family of Diffeomorphisms

In this section we construct a family of diffeomorphisms which map N onto M and derive their representation as functional maps. The point-to-point map which converts the given functional map C will be chosen among this family.

Vector field flow Given a family of tangent vector fields $\{V_i\}_{1 \leq i \leq n}$ on M , we let \mathcal{V} be the space spanned by the linear combinations of the V_i . Any vector field $V \in \mathcal{V}$, defines a one-parameter family of maps $\phi_V^t : M \rightarrow M$ called the flow of V . The flow is formally defined as the unique solution to the differential equation:

$$\frac{d}{dt} \phi_V^t(p) = V(\phi_V^t(p)), \quad \phi_V^0(p) = p \in N. \quad (6.2)$$

Given an arbitrary diffeomorphism $\varphi : N \rightarrow M$ we construct a family of diffeomorphisms \mathcal{T} parametrized by $t \in \mathbb{R}$ and $a \in \mathbb{R}^n$:

$$\varphi_a^t(p) = \phi_{V_a}^t \circ \varphi(p), \quad V_a = \sum_{i=1}^n a_i V_i \quad (6.3)$$

Remark that the orientation of a map $\varphi_a^t \in \mathcal{T}$ is given by the orientation of φ since the flow of a vector field is orientation preserving.

Functional Representation of the family The family of mappings \mathcal{T} has an easy representation in the functional map framework as explained in [6]. This is because, a vector field V on a smooth manifold can be represented as an operator D_V acting on a function f :

$$D_V(f)(p) = \langle V_p, \nabla f(p) \rangle_p. \quad (6.4)$$

Since the action of D_V is linear, the operator is conveniently represented as a matrix in the discrete setting.

It is well known that $g_t = f \circ \phi_V^t$ is the unique solution of the PDE:

$$\frac{\partial g}{\partial t}(t, p) = D_V(g)(t, p), \quad g(0, p) = f(p).$$

A key property of the operator representation of vector fields, introduced in [6] is that for analytic functions the functional map $C_{\phi_V^t}$ is represented by the exponential of the operator D_V since one has:

$$C_{\phi_V^t} f := f \circ \phi_V^t = \exp(tD_V)(f)$$

Since map composition is achieved via matrix multiplication in the functional representation, this yields a simple way of describing our family of diffeomorphisms \mathcal{T} . Let $\varphi_{\mathbf{a}}^t \in \mathcal{T}$ then

$$C_{\varphi_{\mathbf{a}}^t} = C_{\varphi} \exp \left(t \sum_{i=1}^n a_i D_{V_i} \right). \quad (6.5)$$

6.5 Optimal vector field

6.5.1 Optimization Problem

Our main idea, developed in the section, is to project the input functional map C onto the appropriate set of diffeomorphisms \mathcal{T} . Namely our goal is to find a vector field $V \in \mathcal{V}$ such that the operator representation (6.5) of $\varphi_{\mathbf{a}}^t$ is as-close-as possible to C . This projection is easily written thanks to the operator representation, and computationally it reduces to solving the optimization problem:

$$\min_{a \in \mathbb{R}^n} \|C_{\varphi} \exp \left(\sum_{i=1}^n a_i D_{V_i} \right) - C\|_{\phi}, \quad (6.6)$$

for an appropriate choice norm $\|\cdot\|_{\phi}$. Here we note briefly that the norm is chosen to be differentiable as indicated in Section 5.3.2.

In practice, the problem (6.6) can be solved using a first order method such as the L-BFGS algorithm. The main difficulty in finding the gradient of the objective function lies in the computation of derivative of $\exp(\sum_{i=1}^n a_i D_{V_i})$ in the direction V_j . While there exists a vast literature on approximating the exponential of a matrix (for a survey see [77]), to the best of our knowledge few methods address the problem of computing the directional derivative of the matrix exponential, which is conceptually non-trivial. As we show in Lemma 6.5.1, however, the directional derivative can be obtained as a block of the matrix exponential of a bigger operator. Note that if there are n vectors in the family of vector fields we have to compute n matrix exponentials.

Lemma 6.5.1 *The directional derivative $d_H e^{tA}$ defined as*

$$d_H e^{tA} = \lim_{h \rightarrow 0} \left(\frac{1}{h} (\exp(t(A + hH)) - \exp(tA)) \right),$$

is one block in the matrix exponential of a bigger operator:

$$\begin{pmatrix} e^{tA} & B_t \\ d_H e^{tA} & C_t \end{pmatrix} = \exp \begin{pmatrix} tA & 0 \\ tH & tA \end{pmatrix}$$

Proof Let x_0 be an arbitrary vector in \mathbb{R}^n and $x(t) = \exp(tA)x_0$, it is well known that $x(t)$ satisfies the ODE:

$$x'(t) = Ax(t), \quad x(0) = x_0.$$

Moreover $x_h(t) = \exp(t(A + hH))x_0$ is solution of

$$x'_h(t) = (A + hH)x_h(t), \quad x_h(0) = x_0.$$

We denote $y(t)$ the directional derivative in the direction H :

$$\begin{aligned} y(t) &:= \lim_{h \rightarrow 0} \left(\frac{1}{h} (\exp(t(A + hH)) - \exp(tA)) x_0 \right) \\ &:= \lim_{h \rightarrow 0} \left(\frac{1}{h} (x_h(t) - x(t)) \right). \end{aligned}$$

After computing $y'(t)$ we conclude that $y(t)$ is the unique solution of the ODE

$$\begin{cases} x'(t) = Ax(t), & x(0) = x_0 \\ y'(t) = Ay(t) + Hx(t), & y(0) = 0 \end{cases}$$

Expressing the solution as an exponential of a matrix leads to the results. \square

6.5.2 Properties

One of the advantages of the formulation of the problem of finding the optimal point-to-point map from a functional map via Eq. (6.6) is that it makes no assumptions on the input map C . This is particularly important since, as mentioned above, in the presence of intrinsic symmetries the functional map C can, even in the best case, be a linear blending of the functional representation of an orientation-preserving and orientation-reversing map. However, one potential problem is that the presence of the “noisy” part in the functional map can adversely affect the final output map φ obtained by optimizing Eq. (6.6).

Fortunately, both in theory and in practice this is not the case. Namely, under some suitable assumptions, the orientation-preserving ground-truth functional map, must be a local minimum of the problem (6.6) even when the functional map C is given by the symmetry blending defined at (6.1). In particular, as we show in the Lemma 6.5.2, C_φ must be a local minimum of Eq. (6.6) under some assumptions.

Lemma 6.5.2 *If the norm $\|\cdot\|_\phi = \|\cdot\|_F^2$ is the squared Frobenius norm, the set of vector fields considered \mathcal{V} is divergence-free and the initial transformation φ approximately isometric, then C_φ must be a local minimum of Eq. (6.6).*

Proof The first order necessary condition for C_φ to be a local minimum reads:

$$\forall dC_\varphi \in \mathcal{T}(C_\varphi), \quad \langle dC_\varphi, C_\varphi - C_\varphi^\alpha \rangle_F = 0,$$

where $\mathcal{T}(C_\varphi)$ is the tangent space of the set of all functional map at the point C_φ . This tangent space has a simple expression. Let's consider a small perturbation of C_φ by the flow ϕ_V^t induced by the vector field $V \in \mathcal{V}$ applied to an arbitrary function f :

$$\begin{aligned} \lim_{t \rightarrow 0} \frac{1}{t} (C_{\phi_V^t \circ \varphi} f - C_\varphi f) &= \lim_{t \rightarrow 0} \frac{1}{t} (f \circ \phi_V^t \circ \varphi - f \circ \varphi) \\ &= \frac{d}{dt} (f \circ \phi_V^t \circ \varphi) |_{t=0} \\ &= \langle V, \nabla f \rangle \circ \varphi \\ &= C_\varphi(D_V(f)). \end{aligned}$$

Additionally the deformation is nearly isometric so $C_\varphi^\top \approx C_{\varphi^{-1}}$ see Theorem 4.1.4. The necessary condition becomes:

$$\forall V \in \mathcal{V}, \quad (1 - \alpha) \langle D_V, I - C_{\varphi \circ S \circ \varphi^{-1}} \rangle_F \approx 0.$$

Remark that the mapping $\varphi \circ S \circ \varphi^{-1} : M \rightarrow M$ is an intrinsic symmetry on M . Now suppose that the basis function is composed only by even and odd functions with respect to the symmetry S . Therefore the functional map associated to an internal symmetry is a diagonal matrix with 1 and -1 on the diagonal corresponding to the symmetric and antisymmetric eigenfunctions. So $I - C_\varphi^\top C_S C_\varphi$ is approximatively a diagonal matrix. By assumption the vector field V is divergence free therefore represented by a skew-symmetric operator D_V as explained in [6]. Since the result does not depend on the basis C_φ is a critical point of (6.6). \square

6.5.3 Practical Choice of the Norm

As stated before C is not reliable for antisymmetric functions. Therefore there is some function subspace on which C and $C_\varphi \exp(t \sum_{i=1}^n a_i D_{V_i})$ cannot agree. The choice of the norm $\|\cdot\|_\phi$ in the problem (6.6) is of critical importance. Similarly to setting in Chapter 5 above, the naive choice of the squared Frobenius norm is not well-suited for this problem since it is the sum of the squared singular values. As such, it will give a large weight on badly matched function subspace and a small weight on well matched function subspace. However, since typically we have almost no information about antisymmetric functions so the optimization problem based on this problem will put a lot of effort matching functions that we cannot hope to match and few matching interesting subspace. A better choice for $\|\cdot\|_\phi$ is a regularization of the nuclear norm. We choose

$$\|A\|_\phi = \|AY_p\|_{\epsilon, \star} \tag{6.7}$$

where Y_p is a basis of p functions that we want to focus on obtained using the approach described in Chapter 5 and based on Chapter 5. In the unsupervised setting we chose Y_p

to be the identity. The norm $\|\cdot\|_{\epsilon,\star}$ is defined by $\sum_{i=1}^n \sqrt{\sigma_i^2 + \epsilon}$ the σ_i are the singular values of the matrix. With this norm, we give smaller weight to the subspaces that are difficult to align and focus on task we are able to complete.

The parameter ϵ makes the function $\|\cdot\|_{\epsilon,\star}$ differentiable and should be small and is taken at 10^{-3} . Note that the Jacobian matrix of the singular values is easily computable as explained in [92].

6.6 Vector Field Flow on Manifold

Once the optimal vector field V is found using the procedure described above, we obtain the final point-to-point map by composing the initial map φ with the flow of V . To compute this flow, we need to solve the system of equations (6.2) on the given triangle mesh. In principle any advection solver will work with our method. However since computing the flow is known to be potentially difficult, we implemented our own solution. The implementation we use gives a coarse approximation of the flow and might not be accurate for very large deformations. For more accurate solution of this problem we refer to [97, 84] which provide more guaranties of continuity of the flow and faster convergence. In all of our applications we assume that the shapes are given as triangulated meshes and the vector field is given as a single vector per face. Given this representation, we assume that the vector field is constant per face and is interpolated at the edges. Three main situations can occur: the current point could be at inside face, an edge or a vertex.

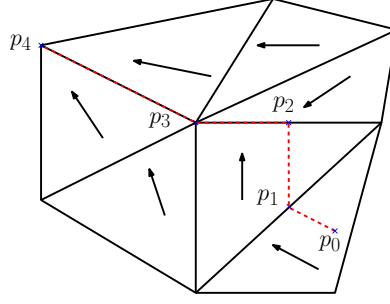
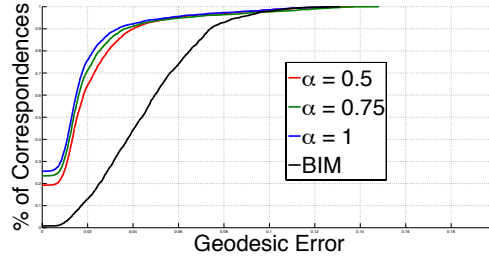
At a Face Since inside a face the vector field is assumed to be constant, we follow it until we reach an edge or a vertex (Figure 6.2 from p_0 to p_1).

At an Edge When a point is at an edge, we try to cross the face we did not come from (Figure 6.2 from p_1 to p_2). If the point did not move we follow the edge by interpolating the vector field from the two neighboring faces and end up at a vertex (Figure 6.2 from p_2 to p_3).

At a Vertex When the point is at a vertex, we try to follow the vector of each of the neighboring faces and choose the one that goes the furthest. If the point cannot move, we try to follow the neighboring edges, using interpolated directions and to potentially end up at another vertex (Figure 6.2 from p_3 to p_4).

6.7 Results

For all the experiments we express all functions in the basis given by the first 150 eigenfunctions of the Laplace-Beltrami operator. We choose a family of 50 tangent vector fields for the V_i given by the first eigenfunctions of the 1-form Laplace-de Rham operator, constructed following the procedure described in [41].

Figure 6.2 – Example of a path trace starting point at p_0 .Figure 6.3 – Impact of the noisy functional map C_φ^α on the point-to-point correspondences for various values of α .

We have evaluated our method for computing point-to-point correspondences on the shapes on the benchmarks of Anguelov and al. [4] and of Bronstein and al. [21]. In all of the cases, the input continuous map φ^0 is the result of the BIM algorithm [62]. This map is most of the time continuous but can be very distorted in some areas. We will show that our method is able to detect the distorted areas and correct them.

6.7.1 Symmetry Blending

As stated above a plausible perturbation for the input functional map C is given by equation (6.1). We test our method when C is the linear blending of the ground-truth functional map and the ground-truth orientation reversing functional map for various values of α . In this experiment Y_p , in Equation 6.7, is the identity matrix. For this experiment we choose a pair of shapes from the SCAPE dataset.

The graph shown in Figure 6.3 shows the percentage of correspondences with a geodesic error smaller than a threshold. Of course the closer C_φ^α is to the ground-truth map the better are the correspondences. However our results are robust even when the target functional map is an exact blending of the direct and symmetric map and are always better than the map coming from BIM. Thus, even when the assumptions of our theoretical observation are not fulfilled, our method can successfully retrieve meaningful information from noisy data.

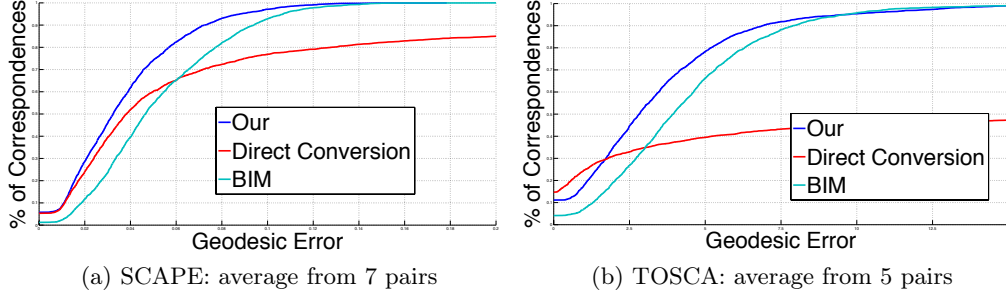


Figure 6.4 – Improvement of the BIM map using our method.

6.7.2 Error using a computed functional map

In a more realistic scenario, rather than using a ground-truth functional map, we compute it via the inference pipeline described in Chapter 5. In this section the experiments are conducted on several pairs of shapes: 7 human pairs (SCAPE) and 5 animal pairs (TOSCA). The functional map C is computed using the least squares problem (5.1), where each functional constraints is weighted. The weights are learned by solving problem (5.3) using the algorithm described in Chapter 5 which also outputs a matrix Y_p corresponding to the p best mapped functions, where we let p equal to 70. The training set is composed of 8 randomly chosen meshes for the SCAPE example and 4 meshes for the TOSCA centaur example. We compute 310 functional constraints equally distributed among these categories:

- Heat Kernel Signature [119],
- Wave Kernel Signature [5] at three different variances,
- Gaussian and Mean Curvature,
- Logarithm of the absolute value of Gaussian and Mean Curvature,
- Mesh Saliency [67].

We compare our approach with BIM, that serves as φ^0 , and with the functional map C converted to point-to-point map using the method proposed in [88]. The graph in Figure 6.4 shows the percent of correspondences which have geodesic error smaller than a threshold in average for SCAPE and TOSCA. In this case, we only accept direct correspondences as correct, and consider symmetric points as wrong. Note that our method shows quality improvement over Blended Intrinsic Maps. The direct conversion of C have some point with very large geodesic error due to points mapped to their symmetric counterparts.

We evaluate the continuity of our map with two measures of distortion. First the maximum radius corresponding to a geodesic ball of given size. For a map φ this is formally given by the function:

$$r(t) \mapsto \max_{d_N(p,q) \leq t} d_M(\varphi(p), \varphi(q)), \quad (6.8)$$

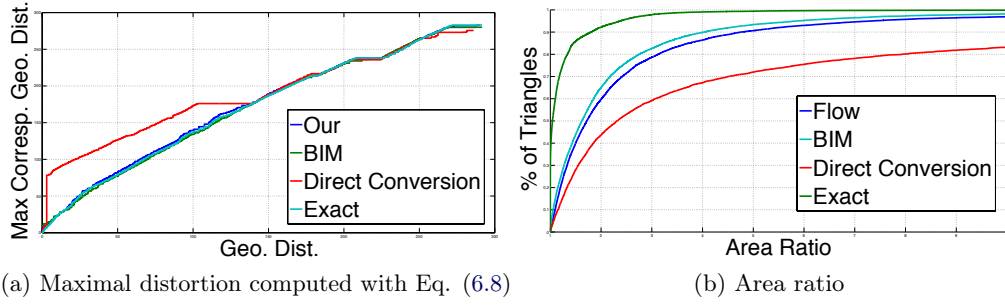


Figure 6.5 – Comparison of the distortion induced by various for a pair of centaur (TOSCA).

where d_N is the geodesic distance on N . If the map is nearly isometric r should be close to identity. We compare this measure for different mapping in Figure 6.5a for one example from TOSCA. Our method is comparable to BIM and to the ground truth in terms of continuity while the direct conversion of C show some very large distortions. Second we compare the ratio between the triangle's area before and after deformation. Since the deformations in our examples are almost isometric this ratio should be close to one. The graph in Figure 6.5b shows the percent of triangles which have an area ration smaller than a threshold. We show only the ratio greater than one since most of the discontinuous behavior is due to large jumps. The area ratio of the exact mapping are concentrated around one which is consistent with the fact that the deformation is nearly isometric. Again the direct conversion of C show some very large area distortions compare to BIM and our method.

This lack of continuity is confirmed by Figure 6.6 which provides on two examples a visualization of the point-to-point mapping using color correspondence. The direct conversion of the functional map shows some artifacts due to the blending between orientation preserving and orientation reversing maps.

Our method successfully repairs the areas distorted by BIM as shown on Figure 6.7 for two different matching problems. In this example the BIM maps transfer poorly functions from the source meshes to the target meshes while our method corrects these incorrect matches by providing a more accurate transfer. A visualization of the optimal vector field is provided on Figure 6.8 for the human example. The vector field on Figure 6.8b corresponds to the displacement needed to repair the BIM map, the action of this correction can be seen on the upper row of Figure 6.7.

6.7.3 Parameters Dependence

In practice we consider only a small family of vector fields based on the first eigenfunctions of 1-form Laplace-de Rham operator. Therefore in this setting our method will be more efficient in repairing low frequency distortion rather than recovering a high frequency deformation that cannot be represented by the flow of low frequency vector field. Of course the bigger is the vector field basis the better will be the repairs, and the slower

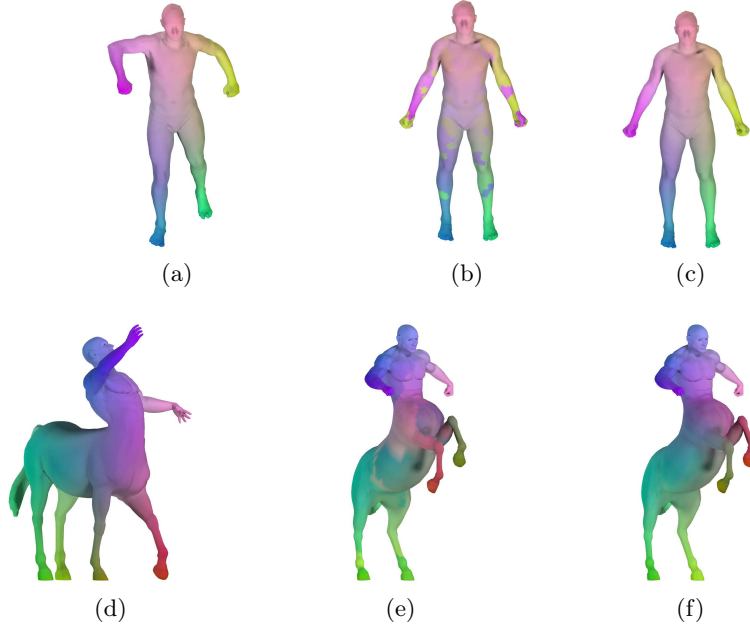


Figure 6.6 – Visualization of the point-to-point mapping through color correspondence. The texture of the first column (6.6a, 6.6d) are transferred to the second using the direct conversion of a functional map (6.6b, 6.6e) and to the third using our method (6.6c, 6.6f).

will be the method. In the experiments we presented the dimension of the vector field family can be reduced to 40 without influencing too much the point-to-point map.

Another critical parameter is the number of eigenfunctions we choose to represent the functional map C and C_{φ^0} . If the deformation is nearly isometric a small number is sufficient as the functional map C is almost diagonal. These considerations also apply to the initial map φ^0 : a very distorted map is badly approximated by a small number of eigenfunctions and can severely influence our method. We found that lowering the size of the function basis under 150 degrades rapidly the quality of the results.

In principle our method should work for non-isometric deformations provided we are given high-quality functional map as input. To obtain such a map, the choice of the functional basis would have to be modified in order to successfully encode the functional map in a reduced basis. This direction is left as an interesting future work.

6.7.4 Performance

For performance evaluation the computation times are given in the Table 6.1 in various cases. All the experiments have been performed on laptop with a 1.4 GHz processor and 4Go memory without parallelization. The timings are given for the two steps of the method: solving the problem (6.6) and tracing the flow lines. The time spent solving the optimization problem is almost independent of the number of vertices. The size of

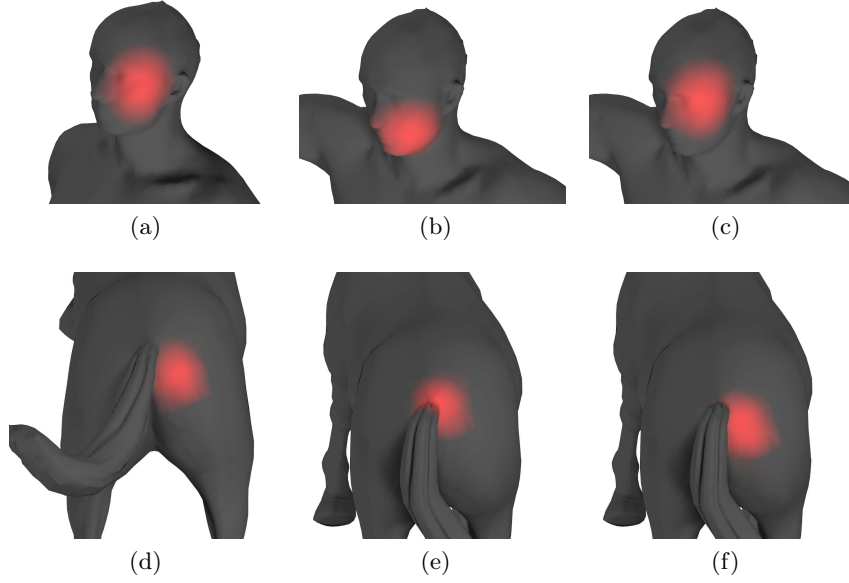


Figure 6.7 – Transfer of a function on the source meshes 6.7a and 6.7d to the target meshes using BIM 6.7b and 6.7e compared to our method 6.7c and 6.7f.

Mesh	Vertices	Optimization	Flow
Horse	19248	369s	29.4s
Dog	25290	300s	20.4s
Centaur	15768	381s	39.0s
SCAPE	12500	231.8s	30.8s

Table 6.1 – Average CPU time of each step for different mesh size.

this problem depends only on the number of computed eigenfunctions of the Laplace-Beltrami operator and on the dimension of the vector field family, which are constant in all experiments. Note that the computation of the flow does not scale linearly with the number of the vertices. This is explained by the fact we compute a composition with the BIM map which may map many vertices to a single point.

6.8 Conclusion, Limitations and Future Work

In this chapter we presented a method for non-rigid shape matching that is designed to output continuous maps. Our approach combines the strengths of conformal-based approaches, which often guarantee continuity with the functional map framework, which can enable low-distortion maps on the space of functions. Key to our method is enforcing continuity via the flow of a vector field, which allows our method to remain efficient by

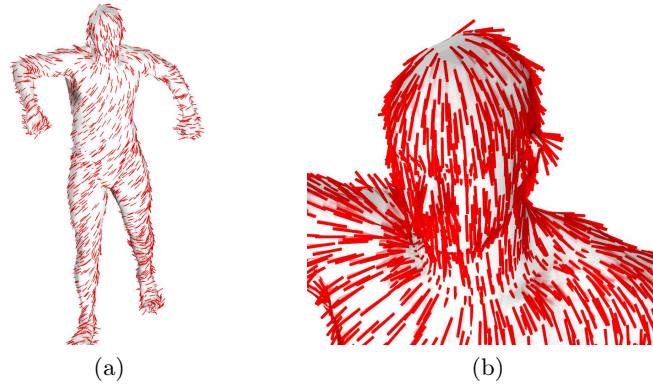


Figure 6.8 – Visualization of the direction of the optimal vector field corresponding to the experiment 6.7c: complete shape 6.8a and close-up on the face 6.8b.

avoiding expensive pairwise vertex constraints. One of the limitations of our method is that we only approximate the flow of a single vector field, whereas in practice, for complex motions, a combination of flows may be necessary. Extending our method to such cases is possible, while taking care of the robustness and non-accumulation of numerical errors. We are also planning to wider arrays of initial maps and ways to incorporate the continuity directly in the optimization of the functional maps.

Part II

Deformation to Shape

In the previous two chapters we showed how one can compute a mapping between deformable shapes. In this part, we take a more careful look at the link between the deformation of shapes and the induced metric distortion. In particular we aim at creating a characterization of surface embeddings within the operator representation framework fit for synthesis and exploration of intrinsic and extrinsic deformations. Namely, in [Chapter 7](#) we propose an algorithm to convert shape differences to embedded surfaces. In [Chapter 8](#) we take a different point of view and introduce a characterization of deformation fields up to isometric deformation.

Functional Characterization of Intrinsic and Extrinsic Geometry

We propose a novel way to capture and characterize distortion between pairs of shapes by extending the recently proposed framework of shape differences built on functional maps. We modify the original definition of shape differences slightly and prove that, after this change, the discrete metric is fully encoded in two shape difference operators and can be recovered by solving two *linear* systems of equations. Then, we introduce an extension of the shape difference operators using offset surfaces to capture extrinsic or embedding-dependent distortion, complementing the purely intrinsic nature of the original shape differences. Finally, we demonstrate that a set of four operators is *complete*, capturing intrinsic and extrinsic structure and fully encoding a shape up to rigid motion in both discrete and continuous settings. We highlight the usefulness of our constructions by showing the complementary nature of our extrinsic shape differences in capturing distortion ignored by previous approaches. We additionally provide examples where we recover local shape structure from the shape difference operators, suggesting shape editing and analysis tools based on manipulating shape differences.

7.1 Introduction

One classic approach to comparing surfaces separates metrics of similarity into *intrinsic* and *extrinsic* measurements. Intrinsic quantities are those that can be expressed exclusively in terms of distances along the surface, whereas extrinsic quantities are those that must be defined using an embedding into space and/or surface normals. A crowning result of classical differential geometry describes local geometry in terms of two quantities: the first and second fundamental forms, which capture the intrinsic Gaussian and extrinsic mean curvatures, respectively [13].

Considerable research in geometry processing has been dedicated to measuring intrinsic and extrinsic curvature in an attempt to replicate this attractive characterization of shape. From a practical standpoint, however, this task remains challenging for potentially noisy or irregular meshes considered in geometry processing. After all, surface curvature is a second-derivative quantity whose approximation on a piecewise-linear mesh requires considerable discretized adaptation and mollification to deal with noise. Measurement of curvature aside, algorithms for recovering geometry from discrete curvatures remain difficult to formulate, leading to potentially non-invertible discretizations.

In this chapter, we formulate an alternative characterization of surface geometry better suited for analysis, comparison, and synthesis tasks in the discrete setting. Several desiderata inform our design; a suitable framework for representing shape should

- capture and distinguish intrinsic and extrinsic geometry,
- express shape properties in a multiscale fashion to distinguish noise and fine-scale detail from large-scale structure,
- come from a smooth theory of shape to provide insensitivity to tessellation,
- be naturally expressible on continuous surfaces and on triangle mesh discretizations, and
- admit an inverse operator for reconstructing the embedded shape.

In short, we wish to pass from pointwise embeddings to a “dual” space featuring a more democratic treatment of intrinsic and extrinsic shape properties.

We approach this task by extending the theory of *shape differences*, introduced by Rustamov et al. [106] for purely intrinsic comparisons of shape structure. Rather than defining a shape in isolation, their construction characterizes shape by considering the distortion or difference of the target shape from a fixed source shape given a functional map between them [88]. Shape differences are couched in the language of functional analysis, indirectly measuring changes in angles and distances through the effects of these changes on inner products of functions and their gradients. They are written as linear operators whose restriction to a multiscale basis like the Laplace–Beltrami eigenfunctions distinguishes features at different levels of detail and allows for straightforward discretization via piecewise-linear finite elements (FEM).

We modify and extend this framework to derive a shape representation that is complete, encoding both the intrinsic and the extrinsic distortion without loss of information in both the continuous and discrete cases. To this end, we begin by reexamining the discretization of shape differences on triangle meshes. We modify the original definition of discrete area-based shape difference and prove an analog of a continuous property mentioned in [106] that shape differences fully capture intrinsic structure. Inspired by this fully-discrete result, we proceed to ask whether shape differences also can capture extrinsic structure. Towards this goal, we define an additional pair of shape differences on a thickened surface that captures extrinsic geometry. We then show that our full set of differences is sufficient to reconstruct a shape up to rigid motion in the discrete setting, under mild assumptions.

Our discussion concludes by closing the loop between shape differences and embeddings. In particular, we provide algorithms for recovering a shape embedding given a combination of intrinsic and extrinsic shape differences. Our algorithms are guaranteed to succeed in the presence of complete information under weak genericity conditions; when shape differences have been truncated to a smaller basis, we suggest regularizers to recover reasonable estimates of the missing data via convex optimization.

To summarize, our main contributions are:

- Theoretical discussion establishing that properly modified shape difference operators from [106] fully encode the intrinsic metric of a triangle mesh. Unlike more direct representations of meshed edge lengths, these operators enjoy connection to smooth theory—providing some degree of tessellation invariance—as well as multiscale approximation in the Laplace–Beltrami basis.

- A novel set of shape differences aimed at capturing and characterizing *extrinsic* or embedding-dependent information, with an associated observation that generically this set of shape differences is complete and encodes shapes up to rigid motion.
- A set of approaches for recovering geometric structure and an embedding from the shape differences that reduce to a sequence of linear solves with theoretical guarantees of recovery in the presence of complete information, and which we apply to shape editing operations based on manipulating shape differences.

We demonstrate the usefulness of these contributions on a variety of tasks, ranging from the analysis of cloth simulations by using our novel shape difference operators, sensitive to changes in extrinsic structure, to the transfer of shape structure such as geodesic distances more accurately, compared to using functional maps, and finally to recovering shape embedding even in the presence of approximate functional correspondences.

7.2 Related Work

Representation and manipulation of extrinsic and intrinsic structure is a vast theme pervading the geometry processing literature. We refer to [16] for discussion of the basic questions of representation and interaction with continuous differential geometry. Here, we highlight research linked to our particular approach.

Functional maps Our goal of using functional maps to characterize local and global geometry builds upon the machinery of *shape differences* [106]; see §7.4 for a summary. Rustamov and colleagues [106] show that in the case of smooth surfaces, shape differences fully encode intrinsic geometry. They do not, however, pursue a corresponding analysis for the discrete case. Furthermore, their work focuses solely on intrinsic geometry and hence cannot characterize extrinsic bending, critical for describing differences between nearly-isometric shapes like articulated bodies and cloth.

Shape-from-Laplacian Recovering structure from intrinsic shape differences is closely linked to recovering structure from Laplacian operators. Both in the continuous [103] and discrete [137] cases, the Laplace-Beltrami operator fully encodes intrinsic surface geometry, namely the Riemannian metric for smooth manifolds and edge lengths for discrete meshes. For triangle meshes, de Goes and colleagues [35] provide convex machinery for recovering the intrinsic structure of the mesh; their encoding of intrinsic structure using only Laplacian matrices is more compact than our pair of area and conformal shape differences, at the cost of a nonlinear objective sensitive to incomplete information.

The theoretical and practical contributions proposed in this chapter provide considerable insight beyond the fundamental mathematical contributions in these other works. Specifically, the convex optimizations in [137, 35] operate in the case of *complete, noise-free information*. They cannot be used for projection-style problems, e.g. finding the closest set of edge lengths to a noisy input Laplacian approximation or to finding an intrinsic

structure *consistent* with a truncated spectral approximation of the full operator. Additionally, we show how to use related machinery to encode extrinsic bending rather than only edge lengths.

Encoding extrinsic geometry A natural question is whether intrinsic structure can be used to reconstruct a surface embedding up to a global rigid transformation. Numerous examples of isometric smooth surface pairs disprove this notion in the continuous case [66]. While exact isometries of triangle meshes are rare with the exception of inward/outward “popping” of valence-three vertices, near-isometries can often arise and have significant differences in the embedding, making shape recovery from intrinsic data like edge lengths a numerically ill-conditioned problem; these near-isometries appear because small variations in the input edge lengths can lead to large changes in the resulting embedding. Nevertheless, Boscaini and colleagues [15] provide an algorithm for recovering a surface embedding in \mathbb{R}^3 from shape differences or equivalent structures. They apply the SMA-COF algorithm [68] for multidimensional scaling to generate an extrinsic embedding that replicates shape differences in a least-squares sense. As an alternative, [91] propose an algorithm for embedding from local approximations of the metric tensor; we will use an extension of this algorithm in §7.7.3. Both of these methods, however, operate using only intrinsic information and are subject to the ambiguity and instability caused by isometry invariance.

Adding extrinsic information to a shape representation allows it to be embedded in \mathbb{R}^3 up to rigid motion. In theory, the Gauss–Codazzi equations fully characterize surfaces from the first and second fundamental forms [13] (see [24, pg. 236]). In geometry processing, [40] reconstructs surfaces from prescribed principal curvatures, while [42] use nonlinear optimization methods to recover shape from dihedral edge lengths. These methods and many subsequent techniques employ nonlinear least-squares fits with few guarantees or characterization of their behavior. Wang, Liu, and Tong [132] propose a linear technique for embedding meshes from their edge lengths, dihedral angles, and axes of rotation across mesh faces.

In this chapter, we make use of offset surfaces to introduce extrinsic information to the shape difference representation. Offset surfaces have appeared in geometry processing for some related tasks, including cage generation [11] and shape optimization for printing [82]. While techniques like [57] are needed to generate “clean” offset surfaces for geometry editing purposes, in our case self-intersection and related artifacts are acceptable since the offset surface is not used for display but rather for geometric computation. [28, 53] provide curvature theories for discrete surfaces using offset geometry.

7.3 Overview

Our two main goals are to modify and extend the definition of the shape difference operators of Rustamov [106] so as to capture extrinsic distortion and to facilitate shape inference, i.e. to recover the metric and potentially the embedding of a target shape, given a base shape and a collection of shape differences.

We achieve these goals in several stages. The main ingredient for constructing both smooth and discretized shape differences is the computation of inner products between functions. So, rather than working directly with shape differences, we largely focus on matrices of functional inner products, which can be constructed on a single shape rather than a pair. Hence, after reviewing the smooth construction of shape differences (§4.2), we reexamine the discretization of intrinsic inner products on triangle meshes and show how a simple modification of the area-based inner product fully encodes intrinsic geometry in an easily-inverted fashion (§7.4.1).

We then capture extrinsic shape structure by introducing two operators built from intrinsic inner products on offset surfaces of a base shape (§7.5). We accompany our construction with theoretical characterization of the new information provided by extrinsic products (§7.5.3) and conclude by making explicit how our constructions involving inner product matrices apply to the construction of differences *between* shapes (§7.6). In this section, we also consider how truncating shape differences written in the Laplace–Beltrami basis affects the linear systems we pose.

With our new definitions and analysis in place, we propose optimization procedures for recovering intrinsic and extrinsic shape structure from the shape difference operators, potentially expressed in a reduced basis (§7.7). While the basic machinery for recovering metric information from shape differences is purely linear, we propose the use of more general convex optimization tools that add resilience to noise and incomplete information by explicitly enforcing the triangle inequality and/or smoothness. We conclude by demonstrating the ability of our constructions to capture and characterize extrinsic distortion ignored by previous approaches (§7.8.1). We furthermore apply our methods to recovering the metric and shape embedding and to facilitating novel shape editing operations via manipulating shape difference operators (§7.8.4).

7.4 Structure of Discrete Inner Products

By examining the derivation of formulas for computing shape differences, we can reveal how they are related to local surface geometry. This analysis not only elucidates the information encoded in a given shape difference but also will inform our design of algorithms for recovering shape embeddings from shape differences.

7.4.1 Discrete Inner Products

Each quantity above is straightforward to discretize in the language of finite elements over a triangle mesh; see [19, 108, 116] for general introductions to this approach. To this end, suppose M is represented using a connected, orientable, and manifold triangle mesh with vertices set \mathcal{X} and triangles set \mathcal{F} . We model functions as vectors $f \in \mathbb{R}^{|\mathcal{X}|}$ interpolated to triangle interiors in piecewise-linear fashion.

We will begin our fine-grained examination of shape differences by posing functional inner products on these meshes in terms of discrete geometry. Our ultimate goal is to show that before truncation in a basis, the area-based and conformal inner product matrices completely encode the intrinsic structure of meshed geometry. This property is also

stated in [106] in the continuous case; their discretization, however, does not admit such completeness due to the use of lumped area weights, as explained below.

Consider a single triangle $T \in \mathcal{F}$ on M , and suppose f and g are affine functions on T ; in other words, f and g are evaluated in the interior of T via barycentric interpolation of the three scalar values f_1, f_2, f_3 and g_1, g_2, g_3 defined on the vertices of the triangle. Multiplying these functions and integrating reveals that the inner product of f and g on T is given by

$$\langle f, g \rangle_{L^2}^T = \frac{\mu(T)}{12} \begin{pmatrix} f_1 & f_2 & f_3 \end{pmatrix} \begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix} \begin{pmatrix} g_1 \\ g_2 \\ g_3 \end{pmatrix},$$

where $\mu(T)$ is the area of T and f_i, g_i denote the values of f, g on vertex v_i . As a sanity check, taking $f_i = g_i = 1 \ \forall i$ recovers the area of T . This is the *exact* L^2 inner product of f and g defined over the meshed surface using piecewise-linear interpolation, without mass lumping commonly introduced in finite element discretizations; this distinction is critical for our construction.

Taking inner products over all of M requires summing over triangles T . If $f, g \in \mathbb{R}^{|\mathcal{X}|}$, then $\langle f, g \rangle_{L^2}^M$ is given by $f^\top A g$, where

$$A_{vw} = \frac{1}{12} \cdot \begin{cases} 2 \sum_{T \sim v} \mu(T) & \text{when } v = w \\ \sum_{T \sim e} \mu(T) & \text{when } e = (v, w) \\ 0 & \text{otherwise,} \end{cases} \quad (7.1)$$

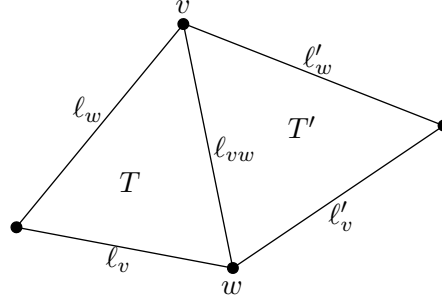
where $T \sim v$ denotes iteration over triangles adjacent to v and $T \sim (v, w)$ denotes iteration over triangles adjacent to edge (v, w) . This $|\mathcal{X}| \times |\mathcal{X}|$ “Galerkin mass matrix” A is nondiagonal but positive definite, integrating products of piecewise linear functions exactly. See e.g. [116, Chapter 10, (32)] for an example of its appearance in finite elements. We can think of A as a linear operator $A(\mu) : \mathbb{R}^{|\mathcal{F}|} \rightarrow \mathbb{R}^{|\mathcal{X}| \times |\mathcal{X}|}$ that constructs the area-based functional inner product matrix A given a vector $\mu \in \mathbb{R}^{|\mathcal{F}|}$ of triangle areas. We can show that $A(\cdot)$ is invertible in the following sense:

Proposition 7.4.1 *Suppose M has a boundary or at least one interior vertex with odd valence. Then, $A(\mu)$ uniquely determines μ , recoverable via a linear solve.*

Proof Equation (7.1) gives A as a linear function of $\mu(\cdot)$. Hence, we must show that this formula is invertible.

First we show how to recover the area of a single triangle on M . By the second row of (7.1), given A we have the sum of triangle areas adjacent to any edge of M . If M has a boundary, we then know the areas of the boundary triangles. Otherwise, take v with odd valence, and enumerate its adjacent triangles as T_1, \dots, T_k for odd k . Since we know the sums of adjacent areas, we have a linear system to recover $\mu(T_1), \dots, \mu(T_k)$:

$$\begin{pmatrix} 1 & 1 & & & \\ & 1 & 1 & & \\ & & \ddots & \ddots & \\ & & & 1 & 1 \\ 1 & & & & 1 \end{pmatrix} \begin{pmatrix} \mu(T_1) \\ \mu(T_2) \\ \vdots \\ \mu(T_k) \end{pmatrix} = \begin{pmatrix} A_{12} \\ A_{23} \\ \vdots \\ A_{k1} \end{pmatrix}.$$

Figure 7.1 – Notation for the conformal product W .

Consider carrying out forward substitution on the matrix. In each iteration, only the bottom row changes, from $(1, 0, \dots, 0, 1)$ to $(0, -1, 0, \dots, 0, 1)$, then to $(0, 0, 1, 0, \dots, 0, 1)$ and so on with alternating sign. When k is odd, in the last step the 1 is augmented to a 2, making the final row $(0, \dots, 0, 2)$. In other words, the matrix reduces to an upper triangular matrix with nonzero diagonal, which is invertible.

Hence, in either case we can recover $\mu(T)$ for at least one T . The remaining areas can be computed by flood filling outward from T ; given the area on one side of an edge and the sum of the adjacent areas, the adjacent area is recovered by subtraction. \square

The proof of this proposition and others below is in the appendix. A proposition of this nature does *not* hold if masses are lumped down the diagonal of A . This observation is intuitive in that a triangle mesh has approximately two times the number of triangles as vertices.

If f and g are piecewise-linear functions on M , then their gradients are piecewise-constant and expressible using one vector per triangle. Taking dot products of these gradients and integrating over M shows that $\langle f, g \rangle_{H_0^1}^M = f^\top W g$, where

$$W_{vw} = \frac{1}{8} \cdot \begin{cases} \mu(T)^{-1}(\ell_{vw}^2 - \ell_v^2 - \ell_w^2) \\ \quad + \mu(T')^{-1}(\ell_{vw}^2 - \ell'_v{}^2 - \ell'_w{}^2) & \text{when } v \sim w \\ - \sum_{u \sim v} W_{uv} & \text{when } v = w \\ 0 & \text{otherwise.} \end{cases} \quad (7.2)$$

Notation for the $v \neq w$ case is shown in Figure 7.1; $e \sim v$ denotes an edge e adjacent to vertex v , and ℓ_{uv} is the length of the corresponding edge. This matrix is the familiar cotangent Laplacian matrix cast in terms of edge lengths and triangle areas; this form also appears e.g. in [15]. Comparing (7.1) and (7.2), scaling the edge lengths of a mesh by some factor α will correspondingly scale A by α^2 while W will be left unchanged; unless otherwise noted, we scale meshes in our experiments to have unit surface area to remove dependence on global scaling.

A crucial observation that we make here is that if the triangle areas encoded in μ are fixed then the mapping $W(\ell^2; \mu) : \mathbb{R}^{|\mathcal{E}|} \rightarrow \mathbb{R}^{|\mathcal{X}| \times |\mathcal{X}|}$ taking squared edge lengths $\ell^2 \in \mathbb{R}^{|\mathcal{E}|}$ to a conformal inner product matrix W is *linear*. Note also that W is fully determined by its values W_{vw} for $v \sim w$. Thus, if we represent the list of inner products W_{vw} as

a vector c in $\mathbb{R}^{|\mathcal{E}|}$ then for a fixed set of area weights μ , there exists a matrix B_μ such that $c = B_\mu \ell^2$. The entries of B_μ are, of course, given in Eq. 7.2. In the pipeline that we propose below, we will first recover the triangle areas and then use those to recover edge lengths from the corresponding inner products. The following proposition shows that “generically” the matrix B_μ is invertible. I.e., the set of weights μ for which B_μ is singular has measure 0.

Proposition 7.4.2 *Assume that the mesh M is manifold without boundary. Then, for almost all choices of areas μ , the map $W(\ell^2; \mu)$ uniquely determines ℓ , which is recoverable via a linear solve.*

Proof By construction $W(\ell; \mu)$ takes squared edge lengths ℓ and outputs the matrix W . Extracting elements of W corresponding to edges on M yields a linear operator $B : \mathbb{R}^{|\mathcal{E}|} \rightarrow \mathbb{R}^{|\mathcal{E}|}$ with matrix

$$B_{ij} = \frac{1}{8} \begin{cases} \mu(T_i)^{-1} + \mu(T'_i)^{-1} & \text{if } i = j \\ -\mu(T)^{-1} & \text{if } i, j \text{ are edges of } T \\ 0 & \text{otherwise.} \end{cases}$$

Here, indices i, j refer to edges on M ; for a given edge i , we label its adjacent triangles T_i and T'_i . Remark that B can be written as a weighted sum: $B = \sum_k \frac{1}{8} \mu(T_k)^{-1} B^k$, where each B^k is a matrix such that:

$$B^k_{ij} = \begin{cases} 1 & \text{when } i = j, \text{ and } i \text{ belongs to triangle } k. \\ -1 & \text{when } i, j \text{ are edges of triangle } k. \\ 0 & \text{otherwise.} \end{cases}$$

It is easy to see that the intersection of the kernels of all B^k is empty, since B^k is non-singular when restricted to the values on edges of triangle k . Moreover, by considering the determinant of B as a multivariate polynomial with real coefficients, we conclude that B is either singular for any choice of values of $\mu(T_k)^{-1}$, or for a finite set of coefficients, which thus have measure zero.

To complete the proof we note that if B is singular for any choice of values of $\mu(T_k)^{-1}$, then the matrix pencil $B = \sum_k a_k B^k$ is singular (i.e., B is singular for any choice of coefficients a_k). Using Lemma 3.4 from [80] and the fact that B^k are symmetric, we see that in that case for every choice of a_k , there must exist a vector x such that $x^T B^k x = 0$ for every B^k , and $Bx = 0$. Now, given the values of x on some triangle, this means that its values on the adjacent triangle are either uniquely determined by the corresponding two equations (one linear, one quadratic), or these equations cannot be satisfied. By inspecting the resulting equalities, it is easy to see that at least two of the values on every triangle must be equal, and by considering any closed loop of triangles, these equations cannot be consistent for every choice of weights a_k . Thus, B cannot be a singular matrix pencil, and therefore B is invertible for almost any choice of values $\mu(T_k)^{-1}$. \square

This proposition implies that the linear map $W(\ell^2; \mu)$ is invertible for a small (possibly zero) perturbation of any set of area weights μ . Nevertheless, there exist cases in which

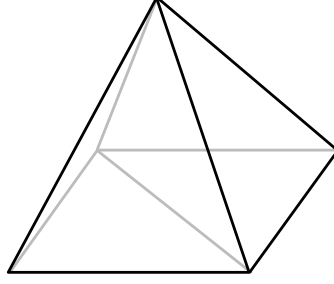


Figure 7.2 – A mesh for which $W(\ell^2; \mu)$ is not invertible when $\mu = \mathbf{1}$.

the squared edge lengths are not recoverable via inversion of the linear map $W(\ell^2; \mu)$ for a fixed set of area weights. One example of such a shape is shown in Figure 7.2, consisting of two tetrahedra glued at their bases. In this case, all the triangles have equal area weights, and it can be seen that the resulting linear system is singular. We also remark that the condition of no boundary is necessary in the Prop. 7.4.2 above, as it is possible to construct meshes for which the map $W(\ell^2; \mu)$ is singular *for all* choices of μ (e.g., a pair of triangles glued along a shared edge). For all the meshes that we tried in practice (§7.8), we have observed that the resulting system is both invertible and typically well-conditioned. We leave the formulation of the necessary and sufficient conditions on the mesh and the weights μ for the invertibility of $W(\ell^2; \mu)$ as a question for future work.

7.5 Encoding Extrinsic Structure

Intrinsic inner products capture the metric tensor (first fundamental form) of a surface, so to complete our representation we show how a related structure can be used to encode its second fundamental form. In keeping with previous discussion, we will use additional inner product matrices to derive a multiscale representation of this missing information. While there exist many possible ways to measure extrinsic distortion, this “functional” language facilitates a connection between continuous and discrete characterizations and unifies our treatment of intrinsic and extrinsic distortion.

These added structures complement the area-based and conformal products by making our representation of a shape unique up to rigid motion. In addition to providing a lossless representation of surface geometry in the presence of complete information, we demonstrate how the new products can capture and encode geometric relationships that are not captured by purely intrinsic analysis.

7.5.1 Extrinsic Alternatives

In discrete language, the inner product matrices $A(\mu)$ and $W(\ell^2; \mu)$ determine the edge lengths of a triangle mesh but not its dihedral angles, illustrated in Figure 7.3(a). Additionally providing dihedral angles is sufficient to recover a mesh up to rigid motion. There are many expressions of extrinsic shape that potentially encode these angles; before

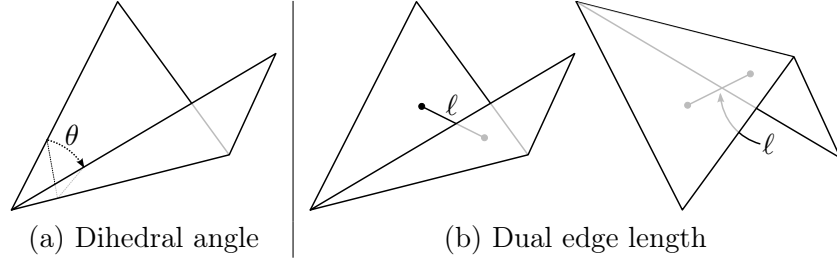


Figure 7.3 – Two potential ways to encode extrinsic mesh structure.

presenting our final solution, we mention a few straightforward alternatives to explain why they are less desirable.

At the most basic level, any technique encoding one value per edge of a triangle mesh could be used to represent dihedral angles. For instance, since the angles are in a vector $\theta \in \mathbb{R}^{|\mathcal{E}|}$, we could use an analog of Proposition 7.4.2 to store them in the matrix $W(\theta; \mu)$. This matrix roughly corresponds to taking products of functional gradients under the second fundamental form h as $\int h(\nabla f, \nabla g) dA$. The resulting matrix is not positive semidefinite, however, which prevents the definition of a smooth analog via the Riesz Representation Theorem (which applies only to positive definite inner products) and causes numerical issues due to departure from the cone of semidefinite matrices. Dihedral angles also are known only up to a period of 2π , providing potential for ambiguity in the expression of the vector θ . Lastly, if the mesh is flat, a definition based on dihedral angles will lead to a degenerate inner product.

In an attempt to bring back the positive definiteness enjoyed by the intrinsic formulation, we might attempt to encode the edge lengths of a *dual* mesh, shown in Figure 7.3(b). These lengths indirectly encode dihedral angles up to sign but are unable to distinguish between inward and outward folding directions, as shown in the figure. Obvious techniques for disambiguating the inward and outward folds generally accompany edge lengths with signs, reintroducing the problems discussed in the previous paragraph.

An alternative construction might define extrinsic shape differences via the *Gauss map*, or map from a surface into the unit sphere based on normal direction; see [75] for an example in geometry processing. While the Gauss map is used in classical differential geometry to derive extrinsic properties of surfaces, we find it to be unstable within the shape difference framework. In particular, the image of the Gauss map is composed of many overlapping spherical triangles that change rapidly from vertex to vertex. Projection of this information into low-frequency Laplace-Beltrami bases tends to remove the majority of the meaningful geometric signal. In a sense, however, we can view the offset surface construction proposed below as a means of smoothing out this construction.

Before proceeding, we should remark that strictly speaking it may not be necessary to provide extrinsic information at all. According to a classical result by Gluck [44], almost all triangulated simply connected closed surfaces are rigid. Although this result might imply that triangle edge lengths are, in general, sufficient to reconstruct the mesh up to rigid motion, this is only true if the metric is known *exactly*; moreover, it is highly

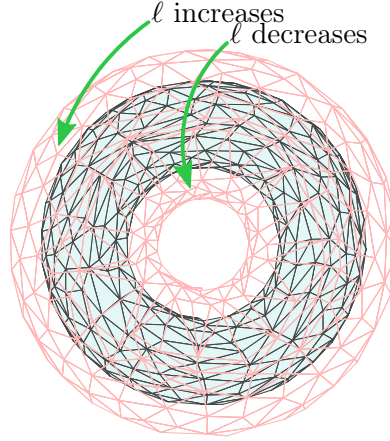


Figure 7.4 – Edge lengths change according to curvature of the offset surface.

nontrivial to recover an embedding even if one is known to exist. When the edge-lengths are perturbed or are approximated, the corresponding embedding might either not exist or be very far from the desired shape. As we show below, the presence of explicit information about the extrinsic distortion can greatly help in both direct and inverse problems and is largely complementary to the intrinsic distortion addressed by prior methods in practice.

7.5.2 Offset Surfaces

Our construction of an extrinsic shape representation is an extension of the dual mesh idea from §7.5.1 that does not suffer from sign ambiguity. Instead, we are able to rely upon the positive definiteness of inner product matrices directly to encode both intrinsic and extrinsic information. In short, rather than encoding a metric and its derivative, we encode a metric and a slightly deformed metric, both of which admit natural positive definite representations.

The intuition for our construction is illustrated in Figure 7.4. Suppose we wish to recover the embedding of the blue torus. As discussed in the previous section, it may be difficult to reconstruct the torus purely from its list of edge lengths. Instead, suppose we generate an *offset surface* by displacing each vertex and face along its outward normal a fixed distance t . The operation is extrinsic, since the mesh moves through the surrounding space, modulating edge lengths ℓ based on the curvature of the surface. The edge lengths in the interior of the torus shrink while the edge lengths on the exterior expand, effectively distinguishing the bend direction.

In the continuous case, we can formalize the effect of offsetting a surface as follows:

Proposition 7.5.1 *Suppose M is a compact orientable Riemannian 2-manifold without boundary. Consider a family of immersions $F_t : U \subset \mathbb{R}^2 \rightarrow M_t \subset \mathbb{R}^3$ satisfying*

$$\frac{\partial F_t}{\partial t}(p) = n_t(p), \forall (p, t) \in U \times \mathbb{R}_+,$$

where n_t denotes the outward unit normal of $M_t := F_t(U)$. At all time t the normal n_t remains equals to the normal n_0 of $M_0 := M$.

Moreover, if $\mathbf{g}_{ij}(t) := \langle \frac{\partial F_t}{\partial x_i}, \frac{\partial F_t}{\partial x_j} \rangle$ is the metric of the embedded surface and $\mathbf{h}_{ij}(t) := \langle \frac{\partial F_t}{\partial x_i}, \frac{\partial n}{\partial x_j} \rangle$ is its second fundamental form, then

$$\mathbf{g}_{ij}(t) = \mathbf{g}_{ij} + 2t\mathbf{h}_{ij} + t^2\mathbf{h}_{il}\mathbf{g}^{lm}\mathbf{h}_{mj} \text{ and } \mu_t = (1 + tH + t^2K)\mu,$$

where $H := (\mathbf{g}^{ij}\mathbf{h}_{ij})_{t=0}$ is the mean curvature, $2K := H^2 - (\mathbf{g}^{ij}\mathbf{h}_{jk}\mathbf{g}^{kl}\mathbf{h}_{il})_{t=0}$ is the Gaussian curvature at $t = 0$ and $\mu_t := \sqrt{\det \mathbf{g}(t)}$.

Proof First let's show that the normal vector is constant during the flow. The vector n is unit norm therefore $\partial_t \langle n_t, n_t \rangle = 2\langle \partial_t n_t, n_t \rangle = 0$ meaning that $\partial_t n_t$ belongs to the tangent of M_t . Since any tangent vector V can be written $V = \langle V, \partial_i F \rangle \mathbf{g}^{ij} \partial_j F$, we have

$$\frac{\partial n_t}{\partial t} = \left\langle \frac{\partial n_t}{\partial t}, \frac{\partial F_t}{\partial x_i} \right\rangle \mathbf{g}^{ij}(t) \frac{\partial F_t}{\partial x_j} = - \left\langle n_t, \frac{\partial n_t}{\partial x_i} \right\rangle \mathbf{g}^{ij}(t) \frac{\partial F_t}{\partial x_j} = 0.$$

As consequence the immersion has the following closed form expression at all time: $F_t = tn_0 + F_0$. Therefore the metric is quadratic with respect to time:

$$\begin{aligned} \mathbf{g}_{ij}(t) &= \left\langle \frac{\partial F_t}{\partial x_i}, \frac{\partial F_t}{\partial x_j} \right\rangle = \left\langle t \frac{\partial n_0}{\partial x_i} + \frac{\partial F_0}{\partial x_i}, t \frac{\partial n_0}{\partial x_j} + \frac{\partial F_0}{\partial x_j} \right\rangle \\ &= \mathbf{g}_{ij} + 2t\mathbf{h}_{ij} + t^2 \left\langle \frac{\partial n_0}{\partial x_i}, \frac{\partial n_0}{\partial x_j} \right\rangle. \end{aligned}$$

The proposition statement is obtained using the Gauss–Weingarten relations linking $\partial_i n$ with the second fundamental form. Because the normal is unit norm $\partial_i n$ is a tangent vector whose expression is given by:

$$\frac{\partial n_0}{\partial x_i} = \left\langle \frac{\partial n_0}{\partial x_i}, \frac{\partial F_0}{\partial x_j} \right\rangle \mathbf{g}^{jk} \frac{\partial F_0}{\partial x_k} = \mathbf{h}_{ij} \mathbf{g}^{jk} \frac{\partial F_0}{\partial x_k}.$$

The evolution of the local area is obtained by a direct computation of the determinant of metric tensor. \square

Results of this nature are fairly well-known for offset surfaces and sometimes referred to as Steiner formula; see e.g. [94] for related discussion. Informally, the proposition shows that the second fundamental form of M is encoded through the change in metric while the surface is being offset along its normal directions. In case of surfaces, a closed form expression for \mathbf{h} can be derived. precisely, there is only one square root of 2-matrices (out of four) satisfying $\partial_t \mathbf{g}_{ij} = 2\mathbf{h}_{ij}$ at time zero:

$$\begin{aligned} \mathbf{h}_{ij}(t) &= \frac{1}{\tau_t t} \mathbf{g}_{ij}(t) + \frac{1}{t} \left(\frac{\delta_t}{\tau_t} - 1 \right) \mathbf{g}_{ij}, \quad -\epsilon < t < \epsilon \\ \text{where } \delta_t &= \mu(t)/\mu, \text{ and } \tau_t = \sqrt{\mathbf{g}^{ij} \mathbf{g}_{ij}(t) + 2\delta_t}. \end{aligned}$$

When M is an oriented triangle mesh, there are many potential constructions of discrete offset surfaces, and several likely would suffice for the proofs in this chapter. For

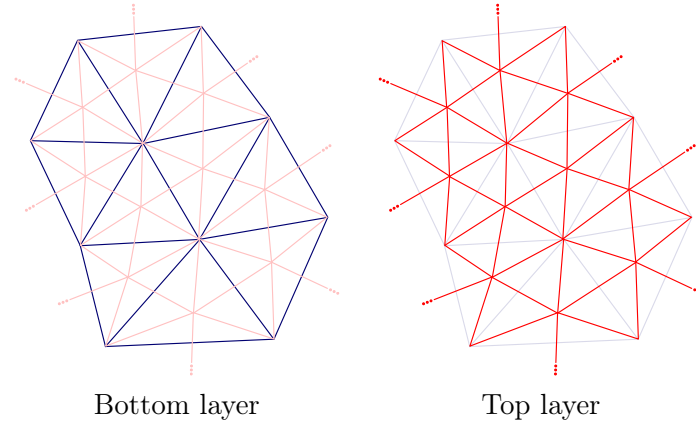


Figure 7.5 – Topology of offset mesh.

mathematical simplicity, we choose the construction in Figure 7.5. On the left we show triangles of the original mesh M in blue. On the right, we define the topology of the offset mesh in red, which contains a vertex for every vertex of M and every triangle of M . For a fixed constant $t > 0$, we place the vertices distance t above M along its face/vertex normals; any reasonable definition of a unit-length vertex normal suffices. Offset vertices associated with triangles are placed directly above the barycenter of the triangle.

7.5.3 Recovery of Embedding

In the end, we encode the geometry accompanying a fixed triangle mesh topology using four structures: the intrinsic area-based and conformal inner product operators and the same operators for the offset surfaces with fixed normal offset distance $t > 0$. We denote the offset surface of M as M_t . In this section, we show—at least before truncation—that these four difference matrices are sufficient for fully reconstructing a shape.

The challenge of reconstructing a triangle mesh from its edge lengths arguably comes from the fact that there are many ways to glue together two adjacent triangles by fixing different dihedral angles. Rigidity may imply that only one such embedding exists, but it is not obvious from local relationships. In contrast, an oriented *tetrahedral* mesh is easy to reconstruct from its list of edge lengths simply by gluing individual tetrahedra face by face.

Hence, our intuition for why a mesh plus its offset are enough to reconstruct the mesh comes from a volumetric perspective. This intuition is confirmed by Proposition 7.5.1 and the Gauss–Codazzi equations, since offset geometry provides extrinsic curvature information, which in turn determines an embedding. We chose the topology of the offset mesh (Figure 7.5) specifically to allow for a canonical “thickening” of the offset slice into a tetrahedral mesh, shown in Figure 7.6. Using mesh-based shape differences, we can recover the edge lengths of the bottom and top layers of the thickening. By construction of the offset mesh, we are able to recover the lengths of the interior edges of the thickening, effectively proving the following proposition:

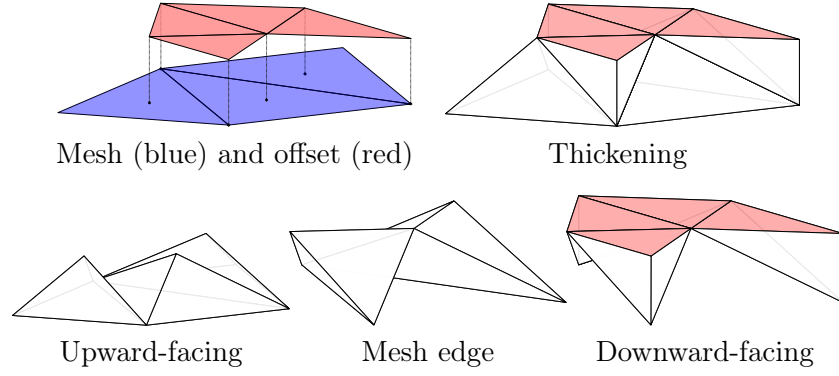


Figure 7.6 – Canonical thickening (top right) of a triangle mesh (top left); types of tetrahedra in the thickening (bottom).

Proposition 7.5.2 *Suppose a mesh M satisfies the criteria in Propositions 7.4.1 and 7.4.2. Given the topology of M , the area-based and conformal product matrices $A(\mu)$ and $W(\ell^2; \mu)$ of M , and the area-based and conformal product matrices $A_t(\mu_t)$ and $W(\ell_t^2; \mu_t)$ of M_t , the geometry of M can (almost always) be reconstructed up to rigid motion.*

Proof The previous propositions show that μ , ν , μ_t , and ν_t are (almost always) sufficient to recover the edge lengths of the base and offset surfaces. The remaining edges of the canonical thickening are between the inner and outer layers and are recoverable essentially by convention. Specifically:

- The edges along surface normals are length t by definition.
- The bottom edge lengths of the “upward-facing” tetrahedra (Figure 7.6) are known because they are on the base surfaces. The remaining edges of these tetrahedra can be computed because the upward-facing tetrahedron is generated via normal offset from the barycenter of the base triangle by a distance t .
- “Mesh edge” tetrahedra are adjacent to “upward-facing” tetrahedra and outer faces of the thickening and hence have edge lengths fixed by their neighbors’ construction.
- Similarly, “downward-facing” tetrahedra have one normal edge of length t , and the remaining edges are on the outer surface or adjacent to an “upward-facing” tetrahedron.

The embedding of a single oriented tetrahedron is fixed up to rigid motion given its edge lengths, so the proposition follows by gluing the tetrahedra of the canonical thickening according to the topology of the construction. \square

7.5.4 Discussion

We pause to summarize the theoretical development in the previous sections. We began by reconsidering the construction of inner products and shape differences from first-order

finite elements. When area elements are not lumped, we showed that inner product matrices fully determine the edge lengths of a mesh and that they can be recovered by solving two linear systems of equations: one for recovering the triangle areas, and the other for recovering the edge lengths. Moreover, generically, both systems are non-singular. In both the continuous and discrete cases, these intrinsic measurements are not enough to distinguish isometric shapes. Even worse, the space of near-isometric shapes can be very large. Hence, we propose generating an offset surface M_t from a mesh or surface M . In the continuous case, the geometry of M_t determines the *extrinsic* structure of M by encoding its second fundamental form. In the discrete case, combining edge lengths of M with edge lengths of M_t fully determines M up to rigid motion. The main development is that we can *completely* determine a shape using functional inner products via the constructions above.

Our theoretical contributions deal with the noise-free, non-truncated case. Roughly, they show that *if* intrinsic/extrinsic shape differences were computed from an embedded mesh M with fixed topology, *then* the embedding of M almost always can be covered from those differences up to rigid motion. We evaluate sensitivity to noise and the possibility of recovering geometry from truncated shape differences empirically in §7.8.

7.6 From Inner Products to Shape Differences

With the goal of working with quantities that exist when meshes are not in vertex-for-vertex correspondence, we shift from working with matrices of inner products to shape differences. This shift is needed to propose algorithms in §7.7 for estimating the dense structure of a target mesh given a source mesh and an approximate relationship between the source and the target, represented as a functional map computed e.g. using assorted correspondence techniques.

7.6.1 Discrete Shape Differences

We begin by considering two meshes M and N in vertex-for-vertex correspondence, with areas $\mu_M, \mu_N \in \mathbb{R}^{|\mathcal{F}|}$ and squared edge lengths $\ell_M^2, \ell_N^2 \in \mathbb{R}^{|\mathcal{E}|}$. Based on the continuous definitions in §4.2, the “full” area-based and conformal shape difference between meshes M and N are [106, §5, “option 1”]

$$\begin{aligned} D_A &= A(\mu_M)^{-1}A(\mu_N) \\ D_C &= W(\ell_M^2; \mu_M)^{-1}W(\ell_N^2; \mu_N). \end{aligned} \tag{7.3}$$

A straightforward corollary of the discussion in §7.4.1 is that these two differences completely determine the edge lengths and triangle areas of N given the geometry of M . Notice the first relationship is still linear in μ_N and the second in ℓ_N^2 , preserving the proposed system of equations for reconstruction.

Similarly, the extrinsic differences are simply the shape differences between the offset surfaces:

$$\begin{aligned} D_A^E &= A(\mu_{M_t})^{-1}A(\mu_{N_t}) \\ D_C^E &= W(\ell_{M_t}^2; \mu_{M_t})^{-1}W(\ell_{N_t}^2; \mu_{N_t}). \end{aligned} \tag{7.4}$$

The discussion in §7.5.3 implies that the tuple (D_A, D_C, D_A^E, D_C^E) is sufficient to reconstruct N up to rigid motion given M .

7.6.2 Source-Truncated Correspondence

More commonly, suppose $\Phi \in \mathbb{R}^{|\mathcal{X}| \times k}$ contains the orthonormal Laplace–Beltrami basis of M , truncated to k functions. Assuming M and N are still in vertex-for-vertex correspondence, we can write “reduced” shape differences as

$$\begin{aligned} D_A^\Phi &= \Phi^\top A(\mu_N) \Phi \\ D_C^\Phi &= \text{diag}(-\{\lambda_i^M\})^+ \Phi^\top W(\ell_N^2; \mu_N) \Phi, \end{aligned} \quad (7.5)$$

where the eigenvalues of the Laplacian on M are λ_i^M . These differences no longer determine angles and edge lengths exactly but still encode a multiscale notion of geometry that is valuable for understanding the relationships between M and N ; extrinsic differences can be defined analogously from the offset surface. We can still define linear systems for computing μ_N and ℓ_N^2 from D_A , D_C , μ_M , and ℓ_N^2 using these relationships, although they are unlikely to be full-rank for small k ; we provide regularizers in the next section. These truncated differences essentially correspond to removing rows and/or columns from the full shape differences after writing them in the Laplace–Beltrami eigenbasis. Such a computation can be useful for multiscale analysis of surface deformations, in which vertex-for-vertex correspondence is known but high-frequency changes may not be useful to analyze. What remains, however, is to consider the case when M and N are not in vertex-for-vertex correspondence and both have incomplete bases.

7.6.3 Source- and Target-Truncated Correspondence

Suppose we are given truncated bases $\Phi^M \in \mathbb{R}^{|\mathcal{X}_M| \times k_M}$ and $\Phi^N \in \mathbb{R}^{|\mathcal{X}_N| \times k_N}$ for the eigenspaces of M and N , respectively, and a functional map matrix $C \in \mathbb{R}^{|\mathcal{X}_N| \times |\mathcal{X}_M|}$ taking functions written in the Φ^M basis on M to functions in the Φ^N basis on N . Following [106, §5], we define shape differences in this case as

$$\begin{aligned} D_A^{\Phi^M, \Phi^N} &= C^\top C \\ D_C^{\Phi^M, \Phi^N} &= \text{diag}(-\{\lambda_i^M\})^+ C^\top \text{diag}(-\{\lambda_i^N\}) C. \end{aligned} \quad (7.6)$$

Whereas the truncated shape differences in (7.5) contain a limited window of values from the full shape difference matrix, in this final case the non-truncated entries of the shape difference matrices also undergo some change. This is because even if a function on M is in the column space of Φ^M , it will not be transported fully to N by the functional map C due to removal of high frequencies.

These shape differences are discretizations of analogous linear operators in the smooth setting. For this reason, even though the differences in (7.6) no longer satisfy exact equality relationships like those in (7.5) for recovering areas μ_N and squared edge lengths ℓ_N^2 from shape differences and the geometry of M , we will pose *approximate* relationships

$$\begin{aligned} D_A^{\Phi^M, \Phi^N} &\approx (\Phi^M)^\top A(\mu_{M \leftarrow N}) \Phi^M \\ D_C^{\Phi^M, \Phi^N} &\approx \text{diag}(-\{\lambda_i^M\})^+ \Phi^{M^\top} W(\ell_{M \leftarrow N}^2; \mu_{M \leftarrow N}) \Phi^M. \end{aligned} \quad (7.7)$$

The unknown variables $\mu_{M \leftarrow N}$ and $\ell_{M \leftarrow N}^2$ can be thought of as “pullbacks” of the metric of N to that of M , in the sense that they attempt to assign areas and edge lengths to the topology of M to mimic inner products on N . The first condition is linear in $\mu_{M \leftarrow N} \in \mathbb{R}^{|\mathcal{F}_M|}$ and the second in $\ell_{M \leftarrow N}^2 \in \mathbb{R}^{|\mathcal{E}_M|}$.

Since the shape differences in (7.6) are the most realistic test cases, we will assume in our experiments that truncated shape differences are computed in this fashion unless noted otherwise. That is, we will assume that we are given a source- and target-*truncated* shape difference. The experiments in §7.8.2 verify that this approximation is reasonable as long as k_M and k_N are sufficiently large.

7.7 Recovery of Intrinsic and Extrinsic Structure

Having established theoretical aspects of intrinsic and extrinsic shape differences, we now provide algorithms for recovering a shape N given a base shape M and shape differences to N and its offset. First, we recover triangle areas from the base and offset surfaces from corresponding area-based shape differences. With these areas fixed, we then recover edge lengths, which were shown in §7.5.3 to completely determine the surface.

Both steps can be carried out using linear solves when shape differences are not truncated. When dealing with truncated or inexact functional maps, we augment the optimization with constraints ruling out unreasonable structures. We also show how to apply existing techniques for recovery of an embedding from edge lengths of the surface and its offset.

7.7.1 Triangle Area Computation

We first show how to recover areas of triangles given an area-based shape difference. Our approach is an extension of the basic linear technique outlined in the proof of Proposition 7.4.1, extended to deal with truncation and noise.

Following §7.6, suppose D_A is the area-based shape difference between M and N in the Laplace–Beltrami basis Φ_M . Recall that our goal is to pull the geometry of N back to the mesh of M . Hence, the the area-based difference from M to the reconstructed target shape N^* should satisfy $D_A^* = \Phi_M^\top A(\mu_N) \Phi_M$. If the reduced basis Φ_M on M has k functions, this linear system for μ_N has k^2 equations and $|\mathcal{F}|$ unknowns. So, we need at least $k \sim \sqrt{|\mathcal{F}|}$ to have a well-posed system.

The quality of the solution found by solving this system without regularization depends on two factors: the quality of D_A and the conditioning of the resulting linear problem. We find that both limitations are improved considerably by introducing a nonnegativity constraint, leading to the following optimization problem for μ_N :

$$\begin{aligned} \min_{\mu_N} \quad & \|\Phi_M^\top A(\mu_N) \Phi_M - D_A\|_{\text{Fro}}^2 \\ \text{s.t.} \quad & \mu_N(T) \geq 0 \quad \forall \text{ triangles } T \in \mathcal{F}. \end{aligned} \quad (7.8)$$

We solve this and other convex programs using the Mosek toolbox [79]. When the system is highly underdetermined, we additionally add a regularizing viscosity term $\varepsilon \|\mu_N - \mu_M\|_2^2$ for small $\varepsilon > 0$, under the assumption that triangle areas should change minimally unless there is evidence to do otherwise; we set $\varepsilon = 10^{-4}$ in all the experiments in this chapter.

7.7.2 Edge Length Computation

Now that we can compute triangle areas, we can recover edge lengths. As in the last section, we start from Proposition 7.4.2 to propose a basic linear system for squared edge lengths and then provide regularization techniques for dealing with inexact or truncated differences.

The conformal shape difference encodes the transformation of the cotangent Laplacian through the deformation. Again borrowing from §7.6, the geometry of N can be pulled back to M via the following linear condition on squared edge lengths ℓ_N^2 given fixed areas μ_N :

$$\text{diag}(-\lambda_i^M) D_C = \Phi_M^\top W(\ell_N^2; \mu_N) \Phi_M. \quad (7.9)$$

Solving this linear system of equations for ℓ_N^2 depends critically on the approximated areas μ_N ; numerical or discretization error from the method in §7.7.1 invalidates this step, regardless of the quality of D_C . To provide resilience to this issue and to noise in D_C , we add constraints to this system ruling out unrealistic edge lengths ℓ_N^2 .

To define a triangulation, the squared edge lengths ℓ_N^2 must be nonnegative; furthermore, $\sqrt{\ell_N^2(T)}$ must respect the triangle inequality in each mesh triangle T . We enforce the latter constraint via the following proposition:

Proposition 7.7.1 *The symmetric matrix \mathbf{E} defined by*

$$\mathbf{E} = \frac{1}{2} \begin{pmatrix} 2x_1 & x_3 - x_1 - x_2 & x_2 - x_1 - x_3 \\ x_3 - x_1 - x_2 & 2x_2 & x_1 - x_2 - x_3 \\ x_2 - x_1 - x_3 & x_1 - x_2 - x_3 & 2x_3 \end{pmatrix}$$

is positive semidefinite if and only if x_1, x_2, x_3 are nonnegative and their square roots satisfy the triangle inequality.

Proof We denote (e_1, e_2, e_3) the canonical basis and the indices $\{i, j, k\} \in \{1, 2, 3\}$.

If $\sqrt{x_k} \leq \sqrt{x_i} + \sqrt{x_j}$, then there exist three points (v_1, v_2, v_3) which define an embedding of a triangle. Let E be the matrix with columns $v_3 - v_2$, $v_1 - v_3$ and $v_2 - v_1$ then $\mathbf{E} = E^\top E$. The matrix \mathbf{E} is therefore positive semidefinite.

Since \mathbf{E} is symmetric positive semidefinite, the Cauchy-Schwartz inequality holds. Expanding the expression $(e_i + e_j)^\top \mathbf{E}(e_i + e_j)$ yields

$$\begin{aligned} (e_i + e_j)^\top \mathbf{E}(e_i + e_j) &= x_i + x_j + 2e_i^\top \mathbf{E}e_j \\ &\leq x_i + x_j + 2\sqrt{x_i x_j} \\ &\leq (\sqrt{x_i} + \sqrt{x_j})^2. \end{aligned}$$

At the same time, a direct computation shows $(e_i + e_j)^\top \mathbf{E}(e_i + e_j) = x_k$ which implies that $\sqrt{x_k} \leq \sqrt{x_i} + \sqrt{x_j}$. \square

We also can link squared edge lengths to the computed triangle areas $\mu_N(T)$. This link is provided by the submatrices \mathbf{E}_k defined as

$$\mathbf{E}_k = \frac{1}{2} \begin{pmatrix} 2x_i & x_k - x_i - x_j \\ x_k - x_i - x_j & 2x_j \end{pmatrix},$$

where $\{i, j, k\} = \{1, 2, 3\}$. With this definition in place, we leverage the following proposition:

Proposition 7.7.2 *\mathbf{E} is positive semidefinite if and only if $x_k \geq 0$ for all $k \in \{1, 2, 3\}$ and $\det(\mathbf{E}_3) \geq 0$. Moreover, if $\mathbf{E} \succeq 0$, then $\det(\mathbf{E}_k) = 4\mu_N(T)^2$.*

Proof Starting with the second statement, direct computation of the determinant shows

$$4\det(\mathbf{E}_k) = \begin{pmatrix} x_1 & x_2 & x_3 \end{pmatrix} \begin{pmatrix} -1 & 1 & 1 \\ 1 & -1 & 1 \\ 1 & 1 & -1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}.$$

When the x_i 's are squared triangle edge lengths, this is a formulation of Heron's area formula.

The first statement is proved using a well-known theorem on positive block matrices (property of Schur complements) [18]: $\mathbf{E} \succeq 0$ if and only if $\mathbf{E}_3 \succeq 0$ and

$$2x_3 - \begin{pmatrix} x_2 - x_1 - x_3 \\ x_1 - x_2 - x_3 \end{pmatrix}^\top \mathbf{E}_3^{-1} \begin{pmatrix} x_2 - x_1 - x_3 \\ x_1 - x_2 - x_3 \end{pmatrix} \geq 0.$$

Notice that

$$\mathbf{E}_k \begin{pmatrix} 1 \\ 1 \end{pmatrix} = -\frac{1}{2} \begin{pmatrix} x_j - x_i - x_k \\ x_i - x_j - x_k \end{pmatrix},$$

and hence the first condition is met whenever $x_3 \geq 0$. Moreover, \mathbf{E}_3 is a 2×2 matrix and therefore is positive semidefinite if and only if $x_1 \geq 0$, $x_2 \geq 0$ and $\det(\mathbf{E}_3) \geq 0$. \square

Enforcing constraints derived from these relationships in the computation of edge lengths from a shape difference leads to the following optimization problem:

$$\begin{aligned} \min_{\ell_N^2} \quad & \|\Phi_M^\top W(\ell_N^2; \mu_N) \Phi_M - \text{diag}(-\lambda_i^M) D_C\|_{\text{Fro}}^2 \\ \text{s.t.} \quad & \ell_N^2 \geq 0 \\ & \det(\mathbf{E}_3(T)) = 4\mu_N(T)^2 \quad \forall \text{ triangles } T \in \mathcal{F}. \end{aligned}$$

This problem, however, is large and non-convex due to the determinant constraint. A convex relaxation is possible by noticing that the cone of symmetric positive semidefinite matrices with determinant ≥ 1 is convex; this observation derives from the convexity of the function $A \mapsto -\log(\det A)$ [18]. So, the former problem can be relaxed to a convex problem:

$$\begin{aligned} \min_{\ell_N^2} \quad & \|\Phi_M^\top W(\ell_N^2; \mu_N) \Phi_M - \text{diag}(-\lambda_i^M) D_C\|_{\text{Fro}}^2 \\ \text{s.t.} \quad & \ell_N^2 \geq 0 \\ & \det(\mathbf{E}_3(T)) \geq 4\mu_N(T)^2 \quad \forall \text{ triangles } T \in \mathcal{F}. \end{aligned} \tag{7.10}$$

The determinant constraint is handled using the rotated quadratic cone optimization in the Mosek toolbox [79]. While (7.10) contains a relaxation of the full set of constraints, we find empirically that this relaxation generally is tight; we leave it to future work to prove conditions for “exact recovery” akin to those in [38] for mesh alignment problems. As in §7.7.1, we can additionally regularize by adding $\varepsilon' \|\ell_N^2 - \ell_M^2\|_2^2$ to the objective; our experiments use $\varepsilon' = 10^{-4}$.

7.7.3 Global Extrinsic Reconstruction

At this point, we have presented algorithms for recovering edge lengths for the entire canonical thickening defined in §7.5.3. As suggested in the proof of Proposition 7.5.2, if these edge lengths are computed without error, the thickening can be reconstructed greedily; then, the embedding of N from M is the inner envelope of this thickening. In reality, the squared edge lengths in ℓ_N^2 likely exhibit numerical error. For this reason, we employ the algorithm in [91] for reconstructing a triangle mesh given its edge lengths. We adapt their approach to take into account the tetrahedra defined by the offset surface, by using the same ARAP-style deformation energy, defined on each triangle facet of each tetrahedron, and using the same alternating optimization strategy. We note, in particular, that this approach does not require embedded surfaces to be manifold, and can easily incorporate edges shared by more than two triangles, which only changes the computation of the gradient of the energy. Hence this allows us to reconstruct the entire set of triangles in the canonical thickening rather than the inner or outer surfaces only. We provide the thickening of M as a starting point for their alternating optimization algorithm. Whereas their method is subject to isometric ambiguity when embedding manifold meshes, reconstructing the entire thickened structure reduces ambiguity and more reliably provides an extrinsically correct embedding.

7.8 Experiments

In this section we illustrate the utility of the constructions presented above in a variety of practical application scenarios. We start by showing how the extrinsic shape differences can be useful for shape exploration and analysis, by complementing the information provided by the intrinsic differences of Rustamov et al. [106]. We then show how our metric and shape recovery methods can be used to both infer shape structure and ultimately recover the embedding from approximate, truncated shape differences.

7.8.1 Shape Space

An example application of shape differences that does *not* rely on exact reconstruction of local geometry involves the extraction of variability within a collection of related shapes. Suppose we choose an arbitrary base shape and compute its shape difference matrices with the remaining shapes in a collection. Then, a simple low-dimensional description of shape variability is to do PCA on the collection of matrices, resulting in the embedding of each shape as a point in PCA space.

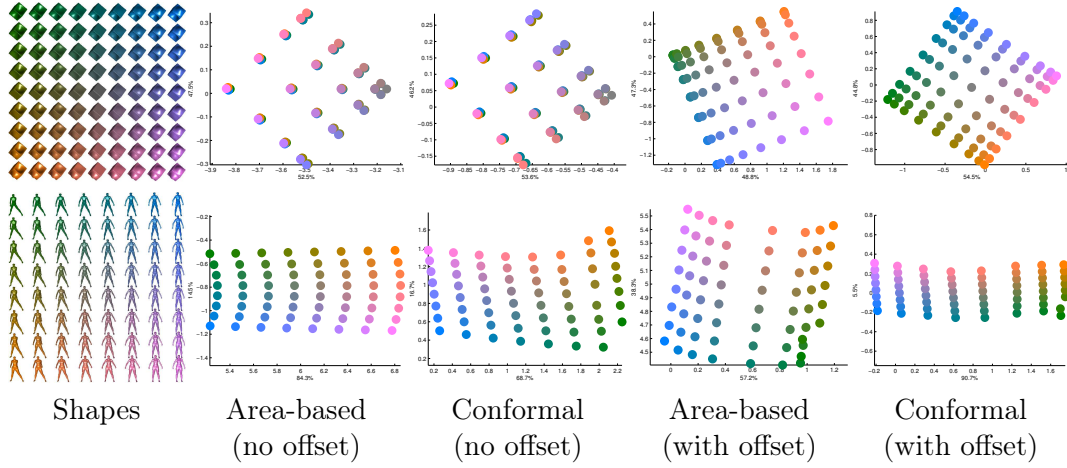


Figure 7.7 – PCA on collections of shape differences reveals the axes of variability within a collection; each shape on the left is colored the same as its corresponding points in the plots. The area-based and conformal differences are unable to distinguish the inward and outward bumps in the top example, leading to clusters of four points.



Figure 7.8 – Human models from Figure 7.7 sorted by the first PCA dimension for area-based shape differences (top) and area-based differences including an offset surface (bottom). The differences without offsets distinguish body type, while the differences with offsets distinguish pose.

We use PCA experiments to illustrate the power of our proposed extrinsic differences. For instance, Figure 7.7 illustrates embeddings of two-parameter shape collections into the plane using the procedure above ($k_M = 50, k_N = 100$). The top row illustrates the need for extrinsic differences most clearly. Here, we generate cubes with smooth bumps, smoothly varying from an inward bump to an outward bump. Intrinsic shape differences are identical for inward and outward bumps, leading to PCA embeddings that cluster sets of four shapes together. Adding extrinsic information disambiguates the embedding problem, separating the clustered points. Similarly, the extrinsic area-based shape difference best separates the parametric human models evenly among the two axes (57.2% variability along the principal axis, 38.3% along the secondary axis); interestingly, conformal shape differences among offset surfaces do not exhibit much variability for this particular class of surfaces.

Figure 7.8 highlights how intrinsic and extrinsic shape differences can measure different properties of shape. We sort the collection of human models by the one-dimensional embeddings (x -axis) of intrinsic (top) and extrinsic (bottom) area-based shape differences ($k_M = 50, k_N = 100$). The intrinsic shape differences distinguish the body type of the

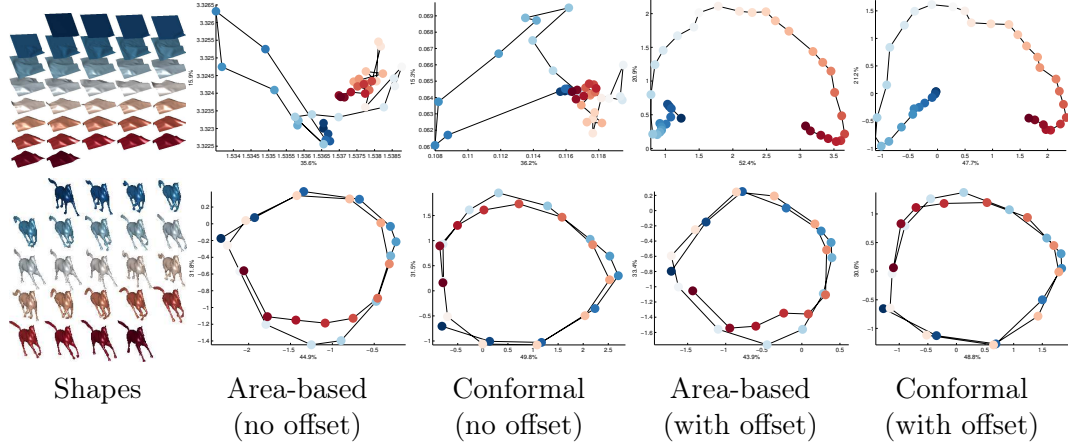


Figure 7.9 – PCA on shape differences applied to recovering the sequence of animated frames for a simulated piece of cloth (top) and galloping horse (bottom).

model and are invariant to the pose of the arms and legs; this ordering reflects the property that articulated deformations of humans are nearly isometric. Complementing this embedding, the extrinsic differences distinguish pose and are less sensitive to body type. This property is also visible in Figure 7.7 since the area-based embeddings without and with the offset are transposed from one another.

Figure 7.9 shows a similar experiment applied to shapes from individual frames of animation sequences. Both intrinsic and extrinsic shape differences are able to recover the cyclic structure of a galloping horse animation; this indicates that the galloping motion contains both intrinsic and extrinsic deformation modes. Contrastingly, the intrinsic differences severely underperform in recovering an animated sequence of deforming cloth. The physics of cloth naturally avoids intrinsic stretching and shearing, maintaining the initial developable structure. Thus, intrinsic shape differences provide little-to-no information, while the extrinsic differences capture the evolution of the animation.

From a wider perspective, the experiments in this section reveal the value of explicitly representing both intrinsic and extrinsic deformation in navigating datasets of 3D surfaces. A sizable fraction of geometry processing algorithms, including the original work on shape differences, focuses on shape exploration based exclusively on intrinsic structure. Yet, motions like the deformation of a piece of cloth cannot be captured by this representation. While cloth deformation may be an extreme example, based on these results we advocate inclusion of both intrinsic and extrinsic structures in shape analysis rather than discarding the extrinsic information.

7.8.2 Effects of Truncation

The propositions in this chapter show that discrete shape differences completely encode geometric structure when they are written in a full basis. For many applications, however, we approximate shape differences in a truncated low-frequency spectrum. While the

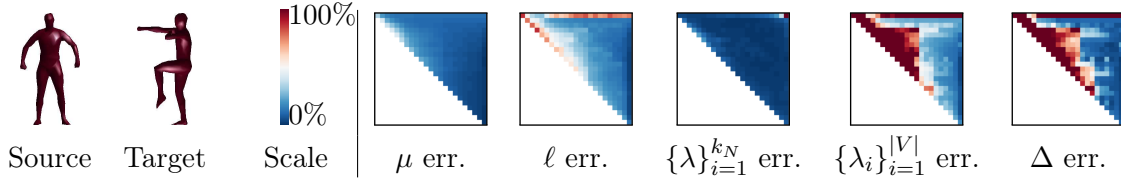


Figure 7.10 – Effects of truncation on computation of mesh structure. See §7.8.2 for discussion.

effects of this truncation are difficult to characterize mathematically, in this section we evaluate the effects of this approximation numerically.

There are two potential sources of truncation error in the twice-truncated differences discussed in §7.6.3: The choice of k_M and the choice of k_N . As mentioned in §7.6.3 these two parameters have slightly different effects; decreasing k_M corresponds to removing rows or columns of the shape difference matrices, while decreasing k_N can affect the values of the entries.

Figure 7.10 illustrates the results of an experiment varying k_M and k_N for intrinsic shape differences and using the pipeline described in §7.7 to recover areas and edge lengths; recall that this technique extracts areas and edge lengths on N using calculations on M . We choose a pair of meshes with a ground-truth map to avoid additional error due to map approximation ($|\mathcal{X}_M| = |\mathcal{X}_N| = 1000$).

Each color plot shows the relative error of assorted quantities extrapolated from the truncated shape differences: face areas (μ), edge lengths (ℓ), truncated eigenvalues ($\{\lambda\}_{i=1}^{k_N}$), full eigenvalues ($\{\lambda\}_{i=1}^{|\mathcal{X}|}$), and entries of the Laplacian (Δ). We assume $k_N \geq k_M$, providing the upper-triangular structure of the plots; the vertical axis represents k_M (range: $k_M \in [60, 500]$) and the horizontal axis represents k_N (range: $k_N \in [60, 500]$). We choose ϵ so that the viscosity regularizer contributes $< 10\%$ of the optimal objective.

These plots show that even truncated shape differences can be used to extract per-face and per-edge information about the mesh using our pipeline. Even with 15% of the Laplacian eigenvectors, we can relatively reliably extract the face areas and edge lengths of the target mesh. Even on challenging tasks like recovering the *full* spectrum of the target mesh—beyond the eigenvalues used to compute the shape difference—our algorithm has some success.

The choice of k_N is particularly important. Intuitively, this phenomenon might be explained by the fact that modulating k_N changes the values in the shape difference matrices rather than just their size. The top row of the matrix also exemplifies a pattern we observed across our experiments; below a certain value for k_M , there is not enough information to get a meaningful indication of local geometry from the shape difference matrices.

7.8.3 Intrinsic Recovery

The experiments in §7.8.2 illustrate a remarkable observation, that we are able to recover local information about the target of a shape difference from a truncated shape difference. That is, the nonnegativity and semidefinite constraints proposed in §7.7 paired with

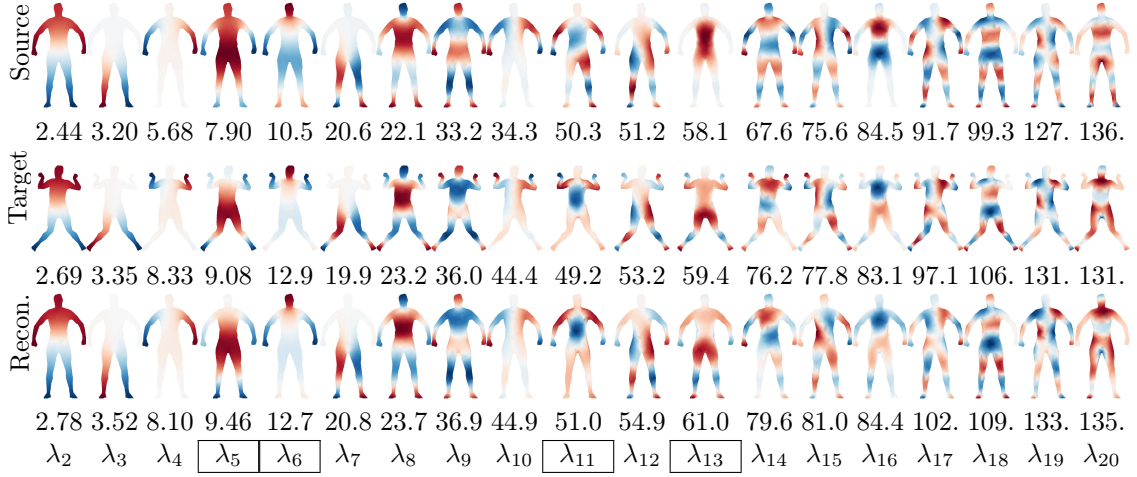


Figure 7.11 – Our machinery can be used to pull back Laplacian operators from a target mesh to a source. Here, we use a truncated functional map (100 Laplace–Beltrami basis functions on source, 200 on target) to compute revised edge lengths on the source mesh. Eigenfunctions of the Laplacian before and after edge length adjustment are shown with eigenvalues; boxed columns provide examples where the eigenfunction changes structure significantly.

regularization are sufficient to avoid the null space of the truncated linear systems for recovering areas and edge lengths.

As the cotangent Laplacian of a triangle mesh (with or without area weights) is an intrinsic structure, we can use our computed vectors μ and ℓ to *pull back* the Laplacian operator from N to M .

This technique is illustrated in Figure 7.11. In this experiment, we compute shape differences from a 100×200 functional map to compute μ and ℓ on the source surface; we then use (7.2) to construct a new Laplacian operator on M using μ and ℓ pulled back from N and show eigenfunctions of the resulting operator. Not only do the eigenvalues of the pulled-back Laplacian better approximate the Laplace–Beltrami eigenvalues of N , but qualitatively the eigenfunctions of the pulled-back Laplacian exhibit more structure in common with the eigenfunctions of N . Boxed examples in Figure 7.11 show particularly striking differences between the source and reconstructed eigenfunctions.

Figure 7.12 illustrates an application of recovering edge lengths from truncated shape differences. Without constructing an embedding, we use pulled-back edge lengths to compute two commonly-used intrinsic functions: single-source geodesic distances and the wave kernel signature [5]. Our edge lengths enable computation of these functions on the source mesh using the metric of the target, given a functional map between them. As a baseline, computing these functions on the target and pulling them back to the source using the functional map (right column) is less accurate; this is due to truncation of the functional map, which removes high frequencies e.g. at the center point of the geodesic function.

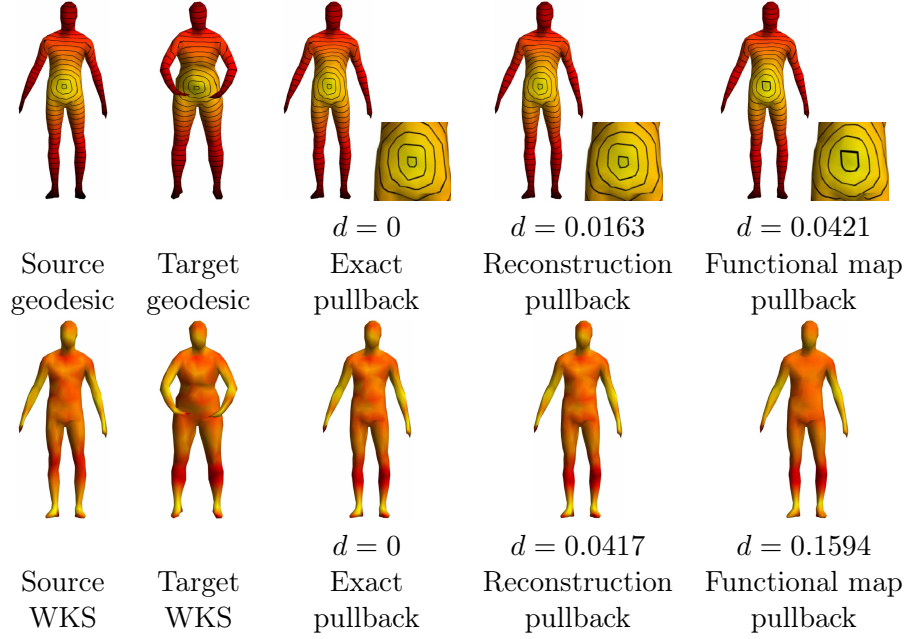


Figure 7.12 – Our technique can be used to recover the pullback metric and therefore compute geodesic distances without direct access to the target mesh. We compare three geodesic pullbacks on the source mesh: the exact pullback using point-to-point correspondence, the geodesic computed by reconstruction of the metric from the shape differences, and the pullback of the geodesic function using a functional map. For each pullback we compute the L^2 distance d to the exact version. Our method achieves better reconstructions than the direct usage of a functional map.

7.8.4 Reconstruction

Figures 7.13, 7.14 and 7.15 illustrate experiments in which geometry is reconstructed after estimating local structure from shape differences. To highlight our method’s effectiveness on extrinsic motion, we show behavior on human shapes and cloth simulation data.

Figure 7.13 applies our method to reconstructing models of humans from shape differences. From a coarse human base mesh ($|\mathcal{X}_M| = |\mathcal{X}_N| = 502$; $M = 100, k_N = 200$ in truncated experiments), we recover various poses. We compare reconstructions using only the intrinsic shape difference (right of each pair) to reconstructions using intrinsic and extrinsic differences together (left of each pair); we also compare using a truncated basis for shape differences (second column) to using a full basis (third column). As a baseline, we compare to [15], which uses only intrinsic geometry (rightmost column). Reconstruction from intrinsic information shows considerable artifacts due to the non-uniqueness of the solution of the embedding problem. Our provably complete intrinsic/extrinsic description is much more stable and close to the solution. The truncation of the basis, discussed in §7.8.2, tends to smooth out the sharp creases as they are represented as high frequency features.

In Figure 7.14, we interpolate between frames of an animation sequence ($|\mathcal{X}_M| = |\mathcal{X}_N| =$






















						
		$d_H = 0.036$	$d_H = 0.054$	$d_H = 0.013$	$d_H = 0.030$	$d_H = 0.058$
						
		$d_H = 0.083$	$d_H = 0.140$	$d_H = 0.023$	$d_H = 0.116$	$d_H = 0.147$
						
		$d_H = 0.064$	$d_H = 0.110$	$d_H = 0.0153$	$d_H = 0.105$	$d_H = 0.132$
Source	Target	Intrinsic	Intrinsic	Intrinsic	Intrinsic	Intrinsic [15]
		Extrinsic	—	Extrinsic	—	—
		<i>Truncated</i>	<i>Truncated</i>	<i>Full</i>	<i>Full</i>	<i>Full</i>

Figure 7.13 – Mesh recovery from a source mesh and shape differences, with (left) and without (middle) the extrinsic shape difference. Intrinsic mesh recovery using a concurrent method (right). The distance to the target d_H is measure by the Hausdorff distance on the prealigned point cloud.

1089, $k_M = 100$, $k_N = 200$). After running a cloth simulation with coarse time steps, we compute the shape difference between subsequent frames ($t \in [0, 1]$). We then use the method in §7.7.3 to construct plausible motion between the frames by interpolating linearly between the computed shape differences ($t = 0.5$). We further extrapolate the motion beyond the $t \in [0, 1]$ range to $t = 1.5$, effectively exaggerating the deformation between the frames. As expected, the extrinsic shape differences allow for reconstruction of largely isometric cloth motion.

Figure 7.15 illustrates a more challenging experiment ($|\mathcal{X}_M| = 669$, $|\mathcal{X}_N| = 1089$, $k_M = 60$, $k_N = 180$). In this case, we reconstruct the same cloth simulation sequence but vary the topology of the source and target meshes. Now it is impossible to pull back the deformation exactly to the new mesh topology, but we still reconstruct plausible motion, with the notable exception of artifacts near the boundary of the patch.

7.8.5 Timings

Figure 7.16 shows timings by stage for our pipeline, applied to meshes of various sizes and topologies. We employ a simplistic single-threaded implementation in MATLAB, using the Mosek toolbox [79] in the CVX library for convex optimization [45]; for this

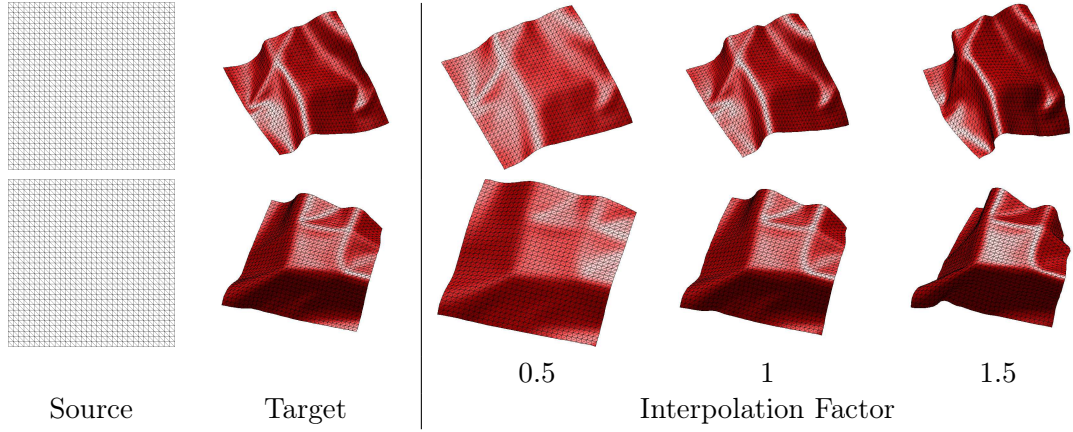


Figure 7.14 – Mesh recovery and interpolation from a source mesh and the intrinsic/extrinsic shape differences. The target meshes come from a cloth simulation sequence.

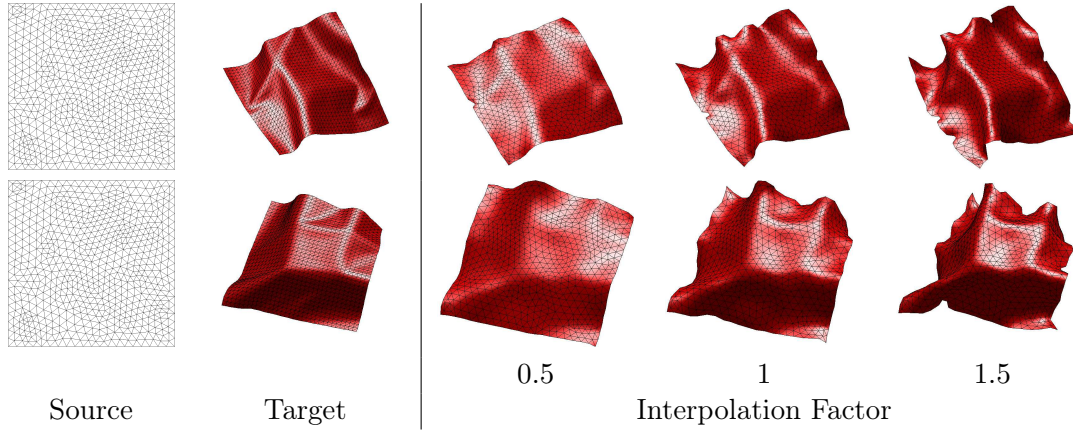


Figure 7.15 – Mesh recovery and interpolation. The source mesh has different connectivity than the target. The target meshes come from a cloth simulation sequence.

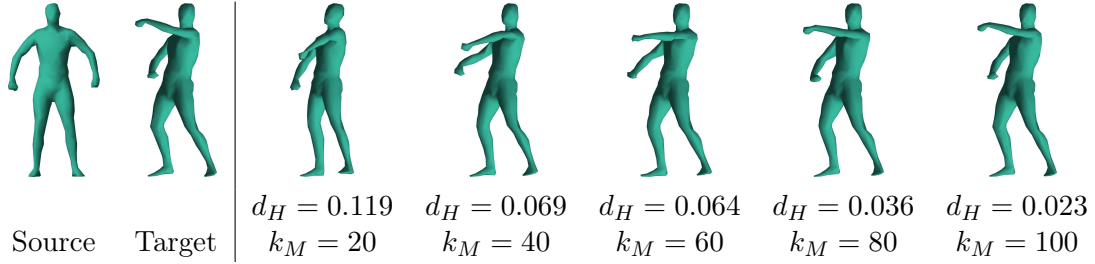
reason, the timings should be viewed as relatively pessimistic upper bounds. Even so, our implementation—including semidefinite constraints, regularization, and the like—can handle meshes of up to several thousand vertices.

7.9 Discussion & Conclusion

In this chapter, we introduced a new way to express intrinsic *and* extrinsic shape information through functional shape differences. Not only do we prove that discrete shape differences can be used to recover shape, but we also extend to characterizing shapes up to rigid motion rather than isometry. Our four shape differences together—two intrinsic and two extrinsic—comprise a powerful description of shape that applies to a wide range of variability, including not only non-isometric shapes but also models

	# Eigen- functions	Intrinsic metric					
		$ V $	$ F $	$ E $	Area	Edge	
Human	100	502	1000	4500	4.7s	9.0s	
Cloth HD	200	1089	2048	3136	20.6s	79.8s	
Cloth LD	60	669	1256	1924	6.0s	3.8s	
Faces	180	588	1097	1687	8.5s	18.8s	
Horse	160	752	1500	2250	10.1s	80.1s	
	# Eigen- functions	Offset metric					Reconst. Tets.
		$ V $	$ F $	$ E $	Area	Edge	
Human	100	1502	3000	4500	16.5s	310.9s	195.6s
Cloth HD	200	3135	6016	9150	430.5s	1833.0s	912.6s
Cloth LD	60	1925	3688	5612	31.5s	73.2s	412.7s
Faces	180	1682	3208	4892	177.3s	709.6s	204.2s
Horse	160	2252	4500	6750	293.1s	666.6s	447.5s

Figure 7.16 – Performance measured on a 2015 iMac 3.3GHz.

Figure 7.17 – Example of failure in mesh recovery from a source mesh and shape differences. As the size of the shape difference increases more details are added to the reconstructed deformation. At $k_M = 100$ and above we achieve a high-quality reconstruction.

obtained from physical simulation and animation. We also show that the inverse problem of recovering shape structure from shape differences can be meaningful even in the under-determined truncated case.

While this work offers the possibility of direct application in pipelines for shape search, embeddings of shape space, and approximate reconstruction, it also suggests myriad avenues for future research. On the theoretical side, a better understanding of the effect of Laplace–Beltrami eigenfunction truncation may provide better guidance for the minimal-sized shape differences needed to reconstruct a shape; spectral truncation is a common part of the geometry processing pipeline, so any relevant theory would have the potential to affect understanding of many existing algorithms.

On the practical side, the primary limitation of our proposed reconstruction methods is the introduction of semidefinite constraints in computing the squared edge lengths ℓ ; multi-scale or lighter-weight optimization methods would enable application to larger-scale meshes. Furthermore, the regularization proposed for recovery of μ and ℓ in §7.7 is very generic and can be ineffective for noisy or highly truncated shape differences. Application

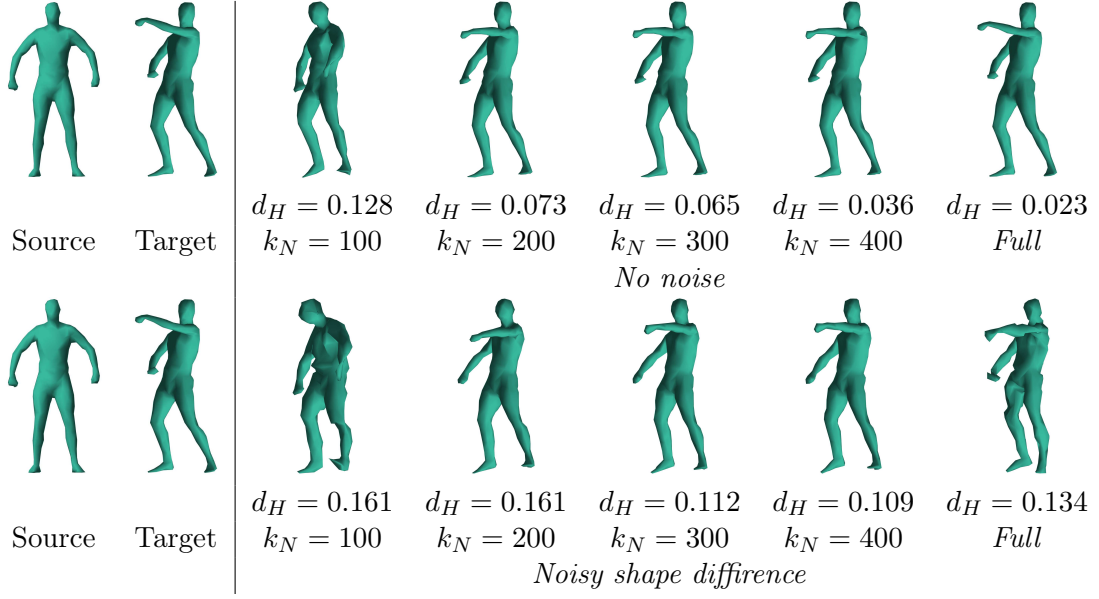


Figure 7.18 – Impact of the basis truncation on from a source mesh and *noisy* shape differences. In this experiment, we fix the number of basis functions on the source shape to $k_M = 100$ and reconstruct the embedding for various k_N . Top row: With no additional noise, the quality of the embedding increases with k_N . Bottom row: With added noise, larger k_N —which normally yields better transfer of high frequency deformation—does not increase the quality of the reconstruction. The noisy shape differences do not correspond to an actual embedding.

of machine learning techniques may allow for the characterization of edge length and triangle area distributions specific to a given class of shapes, considerably reducing the search space for our recovery algorithms.

Figures 7.17 and 7.18 show examples illustrating these potential avenues for improving our pipeline. Figure 7.17 shows how reconstruction can fail when shape differences are over-truncated; stronger regularizers might fill in missing information when truncated shape differences are insufficient to recover edge lengths to high precision. Figure 7.18 shows results of shape reconstruction in the presence of noise. Here, we add noise directly to the shape difference matrix so that it no longer corresponds to an embedded surface. At some point, increasing k_N does not improve the reconstruction result, because noise in the entries dominates added high-frequency shape information.

Functional Characterization of Deformation Fields

In this chapter we present a novel representation for infinitesimal deformations of 3D shapes, by considering the induced changes in the underlying metric. Unlike traditional representations given, for example, by the infinitesimal displacement of each point, our framework allows to consider shape deformations as operators acting on real-valued functions defined on the shapes. This enables a wide variety of applications such as deformation design through precise control of metric distortion, deformation analysis and transfer and even improved shape matching by considering the composition of the deformation with other functional operators. Fundamentally, our approach helps to establish a direct connection between extrinsic deformation fields and changes in intrinsic metric quantities, which can be useful in many deformation processing scenarios.



Figure 8.1 – Example of deformation design using our framework where all objectives are easily expressed as linear constraints. Left: A set of local constraints for the deformation fields on two different shapes. Middle: The deformations are computed separately on each shape to be as-isometric-as possible. Right: Joint deformation design with a soft (functional) map and no pointwise correspondences between the shapes.

8.1 Introduction

Designing and analyzing shape deformations is a central problem in computer graphics and geometry processing, with applications in scenarios such as shape manipulation [135, 114], animation and deformation transfer [117], shape interpolation [61, 130], and even anisotropic meshing [91] among myriad others. Traditionally, shape deformation has been motivated by interactive applications in which the main goal is to design a

deformation that satisfies some user-prescribed handle constraints while preserving the main structural properties of the shape. In other applications, such as shape interpolation and deformation transfer, that lack handle constraints, the goal is to design a global deformation field that would satisfy some structural properties as well as possible.

In both types of applications, most deformation methods are based on specifying a deformation energy and providing a method to optimize it. On the other hand, another very productive line of work has demonstrated that by choosing an appropriate *representation* for shape deformations, many tasks can become significantly easier, and in particular can help to enforce certain properties of the deformation field, which are otherwise very difficult to access and optimize for. In addition to the classical per-vertex displacement vectors, such representations have included gradient-based deformations [135, 136], Laplacian-based approaches [70, 115] and Möbius transformations in the context of conformal deformations [33, 128] among others.

At the same time, several recent works have demonstrated that several basic operations in geometry processing can be represented as linear operators acting on real-valued functions defined on the shape. This includes the functional representation of mappings described in Chapter 4 or correspondences [88, 96], representations of vector fields as derivations [95, 6] and formulation of shape distortion via shape difference operators introduced in [106] and described in Chapter 4. One advantage of these representations is that linear operators can be naturally composed, which makes it easy to define, for example, the push-forward of a vector field with respect to a mapping, if both are represented as linear operators, or to solve for Killing vector fields, by composition between a derivation and the Laplacian operator.

While tangent vector fields are classically understood as operators (derivations) in differential geometry, extrinsic vector fields do not enjoy a similar property. Our main goal is to provide a coordinate-free representation of extrinsic vector fields (infinitesimal deformations) as functional operators, which will prove useful for analysis and design of shape deformations. As we demonstrate below our representation greatly simplifies certain tasks such as deformation transfer, deformation symmetrisation and even the computation of shape correspondences by composition with other operators. Moreover, it provides an explicit link between infinitesimal deformations and the changes in intrinsic metric quantities, which can be useful in a variety of analysis and deformation processing tasks.

For example, consider two shapes shown in Fig. 8.1 (left). By using our framework, it is possible to combine local extrinsic deformation constraints with intrinsic objectives such as constructing a deformation field that is as-isometric-as-possible (center). Moreover, our coordinate-free representation allows to relate deformations on multiple shapes, enabling deformation transfer and joint design even when only soft (functional) correspondences are known.

8.2 Related Work

Shape deformation is one of the oldest and best-researched topics in computer graphics and geometry processing. We therefore only mention works most directly related to ours and refer the interested reader to surveys including [85, 17] and Chapter 9 in [16].

A multitude of methods exists for surface deformation starting with the seminal work of [122], its early follow-ups including [25, 133] and the multi-scale variants, such as [140, 63, 47] among many others. Similarly to our approach, many of these techniques are based on optimizing the so-called elastic thin shell energy that measures stretching and bending, and which is often linearized for efficiency. In the majority of cases, the deformation is represented explicitly as an extrinsic vector field defined on a surface, making deformation transfer difficult in the absence of precise pointwise correspondences. A number of methods have proposed alternative representations for deformation fields, which greatly simplify certain tasks in design and analysis. This includes gradient-based techniques [135, 136] which consider the deformation field by aligning its gradient with a set of local per-triangle transformations. By working in gradient space, constraints can be posed independently on the triangles and then optimized globally by solving the Poisson equation. Similarly, Laplacian-based techniques [115, 70, 86] are based on defining shape deformations by manipulating per-vertex differential coordinates (Laplacians) in order to match some target Laplacian coordinates. Such differential coordinates enable direct editing of the local shape properties, which can be especially beneficial for preserving and manipulating the high-frequency details of the surface. However, these coordinates are typically not rotationally invariant and additional steps are necessary to introduce invariance [115, 70, 93].

More recently, a number of methods have introduced representations for mesh deformations specifically geared towards particular shape manipulations, such as computing conformal transformations by designing special maps into the space of quaternions [33] or by using face-based compatible Möbius transformations [128]. These techniques are rotationally invariant and coordinate-free, while being restricted to special types of manipulations. Another technique, closely related to ours, designs shape deformations by constructing a continuous divergence-free vector field [129], and applying path line integration to obtain a deformed shape. We also consider the effect of the deformation on the metric, but both analyze the distortion of arbitrary extrinsic vector fields and show how they can be represented in coordinate-free way as linear functional operators.

Our approach of considering the deformation via its induced metric distortion is also related to the work of [39] and [110] who manipulate shapes by explicitly editing their curvature properties. Moreover, our use of the strain tensor in characterizing metric distortion is closely related to the applications in various physically based deformation scenarios including [124, 81] among many others (see also the surveys on physically based elastic deformable models [85, 104]). Our approach is also related to the works that aim to design as-isometric-as-possible shape deformations [139, 113, 73]. Similarly to the latter work, however, our framework is general and allows an arbitrary prescribed distortion, although our method works directly on surface representations and moreover enables applications such as joint deformation design.

Finally, our joint deformation design is related to the deformation transfer and interpolation techniques such as [117, 9] and [61] to name a few. However, we place special emphasis on relating infinitesimal deformations between shapes with only soft (or functional) correspondences, which are often much easier to obtain than detailed point matches.

In contrast to the majority of existing techniques our goal is to devise a coordinate-free representation of infinitesimal deformations as linear functional operators, by making an explicit connection between the extrinsic deformations and the change in intrinsic metric quantities. Thus, although we build on classical constructions such as the infinitesimal strain tensor, we show how they can be exploited to create a functional representation of shape deformation, which can be used in conjunction with other operators. As we demonstrate below, our representation is particularly useful for relating deformations that exist on multiple shapes, with only soft correspondences between them. In particular, it enables joint deformation design and transfer without triangle or point-to-point matches, and allows to introduce extrinsic information in the computation of functional maps, greatly increasing the applicability of these techniques in a wide range of application scenarios.

8.3 Overview

Our main strategy for devising a functional representation of infinitesimal deformation is to consider the effect of the deformation on the intrinsic shape metric. Namely, we will characterize the extrinsic vector fields by the isometric distortion that they induce. For this purpose we will start by considering the shape difference operators described above, which provide a way to capture the isometric distortion induced by a map between two shapes as a pair of linear operators acting on functions on one of the shapes. Thus, in this chapter our motivation is to use the information contained in those operators to design and synthesize shape deformations.

The first challenge presented by this approach is the fact that the shape difference operators only contain intrinsic information which is typically not enough to specify an embedding, and thus a deformation. To tackle this problem we restrict our attention to infinitesimal deformations which can be represented by extrinsic vector fields, and which, as we show under certain genericity conditions, are fully encoded by the isometric distortion they induce.

Second, the definition of the shape difference operators introduced in [106] is divided between an area and conformal-based distortions. While this distinction might be desirable for deformation analysis, it is cumbersome for deformation synthesis since it would require us to represent each extrinsic vector field via a pair of functional operators. For this reason we introduce a unified shape difference operator, that characterizes intrinsic distortion fully.

Finally, although the shape difference operators characterise intrinsic distortion, designing a deformation that would reproduce a given shape difference leads to challenging optimization problems as shown in [15]. As we demonstrate below, our approach of

considering infinitesimal deformations greatly simplifies the reconstruction procedure, by allowing us to phrase it with a single linear system of equations.

To summarize, we proceed in three stages: first we introduce a unified shape difference operator that fully characterizes isometric distortion. Second, we show how extrinsic vector fields can be represented by *infinitesimal shape difference operators*, which act linearly on the real-valued functions, in the same way as shape difference operators. Finally, we demonstrate that extrinsic vector fields can be recovered from infinitesimal shape difference operators by solving a linear system of equations.

Our main contribution therefore include:

- Defining a unified shape difference operator that fully characterizes isometric distortion.
- Introducing *infinitesimal shape difference operators* as a way to represent extrinsic vector fields as operators acting on functions, represented as matrices in the discrete setting.
- Showing how infinitesimal shape difference operators can be used to naturally add extrinsic information (second fundamental form) into the computation and analysis of functional maps.
- Presenting various applications that demonstrate the usefulness of our representation, in particular by showing how the infinitesimal deformations can be analysed, designed, transfered and used in map computation by exploiting the operator representation.

8.4 Extrinsic Vector Fields as Operators

In this section we provide a coordinate-free representation of extrinsic vector fields by considering their action on the underlying shape metric. As mentioned in the previous section, our main strategy will be to represent extrinsic vector fields via *infinitesimal shape difference operators*. Therefore, we first define a unified isometric operator in Section 8.4.1. We then define infinitesimal shape difference operators in Section 8.4.2 and list some of their key properties in Section 8.4.3. Throughout this section we assume that we are dealing with smooth surfaces without boundary embedded in \mathbb{R}^3 . The appropriate discretization of all the concepts introduced in this section will be given in Section 8.5.

8.4.1 Isometric Shape Difference Operator

While the two shape difference operators defined in Chapter 4 are very convenient to separate measure (area) and conformal distortion, they also imply that two operators are needed to fully characterize a deformation, which will be cumbersome for representing extrinsic vector fields. Moreover, these two operators are not commensurable since D_A is defined through products of functions, unlike D_C which is a differential operator defined through inner products of gradients. Thus, we introduce a single unified shape difference operator below.

Unified shape difference The main reason for which D_C is only sensitive to conformal changes is that both the inner product and the integration are taken on the target shape. To see this, let us re-write the conformal shape difference D_C by integrating on a fixed domain using the pullback metric tensor as follows:

$$\langle f, D_C(g) \rangle_{H_0^1}^M = \int_M C_{\varphi^{-1}} (\langle \nabla C_{\varphi}(f), \nabla C_{\varphi}(g) \rangle) d(\varphi_* \mu_N).$$

This definition is equivalent to the one given in Eq. (4.3) in Chapter 4, but here we simply highlight the integration with respect to the pushforward measure $d(\varphi_* \mu_N)$ rather than the measure on M itself. Here, the linear operator $C_{\varphi^{-1}}$ is simply the functional map with respect to the inverse diffeomorphism φ^{-1} . Since we are dealing with surfaces, the change in the area will cancel out on the right, leaving D_C only sensitive to conformal changes.

To define a unified shape difference taking into account all intrinsic changes one should compare the pullback metric to the metric on M while keeping the integrating measure fixed. We thus propose a unified shape difference operator D_I that fully characterizes isometric distortion.

Definition 8.4.1 *Assuming that $\varphi : N \rightarrow M$ is a diffeomorphism, the unified shape difference $D_I : H_0^1(M) \rightarrow H_0^1(M)$ is defined implicitly by:*

$$\langle f, D_I(g) \rangle_{H_0^1}^M := \int_M C_{\varphi^{-1}} (\langle \nabla C_{\varphi}(f), \nabla C_{\varphi}(g) \rangle) d\mu_M.$$

The existence of D_I is once again guaranteed by the Lax-Milgram theorem in H_0^1 . Moreover, as we claimed above, the following proposition shows that the unified shape difference fully characterizes isometric deformation.

Proposition 8.4.2 *The following equivalence holds:*

- $D_I(f) = f$ for all $f \in H_0^1(M)$ if and only if φ is an isometry,
- $D_I = D_C$ in $H_0^1(M)$ if and only if φ is an area-preserving.

Proof Using Equation (4.5) we can rewrite Definition 8.4.1:

$$\begin{aligned} \int_M \langle \nabla f, \nabla D_I(g) \rangle d\mu &= \int_M C^{-1} (\langle \nabla C(f), \nabla C(g) \rangle) d\mu \\ &= \int_M \langle \nabla f, \mathbf{A} \nabla g \rangle d\mu \end{aligned}$$

where A is a symmetric (1,1)-tensor defined as $\mathbf{A}_{ij} = ((\varphi^{-1})^* \mathbf{g}^N)^{ik} \mathbf{g}_{kj}^M$. This definition allows us to prove both equivalences.

- If φ is an isometry then $(\varphi^{-1})^* \mathbf{g}^N = \mathbf{g}^M$ so

$$\int_M \langle \nabla f, \nabla D_I(g) \rangle d\mu_M = \int_M \langle \nabla f, \nabla g \rangle d\mu$$

Therefore $D_I(g) = g$ for all function g in H_0^1 .

If $D_I(f) = f$ then $\int_M \langle \nabla f, \nabla (\text{Id} - \mathbf{A})g \rangle d\mu = 0$ for all $f, g \in H_0^1(M)$. Lemma 4.2.2 tells us that φ is an isometry.

- If φ is an area-preserving then $\det((\varphi^{-1})^* \mathbf{g}^N) = \det(\mathbf{g}^M)$ so the tensor field \mathbf{A} is equal to the tensor field \mathbf{B} as defined in Equation 4.6.

If $D_I(f) = D_C(f)$ then, using Lemma 4.2.2, the equality holds between the tensor \mathbf{A} and \mathbf{B} . We conclude that $\det((\varphi^{-1})^* \mathbf{g}^N) = \det(\mathbf{g}^M)$.

□

This new shape difference is defined up to isometric deformation and a pullback operation can be defied to transport a shape difference on another manifold. More precisely, Proposition 4.2.4 can be restated for the unified shape difference.

Proposition 8.4.3 *Assume that $D_I^\varphi : H_0^1(M) \rightarrow H_0^1(M)$ represents the distortion of the metric between the surfaces M and P induced by the diffeomorphism $\varphi : P \rightarrow M$ and $D_I^\phi : H_0^1(P) \rightarrow H_0^1(P)$ the distortion between the surfaces P and N linked through $\phi : N \rightarrow P$. The distortion $D_I^{\varphi \circ \phi} : H_0^1(M) \rightarrow H_0^1(M)$ associated to $\varphi \circ \phi : N \rightarrow M$ is given by*

$$D_I^{\varphi \circ \phi} = D_I^\varphi \circ C_\phi^{-1} \circ D_I^\phi \circ C_\varphi.$$

Proof The proof relies only on Definition 8.4.1:

$$\begin{aligned} \int_P C_\varphi \left(\left\langle \nabla f, \nabla D_I^{\varphi \circ \phi}(g) \right\rangle \right) d\mu &= \int_P C_\phi^{-1} \left(\left\langle \nabla(f \circ \varphi \circ \phi), \nabla(g \circ \varphi \circ \phi) \right\rangle \right) d\mu \\ &= \int_P \left\langle \nabla(f \circ \varphi), \nabla D_I^\phi(g \circ \varphi) \right\rangle d\mu \\ &= \int_P C_\varphi \left(\left\langle \nabla f, \nabla D_I^\varphi \left(D_I^\phi(g \circ \varphi) \circ \varphi^{-1} \right) \right\rangle \right) d\mu. \end{aligned}$$

This yields the equality $D_I^{\varphi \circ \phi}(g) = D_I^\varphi \left(D_I^\phi(g \circ \varphi) \circ \varphi^{-1} \right)$ for all $g \in H_0^1(M)$. □

Example Computing D_I is challenging when studying meshes with different connectivity as it requires the map and its inverse. However, when using it to define an operator representation for infinitesimal deformations, we will only be dealing with shapes with a fixed connectivity, for which we are able to obtain an appropriate discretization. Moreover,

even in the general case, we are able to approximate this quantity in order to compute it using the functional map framework, as shown in Section 8.5.

To illustrate the properties of the three shape differences we use a simple low-dimensional description of a shape collection. Here we choose a fixed base shape and compute the shape difference matrices with respect to the remaining shapes in a collection. Then, we represent each shape by its shape difference matrix and plot them as points in PCA space. Figure 8.2 represents the conformal deformation of a bunny into a sphere as viewed by the three shape differences. As expected the conformal shape difference is almost identity while the area and isometric shape differences both capture the distortion. In the second experiment, shown in Figure 8.2, we explore another collection obtained by the shearing of a plane patch. As this deformation is area preserving, the area-based shape difference provides no information, unlike the other two shape differences which are equal in this case.

8.4.2 Infinitesimal Deformations as Operators

One aspect of the shape difference operators, highlighted in [106], that makes them particularly useful for capturing and representing shape deformations is that both the

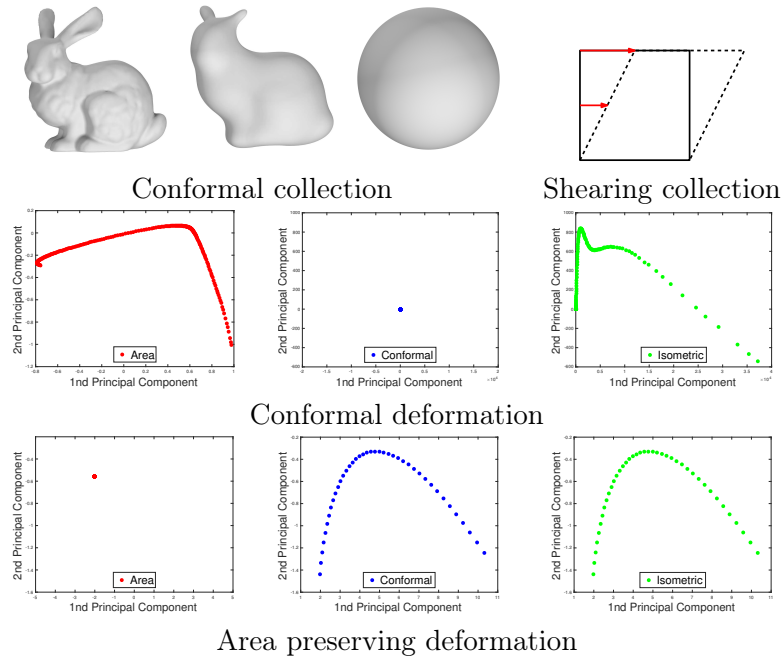


Figure 8.2 – Middle row: Approximately conformal deformation of a bunny into a sphere (top row left). The PCA applied to shape differences confirms the presence of large area and isometric distortion in contrast to small conformal distortion. Bottom row: Area preserving deformation of a plane (top row right). The area shape difference is almost constant while conformal and isometric differences agree.

domain and the range of these operators are functions on a fixed base shape. In other words, although they are obtained by considering the distortion with respect to some target shape, the final functional operators both act on and produce real-valued functions on the base shape. Our main goal in this section is to consider a one-parameter family of shapes M_t , given by displacing the points of a base shape along some fixed deformation field. This family of shapes will result in a one-parameter family of shape difference operators $(D_I)_t$ on the base shape, that we will use to define the infinitesimal shape difference operator $\partial(D_I)_t/\partial t$ at $t = 0$, as a coordinate-free representation of the deformation field.

Deformations via Extrinsic Vector Fields Specifically, given a surface M embedded in \mathbb{R}^3 we consider a family M_t of deformations of M , parameterized by a scalar t and isometrically immersed by the local mappings $F_t : U \subset \mathbb{R}^2 \rightarrow M_t \subset \mathbb{R}^3$. This family of manifolds is generated by the displacement of the points along the smooth vector field $V(p) \in T_p M \times T_p M^\perp \simeq \mathbb{R}^3$:

$$\frac{\partial F_t}{\partial t}(p) = V(p), \quad (p, t) \in M \times \mathbb{R}^+, \quad (8.1)$$

where F_0 is the local immersion of M . We call V a displacement field, or, alternatively, an extrinsic vector field to highlight the distinction with tangent vector fields.

Our main goal below is to characterize extrinsic vector fields as linear functional operators. For this, as mentioned above, we consider the family of diffeomorphisms $\varphi_t : M_t \rightarrow M$, given trivially via $\varphi_t(p) = F_t(p) - tV(p)$, and the associated functional maps C_{φ_t} mapping functions from M_0 to M_t . This creates a one parameter family of shape difference operators D_t^V (which can be taken either to be the area or conformal-based operators or the unified shape difference operator defined above). We then introduce the infinitesimal shape difference, as a functional representation of the deformation field V as follows:

Definition 8.4.4 *The functional representation of an extrinsic vector field V on a surface M is given by the functional operator E^V :*

$$E^V := \left. \frac{\partial D_t^V}{\partial t} \right|_{t=0} \quad (8.2)$$

We will call E^V the infinitesimal shape difference operator associated with the deformation field V .

We will use E_A^V , E_C^V and E_I^V to denote the infinitesimal shape difference operators arising from the area-based, conformal and unified shape differences respectively. Note that although we characterize the properties of all three infinitesimal shape difference operators, in all practical applications we will only use E_I^V as the functional representation of an extrinsic vector field V . This is because this operator allows us to represent an extrinsic vector field using a single linear operator, and moreover, as we show below, under certain genericity conditions, in the discrete case, E^V fully characterizes V up to rigid motion. Remark that since E^V is defined as a derivative of a one-parameter family of linear operators acting on real-valued functions on a surface, both the range and the domain of

E^V are also real-valued functions on M . As such it can naturally be represented in a multi-scale basis, composed with other functional operators, and analyzed thus enabling applications including design, processing and transfer of extrinsic vector fields, as well as their use in other tasks such as improved map (correspondence) computation. Moreover, as we demonstrate below E_I^V is very closely related to classical constructions in computer graphics, geometry processing and mechanics, based on the infinitesimal strain tensor.

8.4.3 Main Properties of Infinitesimal Shape Differences

Below we provide the defining characteristics of the infinitesimal shape difference operators E^V , which will be useful in both deriving the appropriate discretization in the case of shapes represented as triangle meshes, and in the applications shown below.

The Levi-Civita Covariant Derivative We first need to introduce some fundamental notions from differential geometry. In particular, we will use the Levi-Cevita connection to define derivatives on a surface. For this, consider a *tangent vector* u at some point $p \in M$, and an extrinsic vector field V on M . Then, given an arbitrary curve $\gamma(t)$ on M such that $\gamma(0) = p$ and $\gamma'(0) = u$, we obtain $\bar{\nabla}_u V = \left. \frac{\partial V(\gamma(t))}{\partial t} \right|_{t=0}$. Here $\bar{\nabla}_u V$ is the covariant derivative of the ambient space. Note that at a fixed point $p \in M$, $\bar{\nabla}_u V$ is a vector in \mathbb{R}^3 . We can project the covariant derivative onto the tangent plane at p to obtain a vector in the tangent plane, which we denote simply by $\nabla_u V$ where ∇ is the Levi-Cevita connection on M extended naturally to extrinsic vector fields, ([36] p. 126), using the local coordinates $\nabla_i V_j = \langle \partial_i V, \partial_j F \rangle$. We also remark that for any vector x in the tangent space, $\langle \nabla_u V, x \rangle = \langle \bar{\nabla}_u V, x \rangle$, which we will use in our discretization. This definition allows us to express the divergence of the extrinsic vector field V as the trace of the Levi-Civita connection, given, at a fixed point p by $\text{div}(V) = \mathbf{g}^{ij} \nabla_i V_j$.

The fundamental object that we consider below is the infinitesimal strain tensor, which can be understood as a bilinear form, acting on pairs of vectors x, y in the tangent plane of a point $p \in M$. Namely, given an extrinsic vector field V , the infinitesimal strain tensor $\mathcal{L}_V \mathbf{g}(x, y)$ is defined as:

$$\mathcal{L}_V \mathbf{g}(x, y) = \langle x, \nabla_y V \rangle + \langle \nabla_x V, y \rangle \quad (8.3)$$

This quantity has the advantage of being linear in the vector field V which makes it easy to handle for deformation and vector field design and therefore has been used in a wide variety of works in computer graphics [85].

With these definitions in hand we can express the fundamental properties of the infinitesimal shape difference operators described above.

Proposition 8.4.5 *Given a one parameter family of surfaces described in Eq. (8.1), for a fixed point p , the first-order change in the metric tensor \mathbf{g} and in the local area element*

$\mu = \sqrt{\det(\mathbf{g})}$ are given as:

$$\left. \frac{\partial \mathbf{g}(t)}{\partial t} \right|_{t=0} = \mathcal{L}_V \mathbf{g} \quad (8.4)$$

$$\left. \frac{\partial \mu(t)}{\partial t} \right|_{t=0} = \operatorname{div}(V) \mu. \quad (8.5)$$

Proof Those properties are easily proven when using local coordinates. Given a family of diffeomorphisms φ_t the Lie derivative of the metric tensor with respect to the vector field V denoted $\mathcal{L}_V \mathbf{g}$ is by definition:

$$\mathcal{L}_V \mathbf{g} := \left. \frac{\partial}{\partial t} ((\varphi_t^{-1})^* \mathbf{g}(t)) \right|_{t=0}.$$

Since the local immersion F_t use a common chart system, the coordinates of the pullback metric $((\varphi_t^{-1})^* \mathbf{g}^t)_{ij}$ are equal to the metric on M_t in local coordinates $\mathbf{g}_{ij}(t) = \langle \partial_i F_t, \partial_j F_t \rangle$. The computation of derivative is then straightforward:

$$\begin{aligned} \left. \frac{\partial}{\partial t} (((\varphi_t^{-1})^* \mathbf{g}(t))_{ij}) \right|_{t=0} &= \left. \frac{\partial}{\partial t} (\langle \partial_i F_t, \partial_j F_t \rangle) \right|_{t=0} \\ &= \nabla_i V_j + \nabla_j V_i \end{aligned}$$

From here Eq. 8.5 is obtained easily:

$$\begin{aligned} \left. \frac{\partial \mu(t)}{\partial t} \right|_{t=0} &= \left. \frac{\partial}{\partial t} (\sqrt{\det(\mathbf{g}(t))}) \right|_{t=0} \\ &= \frac{1}{2\mu} \det(\mathbf{g}) \mathbf{g}^{ij} (\nabla_i V_j + \nabla_j V_i) = \operatorname{div}(V) \mu \end{aligned}$$

□

This property makes apparent the fact that the strain tensor is equal to the Lie derivative of the metric tensor with respect to the extrinsic vector field V , and that the first-order change in the local area is controlled by the divergence of the vector field, as is similarly the case for tangent vector fields.

Infinitesimal Shape Differences As mentioned in Section 8.4.1, shape differences operators can be expressed using the pullback metric and pushforward measure. Using Proposition 8.4.5 we have all the tools to describe the infinitesimal shape difference operators. The following proposition characterizes these new operators.

Proposition 8.4.6 *Let V be a smooth deformation field on M , the derivatives of D_A , D_C and D_I at time zero satisfy for all smooth functions f, g :*

$$\begin{aligned} \langle f, E_A^V(g) \rangle_{L^2}^M &= \int_M \operatorname{div}(V) f g \, d\mu, \\ \langle f, E_C^V(g) \rangle_{H_0^1}^M &= \int_M \operatorname{div}(V) \langle \nabla f, \nabla g \rangle - \mathcal{L}_V \mathbf{g}(\nabla f, \nabla g) \, d\mu, \\ \langle f, E_I^V(g) \rangle_{H_0^1}^M &= - \int_M \mathcal{L}_V \mathbf{g}(\nabla f, \nabla g) \, d\mu. \end{aligned}$$

Proof The first statement is obtained by using (8.5):

$$\begin{aligned}\langle f, \partial_t D_A(g) \rangle_{L^2}^M &= \frac{\partial}{\partial t} \left(\int_{M_t} C_t(f) C_t(g) \, d\mu(t) \right) \Big|_{t=0} \\ &= \frac{\partial}{\partial t} \left(\int_M f g \sqrt{\frac{\det(\mathbf{g}(t))}{\det(\mathbf{g})}} \, d\mu \right) \Big|_{t=0} \\ &= \int_M \operatorname{div}(V) f g \, d\mu.\end{aligned}$$

The second statement is a consequence of Equation (4.6):

$$\begin{aligned}\langle f, \partial_t D_C(g) \rangle_{H_0^1}^M &= \frac{\partial}{\partial t} \left(\int_{M_t} \langle \nabla C_t(f), \nabla C_t(g) \rangle \, d\mu(t) \right) \Big|_{t=0} \\ &= \frac{\partial}{\partial t} \left(\int_M \partial_i f \mathbf{g}^{ij}(t) \partial_j g \sqrt{\frac{\det(\mathbf{g}(t))}{\det(\mathbf{g})}} \, d\mu \right) \Big|_0 \\ &= \int_M \operatorname{div}(V) \langle \nabla f, \nabla g \rangle \, d\mu \\ &\quad - \int_M (\langle \nabla f, \nabla_{\nabla g} V \rangle + \langle \nabla_{\nabla f} V, \nabla g \rangle) \, d\mu\end{aligned}$$

Starting from Definition 8.4.1:

$$\begin{aligned}\langle f, \partial_t D_I(g) \rangle_{H_0^1}^M &= \frac{\partial}{\partial t} \left(\int_M C_t^{-1} (\langle \nabla C_t(f), \nabla C_t(g) \rangle) \, d\mu \right) \Big|_{t=0} \\ &= \frac{\partial}{\partial t} \left(\int_M \partial_i f \mathbf{g}^{ij}(t) \partial_j g \, d\mu \right) \Big|_{t=0} \\ &= - \int_M \langle \nabla f, \nabla_{\nabla g} V \rangle + \langle \nabla_{\nabla f} V, \nabla g \rangle \, d\mu.\end{aligned}$$

□

As can be seen, the infinitesimal shape differences inherit the properties of the original operators. Namely, E_A^V vanishes if and only if $\operatorname{div}(V)$ is equal to zero, i.e., whenever V infinitesimally preserves the volume form. On the conformal side, finding an extrinsic vector field V such that $E_C^V = 0$ is equivalent to solving the *conformal Killing equation*: $\mathcal{L}_V \mathbf{g} = \operatorname{div}(V) \mathbf{g}$ characteristic of infinitesimal conformal vector field. Both properties combined lead to an infinitesimal isometric deformation induced by the vector field V captured by Eq. (8.4).

Moreover Prop. 8.4.6 reveals a clear link between shape differences:

$$\langle f, E_I^V(g) \rangle_{H_0^1}^M = \langle f, E_C^V(g) \rangle_{H_0^1}^M - \langle 1, E_A^V(\langle \nabla f, \nabla g \rangle) \rangle_{L^2}^M. \quad (8.6)$$

Thus, intuitively, the operator E_I , representing isometric distortion, can be decomposed into an area and conformal parts.

The operator representation of extrinsic vector fields, allows us to derive a some interesting properties specific to this representation and useful for applications, while making a connection between previously proposed shape difference operators and the classical strain tensor.

Linearity The first key property of E_I is its linearity with respect to both vector fields and functions:

$$\begin{aligned} E_I^{\alpha U + \beta V}(f) &= \alpha E_I^U(f) + \beta E_I^V(f), \\ E_I^V(\alpha f + \beta g) &= \alpha E_I^V(f) + \beta E_I^V(g). \end{aligned} \quad (8.7)$$

This makes it easily usable for vector field design, by representing the target deformation field in some fixed basis.

Lie derivative representation From Proposition 8.4.2 we deduce that $E_I^V(f) = 0$ for all f if and only if $\mathcal{L}_V \mathbf{g} = 0$, i.e., if the extrinsic vector field V preserves the intrinsic shape metric to first-order. The operator E_I^V clearly quantifies how the Lie derivative of the metric affects gradients of functions. So the linear dependence between shape operators shown in Eq. (8.6) can be understood as the decomposition of the matrix $\mathcal{L}_V \mathbf{g}$ into a trace free part, linked to the conformal Killing equation, and a divergence part, related to the change in area.

Vector field representation In general the operator E_I^V does not uniquely define an extrinsic vector field. From Prop. 8.4.2 the kernel of $V \mapsto E_I^V$ coincides with the vector fields satisfying $\mathcal{L}_V \mathbf{g} = 0$. In case of a volumetric manifold (i.e. $M \subset \mathbb{R}^3$) the infinitesimal change in the metric $\partial_t \mathbf{g}$ defines the vector field V up to rigid motion (see [27] and all the Korn's inequalities). In the case of a surface embedded in \mathbb{R}^3 the kernel of E_I^V includes infinitesimal isometries such as Killing vector fields but also local normal fields in planar areas. However such fields seem either rare or nonexistent generically. No rigidity result seems to be known for smooth surfaces. However, as we demonstrate below, in the discrete case it can be shown that for almost all surfaces the kernel of $V \mapsto E_I^V$ consists only of rigid deformations (Prop. 8.5.6).

Second-fundamental form representation One interesting special case to consider is the interpretation of E_I when the deformation field is the normal field n . By using Eq. (8.3) it is possible to see that the covariant derivative of the normal yields the second fundamental form denoted by $\mathbf{h}_p : T_p M \times T_p M \rightarrow \mathbb{R}$, more precisely $\mathcal{L}_n \mathbf{g} = 2\mathbf{h}$. Therefore the operator E_I^n captures the action of curvature on functions, since:

$$\int_M \langle \nabla f, \nabla E_I^n(g) \rangle d\mu = -2 \int_M \mathbf{h}(\nabla f, \nabla g) d\mu.$$

From a theoretical point of view the knowledge of the Laplace-Beltrami operator gives access to the first fundamental form and E_I^n yields information about the second. Thus these two operators jointly provide a coordinate-free representation of the embedding.

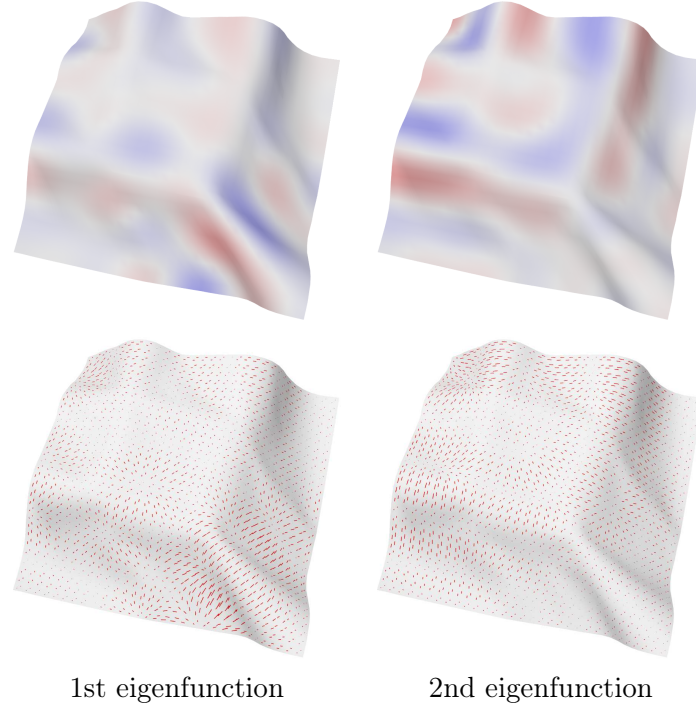


Figure 8.3 – Two eigenfunctions associated with the largest eigenvalues of the operator infinitesimal shape difference E_I^n for the normal field n . The gradients of these functions represent the direction of principal curvature (Bottom row).

The operator E_I^n can be used to obtain a multi-scale representation of curvature information on the triangle mesh, as shown in Figure 8.3. In particular, the eigenfunctions corresponding to the largest eigenvalues of E_I^n , are those that align the best with the maximal principal curvature direction, and can be obtained even if E_I^n is represented in a reduced functional basis, making the computation less sensitive to noise in the triangulation. Moreover, as we demonstrate in Section 8.7.2, the operator E_I^n can be used to inject extrinsic information into the computation of functional maps.

Composition with mappings In many applications, we are interested in the relation between deformation on multiple surfaces related by a mapping. In particular given a deformation field U_N of shape N and a diffeomorphism $\varphi : N \rightarrow M$ with the associated functional map (pullback) C_φ of functions from M to N , one can define an infinitesimal deformation V_M of shape M that produces the same metric distortion. Instead of looking directly at the deformation of the metric, which might require a mapping between individual triangles [117], we account for the action of the metric on functions:

$$E_I^{U_N} C_\varphi(f) = C_\varphi E_I^{V_M}(f) \quad \forall f \in H_0^1(M)$$

In other words, V_M can be obtained by considering an extrinsic vector field, whose operator representation has the same effect on functions when composed with the functional

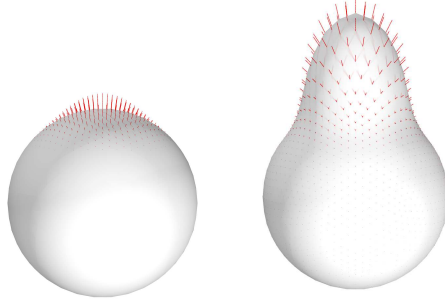


Figure 8.4 – Example of deformation fields that commute with the underlying diffeomorphism. Note that both the direction and the magnitude of the vector field have to adapt to the underlying geometry to produce the same metric distortion.

map C_φ as $E_I^{U_N}$. This property allows us to transfer deformation fields without requiring point-to-point correspondences between shapes, by simply considering the commutativity of the operators C_φ and E_I . We illustrate it in Figure 8.4 and use it in Section 8.7.1 for deformation transfer and deformation symmetrization on meshes with different connectivity with only a functional map known between them. Furthermore, this approach is applicable to design deformations jointly on two shapes, such that they are consistent with the functional map C_φ and even as a regularizer in map computation.

8.5 Discrete Setting

In this section we provide the discretization of the concepts introduced above. For this, we first define a discretization of the unified shape difference by considering Definition 8.4.1. We then obtain a discrete version of the infinitesimal shape difference, by considering a particular discretization of the Levi-Civita connection and the Lie derivative of the metric on the triangle mesh. Finally, we demonstrate that the property stated in Proposition 8.4.6 which links the Lie derivative of the metric and the infinitesimal shape difference is satisfied *exactly* in the discrete case.

Throughout this section, we assume that we are given a manifold triangle mesh. We denote by $(\mathcal{X}, \mathcal{E}, \mathcal{F})$ respectively the set of vertices, edges and faces. To compute the shape differences we start from the discretization of the inner product $\langle \cdot, \cdot \rangle_{H_0^1}$ using standard first order finite elements. We will denote by L the classical cotangent Laplacian matrix, W the inner product of H_0^1 and A the lumped mass matrix such that $L = A^{-1}W$. As before μ is a measure and $\mu(T)$ denotes the area of triangle T .

8.5.1 Discrete unified shape difference

The discretization of the unified shape difference is straightforward when N and M are triangle meshes and share the same connectivity. In Definition 8.4.1 given above, the gradients and the point-wise scalar products are taken on N while the measure $d\mu_M$

comes from M . Therefore the right hand side can be discretized by a modified cotangent weight formula:

$$W_M D_I = W_N^M, \text{ where} \\ (W_N^M)_{i,j} = \frac{1}{2} \left(\frac{\mu_M(T_\alpha)}{\mu_N(T_\alpha)} \cot \alpha_{ij}^N + \frac{\mu_M(T_\beta)}{\mu_N(T_\beta)} \cot \beta_{ij}^N \right). \quad (8.8)$$

Here T_α, T_β are the two triangles adjacent to edge i, j , which is opposite to angles α and β , while μ_M and μ_N are the triangle areas on shapes M and N respectively. Note that W_N^M differs from the standard cotangent weight matrix W_N only by the ratio of weights per triangle. Moreover, one can remark that if the transformation is area preserving for all triangles then D_I reduces to the conformal shape difference defined in [106] (Option 1 in Section 5).

From the expression above it follows that $D_I = W_M^{-1} W_N^M$. For comparison the conformal shape difference has a close formulation $D_C = W_M^{-1} W_N$.

Remarkably, Theorem 4.2.1 admits a discrete parallel as the operator D_I entirely encodes the discrete metric.

Theorem 8.5.1 *The modified cotangent weight matrix $W_N^M(\ell_N)$ uniquely determines the edge length ℓ_N on N . In particular the following equivalence holds true: $D_I = \text{Id}$ if and only if $\ell_M = \ell_N$.*

Proof In [137] the authors prove that W uniquely determines the edge length up to global scaling. Our proof simplifies and adapts their arguments to our problems.

We will first build a convex energy whose gradient corresponds to the modified cotangent weights then show that this energy have a unique minimizer.

Let \mathcal{C} be the set of squared edge length respecting the triangle inequality, namely the set:

$$\mathcal{C} = \left\{ x \in \mathbb{R}^{|\mathcal{E}|} : x_{ij} \geq 0, \forall (i, j) \in \mathcal{E} \quad ; \quad \sqrt{x_{ij}} \leq \sqrt{x_{jk}} + \sqrt{x_{ki}}, \forall (i, j, k) \in \mathcal{F} \right\}.$$

As shown by Proposition 7.7.1, \mathcal{C} is convex.

Thanks to the Heron's formula the triangle areas can be expressed as a function of the squared edge lengths. More precisely we have:

$$\mu(T)^2 = \frac{1}{16} \begin{pmatrix} \ell_{ij}^2 \\ \ell_{jk}^2 \\ \ell_{ki}^2 \end{pmatrix}^\top \begin{pmatrix} -1 & 1 & 1 \\ 1 & -1 & 1 \\ 1 & 1 & -1 \end{pmatrix} \begin{pmatrix} \ell_{ij}^2 \\ \ell_{jk}^2 \\ \ell_{ki}^2 \end{pmatrix}, \quad \ell^2 \in \mathcal{C}.$$

Therefore the gradient of $\mu(T)$ with respect to the squared edge lengths is equal to the vector of cotangent:

$$\nabla_{\ell^2} \mu_T(\ell^2) = \frac{1}{4} \begin{pmatrix} \cot(\alpha_{ij}) \\ \cot(\alpha_{jk}) \\ \cot(\alpha_{ki}) \end{pmatrix}, \quad \ell^2 \in \mathcal{C},$$

where α_{ij} denotes the angle opposite to the edge (i, j) . We introduce the energy term $E : \mathcal{C} \subset \mathbb{R}^{|\mathcal{E}|} \rightarrow \mathbb{R}$ defined as

$$E(\ell^2) := -2 \sum_{T \in \mathcal{F}} \log(\mu_T(\ell^2)) \mu_T(\ell_M^2) + \sum_{(i,j) \in \mathcal{E}} \bar{w}_{ij} \ell_{ij}^2, \quad \ell^2 \in \mathcal{C}.$$

By construction the gradient yields $\nabla_{\ell^2} E(\ell^2) = -w^M(\ell^2) + \bar{w}$, where w^M is the vector containing the modified cotangent weight of Equation (8.8) computed with ℓ instead of ℓ_N and \bar{w} is a target vector of weights.

The triangle areas are also accessible through the determinant of the discrete metric, i.e. $\mu(T) = \sqrt{\det(\mathbf{g}_T)}/2$ in Prop 3.2.2. The function $A \mapsto -\log(\det(A))$ is strictly convex, i.e. [18], and $\ell^2 \mapsto \mathbf{g}_T$ is an injective linear map so E is strictly convex on its domain of definition. Thus the optimization problem:

$$\min_{\ell^2 \in \mathbb{R}^{|\mathcal{E}|}} E(\ell^2) \quad \text{s.t.} \quad \ell^2 \in \mathcal{C},$$

has, when it exists, a unique solution characterized by $w^M(\ell^2) = \bar{w}$. As claimed, there is a one-to-one correspondence between \mathcal{C} and $w^M(\mathcal{C})$.

Suppose $D_I = \text{Id}$ it implies that the equality $w^M(\ell_N^2) = w^M(\ell_M^2)$ holds true, thus the only solution is $\ell_N = \ell_M$. \square

Theorem 8.5.1 suggests that the unified shape difference can be used for recovering intrinsic structure as developed in Chapter 7. However, it will most likely lead to highly non-convex optimization problems as we perform a projection of the weights in the least-squares sense.

Expression in a basis Similarly to the construction given in [106] we can also express the unified shape difference when the basis on the source shape M is given by the eigenfunctions of the Laplace-Beltrami operator Φ_M . In that case, using a diagonal matrix Λ_M of eigenvalues, the expression for D_I becomes:

$$D_I = \Lambda_M^{-1} \Phi_M^T W_N^M \Phi_M.$$

This expression has the advantage of avoiding the inverse of a large sparse matrix, and can be used to analyze deformation of a shape with fixed connectivity in multi-scale basis, which can make the computations resilient to local perturbations (see Option 3 in Section 5 of [106]).

Approximation with a functional map Note that both expressions above assume that the source and target meshes share the same connectivity. When the meshes have different connectivity this discretization requires a map between triangles making it challenging to use in practice. To overcome this problem we approximate this discrete formulation by transferring the weights on triangles to lumped weights on vertices. The approximation then reduces to the usual discrete quantities:

$$(\tilde{W}_N^M)_{ij} \approx \frac{\sum_{t \sim i} \mu_M(T_t)}{\sum_{t \sim i} \mu_N(T_t)} \frac{1}{2} \sum_{j \sim i} (\cot \alpha_{ij}^N + \cot \beta_{ij}^N).$$

We recognize here the cotangent weight Laplacian L_N with lumped area weights, namely $A_M L_N$. In the case of meshes with different connectivity, this remark suggests the following approximation of the isometric shape difference, valid only in a discrete sense, for an arbitrary linear functional map C between M and N :

$$f^\top A_M L_M D_I g \approx f^\top A_M C^{-1} L_N C g.$$

In the reduced basis of the Laplacian eigenvectors, the approximation of the shape difference becomes $D_I \approx \Lambda_M^{-1} C^{-1} \Lambda_N C$, which preserves the principal property of the operator: D_I is identity if and only if the deformation is an isometry since the Laplacian on N has to be equal to the Laplacian on M . We used this discretization in Figure 8.2 and observed that the two expressions given above typically produce similar results.

8.5.2 Infinitesimal shape difference

Throughout this section we will consider the deformation field, which we also call an extrinsic vector field, to be given as a three-dimensional vector per vertex. As we mentioned above, there are two possible ways of discretizing the infinitesimal shape difference operator. First, one can use Definition 8.4.1 and take the derivative at time zero of a discrete shape difference operator. The second possibility is to start from Proposition 8.4.6 and find the discrete equivalent of the Lie derivative of the metric. Remarkably, those two paths lead to the same discretization in our case.

Discrete connection

To build the discrete operator E_I^V we need a consistent discretization of the Levi-Civita connection. While several discrete connections have been proposed (e.g. [7, 72]), because of the special nature of our problem, we choose to build our own. This is because, in applications such as parallel transport it is fundamental that the vectors u , v and $\nabla_u v$ are expressed in the same space (at vertex or face or edge) so often an averaging step has to be introduced to transfer, for example, a face-based representation of a vector to an edge based representation. In our setting such a requirement is not needed and it is easier to distinguish tangent vector fields that will be expressed by one vector per face and extrinsic vector fields expressed at vertices. Thus, our goal is to obtain a connection of the ambient space $\bar{\nabla}_u V$ where u is a tangent vector and V is an extrinsic vector field :

$$\begin{aligned} \bar{\nabla} : \mathbb{R}^{3|\mathcal{F}|} \times \mathbb{R}^{3|\mathcal{V}|} &\rightarrow \mathbb{R}^{3|\mathcal{F}|} \\ (u, V) &\mapsto \bar{\nabla}_u V \end{aligned}$$

We build the connection $\bar{\nabla}$ using finite differences as follows. Since extrinsic vector fields are defined at vertices the differences are taken along the edges.

Definition 8.5.2 *In a given triangle $T \in \mathcal{F}$ the ambient covariant derivative along the edge e_{ij} is defined by*

$$\left(\bar{\nabla}_{\frac{e_{ij}}{\|e_{ij}\|}} V \right)_T = \frac{V_i - V_j}{\|e_{ij}\|}.$$

Thus the ambient connection in the directions $E = (e_{ij}, e_{jk})$ can be stored in a matrix

$$(\bar{\nabla}_E V)_T = \begin{pmatrix} V_i - V_j & V_j - V_k \end{pmatrix}.$$

Then, given any tangent vector $x = E\alpha$, the covariant derivative in its direction can be computed as $\bar{\nabla}_x V = (\bar{\nabla}_E V)\alpha$.

Given the expression above, the discrete Lie derivative of the metric inside triangle T follows immediately, using Eq. (8.3). Namely for any pair of tangent vectors $x = E\alpha, y = E\beta$ in the triangle T , we have:

$$\mathcal{L}_V \mathbf{g}(x, y)_T = \langle x, (\bar{\nabla}_E V)\beta \rangle + \langle (\bar{\nabla}_E V)\alpha, y \rangle. \quad (8.9)$$

After integration we obtain the discrete infinitesimal shape difference:

$$f^\top W_M E_I^V g = - \sum_{T \in \mathcal{F}} \mathcal{L}_V \mathbf{g}(\nabla f, \nabla g)_T \mu(T).$$

Shape difference derivative

The above discrete formulation can also be considered from another point of view. Recall that we first introduced the infinitesimal shape difference as a derivative of a one-parameter family of shape differences. Interestingly, it is possible to reproduce the equivalence shown in Proposition 8.4.6 in the discrete setting using our discretization of the unified shape difference operator.

Suppose that each vertex p_i of the mesh is displaced by the vector V_i by $p_i^t = p_i + tV_i$. This produces a family of triangle meshes $(\mathcal{X}^t, \mathcal{E}, \mathcal{F})$ with identical connectivity. Using this setup properties similar to the continuous case can be derived.

Proposition 8.5.3 *Given a one parameter family of meshes, the first-order change in the metric tensor $\mathbf{g}_T = E^\top E$ and in the area at a triangle $T \in \mathcal{F}$, is given as:*

$$\begin{aligned} \left. \frac{\partial_t \mathbf{g}_T}{\partial t} \right|_{t=0} &= E^\top (\nabla_E V)_T + (\nabla_E V)_T^\top E, \\ \left. \frac{\partial_t \mu(T)}{\partial t} \right|_{t=0} &= \text{Tr} \left(\mathbf{g}_T^{-1} E^\top (\nabla_E V) \right) \mu(T). \end{aligned}$$

Proof First let's remark that the derivative of the discrete metric can be expressed with respect to discrete connection:

$$\left. \frac{\partial \mathbf{g}_T^t}{\partial t} \right|_{t=0} = E^\top (\nabla_E V)_T + (\nabla_E V)_T^\top E.$$

Since the metric is linked to the triangle area by $\mu_t(T) = \frac{1}{2} \sqrt{\det(\mathbf{g}_T^t)}$ the statement obtained by a direct computation of the derivative. \square

In the continuous case the divergence was defined using orthonormal tangent vectors. The formula above is the analog for a pair non orthonormal vectors, therefore we will denote $\text{div}(V)_T = \text{Tr}(\mathbf{g}_T^{-1} E^\top (\nabla_E V))$.

Taking the derivative of the discrete unified shape difference in Eq. 8.8 might be challenging. However, using the formulation of Eq. (3.7) in Chapter 3 leads to an equivalent formulation of Eq. (8.8) is:

$$f^\top W D_I^t g := \sum_{T \in \mathcal{F}} \langle \nabla^t f, \nabla^t g \rangle_T^t \mu(T). \quad (8.10)$$

By taking the derivative of his expression at time $t = 0$, we obtain an alternative discretization of the infinitesimal shape difference E_I .

Proposition 8.5.4 *The discrete infinitesimal shape difference reads $E_I^V(u) = W_M^{-1} H$, where H is a Laplacian matrix whose weights depend on the extrinsic vector field:*

$$(H)_{ij} = \frac{1}{2} \sum_{j \sim i} (c(T_{\alpha_{ij}}) + c(T_{\beta_{ij}})),$$

$$c(T) = (\langle e_{jk}, V_j - V_i \rangle + \langle e_{ij}, V_j - V_k \rangle) \frac{1}{4\mu(T)} - \text{div}(V)_T \frac{\langle e_{jk}, e_{ki} \rangle}{\mu(T)}.$$

Proof Using the FEM gradient, e.g. Equation (3.6), to discretize the unified shape difference written in Eq. 8.10 leads to:

$$f^\top W D_I^t g = \sum_{T \in \mathcal{F}} \frac{1}{4} \begin{pmatrix} f_j - f_k \\ f_j - f_i \end{pmatrix}^\top \frac{\mathbf{g}_T^t}{\mu_t(T)^2} \begin{pmatrix} g_j - g_k \\ g_j - g_i \end{pmatrix} \mu(T). \quad (8.11)$$

We can now compute the derivative with respect to time by using Proposition 8.5.3:

$$f^\top W E_I^V g = \sum_{T \in \mathcal{F}} \frac{1}{4\mu(T)} \begin{pmatrix} f_j - f_k \\ f_j - f_i \end{pmatrix}^\top \mathbf{L}_T \begin{pmatrix} g_j - g_k \\ g_j - g_i \end{pmatrix},$$

$$\mathbf{L}_T = E^\top (\nabla_E V)_T + (\nabla_E V)_T^\top E - 2\text{Tr}(\mathbf{g}_T^{-1} E^\top (\nabla_E V)) \mathbf{g}_T.$$

The matrices \mathbf{L}_T can be written in a form similar to the discrete metric (see Prop. 3.2.2):

$$\mathbf{L}_T = \frac{1}{2} \begin{pmatrix} 2a_{ij} & a_{ki} - a_{jk} - a_{ij} \\ a_{ki} - a_{jk} - a_{ij} & 2a_{jk} \end{pmatrix},$$

(8.12)

where $a_{ij} = \langle e_{ij}, V_i - V_j \rangle - 2\text{div}(V)_T l_{ij}^2$,

leading to the point-wise formulation:

$$(W_M E_I^V)_{ij} = \frac{1}{2} \sum_{j \sim i} (c(T_{\alpha_{ij}}) + c(T_{\beta_{ij}})),$$

$$c(T) = \frac{-a_{ki} + a_{jk} + a_{ij}}{4\mu(T)}.$$

□

Remarkably these two discretizations are strictly identical.

Proposition 8.5.5 *The discretization of E_I based on the discrete Levi-Civita connection is equivalent to the one obtained by differentiating the unified shape difference operator.*

Proof In Equation (8.9) the tangent vectors in a given triangle have to be expressed in the basis form by two edges of the triangle. Following the discussion in Section 3.2.3, the FEM gradient at a face T can be written in two equivalent ways:

$$\nabla f_T = \frac{1}{2\mu(T)} \mathcal{R}^{90^\circ} E \begin{pmatrix} f_j - f_k \\ f_j - f_i \end{pmatrix} = E \mathbf{g}_T^{-1} \begin{pmatrix} f_j - f_i \\ f_k - f_j \end{pmatrix}.$$

Therefore the discrete strain tensor at triangle T follows immediately:

$$\mathcal{L}_V \mathbf{g}(\nabla f, \nabla g) = \begin{pmatrix} f_i - f_j \\ f_j - f_k \end{pmatrix}^\top \mathbf{g}_T^{-1} \left((\bar{\nabla}_E V) + (\bar{\nabla}_E V)^\top \right) \mathbf{g}_T^{-1} \begin{pmatrix} g_i - g_j \\ g_j - g_k \end{pmatrix},$$

From Proposition 8.5.3, we recognize the term the derivative of the inverse metric. Using Equation (3.10), one can further modified the expression to:

$$\mathcal{L}_V \mathbf{g}(\nabla f, \nabla g)_T = -\frac{1}{4} \begin{pmatrix} f_j - f_k \\ f_j - f_i \end{pmatrix}^\top \frac{\partial}{\partial t} \left(\frac{\mathbf{g}_T^t}{\mu_t(T)^2} \right) \Big|_{t=0} \begin{pmatrix} g_j - g_k \\ g_j - g_i \end{pmatrix}. \quad (8.13)$$

The right hand side term appears in the discrete isometric shape difference as written in Equation (8.11). This leads to the equality between the different discretization:

$$\frac{\partial}{\partial t} \left(f^\top W D_{Ig}^t \right) \Big|_{t=0} = - \sum_{T \in \mathcal{F}} \mathcal{L}_V \mathbf{g}(\nabla f, \nabla g)_T \mu(T).$$

□

Interestingly, the discrete metric tensor can be defined for higher dimension simplicial complexes in accordance with the FEM discretization. Therefore Proposition 8.5.5 holds in more general cases.

8.5.3 Properties

Interestingly, many of the properties of the continuous operators are satisfied exactly by their discrete counterparts.

Linearity The discretization $E_I^V(f)$ naturally preserves the linearity with respect to both V and f .

Shape difference decomposition Since the discretization using a discrete connection and through a the time derivative agree, the decomposition described by Eq. (8.6) is also satisfied exactly. With the family of meshes conformal shape difference is:

$$f^\top W D_C^t g := f^\top W_t g = \sum_{T \in \mathcal{F}} \langle \nabla^t f, \nabla^t g \rangle_T^t \mu_t(T),$$

where W_t is the cotangent weight matrix associated to the deformed mesh at time t . As noted above the derivative of $\langle \nabla^t f, \nabla^t g \rangle_T^t$ will lead to E_I^V . The derivative of the area will produce a term close to $\int_M \text{div}(u) \langle \nabla f, \nabla g \rangle d\mu$ thanks to Proposition 8.5.3. Namely:

$$f^\top W E_C^V g = f^\top W E_I^V g + \sum_{T \in \mathcal{F}} \text{div}(V)_T \langle \nabla f, \nabla g \rangle_T \mu(T).$$

Thus, the decomposition of E_I^V , representing isometric distortion, into area and conformal parts given in Eq. (8.6) in the continuous case holds exactly in the discrete case as well.

Vector Fields representation In the continuous setting the kernel of $V \mapsto E_I^V$ is the set of infinitesimal isometries. However, to the best of our knowledge, there is no characterization of how often this set is reduced to rigid motion. In the particular setting of our discretization some standard results can be applied, however.

Proposition 8.5.6 *For almost all triangle meshes M without boundary, the operator E_I^V uniquely defines the extrinsic vector field V up to rigid motion.*

Proof The proof is organized as follows: we show that we can recover the matrices \mathbf{L}_T from the infinitesimal shape difference then we use a standard results in combinatorix to prove that $\mathbf{L}_T = 0$ if and only if the extrinsic vector field is a rigid motion.

- *Kernel of $\mathbf{L}_T \mapsto E_I^V$.* Information about the extrinsic vector field are solely contained by the matrices \mathbf{L}_T . Those matrices agree on edges so they can be reduced to the vector $a \in \mathbb{R}^{|\mathcal{E}|}$ as defined in Eq. (8.12). The mapping $a \mapsto E_I^V$ is linear and almost always invertible as proven by Proposition 7.4.2.
- *Rigidity Theorem.* As shown previously the kernel of $a \mapsto E_I^V$ is almost always reduced to the zero element. Going back to the matrices \mathbf{L}_T , the extrinsic vector field in the kernel should satisfy:

$$\mathbf{g}_T^{-1} \mathbf{L}_T = \mathbf{g}_T^{-1} E^\top (\nabla_E V)_T + \mathbf{g}_T^{-1} (\nabla_E V)_T^\top E - 2 \text{Tr} \left(\mathbf{g}_T^{-1} E^\top (\nabla_E V) \right) \text{Id} = 0.$$

Taking the trace in both sides implies that $\text{div}(V)_T = \text{Tr}(\mathbf{g}_T^{-1} \partial_t \mathbf{g}_T^t|_0)$ should vanish so it is equivalent to have all matrices $E^\top (\nabla_E V)_T + (\nabla_E V)_T^\top E$ equal zero. It follows that the extrinsic vector field satisfies $\langle e_{ij}, V_i - V_j \rangle = 0$ at all edges. It has been proved in [44] that almost all simply connected closed surfaces only admit rigid deformation as solution of this equation.

□

Therefore, unlike continuous setting, we can guarantee that E_I^V is a coordinate-free representation of almost any extrinsic vector field V .

8.6 Representation in basis

8.6.1 Basis for the Function Space

Our infinitesimal shape difference is a linear operator acting on smooth functions defined on the surface. In practice, it is handy to use a basis for the function space, so any function can be represented as a linear combination of some basis functions ϕ_i . In this basis, the operator E_I^V can be seen as a (possibly infinite) matrix. For a function $f = \sum_i \alpha_i \phi_i$ we use the linearity property $E_I^V(f) = E_I^V(\sum_i \alpha_i \phi_i) = \sum_i \alpha_i E_I^V(\phi_i)$. The choice of basis depends on the application. Since we want to represent smooth deformations of a surface, we take a subset of the smoothest functions given by the first eigenfunctions of the Laplace-Beltrami operator.

8.6.2 Vector field basis

As mentioned in Section 8.4.3 the infinitesimal shape difference operator E_I^V is linear with respect to V . Therefore, if the vector field is given in some basis $V = \sum_i \beta_i X_i$ then the operator reads $E_I^V = \sum_i \beta_i E_I^{X_i}$. This means that when designing infinitesimal deformation V we can consider an objective as a function of the coefficients β , and moreover if the function is quadratic in β then recovering V can be done by solving a single linear system of equations. Note that this allows us to combine constraints both on the vector field itself (e.g., handle constraints) and on its operator representation, such as enforcing commutativity with other operators, while remaining easy to optimize for.

Choice of basis In practice we use two alternatives for the basis of extrinsic vector fields. The simplest option is to take the eigenfunctions of the Laplace-Beltrami operator as the basis for each component of the deformation field. While simple, this basis might not preserve rotation invariance. Thus, alternatively we construct a basis via modal analysis of a deformation energy. In particular we consider an energy of the form $V \mapsto \int_M \|\bar{\nabla} V\|^2 d\mu$. This corresponds to the Dirichlet energy on a particular discretization of the Bochner Laplacian of extrinsic vector fields. To obtain the basis we take the eigenvectors of the Hessian of the energy function, which correspond to smooth deformation fields.

8.7 Experiments

In this section we apply our constructions to various tasks in deformation design and analysis. As our framework relies on manipulating and inverting moderately-sized matrices, all of the applications are very efficient and even when combining multiple objectives our method remains near interactive.

8.7.1 Deformation transfer

Pose transfer Given frames from an animation sequence and a functional map we can use our method to transfer the deformation to an arbitrary mesh. At each step we take a given deformation field U and find the transferred deformation V by minimizing

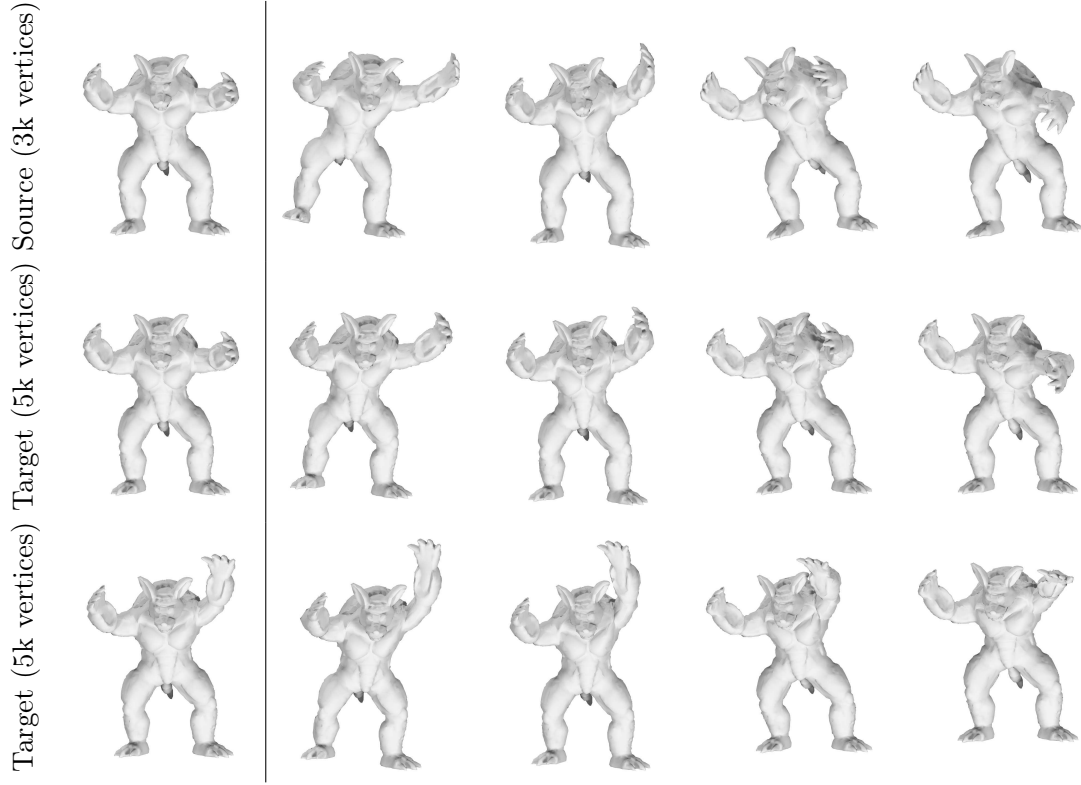


Figure 8.5 – Deformation transfer from a given animation sequence (top row) to: another mesh with different connectivity (middle row), and in a different pose (bottom row) in an orientation-invariant way without using point or triangle correspondences.

$V \mapsto \|E_I^U C_\varphi - C_\varphi E_I^V\|^2$ for some given functional map C_φ , which is represented in a reduced basis. We parameterize the space of deformations by using the 70 principal eigenfunctions of the Bochner-Laplacian as explained in Section 8.6.2. Figure 8.5 shows an example of transfer from an animation consisting of 20 frames of a waving Armadillo to Armadillos with different connectivity and in different starting position. The transferred animation induces a similar metric distortion across different shapes resulting in a consistent deformation. In Figure 8.6 a similar experiment is presented for a collection of faces. The facial expressions of the first row are transferred to a another face on the second row. The results are compared to the corresponding faces present in the collection. Unlike [117] the transfer does not require a triangle-to-triangle map only an approximate functional map. Moreover the meshes do not need to be oriented or positioned consistently since only the metric changes are transferred as illustrated in the third row of Figure 8.5.

Style transfer In a similar experiment we use our approach to transfer style across the poses of different face shapes, shown in Figure 8.7, as we all shapes in the FAUST dataset [12], shown in Figure 8.8. Here we first consider the deformation field U given by the point displacements across two different shapes in approximately the same pose.

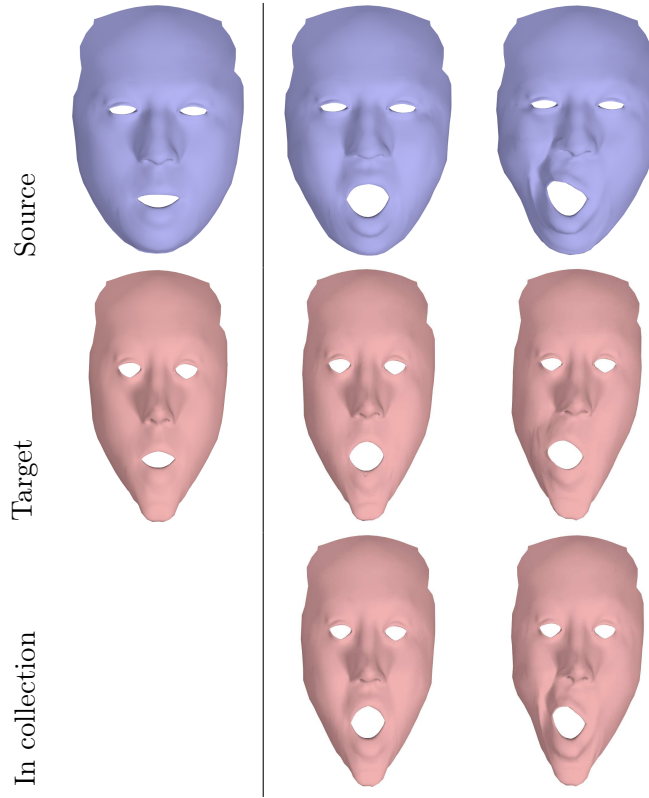


Figure 8.6 – Transfer of expressions in a faces collection. The expressions of the reference face (top row) are transferred to a different face (middle row) and compared to the corresponding faces of the collection (third row).

We then use our framework to transfer U to another shape in a different pose and with different mesh structure. In Figure 8.8 our method consistently preserves the global structure, although some high frequency details of the deformation are lost because of the basis reduction. The deformation transfer for faces in Figure 8.7 is successful because the deformation is very smooth and therefore well represented in the basis.

Symmetry transfer One interesting feature of the functional representation is that it is “shape aware.” For example in Figure 8.9 we transfer the shrinking of the right leg to the left leg by looking for the operator which commutes with the operator representation of the symmetry map. Since both legs are in different positions this transfer is not easy by a simple point-to-point transfer of the vector field or even by transferring it using local coordinates. As shown by the vector field representation in Figure 8.9 bottom row, the transferred vector field adapts to the geometry.

Deformations design Since our operator is linear with respect to the deformation field one can easily enforce additional deformation constraints to the extrinsic vector

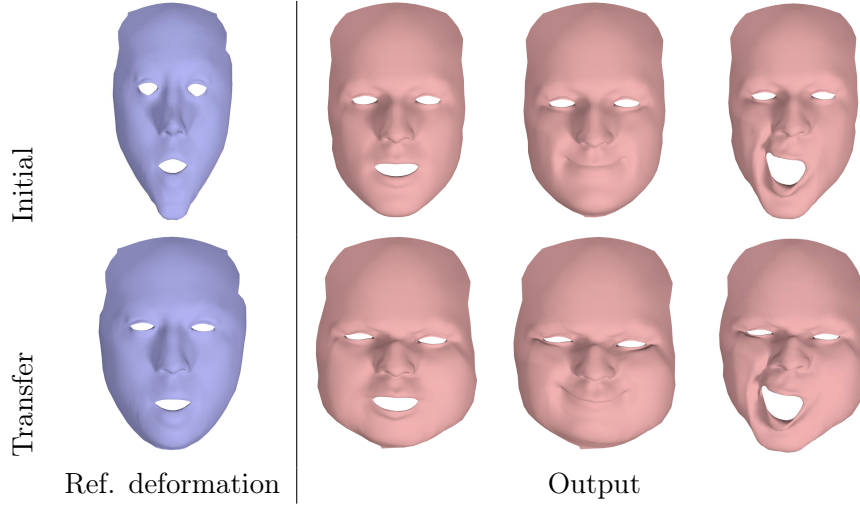


Figure 8.7 – The deformation between the two shapes in the *first column* is transferred to a face in various expressions (red faces top row). The obtained deformation is consistent across the poses (red faces bottom row) even if the meshes have different connectivity and only an approximate functional map is used for the transfer.

field. In Figure 8.10 we require that at a point p the deformation field matches a given vector u : $V(p) = u$ in addition to other global constraints. We find the most isometric deformation by minimizing $V \mapsto \|E_I^V\|_F^2$. Given a self-map S , we design a symmetric vector field imposing a constraint of the form $\|E_I^V C_S - C_S E_I^V\| = 0$. We observe a similar deformation on each tentacles. We can also impose an anti-symmetry constraint with $\|E_I^V C_S + C_S E_I^V\| = 0$. Alternatively, regularization technique for deformation can be tested by imposing the commutativity with the Laplace-Beltrami operator, interestingly the deformation tends to spread to the entire shape.

Figure 8.11 presents an example of joint deformation design. Namely, we impose a set of directional constraints on two different shapes and want a deformation on each shapes that is aware in some way of the deformation of the other shape. To do so, we find U and V minimizing $(U, V) \mapsto \|E_I^{U^M} C_\varphi - C_\varphi E_I^{V^N}\|^2$ and respecting the local constraint on their respective shape. As a result, the constraints on one shape are transferred to the other. Moreover the area that could lead to contradictory deformation remains still.

8.7.2 Functional map inference

The infinitesimal shape difference is not only useful for design and analysis of deformations, it can also be used as a regularization in a shape matching problem. Below we show how this representation can be used to add extrinsic information to the computation of functional maps [88]. In this framework it is challenging to use extrinsic information in another way than just using extrinsic based descriptors and while remaining rotationally invariant.

In this experiment we propose to solve the following problem: given two shapes and a

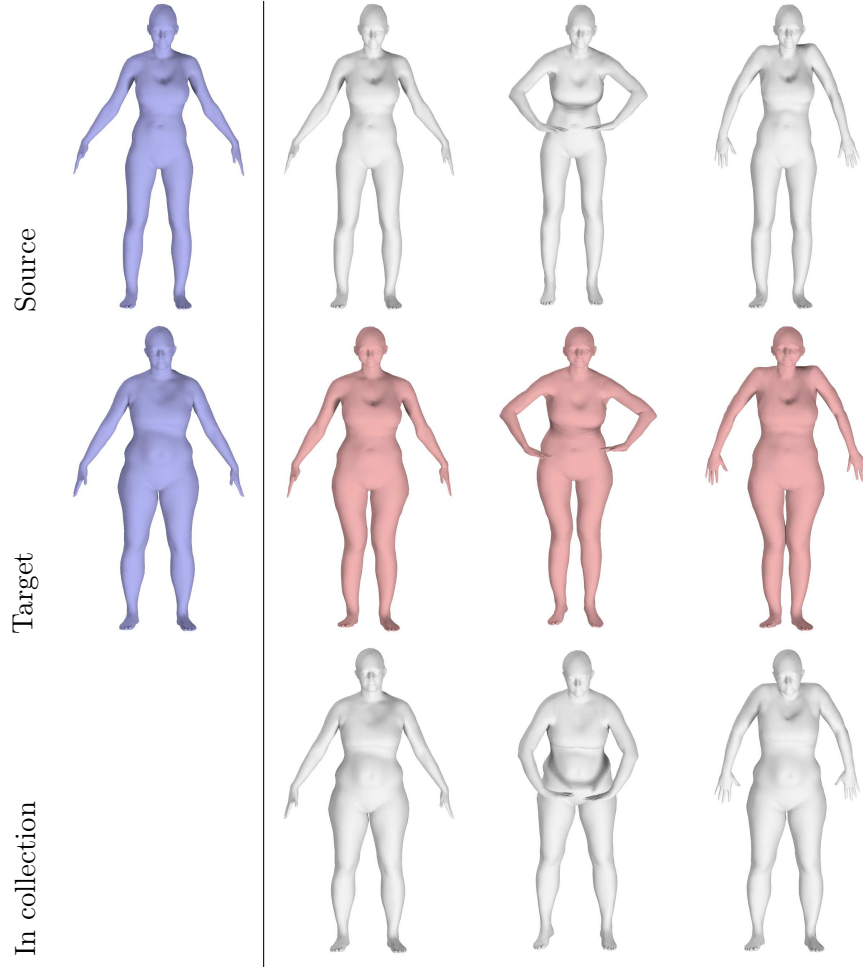


Figure 8.8 – The deformation field defined by the blue shapes (first column) is transferred to the same shape in different poses (top white shapes). While the style is consistent across the poses (red shapes) some details of the deformation are lost due to the basis representation. The style transfer are compared to the corresponding shape in the collection (bottom white shapes).

sparse set of correspondences recover a dense map. The shapes come from the Faust dataset [12] and we are given five corresponding landmarks at the hands, feet and head. The baseline method following the logic of the original paper is to represent the landmark points as delta functions δ_M and δ_N and look for the most isometric functional map $C : L^2(M) \rightarrow L^2(N)$, by enforcing commutativity with the Laplace-Beltrami operator. Thus, the straightforward approach would be to solve the optimization problem:

$$\min_C \|C\Delta_M - \Delta_N C\|_F^2 \quad \text{s.t.} \quad C\delta_M = \delta_N.$$

In order to add extrinsic information to this problem, we would like C to commute with the operator E_I^n representing the information about the second fundamental form. Note

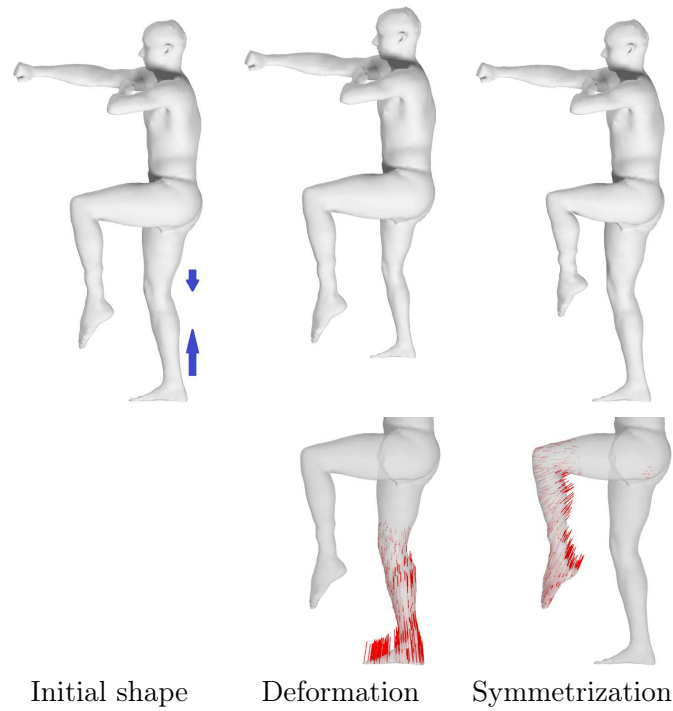


Figure 8.9 – An initial deformation (first two columns), corresponding to the shrinking of the right leg of a human model, is transferred to the left leg by imposing the commutativity between the infinitesimal shape difference and the symmetry map. Both legs are in different position so the transfer has to adapt to the geometry.

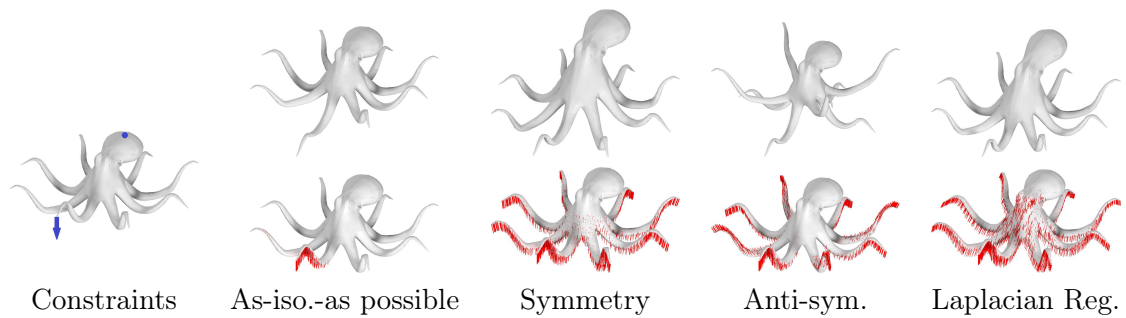


Figure 8.10 – We design deformations respecting the directional constraints shown on the far left and minimizing various criteria (from right to left): the infinitesimal shape difference leading to the most isometric vector field, the commutativity with the a self-map, the anti-commutativity with the same self-map and the commutativity with the Laplace-Beltrami operator.

that if a diffeomorphism commutes with the Laplace-Beltrami operator and E_I^n then the

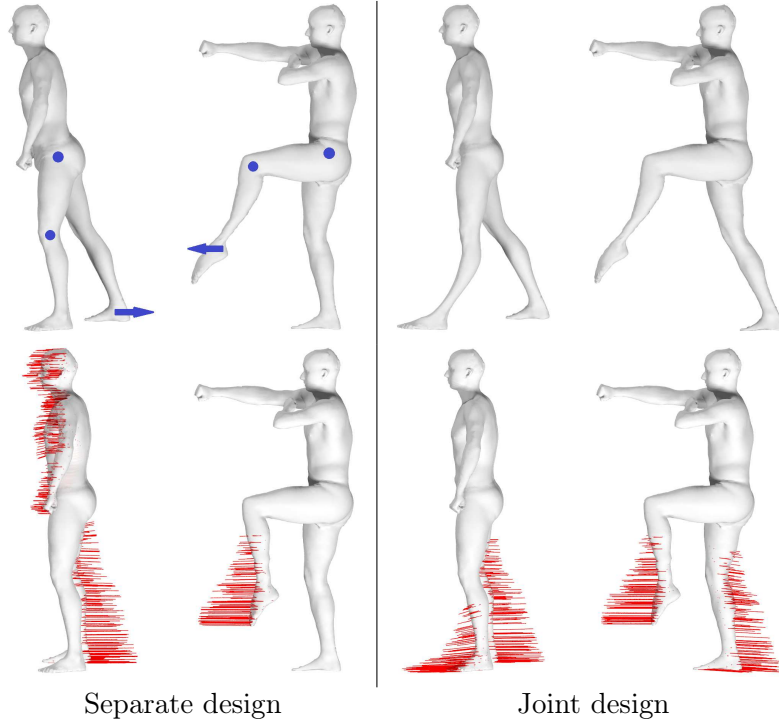


Figure 8.11 – Left: The deformations are designed separately by minimizing the norm of E_I^V . Right: joint deformation design by adding the commutativity with the mapping to the optimization. Note that the constraints on one shape tend to be transferred to the other.

shapes admit the same embedding. Our new optimization problem thus reads:

$$\begin{aligned} \min_C \quad & \|C\Delta_M - \Delta_NC\|_F^2 + \|CE_I^{nM} - E_I^{nN}C\|_F^2 \\ \text{s.t.} \quad & C\delta_M = \delta_N. \end{aligned}$$

Once the functional maps are obtained they are converted to a point-to-point map using the knn-algorithm as described in [88]. The results are shown for two shape matching problems a nearly-isometric and a non-isometric. Figure 8.12 shows the percentage of correspondence within a given geodesic distance, and it can be seen clearly that the additional extrinsic information provides valuable information since the correspondence exhibits less error. Figure 8.13 provides a visualization of the point-to-point correspondences by transferring the coordinates functions encoded as RGB channels. Interestingly our new constraint make the map smoother in both cases.

We preform a similar experiment in Figures 8.14 and 8.15, where instead of using the normal field as regularization we deform by hand the shapes to be matched with an equivalent deformation. This extrinsic vector field is used to regularized the shape matching problem.

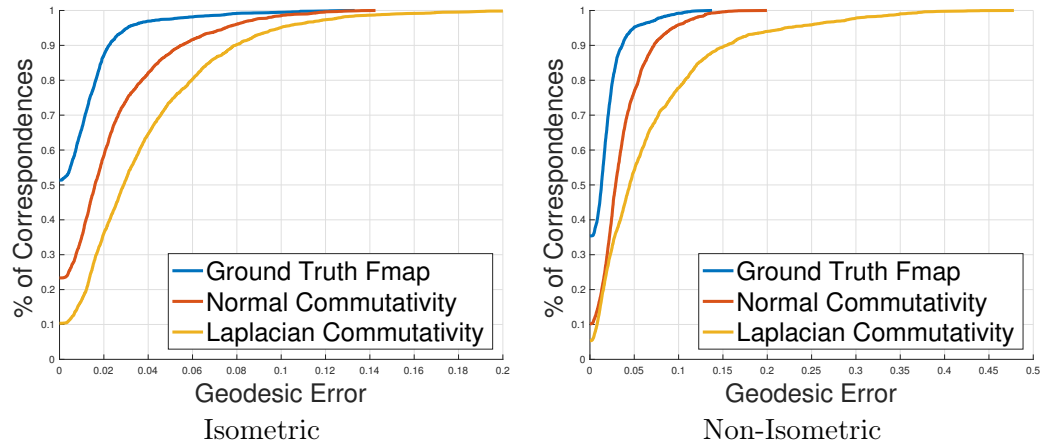


Figure 8.12 – Percentage of correspondences within a geodesic ball for a nearly-isometric and a non-isometric shape matching problems. Note that adding the commutativity constraint with E_I^n (red lines) improves the correspondences.

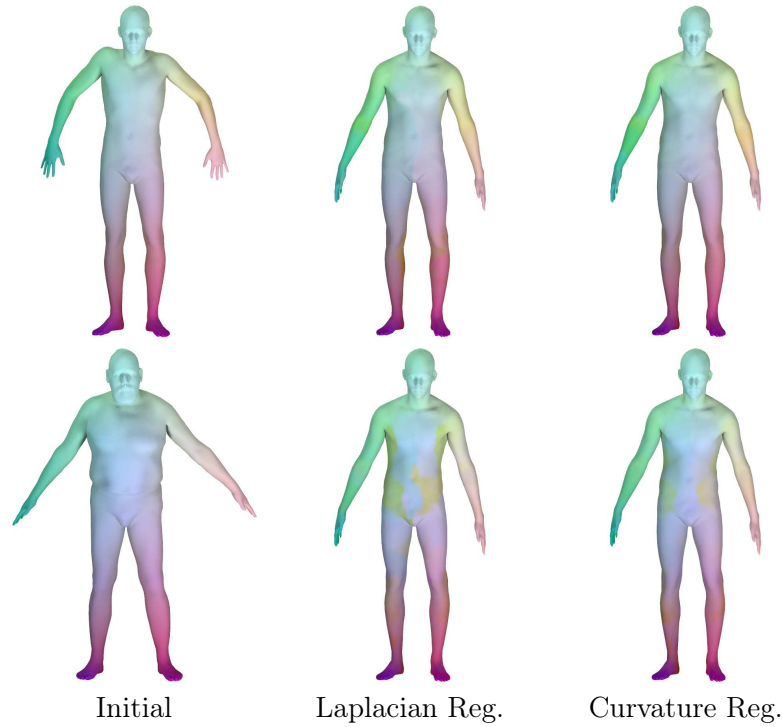


Figure 8.13 – Representation of the point-to-point map evaluated in Figure 8.12. The RGB channel (left column) represents the xyz -coordinates, which are transferred using the recovered point-to-point map. In both cases, the maps obtained using commutativity with E_I^n (right column) tend to be smoother than the naive approach (middle column).

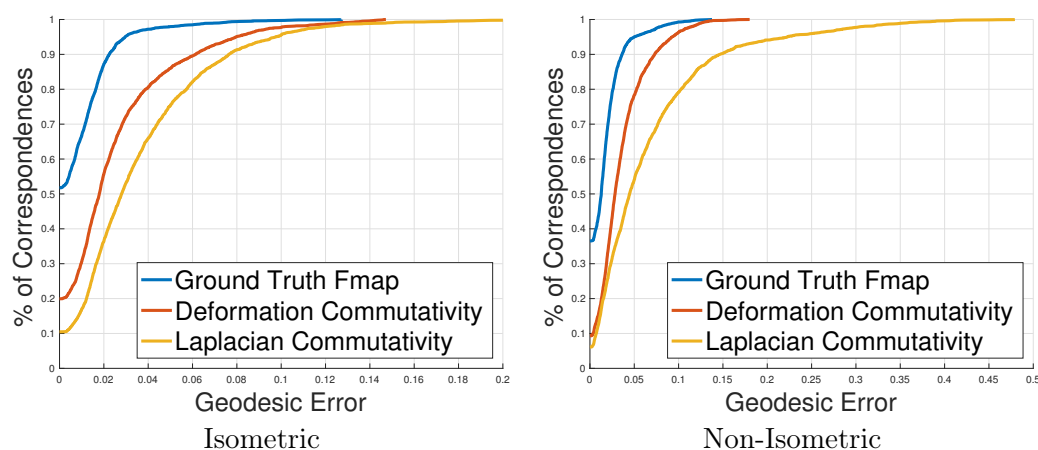


Figure 8.14 – Percentage of correspondences within a geodesic ball for an nearly-isometric and a non-isometric shape matching problems. Note that adding the commutativity constraint with a deformation field (red lines) improves the correspondences.

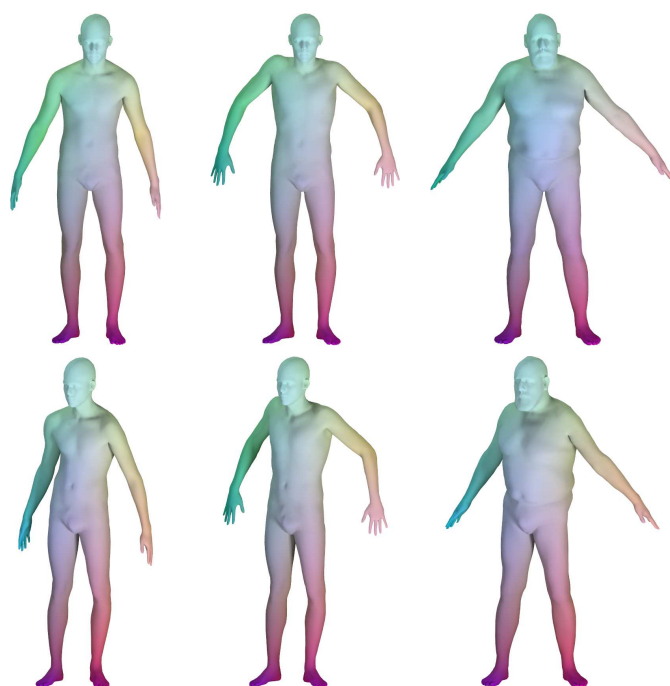


Figure 8.15 – Representation of the point-to-point map evaluated in Figure 8.14. The RGB channel (left column) represents the xyz -coordinates, they are transferred using the point-to-point map. In both case, the maps obtained using commutativity constraint (right column) tend to be smoother than the naive approach (middle column).

8.8 Conclusion and Future Work

In this chapter we presented a method for representing extrinsic vector fields as linear operators acting on functions on the shapes, by considering the metric distortion induced by the deformation. For this we adapted the previously proposed shape difference operators by first introducing a unified shape difference, which fully encodes isometric distortion. We then defined infinitesimal shape difference operators, which provide a compact functional representation of deformation fields. We showed how this representation can be used to analyze, transfer, and design deformations and to introduce extrinsic information into the computation of functional correspondences.

In the future, we are planning to use the newly introduced functional representation for shape animation, by exploiting its simplicity and ability to relate deformations on multiple shapes even without pointwise correspondences.

Conclusion

In this thesis we have studied functional characterization of diffeomorphisms and embedded surfaces. The field of operator representation is relatively new in computer graphics and many questions remain open. Here we summarize some possible improvements and future works.

Shape to Deformation

This thesis provides a whole pipeline for shape matching. The first stage is the computation of an unknown functional map between shapes. The descriptor selection and stable subspace learning prove to be resilient to noise and robust to non-isometric matching. The learning procedure is guided by given correspondences in a collection. However, this type of data might be difficult to obtain, whereas non-supervised optimization seems an achievable goal with functional maps. For example spectral descriptors are often related to geodesic distances. So tracking inconsistencies among descriptors might be a way of learning common intrinsic changes and moving parts in models.

The second stage, which consists of converting the functional map into a continuous point-to-point map attempts to improve currently available methods by adding continuity. The method relies on the functional representation of vector field flow. However some diffeomorphisms are badly or not at all represented due to the use of a reduced function basis. Therefore, only a subset of vector fields generates representable maps. Currently we restrict vector fields to smooth bases but a more careful analysis of representability is needed to adapt the size of the vector field space to the size of the function space. Besides the hierarchical nature of eigenfunctions may be exploited into a “spectral multi-grid” algorithm.

A more long term challenge is to rethink this pipeline to solve non-isometric matching. The conversion to a point-to-point map do not make any assumptions on the nature of the deformation as long as a valid functional map is given as input. The computation of the functional map however depends on function correspondences which are most of the time unreliable for large deformations. Moreover, the function basis, based on the Laplacian eigenfunctions, can be quite different on each mesh, meaning that large functional maps are needed. It seems that improvements can be obtained by considering adaptive basis function and localized descriptors.

Deformation to Shape

A functional characterization of deformable surfaces offers many possible future works in particular it defines an alternative *shape space* (e.g. [61, 50]).

Infinitesimal shape differences constitute a tangent space of a shape difference. This is a first step toward building a complete representation of a manifold of Riemannian metrics. We can then explore the shape space by following geodesics between surfaces or finding barycentric shapes. The most straightforward method to do so is by linear interpolation of the edge length and dihedral angle [134] – discrete analogs of the metric and mean curvature. However, it is challenging to extent this view to other types of data whereas operator based methods rely on well-studied discretizations. To reach this goal, however, we are lacking a notion of metric between infinitesimal shape difference operators. The naive Frobenius norm does not have meaning outside the discrete setting and may lead to ill-conditioned problems. An analysis of the space of shape differences would help defining a suitable metric and geodesic distances as different metric and curvature may lead to different notion of interpolation of meshes.

Defining a functional curvature representation stands as an unsolved problem. The solution provided by Chapter 7 is not entirely satisfactory since it requires to transform a surface to a tetrahedral mesh increasing the size of the problem. Moreover, global constraints linking the shape differences of the bottom layer to the top layer so it represents an embedded surface, are lacking and may not be easy to use in practice. Chapter 8 also suggests an answer but without theoretical proof of complete representation of triangle meshes. Nevertheless, this issue is fundamental in order to perform deformation based on curvature analysis (parametrization is an extreme example) or curvature interpolation. Two properties are desirable: the completeness of information, in the sense that we are able to recover the embedding from the shape differences and the curvature representation, and the insensitivity to noise and tessellation.

Bibliography

- [1] N. AIGERMAN AND Y. LIPMAN, *Orbifold tutte embeddings*, ACM Trans. Graph, 34 (2015), p. 190. (Cited on pages 2 and 8.)
- [2] N. AIGERMAN, R. PORANNE, AND Y. LIPMAN, *Seamless surface mappings*, ACM Transactions on Graphics (TOG), 34 (2015), p. 72. (Cited on pages 2 and 8.)
- [3] M. ALEXA AND M. WARDETZKY, *Discrete laplacians on general polygonal meshes*, in ACM Transactions on Graphics (TOG), vol. 30, ACM, 2011, p. 102. (Cited on pages 1 and 7.)
- [4] D. ANGUELOV, P. SRINIVASAN, D. KOLLER, S. THRUN, J. RODGERS, AND J. DAVIS, *Scape: shape completion and animation of people*, in ACM TOG (Proc. SIGGRAPH), vol. 24, 2005, pp. 408–416. (Cited on pages 51 and 75.)
- [5] M. AUBRY, U. SCHLICKWEI, AND D. CREMERS, *The wave kernel signature: A quantum mechanical approach to shape analysis*, in ICCV Workshops, 2011, pp. 1626–1633. (Cited on pages 1, 7, 21, 23, 52, 58, 76 and 106.)
- [6] O. AZENCOT, M. BEN-CHEN, F. CHAZAL, AND M. OVSJANIKOV, *An operator approach to tangent vector field processing*, in Computer Graphics Forum, vol. 32, 2013, pp. 73–82. (Cited on pages 31, 50, 66, 68, 70, 71, 73 and 114.)
- [7] O. AZENCOT, M. OVSJANIKOV, F. CHAZAL, AND M. BEN-CHEN, *Discrete derivatives of vector fields on surfaces—an operator approach*, ACM Transactions on Graphics (TOG), 34 (2015), p. 29. (Cited on page 130.)
- [8] O. AZENCOT, S. WEISSMANN, M. OVSJANIKOV, M. WARDETZKY, AND M. BEN-CHEN, *Functional fluids on surfaces*, in Computer Graphics Forum, vol. 33, Wiley Online Library, 2014, pp. 237–246. (Cited on page 31.)
- [9] I. BARAN, D. VLASIC, E. GRINSUN, AND J. POPOVIĆ, *Semantic deformation transfer*, in ACM Transactions on Graphics (TOG), vol. 28, ACM, 2009, p. 36. (Cited on page 116.)
- [10] M. F. BEG, M. I. MILLER, A. TROUVÉ, AND L. YOUNES, *Computing large deformation metric mappings via geodesic flows of diffeomorphisms*, International journal of computer vision, 61 (2005), pp. 139–157. (Cited on page 67.)
- [11] M. BEN-CHEN, O. WEBER, AND C. GOTSMAN, *Spatial deformation transfer*, in Proc. SCA, 2009, pp. 67–74. (Cited on page 86.)
- [12] F. BOGO, J. ROMERO, M. LOPER, AND M. J. BLACK, *FAUST: Dataset and evaluation for 3D mesh registration*, in Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), Piscataway, NJ, USA, June 2014, IEEE. (Cited on pages 136 and 139.)

- [13] O. BONNET, *Mémoire sur la theorie des surfaces applicables sur une surface donnée*, Journal de l'École Polytechnique, 25 (1867), pp. 313–151. (Cited on pages 83 and 86.)
- [14] F. L. BOOKSTEIN, *Landmark methods for forms without landmarks: morphometrics of group differences in outline shape*, Medical image analysis, 1 (1997), pp. 225–243. (Cited on pages 50 and 51.)
- [15] D. BOSCAINI, D. EYNARD, D. KOUROUNIS, AND M. M. BRONSTEIN, *Shape-from-operator: Recovering shapes from intrinsic operators*, Computer Graphics Forum, (2015). (Cited on pages 3, 9, 86, 89, 107, 108 and 116.)
- [16] M. BOTSCH, L. KOBELT, M. PAULY, P. ALLIEZ, AND B. LEVY, *Polygon Mesh Processing*, Taylor & Francis, 2010. (Cited on pages 25, 85 and 115.)
- [17] M. BOTSCH AND O. SORKINE, *On linear variational surface deformation methods*, Visualization and Computer Graphics, IEEE Transactions on, 14 (2008), pp. 213–230. (Cited on page 115.)
- [18] S. BOYD AND L. VANDENBERGHE, *Convex Optimization*, Cambridge University Press, 2004. (Cited on pages 101 and 129.)
- [19] S. BRENNER AND R. SCOTT, *The mathematical theory of finite element methods*, vol. 15, Springer, 2007. (Cited on pages 1, 7 and 87.)
- [20] H. BREZIS, *Functional analysis, Sobolev spaces and partial differential equations*, Springer, 2010. (Cited on page 38.)
- [21] A. BRONSTEIN, M. BRONSTEIN, AND R. KIMMEL, *Numerical geometry of non-rigid shapes*, Springer, 2008. (Cited on pages 50, 58 and 75.)
- [22] A. M. BRONSTEIN, M. M. BRONSTEIN, AND R. KIMMEL, *Generalized multidimensional scaling: a framework for isometry-invariant partial surface matching*, PNAS, 103 (2006). (Cited on pages 49, 50, 65 and 67.)
- [23] A. BRUNTON, M. WAND, S. WUHRER, H.-P. SEIDEL, AND T. WEINKAUF, *A low-dimensional representation for robust partial isometric correspondences computation*, Graphical Models, 76 (2014), pp. 70–85. (Cited on pages 65 and 67.)
- [24] M. P. D. CARMO, *Differential Geometry of Curves and Surfaces*, Pearson, 1976. (Cited on page 86.)
- [25] G. CELNIKER AND D. GOSSARD, *Deformable curve and surface finite-elements for free-form shape design*, in ACM SIGGRAPH computer graphics, vol. 25, ACM, 1991, pp. 257–266. (Cited on page 115.)
- [26] X. CHEN, A. SAPAROV, B. PANG, AND T. FUNKHOUSER, *Schelling points on 3d surface meshes*, ACM Trans. Graph. (TOG), 31 (2012), p. 29. (Cited on page 51.)
- [27] P. G. CIARLET, *Theory of shells*, vol. 3, Elsevier, 2000. (Cited on page 125.)

- [28] D. COHEN-STEINER AND J.-M. MORVAN, *Restricted delaunay triangulations and normal cycle*, in Proceedings of the nineteenth annual symposium on Computational geometry, ACM, 2003, pp. 312–321. (Cited on page 86.)
- [29] T. F. COOTES, C. J. TAYLOR, D. H. COOPER, AND J. GRAHAM, *Training models of shape from sets of examples*, in BMVC92, Springer, 1992, pp. 9–18. (Cited on pages 50 and 51.)
- [30] ———, *Active shape models-their training and application*, Computer vision and image understanding, 61 (1995), pp. 38–59. (Cited on pages 50 and 51.)
- [31] E. CORMAN, M. OVSJANIKOV, AND A. CHAMBOLLE, *Supervised descriptor learning for non-rigid shape matching*, in ECCV 2014 Workshops, Part IV, Springer International Publishing, 2014. (Cited on pages 5 and 11.)
- [32] ———, *Continuous matching via vector field flow*, in Proceedings of the Eurographics Symposium on Geometry Processing, vol. 34, Eurographics Association, 2015, pp. 129–139. (Cited on pages 5 and 11.)
- [33] K. CRANE, U. PINKALL, AND P. SCHRÖDER, *Spin transformations of discrete surfaces*, in ACM Transactions on Graphics (TOG), vol. 30, ACM, 2011, p. 104. (Cited on pages 114 and 115.)
- [34] K. CRANE, C. WEISCHEDEL, AND M. WARDETZKY, *Geodesics in Heat: A New Approach to Computing Distance Based on Heat Flow*, ACM Trans. Graph., 32 (2013). (Cited on pages 1, 7, 18 and 21.)
- [35] F. DE GOES, P. MEMARI, P. MULLEN, AND M. DESBRUN, *Weighted triangulations for geometry processing*, ACM Trans. Graph., 33 (2014), pp. 28:1–28:13. (Cited on page 85.)
- [36] M. DO CARMO, *Riemannian Geometry*, Mathematics: Theory & Applications, Birkhäuser Boston, 2013. (Cited on page 122.)
- [37] I. L. DRYDEN AND K. V. MARDIA, *Statistical shape analysis*, vol. 4, John Wiley & Sons New York, 1998. (Cited on pages 50 and 51.)
- [38] N. DYM AND Y. LIPMAN, *Exact recovery with symmetries for Procrustes matching*, arXiv:1606.01548, (2016). (Cited on page 102.)
- [39] M. EIGENSATZ AND M. PAULY, *Positional, metric, and curvature control for constraint-based surface deformation*, in Computer Graphics Forum, vol. 28, Wiley Online Library, 2009, pp. 551–558. (Cited on page 115.)
- [40] M. EIGENSATZ, R. W. SUMNER, AND M. PAULY, *Curvature-domain shape processing*, Computer Graphics Forum, 27 (2008), pp. 241–250. (Cited on page 86.)

- [41] M. FISHER, P. SCHRÖDER, M. DESBRUN, AND H. HOPPE, *Design of tangent vector fields*, in ACM Transactions on Graphics (TOG), vol. 26, ACM, 2007, p. 56. (Cited on page 74.)
- [42] S. FRÖHLICH AND M. BOTSCH, *Example-driven deformations based on discrete shells*, Computer Graphics Forum, 30 (2011), pp. 2246–2257. (Cited on page 86.)
- [43] X. GAO, Y. SU, X. LI, AND D. TAO, *A review of active appearance models*, Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on, 40 (2010), pp. 145–158. (Cited on pages 50 and 51.)
- [44] H. GLUCK, *Almost all simply connected closed surfaces are rigid*, in Geometric topology, Springer, 1975, pp. 225–239. (Cited on pages 67, 92 and 134.)
- [45] M. GRANT AND S. BOYD, *CVX: Matlab software for disciplined convex programming, version 2.1*. <http://cvxr.com/cvx>, Mar. 2014. (Cited on page 108.)
- [46] M. GROSS AND H. PFISTER, *Point-based graphics*, Morgan Kaufmann, 2011. (Cited on pages 1 and 7.)
- [47] I. GUSKOV, W. SWELDENS, AND P. SCHRÖDER, *Multiresolution signal processing for meshes*, in Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '99, 1999, pp. 325–334. (Cited on page 115.)
- [48] S. HAKER, S. ANGENENT, A. TANNENBAUM, R. KIKINIS, G. SAPIRO, AND M. HALLE, *Conformal surface parameterization for texture mapping*, IEEE Transactions on Visualization and Computer Graphics, 6 (2000), pp. 181–189. (Cited on page 67.)
- [49] N. HASLER, C. STOLL, M. SUNKEL, B. ROSENHAHN, AND H.-P. SEIDEL, *A statistical model of human pose and body shape*, in Computer Graphics Forum, vol. 28, Wiley Online Library, 2009, pp. 337–346. (Cited on pages 49, 51 and 65.)
- [50] B. HEEREN, M. RUMPF, P. SCHRÖDER, M. WARDETZKY, AND B. WIRTH, *Splines in the space of shells*, in Computer Graphics Forum, vol. 35, Wiley Online Library, 2016, pp. 111–120. (Cited on page 146.)
- [51] D. HENRION, I. MEZIC, AND M. PUTINAR, *Applied koopmanism*, (2016). (Cited on page 31.)
- [52] A. N. HIRANI, *Discrete exterior calculus*, PhD thesis, Citeseer, 2003. (Cited on pages 1 and 7.)
- [53] T. HOFFMANN, A. O. SAGEMAN-FURNAS, AND M. WARDETZKY, *A discrete parametrized surface theory in \mathbb{R}^3* , ArXiv e-prints, (2014). (Cited on page 86.)
- [54] Q. HUANG, F. WANG, AND L. GUIBAS, *Functional map networks for analyzing and exploring large shape collections*, ACM Transactions on Graphics (TOG), 33 (2014), p. 36. (Cited on pages 3, 10, 50 and 55.)

- [55] Q.-X. HUANG, B. ADAMS, M. WICKE, AND L. J. GUIBAS, *Non-rigid registration under isometric deformations*, CGF (Proc. SGP), (2008), pp. 1449–1457. (Cited on pages 50 and 67.)
- [56] M. K. HURDAL AND K. STEPHENSON, *Discrete conformal methods for cortical brain flattening*, Neuroimage, 45 (2009), pp. S86–S98. (Cited on page 67.)
- [57] W. JUNG, H. SHIN, AND B. K. CHOI, *Self-intersection removal in triangular mesh offsetting*, Computer-Aided Design and Applications, 1 (2004), pp. 477–484. (Cited on page 86.)
- [58] E. KALOGERAKIS, A. HERTZMANN, AND K. SINGH, *Learning 3d mesh segmentation and labeling*, ACM Trans. Graph., 29 (2010), p. 102. (Cited on page 51.)
- [59] Z. KARNI AND C. GOTSMAN, *Spectral compression of mesh geometry*, in Proceedings of the 27th annual conference on Computer graphics and interactive techniques, ACM Press/Addison-Wesley Publishing Co., 2000, pp. 279–286. (Cited on page 21.)
- [60] P. KHAVARI, *Regge Calculus as a Numerical Approach to General Relativity*, PhD thesis, University of Toronto, 2009. (Cited on page 26.)
- [61] M. KILIAN, N. J. MITRA, AND H. POTTMANN, *Geometric modeling in shape space*, in ACM Transactions on Graphics (TOG), vol. 26, ACM, 2007, p. 64. (Cited on pages 49, 65, 113, 116 and 146.)
- [62] V. G. KIM, Y. LIPMAN, AND T. FUNKHOUSER, *Blended intrinsic maps*, ACM TOG (Proc. SIGGRAPH), 30 (2011). (Cited on pages 49, 51, 65, 67, 68 and 75.)
- [63] L. KOBBELT, S. CAMPAGNA, J. VORSATZ, AND H.-P. SEIDEL, *Interactive multi-resolution modeling on arbitrary meshes*, in Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '98, 1998, pp. 105–114. (Cited on page 115.)
- [64] A. KOVNATSKY, M. M. BRONSTEIN, X. BRESSON, AND P. VANDERGHEYNST, *Functional correspondence by matrix completion*, in Proc. CVPR, June 2015. (Cited on pages 2, 3, 8 and 10.)
- [65] A. KOVNATSKY, M. M. BRONSTEIN, A. M. BRONSTEIN, K. GLASHOFF, AND R. KIMMEL, *Coupled quasi-harmonic bases*, in Computer Graphics Forum, vol. 32, Wiley Online Library, 2013, pp. 439–448. (Cited on pages 53 and 67.)
- [66] E. KREYSZIG, *Differential Geometry*, Dover, 1959. (Cited on page 86.)
- [67] C. H. LEE, A. VARSHNEY, AND D. W. JACOBS, *Mesh saliency*, in ACM Transactions on Graphics (TOG), vol. 24, ACM, 2005, pp. 659–666. (Cited on pages 58 and 76.)

- [68] J. D. LEEUW, I. J. R. BARRA, F. BRODEAU, G. ROMIER, AND B. V. CUTSEM, *Applications of convex analysis to multidimensional scaling*, in Recent Developments in Statistics, 1977, pp. 133–146. (Cited on page 86.)
- [69] Y. LIPMAN AND T. FUNKHOUSER, *Mobius voting for surface correspondence*, ACM Transactions on Graphics (Proc. SIGGRAPH), 28 (2009). (Cited on pages 49, 51, 65 and 67.)
- [70] Y. LIPMAN, O. SORKINE, D. COHEN-OR, D. LEVIN, C. ROSSI, AND H.-P. SEIDEL, *Differential coordinates for interactive mesh editing*, in Shape Modeling Applications, 2004, pp. 181–190. (Cited on pages 114 and 115.)
- [71] R. LITMAN AND A. M. BRONSTEIN, *Learning spectral descriptors for deformable shape correspondence*, IEEE transactions on pattern analysis and machine intelligence, 36 (2014), pp. 171–180. (Cited on pages 51, 53 and 56.)
- [72] B. LIU, Y. TONG, F. D. GOES, AND M. DESBRUN, *Discrete connection and covariant derivative for vector field analysis and design*, ACM Transactions on Graphics (TOG), 35 (2016), p. 23. (Cited on page 130.)
- [73] J. MARTINEZ ESTURO, C. RÖSSL, AND H. THEISEL, *Generalized metric energies for continuous shape deformation*, Springer LNCS (Proc. Curves and Surfaces 2012), 8177 (2013), pp. 135–157. (Cited on page 115.)
- [74] F. MÉMOLI, *On the use of Gromov-Hausdorff Distances for Shape Comparison*, in Symposium on Point Based Graphics, 2007, pp. 81–90. (Cited on pages 49, 50 and 67.)
- [75] M. MEYER, M. DESBRUN, P. SCHRÖDER, AND A. H. BARR, *Discrete differential-geometry operators for triangulated 2-manifolds*, in Visualization and mathematics III, Springer, 2003, pp. 35–57. (Cited on page 92.)
- [76] M. I. MILLER, A. TROUVÉ, AND L. YOUNES, *Geodesic shooting for computational anatomy*, Journal of mathematical imaging and vision, 24 (2006), pp. 209–228. (Cited on page 67.)
- [77] C. MOLER AND C. VAN LOAN, *Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later*, SIAM review, 45 (2003), pp. 3–49. (Cited on page 71.)
- [78] S. MORITA, *Geometry of differential forms*, vol. 201, American Mathematical Soc., 2001. (Cited on pages 13, 19 and 20.)
- [79] MOSEK APS, *The MOSEK optimization toolbox for MATLAB manual. Version 7.1 (Revision 28)*., 2015. (Cited on pages 100, 102 and 108.)
- [80] A. MUHIC AND B. PLESTENJAK, *On the singular two-parameter eigenvalue problem*, Electron. J. Linear Algebra, 18 (2009), p. 42. (Cited on page 90.)

- [81] M. MÜLLER, N. CHENTANEZ, T.-Y. KIM, AND M. MACKLIN, *Strain based dynamics*, in Proc. SCA, 2014, pp. 149–157. (Cited on page 115.)
- [82] P. MUSIALSKI, T. AUZINGER, M. BIRSAK, M. WIMMER, AND L. KOBELT, *Reduced-order shape optimization using offset surfaces*, ACM Trans. Graph., 34 (2015), pp. 102:1–102:9. (Cited on page 86.)
- [83] S. B. MYERS AND N. STEENROD, *The group of isometries of a riemannian manifold*, Annals of Mathematics, (1939), pp. 400–416. (Cited on page 19.)
- [84] A. MYLES, N. PIETRONI, AND D. ZORIN, *Robust field-aligned global parametrization*, ACM Transactions on Graphics (TOG), 33 (2014), p. 135. (Cited on page 74.)
- [85] A. NEALEN, M. MÜLLER, R. KEISER, E. BOXERMAN, AND M. CARLSON, *Physically based deformable models in computer graphics*, in Computer graphics forum, vol. 25, Wiley Online Library, 2006, pp. 809–836. (Cited on pages 115 and 122.)
- [86] A. NEALEN, O. SORKINE, M. ALEXA, AND D. COHEN-OR, *A sketch-based interface for detail-preserving mesh editing*, ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH), 24 (2005), pp. 1142–1147. (Cited on page 115.)
- [87] T. NEUMANN, K. VARANASI, C. THEOBALT, M. MAGNOR, AND M. WACKER, *Compressed manifold modes for mesh processing*, in Computer Graphics Forum, vol. 33, Wiley Online Library, 2014, pp. 35–44. (Cited on page 34.)
- [88] M. OVSJANIKOV, M. BEN-CHEN, J. SOLOMON, A. BUTSCHER, AND L. GUIBAS, *Functional maps: a flexible representation of maps between shapes*, ACM TOG (Proc. SIGGRAPH), 31 (2012). (Cited on pages 1, 2, 3, 7, 8, 9, 31, 32, 37, 50, 51, 52, 53, 58, 59, 60, 65, 66, 67, 68, 76, 84, 114, 138 and 141.)
- [89] M. OVSJANIKOV, Q. MÉRIGOT, F. MÉMOLI, AND L. GUIBAS, *One point isometric matching with the heat kernel*, CGF, 29 (2010), pp. 1555–1564. (Cited on pages 50, 65 and 67.)
- [90] M. OVSJANIKOV, Q. MÉRIGOT, V. PĂTRĂUCEAN, AND L. GUIBAS, *Shape matching via quotient spaces*, Computer Graphics Forum, 32 (2013), pp. 1–11. (Cited on pages 31, 50 and 58.)
- [91] D. PANOZZO, E. PUPPO, M. TARINI, AND O. SORKINE-HORNUNG, *Frame fields: Anisotropic and non-orthogonal cross fields*, ACM Transactions on Graphics (TOG), 33 (2014), p. 134. (Cited on pages 86, 102 and 113.)
- [92] T. PAPADOPOULOU AND M. I. LOURAKIS, *Estimating the jacobian of the singular value decomposition: Theory and applications*, in Computer Vision-ECCV 2000, Springer, 2000, pp. 554–570. (Cited on pages 56 and 74.)

- [93] N. PARIES, P. DEGENER, AND R. KLEIN, *Simple and efficient mesh editing with consistent local frames*, in Computer Graphics and Applications, 2007. PG'07. 15th Pacific Conference on, IEEE, 2007, pp. 461–464. (Cited on page 115.)
- [94] N. PATRIKALAKIS AND T. MAEKAWA, *Shape Interrogation for Computer Aided Design and Manufacturing*, Springer, 2009. (Cited on page 94.)
- [95] D. PAVLOV, P. MULLEN, Y. TONG, E. KANSO, J. E. MARSDEN, AND M. DESBRUN, *Structure-preserving discretization of incompressible fluids*, Physica D: Non-linear Phenomena, 240 (2011), pp. 443–458. (Cited on page 114.)
- [96] J. POKRASS, A. M. BRONSTEIN, M. M. BRONSTEIN, P. SPRECHMANN, AND G. SAPIRO, *Sparse modeling of intrinsic correspondences*, in Computer Graphics Forum, vol. 32, 2013, pp. 459–468. (Cited on pages 2, 3, 8, 10, 67 and 114.)
- [97] N. RAY AND D. SOKOLOV, *Robust polylines tracing for n -symmetry direction field on triangulated surfaces*, ACM Transactions on Graphics (TOG), 33 (2014), p. 30. (Cited on page 74.)
- [98] T. REGGE, *General relativity without coordinates*, Il Nuovo Cimento (1955-1965), 19 (1961), pp. 558–571. (Cited on page 26.)
- [99] M. REUTER, F.-E. WOLTER, AND N. PEINECKE, *Laplace–beltrami spectra as ‘shape-dna’ of surfaces and solids*, Computer-Aided Design, 38 (2006), pp. 342–366. (Cited on page 21.)
- [100] E. RODOLÀ, S. R. BULÒ, T. WINDHEUSER, M. VESTNER, AND D. CREMERS, *Dense non-rigid shape correspondence using random forests*, in Proc. CVPR, 2014, pp. 4177–4184. (Cited on pages 2, 3, 8, 10, 31, 51 and 67.)
- [101] E. RODOLÀ, L. COSMO, M. M. BRONSTEIN, A. TORSSELLO, AND D. CREMERS, *Partial functional correspondence*, in Computer Graphics Forum, Wiley Online Library, 2016. (Cited on pages 2, 3, 8, 10, 31 and 36.)
- [102] E. RODOLA, M. MOELLER, AND D. CREMERS, *Point-wise map recovery and refinement from functional correspondence*, arXiv preprint arXiv:1506.05603, (2015). (Cited on pages 2, 3, 8 and 10.)
- [103] S. ROSENBERG, *The Laplacian on a Riemannian Manifold*, Cambridge University Press, 1997. Cambridge Books Online. (Cited on pages 13, 21, 24 and 85.)
- [104] M. RUMPF AND M. WARDETZKY, *Geometry processing from an elastic perspective*, GAMM-Mitteilungen, 37 (2014), pp. 184–216. (Cited on page 115.)
- [105] R. M. RUSTAMOV, *Laplace-beltrami eigenfunctions for deformation invariant shape representation*, in Proceedings of the fifth Eurographics symposium on Geometry processing, Eurographics Association, 2007, pp. 225–233. (Cited on pages 1 and 7.)

- [106] R. M. RUSTAMOV, M. OVSJANIKOV, O. AZENCOT, M. BEN-CHEN, F. CHAZAL, AND L. GUIBAS, *Map-based exploration of intrinsic shape differences and variability*, ACM Transactions on Graphics (TOG), 32 (2013), p. 72. (Cited on pages 2, 9, 31, 38, 39, 40, 42, 43, 49, 51, 84, 85, 86, 88, 97, 98, 102, 114, 116, 120, 128 and 129.)
- [107] Y. SAHILLIOĞLU AND Y. YEMEZ, *Coarse-to-fine combinatorial matching for dense isometric shape correspondence*, Computer Graphics Forum, 30 (2011), pp. 1461–1470. (Cited on pages 50, 65 and 67.)
- [108] F.-J. SAYAS, *A gentle introduction to the finite element method*, Lecture notes, University of Delaware, (2008). (Cited on page 87.)
- [109] H. SCHUMACHER, *Conformal maps and p-dirichlet energies*, tech. rep., Citeseer, 2013. (Cited on page 39.)
- [110] M. SELA, Y. AFLALO, AND R. KIMMEL, *Computational caricaturization of surfaces*, Computer Vision and Image Understanding, 141 (2015), pp. 1–17. (Cited on page 115.)
- [111] N. SHAPIRA AND M. BEN-CHEN, *Cross-collection map inference by intrinsic alignment of shape spaces*, in Computer Graphics Forum, Wiley Online Library, 2014. (Cited on page 55.)
- [112] R. SINGH AND J. MANHAS, *Composition Operators on Function Spaces*, no. no. 179 in Composition operators on function spaces, North-Holland, 1993. (Cited on pages 31 and 32.)
- [113] J. SOLOMON, M. BEN-CHEN, A. BUTSCHER, AND L. GUIBAS, *As-killing-as-possible vector fields for planar deformation*, in Computer Graphics Forum, vol. 30, Wiley Online Library, 2011, pp. 1543–1552. (Cited on page 115.)
- [114] O. SORKINE AND M. ALEXA, *As-rigid-as-possible surface modeling*, in Symposium on Geometry processing, vol. 4, 2007. (Cited on page 113.)
- [115] O. SORKINE, D. COHEN-OR, Y. LIPMAN, M. ALEXA, C. RÖSSL, AND H.-P. SEIDEL, *Laplacian surface editing*, in Proc. SGP, ACM, 2004, pp. 175–184. (Cited on pages 114 and 115.)
- [116] G. STRANG AND G. J. FIX, *An analysis of the finite element method*, vol. 212, Wellesley-Cambridge, 2 ed., 2008. (Cited on pages 87 and 88.)
- [117] R. W. SUMNER AND J. POPOVIĆ, *Deformation transfer for triangle meshes*, in ACM Transactions on Graphics (TOG), vol. 23, 2004, pp. 399–405. (Cited on pages 113, 116, 126 and 136.)
- [118] R. W. SUMNER, M. ZWICKER, C. GOTSMAN, AND J. POPOVIĆ, *Mesh-based inverse kinematics*, in ACM SIGGRAPH, 2005, pp. 488–495. (Cited on pages 49 and 65.)

- [119] J. SUN, M. OVSJANIKOV, AND L. GUIBAS, *A concise and provably informative multi-scale signature based on heat diffusion*, in Computer Graphics Forum, vol. 28, Wiley Online Library, 2009, pp. 1383–1392. (Cited on pages [1](#), [7](#), [21](#), [23](#), [24](#), [52](#), [58](#) and [76](#).)
- [120] G. K. TAM, Z.-Q. CHENG, Y.-K. LAI, F. C. LANGBEIN, Y. LIU, D. MARSHALL, R. R. MARTIN, X.-F. SUN, AND P. L. ROSIN, *Registration of 3d point clouds and meshes: A survey from rigid to nonrigid*, Visualization and Computer Graphics, IEEE Transactions on, 19 (2013), pp. 1199–1217. (Cited on page [49](#).)
- [121] G. TAUBIN, *A signal processing approach to fair surface design*, in Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, ACM, 1995, pp. 351–358. (Cited on page [22](#).)
- [122] D. TERZOPOULOS, J. PLATT, A. BARR, AND K. FLEISCHER, *Elastically deformable models*, in ACM Siggraph Computer Graphics, vol. 21, ACM, 1987, pp. 205–214. (Cited on page [115](#).)
- [123] A. TEVS, M. BOKELOH, M. WAND, A. SCHILLING, AND H.-P. SEIDEL, *Isometric registration of ambiguous and partial data*, in Proc. CVPR, 2009, pp. 1185–1192. (Cited on pages [50](#), [65](#) and [67](#).)
- [124] B. THOMASZEWSKI, S. PABST, AND W. STRASSER, *Continuum-based strain limiting*, in Computer Graphics Forum, vol. 28, Wiley Online Library, 2009, pp. 569–576. (Cited on page [115](#).)
- [125] B. VALLET AND B. LÉVY, *Spectral geometry processing with manifold harmonics*, in Computer Graphics Forum, vol. 27, Wiley Online Library, 2008, pp. 251–260. (Cited on page [35](#).)
- [126] O. VAN KAICK, A. TAGLIASACCHI, O. SIDI, H. ZHANG, D. COHEN-OR, L. WOLF, AND G. HAMARNEH, *Prior knowledge for part correspondence*, Computer Graphics Forum (Proc. Eurographics), 30 (2011), pp. 553–562. (Cited on page [51](#).)
- [127] O. VAN KAICK, H. ZHANG, G. HAMARNEH, AND D. COHEN-OR, *A survey on shape correspondence*, Computer Graphics Forum, 30 (2011), pp. 1681–1707. (Cited on page [50](#).)
- [128] A. VAXMAN, C. MÜLLER, AND O. WEBER, *Conformal mesh deformations with möbius transformations*, ACM Transactions on Graphics (TOG), 34 (2015), p. 55. (Cited on pages [114](#) and [115](#).)
- [129] W. VON FUNCK, H. THEISEL, AND H.-P. SEIDEL, *Vector field based shape deformations*, in ACM Transactions on Graphics (TOG), vol. 25, ACM, 2006, pp. 1118–1125. (Cited on page [115](#).)
- [130] C. VON-TYCOWICZ, C. SCHULZ, H.-P. SEIDEL, AND K. HILDEBRANDT, *Real-time nonlinear shape interpolation*, ACM Transactions on Graphics (TOG), 34 (2015), p. 34. (Cited on page [113](#).)

- [131] S. WANG, Y. WANG, M. JIN, X. D. GU, AND D. SAMARAS, *Conformal geometry and its applications on 3d shape matching, recognition, and stitching*, Pattern Analysis and Machine Intelligence, IEEE Transactions on, 29 (2007), pp. 1209–1220. (Cited on pages 65 and 67.)
- [132] Y. WANG, B. LIU, AND Y. TONG, *Linear surface reconstruction from discrete fundamental forms on triangle meshes*, Computer Graphics Forum, (2012). (Cited on page 86.)
- [133] W. WELCH AND A. WITKIN, *Variational surface modeling*, in ACM SIGGRAPH computer graphics, vol. 26, ACM, 1992, pp. 157–166. (Cited on page 115.)
- [134] T. WINKLER, J. DRIESEBERG, M. ALEXA, AND K. HORMANN, *Multi-scale geometry interpolation*, in Computer graphics forum, vol. 29, Wiley Online Library, 2010, pp. 309–318. (Cited on page 146.)
- [135] Y. YU, K. ZHOU, D. XU, X. SHI, H. BAO, B. GUO, AND H.-Y. SHUM, *Mesh editing with poisson-based gradient field manipulation*, ACM Transactions on Graphics (TOG), 23 (2004), pp. 644–651. (Cited on pages 113, 114 and 115.)
- [136] R. ZAYER, C. RÖSSL, Z. KARNI, AND H.-P. SEIDEL, *Harmonic guidance for surface deformation*, in Computer Graphics Forum, vol. 24, Wiley Online Library, 2005, pp. 601–609. (Cited on pages 114 and 115.)
- [137] W. ZENG, R. GUO, F. LUO, AND X. GU, *Discrete heat kernel determines discrete Riemannian metric*, Graph. Models, 74 (2012), pp. 121–129. (Cited on pages 35, 85 and 128.)
- [138] H. ZHANG, A. SHEFFER, D. COHEN-OR, Q. ZHOU, O. VAN KAICK, AND A. TAGLIASACCHI, *Deformation-driven shape correspondence*, in Proc. SGP, 2008, pp. 1431–1439. (Cited on pages 49 and 51.)
- [139] Z. ZHANG, G. LI, H. LU, Y. OUYANG, M. YIN, AND C. XIAN, *Fast as-isometric-as-possible shape interpolation*, Computers & Graphics, 46 (2015), pp. 244–256. (Cited on page 115.)
- [140] D. ZORIN, P. SCHRÖDER, AND W. SWELDENS, *Interactive multiresolution mesh editing*, in Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '97, 1997, pp. 259–268. (Cited on page 115.)

Titre : Représentation fonctionnelle des surfaces déformables pour l'analyse et la synthèse géométrique

Mots clés : Imagerie 3D, Association forme, Algorithmes géométriques

Résumé :

La création et la compréhension des déformations de surfaces sont des thèmes récurrents pour le traitement de géométrie 3D. Comme les surfaces lisses peuvent être représentées de multiples façons allant du nuage de points au maillage polygonal, un enjeu important est de pouvoir comparer ou déformer des formes discrètes indépendamment de leur représentation. Une réponse possible est de choisir une représentation flexible des surfaces déformables qui peut facilement être transportée d'une structure de données à une autre.

Dans ce but, les "functional map" proposent de représenter des applications entre les surfaces et, par extension, des déformations comme des opérateurs agissant sur des fonctions. Cette approche a été introduite récemment pour le traitement de modèle 3D, mais a été largement utilisée dans d'autres domaines tels que la géométrie différentielle, la théorie des opérateurs et les systèmes dynamiques, pour n'en citer que quelques-uns. Le principal avantage de ce point de vue est de détourner les problèmes encore non-résolus, tels que la correspondance forme et le transfert de déformations, vers l'analyse fonctionnelle dont l'étude et la discrétisation sont souvent mieux connues. Cette thèse approfondit l'analyse et fournit de nouvelles applications à ce cadre d'étude. Deux questions principales sont discutées.

Premièrement, étant donné deux surfaces, nous analysons les déformations sous-jacentes. Une façon de procéder est de trouver des correspondances qui minimisent la distorsion globale. Pour compléter l'analyse, nous identifions les parties les moins fiables du difféomorphisme grâce à une méthode d'apprentissage. Une fois repérés, les défauts peuvent être éliminés de manière différentiable à l'aide d'une représentation adéquate des champs de vecteurs tangents.

Le deuxième développement concerne le problème inverse : étant donné une déformation représentée comme un opérateur, comment déformer une surface en conséquence ? Dans une première approche, nous analysons un encodage de la structure intrinsèque et extrinsèque d'une forme en tant qu'opérateur fonctionnel. Dans ce cadre, l'objet déformé peut être obtenu, à rotations et translations près, en résolvant une série de problèmes d'optimisation convexe. Deuxièmement, nous considérons une version linéarisée de la méthode précédente qui nous permet d'appréhender les champs de déformation comme agissant sur la métrique induite. En conséquence la résolution de problèmes difficiles, tel que le transfert de déformation, sont effectués à l'aide de simple systèmes linéaires d'équations.

Title : Functional representation of deformable surfaces for geometry processing

Keywords : Geometry processing, Shape matching, Geometry algorithm

Abstract :

Creating and understanding deformations of surfaces are recurring themes in geometry processing. As smooth surfaces can be represented in many ways from point clouds to triangle meshes, one of the challenges is being able to compare or deform consistently discrete shapes independently of their representation. A possible answer is choosing a flexible representation of deformable surfaces that can easily be transported from one structure to another.

Toward this goal, the functional map framework proposes to represent maps between surfaces and, to further extents, deformation of surfaces as operators acting on functions. This approach has been recently introduced in geometry processing but has been extensively used in other fields such as differential geometry, operator theory and dynamical systems, to name just a few. The major advantage of such point of view is to deflect challenging problems, such as shape matching and deformation transfer, toward functional analysis whose discretization has been well studied in various cases. This thesis investigates further analysis and novel applications in this framework. Two aspects of the functional representation fra-

mework are discussed.

First, given two surfaces, we analyze the underlying deformation. One way to do so is by finding correspondences that minimize the global distortion. To complete the analysis we identify the least and most reliable parts of the mapping by a learning procedure. Once spotted, the flaws in the map can be repaired in a smooth way using a consistent representation of tangent vector fields.

The second development concerns the reverse problem : given a deformation represented as an operator how to deform a surface accordingly ? In a first approach, we analyse a coordinate-free encoding of the intrinsic and extrinsic structure of a surface as functional operator. In this framework a deformed shape can be recovered up to rigid motion by solving a set of convex optimization problems. Second, we consider a linearized version of the previous method enabling us to understand deformation fields as acting on the underlying metric. This allows us to solve challenging problems such as deformation transfer are solved using simple linear systems of equations.