Robust shape approximation and mapping between surfaces

Manish Mandad

Université Côte d'Azur - UFR Sciences
Ecole Doctorale STIC
INRIA Sophia Antipolis (TITANE)

Thèse de doctorat

présentée en vue de l'obtention du
grade de docteur en Informatique

par

Manish Mandad

Robust Shape Approximation and Mapping between Surfaces

Dirigée par Pierre Alliez

Soutenue le Novembre 29, 2016

Jury:

Marc Alexa	*Professeur, Technische Universität Berlin*	Rapporteur
Pierre Alliez	*Senior Researcher, Inria Sophia Antipolis*	Directeur de thèse
Paolo Cignoni	*Senior Researcher, CNR-ISTI, Pisa*	Rapporteur
David Cohen-Steiner	*Researcher, Inria Sophia Antipolis*	Examinateur
Marco Cuturi	*Professeur, ENSAE / CREST*	Examinateur

Contents

Acknowledgments

First of all, I am most thankful to my advisor, Pierre Alliez, for his thorough involvement and patience. His incredible zeal for research and a near supernatural ability of fulfilling multiple deadlines at the same time has been a source of motivation. He pushed me towards ideas which seemed unreasonable at times, but later formed the backbone of the algorithms presented in this thesis.

I am grateful to David Cohen-Steiner for being my co-advisor (though unofficial). Every discussion with him brought me some new insights. His critical thinking and reasoning always amused as well as motivated me to perform better.

I owe many thanks to Mathieu Desbrun for helping me out with every minute issue, always excited and being available for discussions, and most importantly, bearing up with my horrible explanation and coding skills. I am also thankful to Leif Kobbelt for his guidance, broad discussions and providing me with a postdoctoral fellowship opportuinity.

I must thank Florent Lafarge, David Bommes, Bishesh Khanal, Yannick Verdie, Sonali, Liuyun Duan, Deepesh Agarwal and Sven Oesau for always being there and helping me out whenever I needed them.

I thank the members of the jury: Marc Alexa, Paolo Cignoni and Marco Cuturi for spending their precious time on my dissertation.

Finally, I attribute some of the shortcomings of this thesis to my family, friends and *la belle France* who gave me enough reasons to escape work every now and then.

Abstract

This thesis is divided into two independent parts.

In the first part, we introduce a method that, given an input tolerance volume, generates a surface triangle mesh guaranteed to be within the tolerance, intersection free and topologically correct. A pliant meshing algorithm is used to capture the topology and discover the anisotropy in the input tolerance volume in order to generate a concise output. We first refine a 3D Delaunay triangulation over the tolerance volume while maintaining a piecewise-linear function on this triangulation, until an isosurface of this function matches the topology sought after. We then embed the isosurface into the 3D triangulation via mutual tessellation, and simplify it while preserving the topology. Our approach extends to surfaces with boundaries and to non-manifold surfaces. We demonstrate the versatility and efficacy of our approach on a variety of data sets and tolerance volumes.

In the second part we introduce a new approach for creating a homeomorphic map between two discrete surfaces. While most previous approaches compose maps over intermediate domains which result in suboptimal inter-surface mapping, we directly optimize a map by computing a variance-minimizing mass transport plan between two surfaces. This non-linear problem, which amounts to minimizing the Dirichlet energy of both the map and its inverse, is solved using two alternating convex optimization problems in a coarse-to-fine fashion. Computational efficiency is further improved through the use of Sinkhorn iterations (modified to handle minimal regularization and unbalanced transport plans) and diffusion distances. The resulting inter-surface mapping algorithm applies to arbitrary shapes robustly and efficiently, with little to no user interaction.

Résumé

Cette thèse comprend deux parties indépendantes.

Dans la première partie nous contribuons une nouvelle méthode qui, étant donnée un volume de tolérance, génère un maillage triangulaire surfacique garanti d'être dans le volume de tolérance, sans auto-intersection et topologiquement correct. Un algorithme flexible est conçu pour capturer la topologie et découvrir l'anisotropie dans le volume de tolérance dans le but de générer un maillage de faible complexité.

Dans la seconde partie nous contribuons une nouvelle approche pour calculer une fonction de correspondance entre deux surfaces. Tandis que la plupart des approches précédentes procède par composition de correspondance avec un domaine simple planaire, nous calculons une fonction de correspondance en optimisant directement une fonction de sorte à minimiser la variance d'un plan de transport entre les surfaces.

Representative Publications

MANDAD, M., Cohen-Steiner, D., and Alliez, P., 2015. Isotopic Approximation within a Tolerance Volume. Proceedings of ACM SIGGRAPH, *ACM transactions on Graphics*, 34(4):64:1–64:12. ISSN 0730-0301.

MANDAD, M., Cohen-Steiner, D., Desbrun, M., Alliez, P., and Kobbelt L., 2016. Low Distortion Inter-surface Mapping via Variance-Minimizing Mass Transport. (Submitted).

Part I

Isotopic Approximation within a Tolerance Volume

Chapter 1

Introduction

Faithful approximation of complex shapes with simplicial meshes is a multifaceted problem, involving geometry, topology and their discretization. This problem has received considerable interest due to its wide range of applications and the ever-increasing accessibility of geometric sensors. Increased availability of scanned geometric models, however, does not mean improved quality: while many practitioners have access to high-end acquisition systems, a recent trend is to replace these expensive systems with a combination of consumer-level acquisition devices. Measurement data generated by, and merged from, these heterogeneous devices are reputedly unfit for direct processing.

Similarly, the growing variety of geometry processing tools often increases the net amount of defects in data: conversion to and from various geometry representations often degrades the input, and rare are the algorithms that have stronger guarantees on their output than they have requirements on their input. As we deal with ever finer discretizations to capture intricate geometric features, this issue of offering strict geometric guarantees to be robust to the occurrence of artifacts is becoming more prevalent.

Geometric guarantees usually refer to upper bounds on the approximation error and to the absence of self-intersections. Topological guarantees refer to homotopy, homeomorphism or isotopy. Surface meshes with such guarantees are required for artifact-free rendering, computational engineering, reverse engineering, manufacturing and 3D printing. While *geometric* simplification can reduce the number of primitives, *topological* simplification can repair holes and degeneracies in existing discretizations. Combined, the two may also be used for reconstructing clean shapes from raw geometric data such as point sets or polygon soups with strict guarantees.

1.1 Background

Before delving into details, we discuss the existing theory required to understand the topic and our problem statement. As mentioned before, the problem of faithful approximation deals with two sets of theoretical guarantees: Topological and Geometric. We will detail them one by one.

1.1.1 Topology

In the context of this thesis, topological guarantees reflect whether the final (reconstructed) output has same topology as that of the input or not. Intuitively, topology refers to study of the properties that are preserved through deformations, twistings, and stretchings of objects. Tearing and stitching, however, are not allowed. Based on these properties and how strongly they are preserved, topological guarantees can be classified as homotopy, homeomorphism and isotopy.

Homotopy. Mathematically, two given spaces X and Y, are *homotopy equivalent* or of same *homotopy type* if there exist continuous maps $f : X \to Y$ and $g : Y \to X$ such that $f \circ g$ is the identity map on Y, and, $g \circ f$ is the identity map on X. Having the same homotopy type enforces the existence of isomorphisms between respective homology and homotopy groups. In other words, as explained by Chazal and Lieutier [2008], having the same homotopy type enforces a one-to-one correspondence between cycles, holes, connected components, tunnels, cavities and/or higher dimensional topological features of the two sets X and Y such that one set can be continuously deformed into the other set. These correspondences along with the relation between these features is defined by the maps f and g. For example, a solid disk or solid ball is homotopy equivalent to a point.

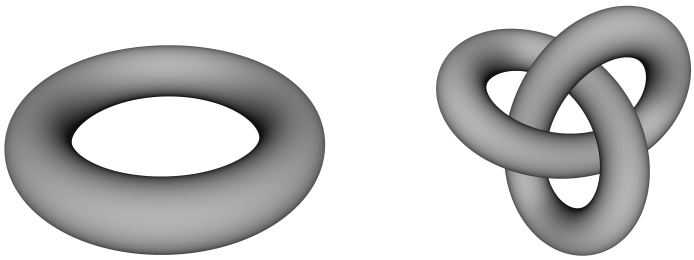


Figure 1.1: Unknot and knot. A torus and a trefoil knot are homeomorphic. Note that there is no need for the existence of a continuous deformation for the two spaces to be homeomorphic. Furthermore, they are not isotopic as the knots will have to pass through each other to form an unknot and it will not be possible to maintain a homeomorphism throughout the deformation.

Homeomorphism. Being a stronger condition, homeomorphism is generally considered as the notion of equality in topology. Two spaces with a homeomorphism between them are called homeomorphic. For example, a classic example in topology suggests that an unknot (torus) and a knot (trefoil knot) are homeomorphic (Figure 1.1). This is because one of the geometric objects can be stretched and bent continuously to form the other object. Mathematically, given two topological spaces X and Y, a function $f : X \to Y$ between these spaces is a homeomorphism if

1. f is a bijection and is continuous, and
2. the inverse of f exists and is continuous.

Isotopy. Isotopy is a finer relation than homeomorphism and the strongest condition for topological similarity between two objects. In our context, isotopy means that there exists a smooth deformation that maps one shape to another while maintaining a homeomorphism between the two. Note that maintaining a homeomorphism during deformation is not a necessary criteria for two objects to be homeomorphic. As a result, a coffee mug and donut are isotopic (Figure 1.2), but they are not isotopic to a trefoil knot (Figure 1.1).

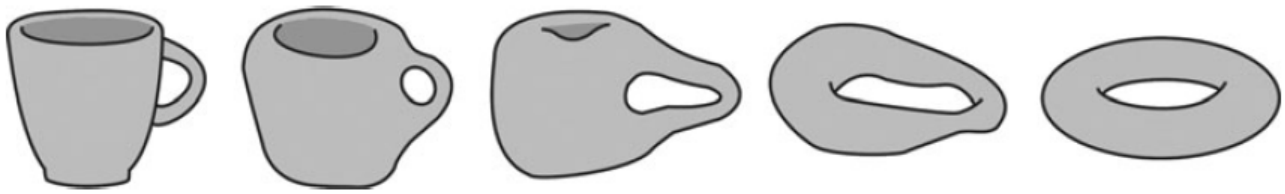


Figure 1.2: Isotopy. A coffee mug can be continuously deformed into a donut (torus). Intuitively, two surfaces are isotopic if they can be continuously deformed into the other without introducing self-intersections. Note that all the objects including the intermediate deformations are homeomorphic.

1.1.2 Geometry

Geometric guarantees mainly deal with upper bounds on the approximation error of the reconstructed output and with the absence of self-intersections in it. In this thesis we look at these guarantees only in the context of triangle meshes.

Approximation

It is usually desirable to significantly reduce the complexity (mainly redundancies) of the discrete surface representation while preserving, as much as possible, the geometry (features) of the object. This results in a tradeoff between the mesh complexity and mesh quality as illustrated in Figure 1.3. While the mesh complexity can be easily defined in terms of the vertex count,

the mesh quality has many aspects and is, in general, computed in terms of *Hausdorff* distance. Along with Hausdorff distance, other fairness criteria for example based on normal deviation, triangle shape, dihedral angle, valence etc. can also be used.

A *good* approximation can either be achieved by reconstructing an anisotropic mesh directly or via mesh simplification. Such an approximation can drastically reduce the size of an oversampled 3D data (for example, scan data or an isotropic mesh) and the output mesh can be efficiently used for geometry processing, multi-resolution hierarchies, level-of-detail (LOD) rendering and adaptation to hardware capabilities.

Figure 1.3: Size vs Quality tradeoff. As we gradually decrease the number of vertices, the quality of mesh (features) degrades leading to coarser approximation of original mesh.

Hausdorff Distance. The Hausdorff distance is a measure of (dis-)similarity between two sets and is widely used in various applications. It is measured as the maximum distance of a set to the nearest point in the other set. More formally, given two sets A and B, the (one-sided) Hausdorff distance $d_H(A \to B)$ from a set A to set B is a maximin function (Figure 1.4), defined as:

$$d_H(A \to B) = \sup_{\forall a \in A} \inf_{\forall b \in B} \|a - b\|. \tag{1.1}$$

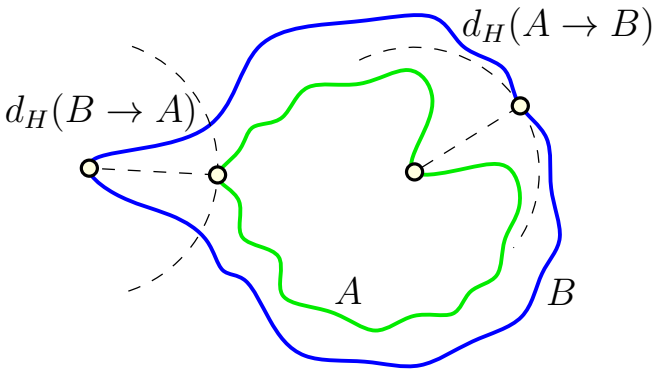


Figure 1.4: Hausdorff Distance. The one-sided Hausdorff distance is defined as the maximum distance of a set to the nearest point in the other set. Note that this distance is not a metric because it is not always symmetric, i.e. $d_H(A \to B) \neq d_H(B \to A)$ and $d_H(A \to B) = 0$ does not imply that both sets are identical.

Such distance between a given pair of surfaces can be easily computed via densely sampling them and finding the nearest neighbours [Cignoni *et al.*, 1996].

On a general note, asymmetry is a property of maximin functions, while minimin functions are symmetric. As a result, the two-sided Hausdorff distance is symmetric (assuming that the two sets do not have any repetitive elements), which can be calculated as the maximum of the two one-sided distances:

$$\max\{d_H(A \to B), d_H(B \to A)\}. \tag{1.2}$$

Mesh Simplification. Given a mesh $M = (V, F)$, where V and F denote the vertices and faces respectively, a mesh approximation problem can be defined as finding a new mesh $M' = (V', F')$ such that either

1. $|V'| = n < |V|$ and $\|M - M'\|$ is minimal, or
2. $\|M - M'\| < \delta$ and $|V'|$ is minimal.

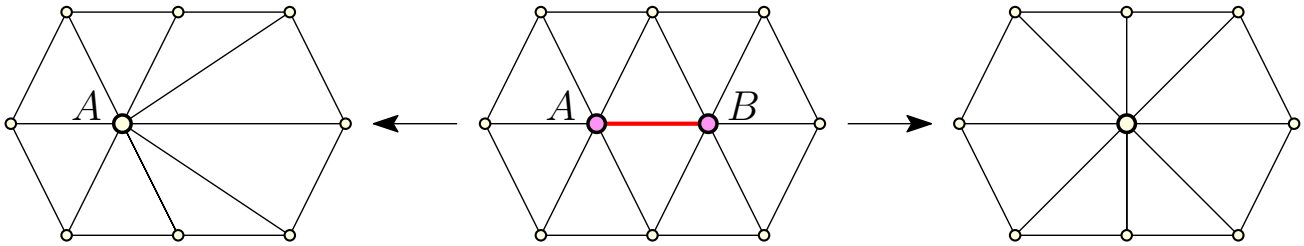


Figure 1.5: Edge collapse. Given an edge AB (middle), a half-edge collapse (left) locates the target vertex at one of the vertices of the edge. A general edge collapse (right), relocates the target vertex based on the user defined criteria. Note that an edge collapse always has a unique triangulation (combinatorially, depending on its one-ring neighbours), although the final vertex location may change.

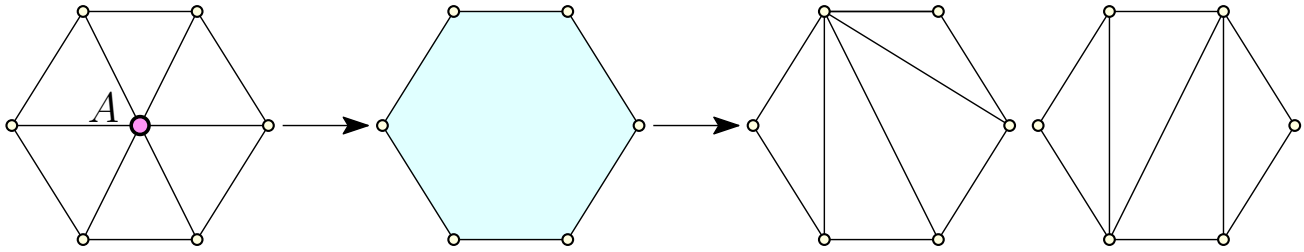


Figure 1.6: Vertex removal. From left to right: Given a vertex A, it is eliminated leading to a hole in the triangulation which is then filled via re-triangulation. Note that unlike an edge collapse operator (which has vertex location as the degree of freedom), a vertex removal operator has degrees of freedom in terms of triangulation.

In general, incremental decimation algorithm is used to implement mesh decimation. Using a greedy optimization approach, a decimation operator is applied on a mesh region until no further reduction is possible. Half-edge collapse is commonly used as a decimation operator due

to its simplicity, but, other methods like general edge collapse (Figure 1.5) and vertex removal with re-triangulation also exist (Figure 1.6). Note that both of them are **Euler** operations as they do not change the genus of the mesh.

Self-Intersection

A mesh with self-intersections does not necessarily mean that its topology (homeomorphism) has been altered. In fact, most of the time, purely based on connectivity, the mesh does not lose its homeomorphic property at all. However, the mesh seen as a function loses the bijective property and behaves as a one-to-many relation. Such a mesh cannot be used for practical applications, for example, 3D printing, as at self-intersections, the Euclidean distance is zero for points with non-zero geodesic distance. Note that an isotopic mesh is always free from (extra, if any) self-intersections. This, in fact, bridges this geometric property back to topological properties mentioned above.

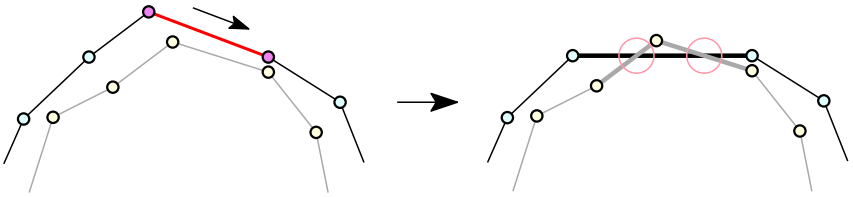


Figure 1.7: Self-Intersection. Intersection with the nearby mesh can be caused due to vertex removal or edge collapse operator in regions where the feature size is relatively smaller as compared to the desired Hausdorff distance error in the approximation.

The decimation operators discussed above can cause self-intersections. This is because these operators alter the triangulation which may end up intersecting with the nearby surface (Figure 1.7) or creating a local flip (Figure 1.8). They may also even change the topology of the mesh as we will see in the later sections.

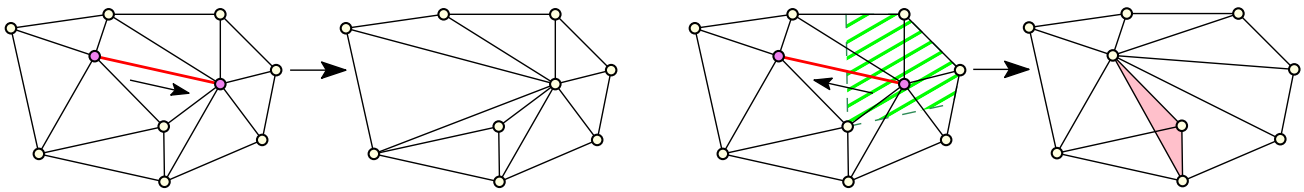


Figure 1.8: Flips. An edge collapse may not always result in a valid triangulation. For a triangulation to be valid, the target location of vertex must lie in the visibility kernel (green shaded region) of the one-ring. Left: a valid triangulation in 2D. Right: A self-intersection can also occur in the form of flips due to invalid triangulation. The same can also be extended to higher dimensions.

1.2 Related Work

A vast array of methodologies have been proposed for surface reconstruction and shape approximation over the years, ranging from Delaunay triangulation to implicit surfaces and decimation to optimization through clustering and refinement. Fewer, however, provide error bounds. In addition, they only apply to specific types of input geometry, and often fail to satisfy geometric *and/or* topological guarantees as we now review.

1.2.1 Reconstruction with Topological Guarantees

A great variety of methods have been proposed for surface reconstruction. Among them, while only a limited come with the theoretical guarantee of homeomorphism, only a *few* recent articles tackle the more difficult problem of ensuring isotopy.

The credit for the first reconstruction algorithm to provide theoretical guarantees goes to Amenta for Crust [Amenta *et al.*, 1998; Amenta and Bern, 1998]. This algorithm exploits the structures of the Voronoi diagram of the input point set to reconstruct the surface, and converges to the original surface as the sampling density increases. The output mesh of this combinatorial algorithm interpolates the input points rather than approximating them to form a surface and hence, can handle variable sampling density. A 2-dimensional curve reconstruction example is illustrated in Figure 1.9.

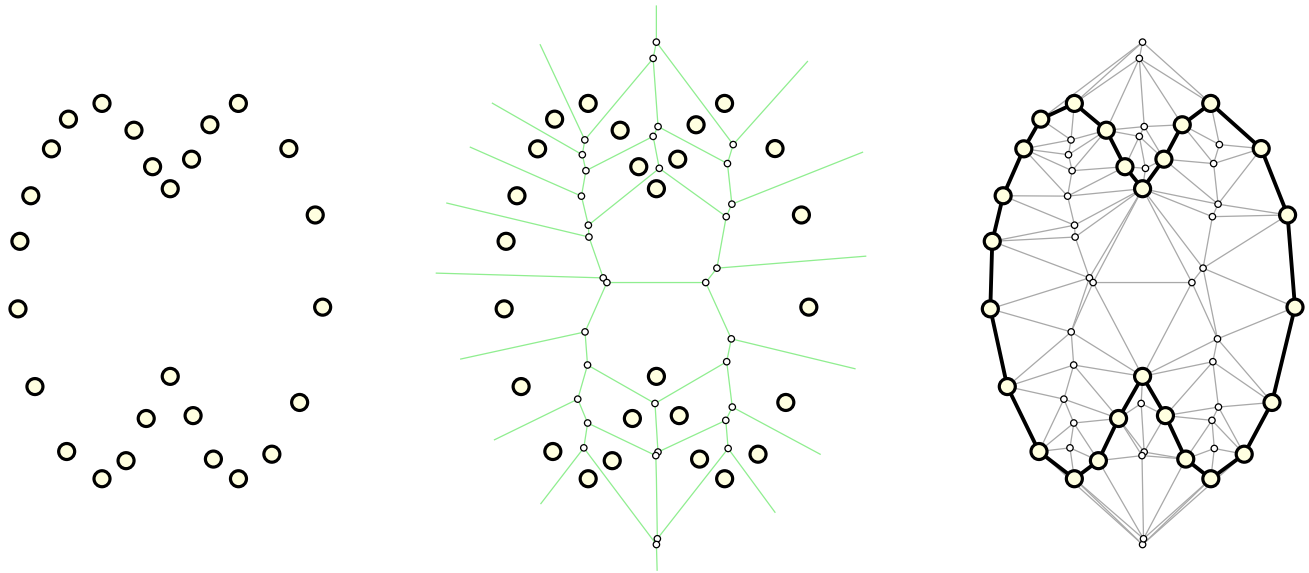


Figure 1.9: Reconstruction using Crust algorithm. From left to right: input point set; its Voronoi diagram along with Voronoi vertices and; the Delaunay triangulation of the union of input points and Voronoi vertices. The crust edges connecting the input points are depicted in black. In general, all Delaunay based algorithms attempt to identify or extract a correct subset from Delaunay triangulation for the desired curve or surface through various filtering methods or geometric heuristics.

The resulting mesh is homeomorphic and geometrically close to the original (smooth) surface when the prescribed sampling condition is satisfied. Furthermore, they proposed the definition of *poles*, formed by a subset of Voronoi vertices of the input point set which can represent an approximated medial axis when input data set is sufficiently dense.

Homeomorphism. Amenta *et al.* [2000] introduced the COCONE algorithm which improved the previous algorithm [Amenta and Bern, 1998] both in theory and in practice.

Following CRUST, Boissonnat and Cazals [2000] designed another algorithm based on natural neighbors and reconstruction using signed distance functions. The algorithm works in any dimension. Another Delaunay based approach Cohen-Steiner and Da [2004] provided a greedy algorithm for surface reconstruction form unorganized point sets. At each iteration, the algorithm ensured that the reconstructed surface is an orientable manifold, possibly with a boundary.

Note that the fundamental property used for proofs is that a *closed* surface is homeomorphic (and in fact isotopic) to the restricted Delaunay triangulation of a sufficiently dense point sample. However, the same does not hold for surfaces with boundaries. Although all these algorithms work nicely on dense data sets, they have difficulty when the data is undersampled resulting into a reconstructed surface that may not even be a manifold. In fact, in most such cases, these algorithms produce *holes* or other artifacts in the regions of undersampling. The reason for these artifacts can be attributed to relying totally on Delaunay triangulation or Voronoi diagram which, in practice, fails to form a topologically correct surface due to noise, undersampling or sharp surface features.

Watertight Reconstruction. To deal with undersampling, Amenta *et al.* [2001a,b] provided the POWER CRUST algorithm based on approximating the medial axis transform of the object and then using an inverse transform to produce the surface. The algorithm uses a weighted Voronoi diagram of the poles, referred to as a power diagram, instead of the Delaunay triangulation (as in Amenta and Bern [1998]) of the point set. The algorithm is guaranteed to produce a watertight surface (along with producing an approximate medial axis) and does not depend in any way on the quality (sampling density) of the input points. However, it introduces many extra points in the output and does not produce a triangulated surface.

Edelsbrunner [2003] proposed a Morse theoretic approach to determine the surfaces which also produces watertight surfaces. It also provides an algorithm to reconstruct the surface based on Delaunay complexes and extracting a subcomplex through repeated collapsing. All ideas and the results generalize to arbitrary dimensions. However, the algorithm may not recover small features of the surface even if it is densely sampled.

Adamy *et al.* [2002] introduced an *umbrella filter* algorithm coupled with a linear-programming based topological post-processing step designed for topologically correct watertight triangle mesh reconstruction. Dey and Goswami [2003] extended Amenta *et al.* [2000] to provide the TIGHT COCONE algorithm. The algorithm first computes a preliminary surface using COCONE which almost completes the reconstruction except in the vicinity of undersampled areas. A subsequent marking and peeling phase completes the reconstruction by filling holes and computes the output surface as the boundary of the union of Delaunay tetrahedra. The algorithm guarantees that the surface cannot have any holes. However, it relies on the locality of undersampling and may produce a non-manifold (though watertight) or empty output if this property is violated. Furthermore, it cannot reconstruct surfaces with boundaries.

Another class of approaches based on implicit surfaces have the advantage that the output surface always remains watertight and they work well on noisy dataset. Hoppe *et al.* [1992] pioneered an approach to explicitly construct the geometry of an implicit manifold. The algorithm computes a signed distance field to the underlying surface using local tangent plane construction, and extracts an isosurface from the distance field using the marching cubes algorithm. Mullen *et al.* [2010] proposed a method that improves the correctness of the local surface orientation in order to fill gaps in regions where data is missing. Schnabel *et al.* [2009] used implicit surfaces and combined this with graph-cuts to get a watertight model. Similarly, Shalom *et al.* [2010] used generalized cones with a data point at their apex to indicate exterior regions. Alliez *et al.* [2007] created an implicit surface using the anisotropy of the Voronoi cells to estimate the normals of the surface (Figure 1.10). However, such implicit surface methods have problems in reconstructing sharp features and are in general, not suited to deal with massive data sets.

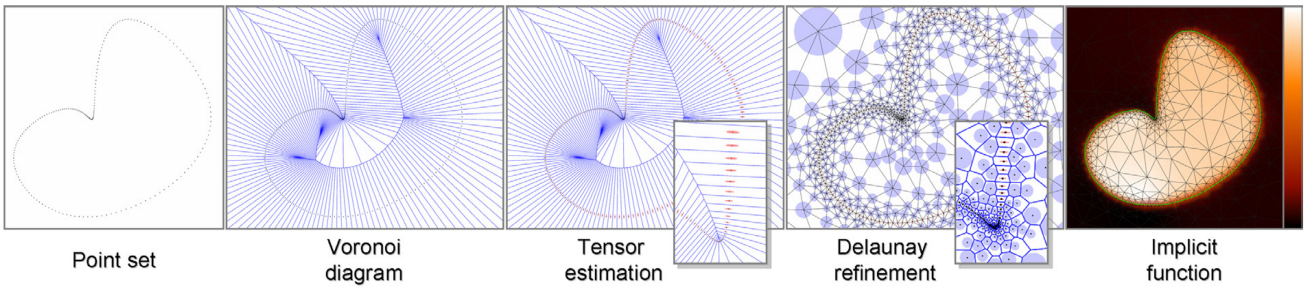


Figure 1.10: Voronoi-based variational reconstruction of unoriented point sets. From left to right: input point set; its Voronoi diagram; covariance matrices of the cells; Steiner points added through Delaunay refinement (isotropic tensors are assigned to Steiner points); piecewise linear function f (solution of a generalized eigenvalue problem). *(Image taken from Alliez* et al. *[2007])*

Isotopy. Given a nonsingular compact 2-manifold surface S without boundary, Sakkalis and Peters [2003] introduced algorithm for creating a family of approximating surfaces to S while

ensuring that each approximant (S') is ambient isotopic to the original. The algorithm proceeds by constraining the approximant to lie within a bounded offset of S. While most existing reconstruction methods provide only for piecewise-linear approximations, this method is also suitable for higher order approximations. The main result of Sakkalis and Peters [2003] is that S and S' are isotropic if projection on S defines a homeomorphism from S' to S (Figure 1.11).

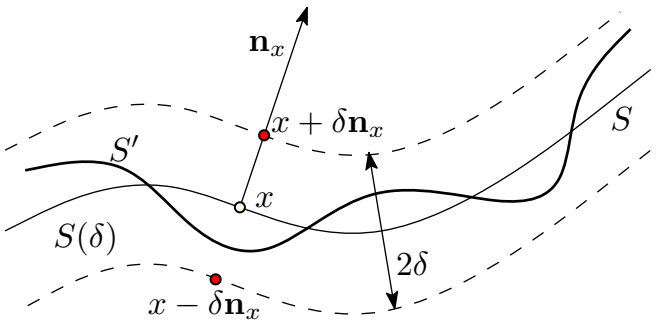


Figure 1.11: Conditions for ambient isotopy. Assuming S to be a compact nonsingular 2-manifold without boundary and a normal tubular neighborhood $S(\delta)$ for a given δ (refer to Sakkalis and Peters [2003] for details). If (a) $S' \subset S(\delta)$, and (b) $\forall x \in S$, the line segment $(x - \delta n_x, x + \delta n_x)$ intersects S' precisely at one point, then, S and S' are ambient isotopic.

Amenta *et al.* [2003] showed that a specific piecewise linear approximation of a closed surface is isotopic, using the same condition as in Sakkalis and Peters [2003] indirectly. Both these approaches involve not only the topology of the surfaces, but also the geometry in providing topological guarantees. Furthermore, since projection is involved, the conditions cannot be met when S is not smooth. Chazal and Cohen-Steiner [2004] proposed a purely topological condition which is sufficient to ensure isotopy but a practical algorithm is missing for this theoretical result. Another work Chazal and Lieutier [2008], provided reconstruction with topological guarantees using unions of balls centered at the data set. Attali *et al.* [2011] showed that Rips complexes can also be used to provide topologically correct reconstruction of shapes which might be of some computational interest in higher dimensions.

Isotopic surface reconstruction from noisy point sets, provided the sampling is dense enough, has also been investigated well [Dey, 2006]. Another approach which can handle noisy data set by estimating normals from *big* Delaunay balls [Dey and Sun, 2005] (Figure 1.12), extended the classical Moving Least Squares (MLS) [Shepard, 1968] (out-of-core extension [Fiorin *et al.*, 2007]) which is a popular method for functional approximation. A recent approach, Dey *et al.* [2009] showed that a restricted Delaunay triangulation of a sufficiently dense point sample of a surface is isotopic to a *collar extension*. Although this approach is able to deal with boundaries and even allows reconstruction of non-orientable surfaces, it cannot handle noisy data.

Isotopic approximation in the context of implicit curves and surfaces has also been studied.

Lately, Burr *et al.* [2012]; Lin and Yap [2009] introduced new algorithms by combining the parametrizability (as in Snyder [1992]) and the nonlocal isotopy (as in Plantinga and Vegter [2004]).

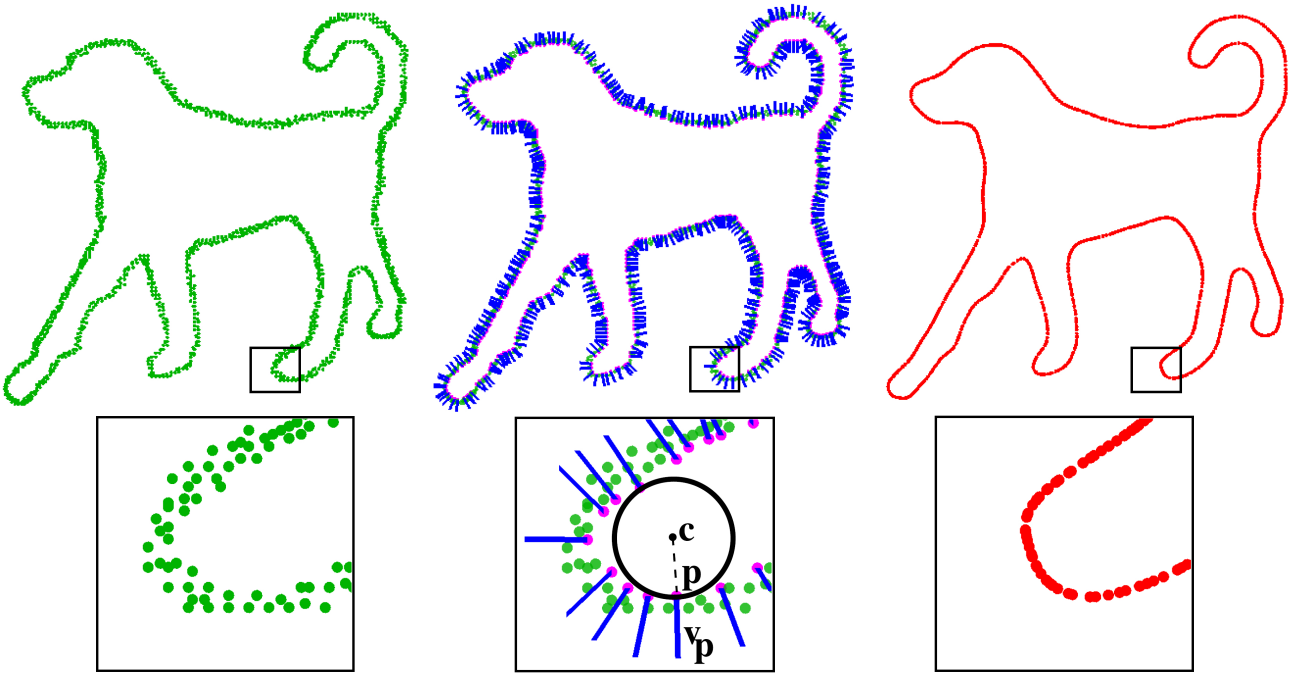


Figure 1.12: Adaptive moving least squares surface. From left to right: To handle noisy datasets, outward normals are estimated from big Delaunay balls at a subset of sample points and then the points are projected using these normals. *(Image taken from Dey and Sun [2005])*

Discussion. Most of the algorithms discussed above cannot handle noisy data and, in fact, none can handle *outliers*. Although some of them provide isotopic guarantee, they fail to provide an approximation with a tradeoff between complexity and quality. All these methods generate isotropic meshes, overly complex, which would require another algorithm for simplification with geometric guarantees.

1.2.2 Other Reconstruction/Mesh-Repair Algorithms

Apart from the algorithms discussed above, there are many reconstruction algorithms that do not provide theoretical guarantees over the topology but we highlight them to complete the state of the art in the context of this thesis.

Heterogeneous Data. Hornung and Kobbelt [2006] used unsigned distance function to avoid the topological noise artifacts caused by misalignment of 3D scans, which are common to most

volumetric reconstruction techniques. The proposed volumetric algorithm estimates local surface confidence values within a dilated crust around the input samples and the surface which maximizes the global confidence is then extracted by computing the minimum cut of a weighted spatial graph structure. the method generates a watertight mesh even for noisy and highly irregular data containing large holes, without loosing fine details in densely sampled regions (Figure 1.13). Lately, Argudo *et al.* [2015] proposed a volumetric method based on bi-harmonic fields. The algorithm operates locally, within an expanded bounding box of each hole, and therefore scales well with the number of holes in a single, complex model.

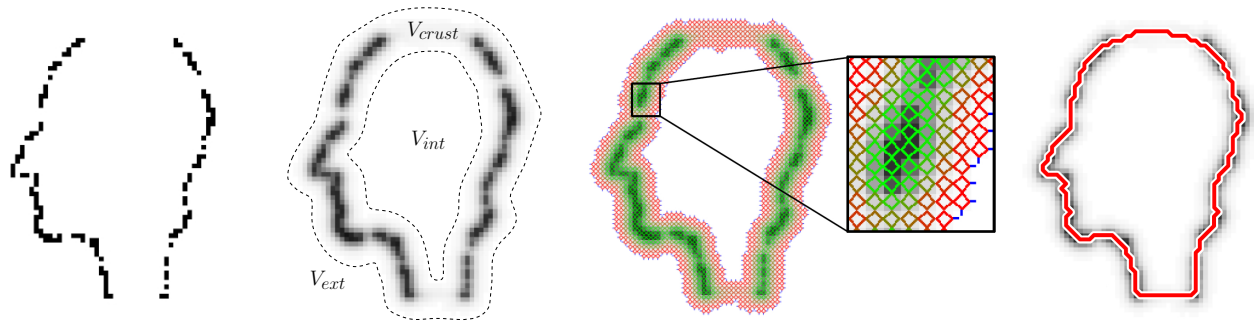


Figure 1.13: Hole filling. From left to right: Given an input point cloud, an unsigned distance function by volumetric diffusion is computed. The output surface is supposed to lie in the voxel crust V_{crust} between the outer and the inner boundary. A spatial graph structure is then embedded within the voxel grid, with small edge weights for high confidence voxels and vice versa. The boundaries are connected to a sink and a source node, respectively. The output surface is finally computed through the min-cut of this graph. Note that the algorithm does not require any information about the local surface orientation. *(Image taken from Hornung and Kobbelt [2006])*

Kazhdan *et al.* [2006] casted the surface reconstruction from oriented points as a spatial Poisson problem. The approach considers all the points at once, without resorting to heuristic spatial partitioning or blending, and hence, is highly resilient to data noise. Recently, Giraudot *et al.* [2013] extended the robust distance function [Chazal *et al.*, 2011] and proposed a noise-adaptive robust distance function and a surface reconstruction algorithm. The resulting algorithm is resilient to variable noise, outliers and missing data as illustrated in Figure 1.14. The algorithm only assumes that the inferred shape is a smooth closed submanifold of known dimension.

Reconstruction and simplification algorithms based on optimal transportation [Goes *et al.*, 2011; Digne *et al.*, 2014] has also been studied as they are robust to both noise and outliers. However, they lack topological guarantees and are too compute intensive to be practical.

Methods involving repairing using a volumetric approach [Ju, 2004] have also been targeted. Other methods involve identifying consistent sub-meshes in the original triangle data which are then merged together by snapping corresponding boundary segments or by stitching

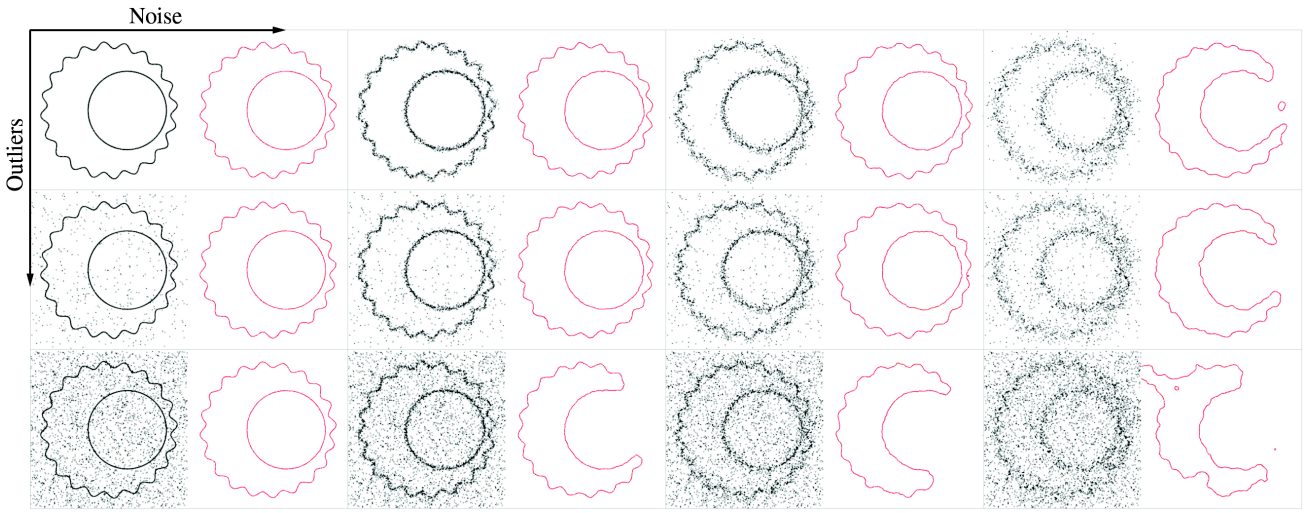


Figure 1.14: Robustness to noise and outliers. Input point set and the reconstructed curve are depicted in black and red respectively. *(Image taken from Giraudot et al. [2013])*

small patches into the remaining gaps and holes [Bohn, 1993; Guéziec *et al.*, 1998; Liepa, 2003; Attene, 2010]. Other local inconsistencies, for example small handles or tunnels can also be repaired [Guskov and Wood, 2001]. We refer the readers to Attene *et al.* [2013] for a comprehensive survey on polygon mesh repairing. Bischoff *et al.* [2005] used a combination of a volumetric geometry representation and the original triangle data in order to exploit the advantages of both (Figure 1.15). Shen *et al.* [2004] introduced a method for building interpolating or approximating implicit surfaces from polygonal data which can preserve sharp features while smoothing noise, generate envelopes around input data and produce mesh with low polygon count.

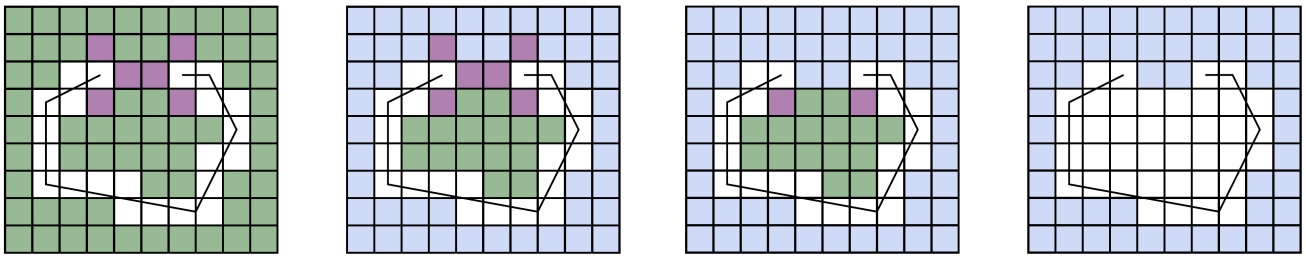


Figure 1.15: Hole-fixing by morphological operations. From left to right: The boundary cells are dilated (magenta) into the empty cells (green) and the outside component (blue) is determined. The outside component is then dilated back into the already dilated cells (magenta). This results in a clean separation of outside (blue) and inside cells (white) from which a surface topology of the output mesh can be deduced. *(Image taken from Bischoff et al. [2005])*

Anisotropic Reconstruction. Instead of a two step process – reconstruction followed by simplification – direct anisotropic reconstruction has been targeted via particle-based meshing [Bossen and Heckbert, 1996; Zhong *et al.*, 2013], Riemannian Voronoi diagrams [Leibon and Letscher,

2000], Centroidal Voronoi Tessellation (CVT) [Du and Wang, 2005; Sun *et al.*, 2011; Richter and Alexa, 2015], Delaunay Triangulation [Borouchaki *et al.*, 1997] and parameterization [Zhong *et al.*, 2014]. Meshing in higher dimensional space has also been explored by embedding in a 6D space and efficiently computing CVT [Lévy and Bonneel, 2013]. Based on Delaunay refinement Boissonnat *et al.* [2008, 2014] made the star around each vertex to be consisting of the triangles that are exactly Delaunay for the given metric. These stars are later stitched together gradually using a refinement algorithm. As a totally different approach, Azernikov and Fischer [2005] proposed a grid-based approach to mesh implicit surfaces according to their curvature tensor. The method uses a deformed grid and produces quad-meshes. Recently, Fu *et al.* [2014] transformed the anisotropic meshing problem into a functional (convex) approximation problem which generalizes optimal Delaunay triangulation [Long and Jin-chao, 2004; Chen *et al.*, 2007].

Discussion. When dealing with imperfect or heterogeneous data, various methods involving repairing, conversion, or reconstruction are designed to generate clean meshes but they do not yield low-polygon-count approximations with bounded error. Moreover, the topological guarantees are missing, omitting a few which provide watertight meshes as output.

Anisotropic reconstruction algorithms are too slow and are not effective enough to provide low-polygon count as compared to other state of the art approaches, for example, mesh simplification algorithms. Furthermore, handling noisy or undersampled data is often out of scope.

1.2.3 Guaranteed Error

Hausdorff. In order to build approximation with bounded error, a large collection of simplification algorithms exist. Approximation with bounded error has been targeted through clustering [Kalvin and Taylor, 1996] and mesh decimation [Bajaj and Schikore, 1996; Cohen *et al.*, 1996; Klein *et al.*, 1996; Guéziec, 1996; Ciampalini *et al.*, 1997; Cignoni *et al.*, 2000; Cohen *et al.*, 2003; Botsch *et al.*, 2004; Ovreiu *et al.*, 2012]. A combination of both, clustering and mesh decimation has also been studied [Zelinka and Garland, 2002], where the tolerance volume was discretized to accelerate the validity tests for the atomic simplification operations. Typically, the methods only check for topology preservation and the normal-flip test [Ronfard and Rossignac, 1996] which only prevents local self-intersection. The simplification algorithms use a distance metric to estimate the error, Garland and Heckbert [1997]; Hoppe [1999] being the popular ones. We further refer the readers to Cignoni *et al.* [1998] and Luebke [2001] for comprehensive surveys on simplification algorithms.

Other Fairness Schemes. In general, the error metric considered is the one-sided Hausdorff distance to the input mesh, but the normal deviation has also been considered [Borouchaki and Frey, 2005]. Although the distance metric is very efficient in measuring geometric errors, it has difficulties in distinguishing important shape features such as high-curvature regions. Measuring geometric error for such features has been targeted via curvature tensor [Coll *et al.*, 2011] and discrete curvature norm [Kim *et al.*, 2002]. Kobbelt *et al.* [1998] generalized the error metric by combining the local surface properties – the approximation error (function value), the local distortion (1st order derivatives) and the local curvature (2nd order derivatives) – into one linear functional. Recently, Morigi and Rucci [2014] presented new approach based on the evolution of surfaces under p-Laplacian operators which provides a natural geometric clustering.

Discussion. All these approaches, in general, need a clean mesh as an input. Furthermore, most of them are not generic enough to handle heterogeneous input data, and they are not designed to guarantee a valid, intersection-free output.

1.2.4 Self-intersection Free

By Prevention. Most of the algorithms provide guarantees for intersection-free output during mesh decimation by building an envelope volume around the surface. Cohen *et al.* [1996] provided a generic genus-preserving simplification algorithm which guarantees a two-sided Hausdorff error bound along with self-intersection prevention and sharp feature preservation. The envelope is created by simply offsetting each vertex of the original surface in the direction of its normal vector to transform the fundamental triangles into those of the envelope. Offset vertices are not allowed to go beyond the Voronoi regions of its adjacent fundamental triangles to avoid self-intersection in within the envelope (Figure 1.16). The original mesh is then decimated within the envelope.

The main disadvantage of the tolerance volume based approaches is that the space needs to be split. As a result, we cannot cross the tolerance even if the operation corresponds to an optimal target location with no self-intersection. Gumhold *et al.* [2003] proposed a method based on preventing and avoiding self-intersections during the mesh decimation. However, searching for the locus of points in space that avoids intersections when applying a decimation operator, and tunneling out of situations where every operator is forbidden is often too labor-intensive to be considered a practical solution.

By Construction. Another class of approaches based on Delaunay filtering and refinement, instead, provide intersection-free approximations by construction. Boissonnat and Oudot [2005]

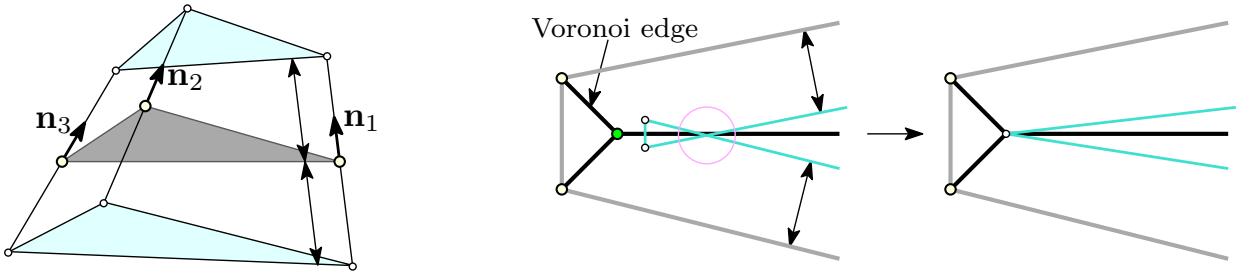


Figure 1.16: Creating envelope using offset vertices. Left: The vertices are offsetted in the normal direction by the desired Hausdorff error to create the Simplification envelope [Cohen *et al.*, 1996]. Right: To avoid self-intersection within the envelope, the offset vertices are bounded by the Voronoi regions. This however, might reduce the level of approximation in certain regions. Note that the boundaries of this envelope will be isotopic to original surface.

presented an algorithm that constructs provably good surface samples and meshes. A notable feature of the algorithm is that the surface needs only to be known through an oracle that, given a line segment, detects whether the segment intersects the surface and, in the affirmative, returns the intersection points making the algorithm useful in a wide variety of contexts and for a large class of surfaces. Note that all the methods discussed above in Section 1.2.1 which provide topological guarantees obviously fall in this catogery.

Discussion. The algorithms which provide self-intersection free guarantee by prevention need a clean mesh to start with. Moreover, reconstructing a clean surface first and then using these algorithms will accumulate the Hausdorff error and the output will not be within the desired bounds. Other algorithms which guarantee intersection-free by construction, unfortunately, generate only isotropic meshes (as we saw before) and do not target very coarse approximations and, as such, cannot be used for shape simplification.

1.3 Positioning

In $\mathbb{R}^3$, Chazal and Cohen-Steiner [2004] showed that when seeking a homeomorphic approximation S' of a connected surface S, a simple topological condition is sufficient to guarantee that the two surfaces are isotopic (Figure 1.17). S and S' are isotopic iff:

1. S' is contained in a *topological thickening* of S and separates the boundary components of this thickening, and,
2. S and S' are homeomorphic.

We contribute next a constructive approach for this theoretical result in the form of an algorithm that matches these conditions in order to ensure that the output surface mesh is an isotopic approximation.

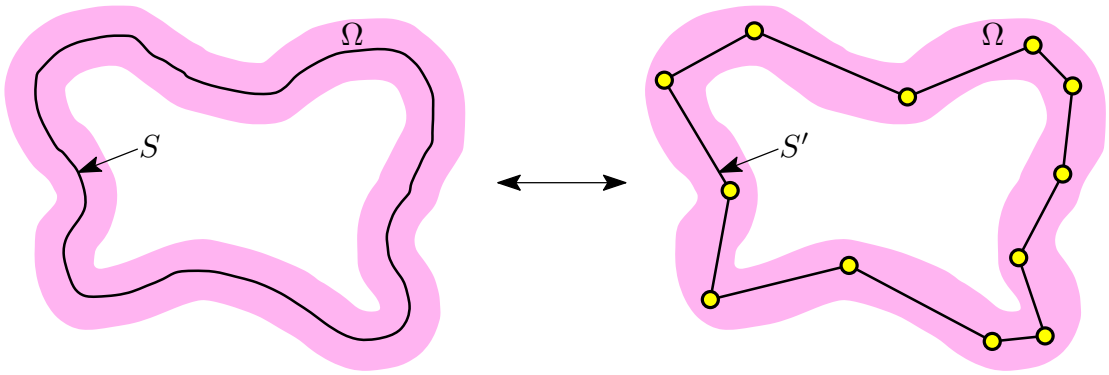


Figure 1.17: Isotopic conditions. Two connected and closed surfaces S and S' are isotopic iff (a) S' is included in a topological thickening of S and separates its boundary components, and (b) the genus of S' does not exceed the genus of S. Note that if S' is closed and separates boundary components of topological thickening of S, then its genus cannot be lower than the genus of S.

1.3.1 Problem Statement

We state the problem as follows. The input is a tolerance volume Ω (Figure 1.17) that is a topological thickening of a surface S which we want to approximate. By topological thickening of S we mean a compact subset of $\mathbb{R}^3$ homeomorphic to $S \times [0, 1]$. Our goal is to generate as output, a surface triangle mesh:

1. located within Ω,
2. isotopic to the boundary components of Ω, and
3. with a low triangle count.

Note that our problem statement contains all the features which were missing from the current state of the art. The first criteria ensures that the approximation error is bounded by the Hausdorff distance of the topological thickening. It also ensures that the output separates the boundary components of Ω. This will later help us in providing theoretical guarantees over its topology (isotopy). Although we have explicitly mentioned only Hausdorff distance in the problem statement, other fairness schemes, for example, normal deviation (as discussed in Section 3.1) can always be added. Secondly, by definition, the boundary components of a topological thickening are always isotopic to the original surface. Furthermore, assuming that the input is provided in the form of tolerance volume ensures that original surface S need not be provided in a specific form, and hence, our algorithm can handle heterogeneous data. The second criteria also ensures that our output is free from self-intersections. Finally, the third criteria ensures that we solve the mesh simplification problem (Section 1.1.2) and the output size is as low as possible.

This approximation problem was originally stated by Klee for polytopes in arbitrary dimensions. In 2D, the problem is commonly referred to as the *minimum nested polygon* problem, and

has been investigated well [Aggarwal *et al.*, 1985]. The 3D instance of this problem, referred to as *minimum nested polyhedron* problem has been shown to be NP-hard [Agarwal and Suri, 1998].

1.3.2 Discussion

Despite being a long standing problem, there is still no robust and practical solution to this enduring scientific challenge. Yet, it is both relevant to, and timely for, the increasing variety of industrial applications that involve raw geometric data. In this part of the thesis, we develop an algorithm for the above problem that yields approximations with very low triangle count, while enjoying topological guarantees under relatively mild assumptions on the tolerance volume.

Note that while the assumption that Ω is a proper thickening makes the analysis easier, it is not always necessary and our approach may also work when boundary components of Ω have, for instance, additional spurious handles. We also extend our algorithm to non-closed and non-manifold surfaces. If Ω is not provided as input, we may generate it from a possibly defect-laden approximation of S (Σ, *e.g.*, a point cloud or a polygon soup) using either simple offsets in the noise-free case, or sublevel sets of a robust distance function (e.g. Chazal *et al.* [2011]). Hence, under relatively mild conditions, our algorithm is able to solve the problem of robust reconstruction, repair and simplification concurrently.

Chapter 2

Algorithm and Guarantees

2.1 Algorithm Overview

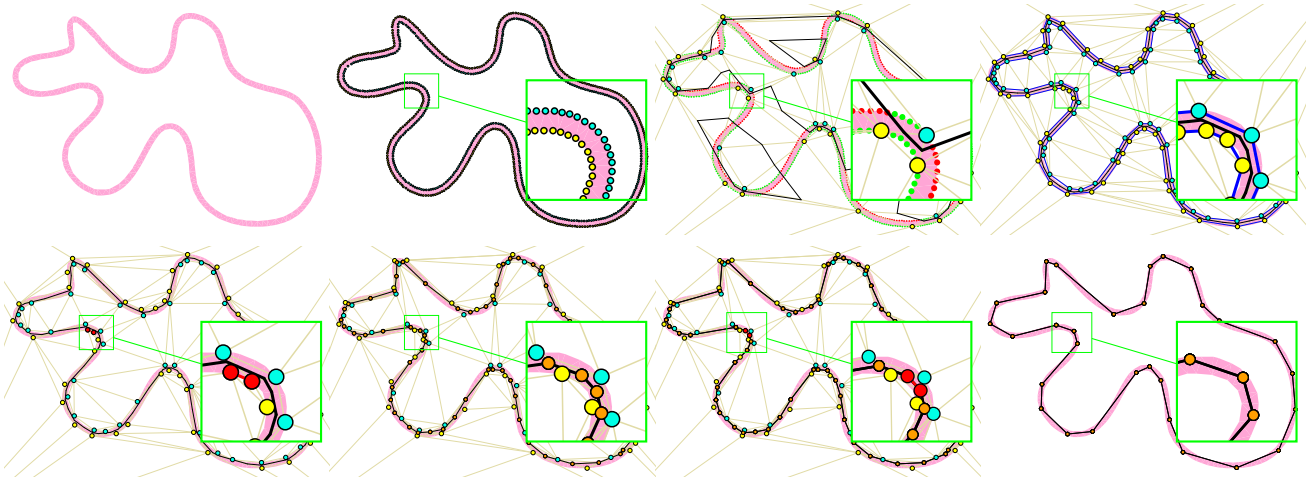


Figure 2.1: Overview of our algorithm. Top: input tolerance Ω, sampling of $\partial\Omega$, mesh refinement by inserting a subset of the sample points, and topology condition met. Samples that are well classified are depicted in green, and in red otherwise. The boundary of the simplicial tolerance volume $\partial\Gamma$ is depicted with blue edges. Bottom: simplification of $\partial\Gamma$, mutual tessellation of zero-set, simplification of zero-set, and final output.

Figure 2.1 depicts the three main steps of our approach: First, the initialization step generates a dense point sample $\mathcal{S}$ on the boundary of the tolerance volume $\partial\Omega$. Second, we proceed *coarse-to-fine* through refinement of a 3D Delaunay triangulation by inserting one sample of $\mathcal{S}$ at a time, and while maintaining a piecewise-linear function interpolated on the triangulation. The function value at the triangulation vertices is set in accordance to the index of each boundary component $\partial\Omega_i$ ($+1$ or -1). The term zero-set refers to the isosurface where the interpolated function evaluates to zero. Refinement is performed until the zero-set is entirely contained into Ω and

matches the topology of Ω. All samples are then well classified, and the tolerance volume is approximated by Γ, referred to as the simplicial tolerance volume. Third, we proceed mainly *fine-to-coarse* through simplifying Γ, inserting the zero-set into Γ via mutual tessellation, and simplifying the zero-set while preserving the validity of the embedding.

2.2 Initialization

For initialization, we generate a σ-dense set $\mathcal{S}$ sampled on the tolerance boundary $\partial\Omega$, σ being typically set to a fixed fraction of the minimum separation δ between the $\partial\Omega_i$. That is, the balls of radius σ centered on $\mathcal{S}$ cover $\partial\Omega$.

For the base algorithm we assume that $\partial\Omega$ has only two components $\partial\Omega_1$ and $\partial\Omega_2$. We assign to each sample s of $\mathcal{S}$ a function value: $\mathcal{F}(s) = +1$ if $s \in \partial\Omega_1$, and $\mathcal{F}(s) = -1$ if $s \in \partial\Omega_2$.

We then construct an initial 3D Delaunay triangulation ($\mathcal{T}$) with the eight corners of a loose bounding box of $\mathcal{S}$. We assign to these eight vertices the same function value as that of the samples of the outer boundary of Ω. We maintain a piecewise-linear function f interpolated on $\mathcal{T}$, and its zero-set, denoted by $\mathcal{Z}$.

At each sample point $s \in \mathcal{S}$ we define an error $\epsilon(s)$:

$$\epsilon(s) = |\mathcal{F}(s) - f(s)|, \tag{2.1}$$

where $f(s)$ denotes the interpolated function at s calculated using the function value $\mathcal{F}$ of the vertices of the tetrahedron containing s. Each sample point $s \in \mathcal{S}$ is classified as bad if $\epsilon(s) \geq 1$, and as good (or well classified) otherwise.

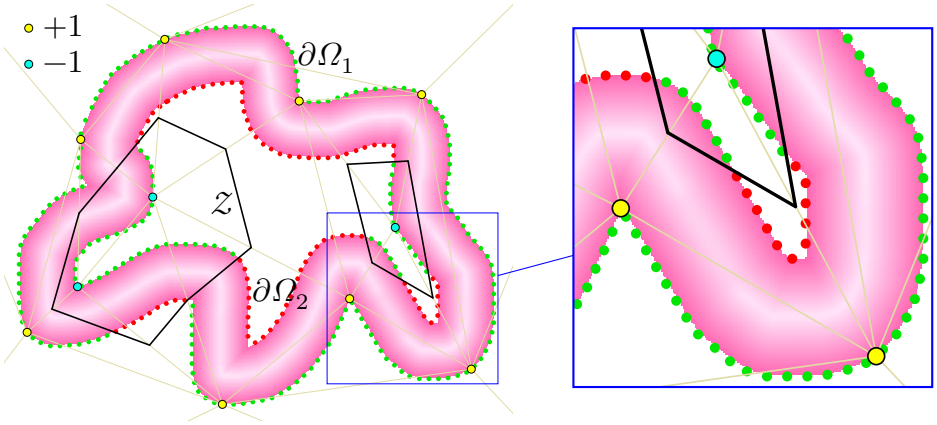


Figure 2.2: Classification of $\mathcal{S}$. The black solid edges depict the zero-set $\mathcal{Z}$ of f. A sample classified as good is depicted in green, and in red otherwise.

During refinement of $\mathcal{T}$ with a subset of $\mathcal{S}$ (described next), the classification of $\mathcal{S}$ provides

us with a means to detect when $\mathcal{Z}$ lies within Ω, with a safety margin (defined in Section 2.6). Figure 2.2 illustrates in 2D a refinement on $\mathcal{T}$, the corresponding zero-set $\mathcal{Z}$ and the classification of $\mathcal{S}$.

2.3 Refinement

We now refine the triangulation $\mathcal{T}$ through inserting Steiner points selected from $\mathcal{S}$, until the correct topology is met. More specifically, we insert one sample point at a time into $\mathcal{T}$ and update the Delaunay property, until $\mathcal{Z}$ classifies all samples of $\mathcal{S}$ as good, or equivalently, until $\mathcal{Z}$ separates the boundaries $\partial\Omega_i$ of Ω.

Greedily inserting the sample s with maximum error at each step is a natural idea for achieving the above goal with few samples. For each tetrahedron we maintain a list of sample points ($\subset \mathcal{S}$) contained in it, and a global modifiable priority queue during refinement with the maximum error points of these tetrahedra. Figure 2.3 illustrates several steps of a refinement sequence in 2D, until complete classification of $\mathcal{S}$.

Unfortunately, the above basic refinement algorithm is not sufficient for at least two reasons.

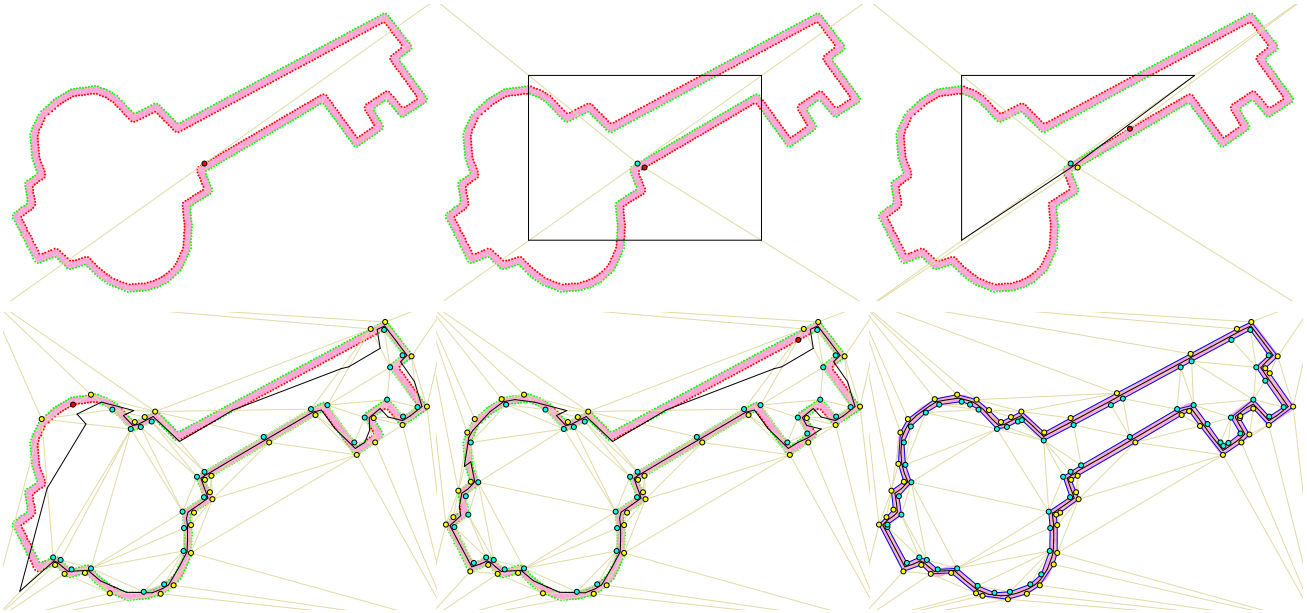


Figure 2.3: Refinement of $\mathcal{T}$. Top: initial triangulation and one Steiner point inserted. Middle: more Steiner points inserted. Bottom: more Steiner points inserted, and complete classification of samples. The zero-set is depicted with black solid edges. Samples classified as good are depicted in green, and in red otherwise. A Steiner point to be inserted at the next iteration is depicted in red. Upon termination the edges of $\partial\Gamma$ are depicted in blue.

The first reason relates to the fact that we are dealing with a finite sample of $\partial\Omega$. Even if all sample points end up being well classified, this still leaves the possibility that $\mathcal{Z}$ crosses

$\partial\Omega$ in-between the samples. To prevent this from happening, we enforce that all samples are well classified with an α margin, as well as an upper bound on the Lipschitz constant of the piecewise-linear function.

The second reason relates to the quality of normals. In certain configurations (*e.g.*, Figure 2.4), the normal directions are grossly wrong even in locally smooth areas. To alleviate this issue we detect so-called misoriented tetrahedra by checking that the piecewise linear function they define is locally well adapted to the geometry of Ω (condition 3 below). We note that this condition is not required for the topological correctness of the algorithm.

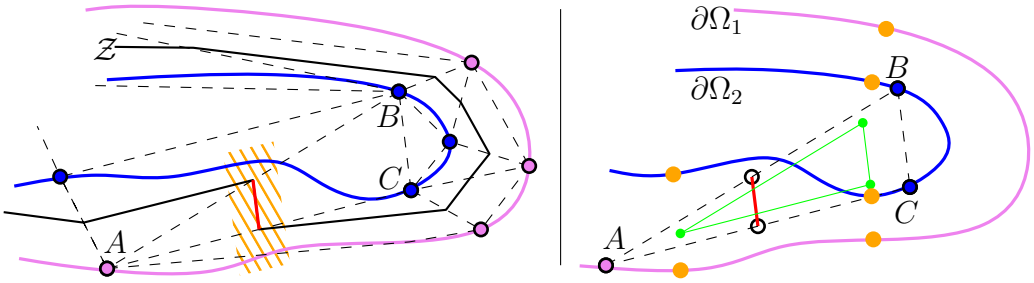


Figure 2.4: Misoriented element. Left: The edges of $\mathcal{Z}$ are depicted with solid black lines. The zero-set of $\triangle ABC$ (red) has an incorrect normal. Right: The piecewise-linear function defined on $\triangle ABC$ should classify well the samples of $\mathcal{S}$ (on both $\partial\Omega_i$ forming $\triangle ABC$) which are nearest (orange) to the vertices of a shrunk triangle (green).

Our modified refinement algorithm iteratively refines the triangulation until all the following criteria are met in order:

1. For some given $0 < \alpha < 1 : \forall s \in \mathcal{S}, \epsilon(s) \leq 1 - \alpha$ (α is set to 0.2 in all experiments).
2. The *height* of every tetrahedron contributing to $\mathcal{Z}$ is at least $2\sigma/\alpha$. The height is defined as the distance between the supporting lines or planes of the maximal faces with different labels (Figure 2.5).
3. The piecewise-linear function defined by each tetrahedron t classifies well the samples of $\mathcal{S}$ on both $\partial\Omega_i$ that are nearest to the vertices of a shrunk copy of t. The size of this shrunk copy is set to 70% of the size of t in all experiments.

The term "in order" herein means that at each iteration, we look at the first condition that is violated and attempt to satisfy it by inserting a Steiner point as described below. If the condition is not satisfied after exhausting all candidate Steiner points, we move to the next condition.

Since we are dealing with a σ-dense sample, for an α margin (condition 1), an upper bound of α/σ on the Lipschitz constant of the piecewise-linear function suffices to ensure that the zero-set does not cross $\partial\Omega$. Noticing that the Lipschitz constant is nothing but twice the inverse height of a tetrahedron, we get an easy-to-check criterion (condition 2). The first criterion is met by adding the sample point with maximum error while the two other criteria are met by adding the

sample point nearest to the circumcenter of a bad tetrahedron.

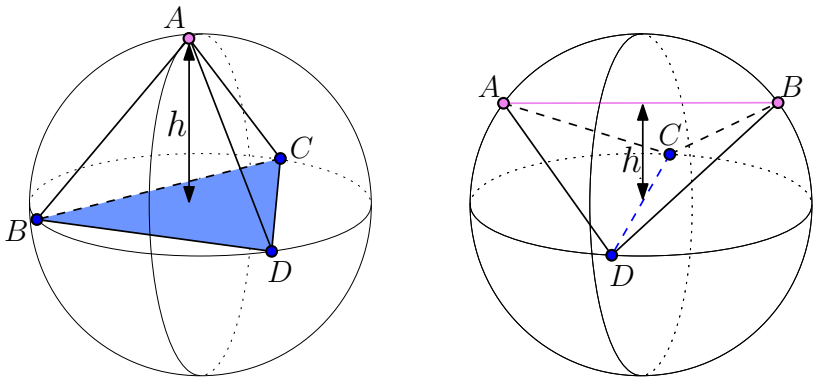


Figure 2.5: Height of a tetrahedron contributing to $\mathcal{Z}$. The height is defined as the distance between the supporting primitive of the maximum dimension simplices formed by the tetrahedron vertices with common labels. Left: distance between point A and supporting plane of $\triangle BCD$. Right: distance between supporting lines of edges AB and CD.

The full refinement algorithm incorporates one more condition (condition 4): while the output $\mathcal{Z}$ of the above algorithm does not have the expected genus, we refine the heterogeneous tetrahedron with the largest circumradius by adding the sample point closest to its circumcenter. This additional layer is needed to get topological guarantees on the result. However, in practice, we did not encounter a single case where the genus was not correct after the first iteration. Also note that the circumradius criterion for refining tetrahedra is blind in the sense that it does not necessarily refines the mesh where topological defects are present. While it is possible to improve the criterion from this point of view, we did not pursue this goal since it has no practical relevance.

Upon termination of the refinement step, the union of all tetrahedra of $\mathcal{T}$ which contribute to $\mathcal{Z}$, bound a *simplicial tolerance* volume (Γ), seen as an approximation of Ω. The boundary facets of Γ are denoted by $\partial\Gamma$.

2.4 Simplification

The zero-set $\mathcal{Z}$ is now topologically correct. In the simplification step we reduce its complexity via decimation of $\mathcal{T}$ combined with a mutual tessellation with $\mathcal{Z}$. Note that we stop enforcing that $\mathcal{T}$ is a Delaunay triangulation, which allows for increasingly anisotropic triangulations.

Simplification is achieved through performing a series of edge-collapse operators on $\mathcal{T}$. These operators are made conservative to preserve a valid triangulation $\mathcal{T}$, the classification of $\mathcal{S}$ and the normals achieved in previous step.

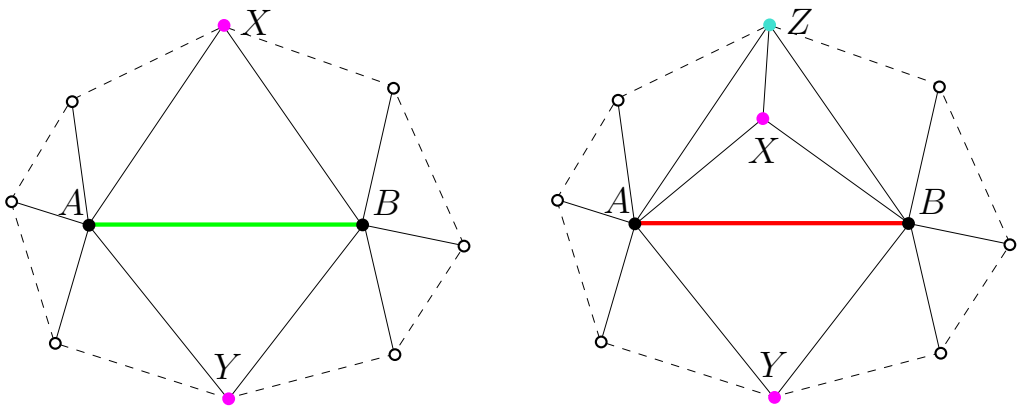


Figure 2.6: Link condition in 2D. Left: the edge AB is collapsible as $Lk(A) \cap Lk(B) = Lk(AB)$. Right: the edge AB is not collapsible as $Lk(A) \cap Lk(B) \neq Lk(AB)$.

The validity of $\mathcal{T}$ requires checking for two conditions. The combinatorial topology of $\mathcal{T}$ is preserved via the *link condition* [Dey *et al.*, 1998] (Figure 2.6).

The valid embedding of $\mathcal{T}$ is preserved by computing the visibility kernel $(\mathcal{K}_{\mathcal{T}})(PQ)$ of the polyhedron formed by the one-ring of the edge PQ (Figure 2.7). If the visibility kernel is non empty then locating the target vertex into this kernel preserves a valid embedding.

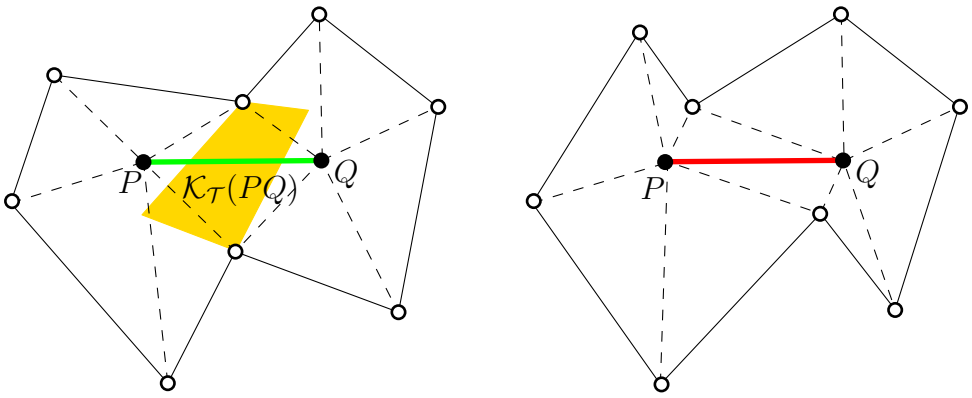


Figure 2.7: Visibility kernel condition in 2D. Left: the edge PQ is collapsible and a valid embedding is preserved when the target vertex is located within the kernel $\mathcal{K}_{\mathcal{T}}(PQ)$ (orange) of the polygon formed by the one-ring of the edge. Right: the kernel is empty and hence the edge PQ is not collapsible.

Preserving the classification of $\mathcal{S}$ requires further restricting the visibility kernel of edge. As this problem is non-convex, we resort to a point sampling of the kernel during the simulation of each edge-collapse operator. To obtain faithful normals locally in a smooth area, we use the same method as before (Figure 2.4) and check in advance whether the final solution is locally well adapted to the geometry of Ω or not.

In order to improve efficiency we always perform simpler halfedge collapse before general edge collapse operators. A halfedge collapse operator locates the target vertex at one of the vertices of the edge. In addition, we adopt a multi-staged decimation approach with the following steps:

1. Collapse edges of $\partial\Gamma$.
2. Mutual Tessellation of $\mathcal{Z}$ into $\mathcal{T}$.
3. Collapse edges of $\mathcal{Z}$.
4. Collapse edges between Γ and $\mathcal{Z}$, which may induce further edge collapses of $\mathcal{Z}$ (previous step).

Intuitively, we perform the steps in increasing order of computational complexity: first the operations with low number and discrete degrees of freedom, then with higher or continuous degrees of freedom. As for other decimation algorithms we need to define an error to sort the operators and to optimize the target vertex placement when performing a general edge collapse operator. In order to preserve fidelity to the initial zero-set, we use as error the sum of square distances between the target vertex and the set of supporting planes of the zero-set facets located in the 2−ring of the collapsed edge. The edge collapse operators are sorted via a priority queue sorted by increasing error.

2.4.1 Simplicial Tolerance

In this step we collapse only a subset of the edges of the simplicial tolerance boundary $\partial\Gamma$. Denote by PQ such an edge (Figure 2.8). We select as target vertex a sample point (1) from $\mathcal{S}$ (2) located within the visibility kernel $\mathcal{K}_{\mathcal{T}}(PQ)$ of PQ, (3) inducing a zero-set that preserves the classification of $\mathcal{S}$ along with normals and (4) that minimizes the aforementioned error.

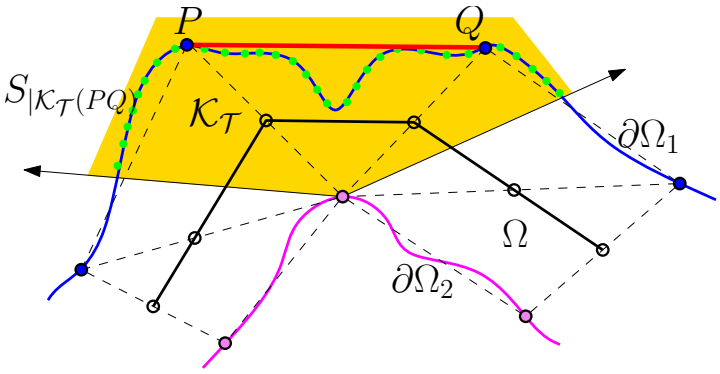


Figure 2.8: Edge PQ of $\partial\Gamma$. PQ is candidate to be collapsed. The edges of $\mathcal{T}$ are depicted with dashed black lines. Visibility Kernel $\mathcal{K}_{\mathcal{T}}(PQ)$ is depicted (partially) in orange. The edges of $\mathcal{Z}$ (black solid lines) are not part of $\mathcal{T}$. The green dots ($S_{|\mathcal{K}_{\mathcal{T}}(PQ)}$) depict the subset of samples from $\mathcal{S}$ located within $\mathcal{K}_{\mathcal{T}}(PQ)$.

Denote by $S_{|\mathcal{K}_{\mathcal{T}}(PQ)}$ the initial set of candidate sample points from $\mathcal{S}$ located within the visibility kernel $\mathcal{K}_{\mathcal{T}}(PQ)$. To avoid exhaustive search, we discard the sample points leading to errors in the classification of $\mathcal{S}$, as located in invalid regions, denoted Ψ. Figure 2.9 illustrates $\Psi = a \cap b \cap m$ where the point with maximum error is chosen only over $\partial\Omega_1$. A similar invalid

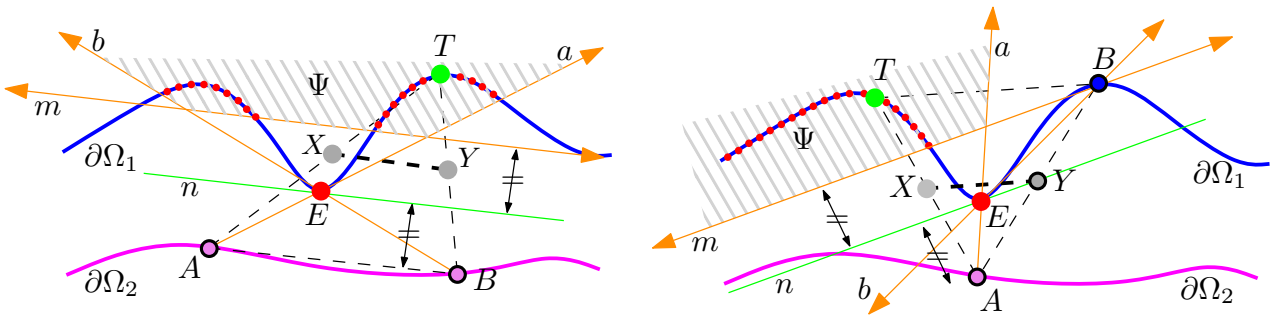


Figure 2.9: Invalid region. Assume an edge of $\partial\Gamma$ is collapsed into the target point T. Line segment XY denotes the zero-set of $\triangle ABT$ after collapse. Line n represents the extreme zero-set of $\triangle ABT$ which preserves the classification of the point with maximum error E. Line m delineates the corresponding locus for T. The intersection of the two half-spaces delineated by a and b represents the locus of T which keeps E within $\triangle ABT$. If T is located in the invalid region Ψ (gray) then the classification of E is not preserved. Left: case where A and B belong to the same $\partial\Omega_i$. Line n is parallel to AB and passes through E. Right: case where A and B belong to two different $\partial\Omega_i$. Notice that Y is fixed and n is the supporting line of EY.

region is computed by considering the point of maximum error on $\partial\Omega_2$ within $\triangle ABT$. We then collapse iteratively all edges of $\partial\Gamma$ to the point which exhibits the minimum error, as discussed above.

2.4.2 Mutual Tessellation

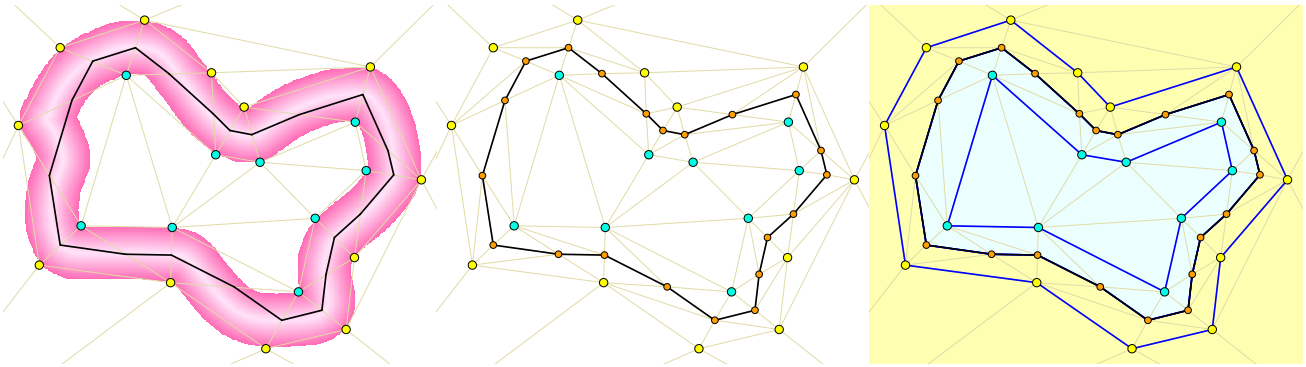


Figure 2.10: Mutual tessellation. Left: before mutual tessellation. Middle: after mutual tessellation. Right: classification of tetrahedra in accordance to $\partial\Omega_i$. The edges of $\mathcal{Z}$ and $\partial\Gamma$ are depicted with solid black and blue lines, respectively.

When no more edges of $\partial\Gamma$ are collapsible, we perform a mutual tessellation between $\mathcal{Z}$ and $\mathcal{T}$ by inserting all vertices and faces of $\mathcal{Z}$ into $\mathcal{T}$. The newly inserted vertices are assigned the function value $\mathcal{F} = 0$. We then label all tetrahedra of $\mathcal{T}$ in accordance to their associated tolerance boundary component $\partial\Omega_i$. This provides us with a means to preserve the classification in the next simplification steps. A sample $s \in \partial\Omega_i$ is constrained to lie within a tetrahedron

with label i. Intuitively, this step implements a transition from a function embedded in a volume mesh of the tolerance, to a surface mesh embedded within the 3D triangulation. Figure 2.10 illustrates such mutual tessellation in 2D.

2.4.3 Zero-set

After mutual tessellation we collapse the edges of $\mathcal{Z}$. Figure 2.11 illustrates the visibility kernel of an edge that preserves a valid embedding upon a collapse operator.

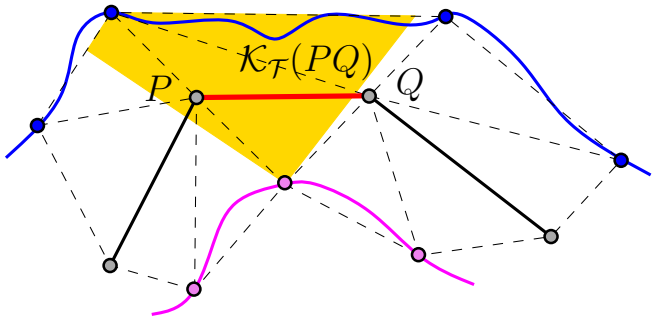


Figure 2.11: Kernel of an edge PQ of $\mathcal{Z}$. PQ is candidate to be collapsed. The edges of $\mathcal{Z}$ are depicted with solid black lines. The visibility kernel $\mathcal{K}_{\mathcal{F}}(PQ)$ of PQ is depicted in orange.

Two important differences with the previous step are that we collapse an edge to an arbitrary target vertex location within the valid area ($\subset \Omega$, that minimizes the aforementioned error), and the target vertex is assigned the function value $\mathcal{F} = 0$.

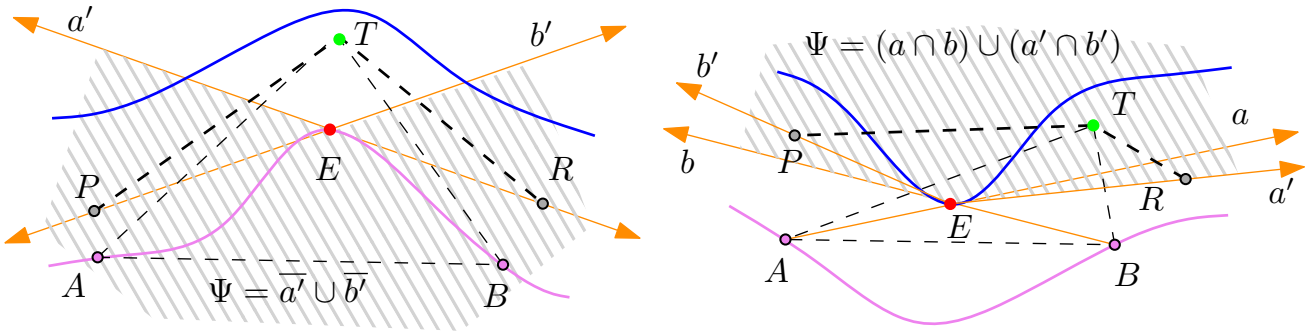


Figure 2.12: Invalid region. Assume an edge of $\mathcal{Z}$ is collapsed into the target point T (PTR represents the zero-set after collapse). The intersection of the two half-spaces delineated by a and b represents the locus of T which keeps E within $\triangle ABT$. Lines a' and b' represent the extreme zero-set originating from E that preserves the classification of point E. If $T \in \Psi$ (gray), the classification of E is not preserved.

To accelerate the computations, we compute invalid regions as described above. Figure 2.12 illustrates the invalid region by considering the point of maximum error on both $\partial\Omega_i$ for a $\triangle ABT$ when it contains one zero-set vertex. The invalid regions Ψ are constructed similarly

when $\triangle ABT$ contains several zero-set vertices. To further reduce the computational time when simulating general edge collapse operators, we use an octree for hierarchical sampling of $\mathcal{K}_{\mathcal{T}}$ to find the best target location and ignore further sampling of $\mathcal{K}_{\mathcal{T}}$ for the octree cells lying inside Ψ. Figure 2.13 provides a summary of the conditions required for a valid edge collapse.

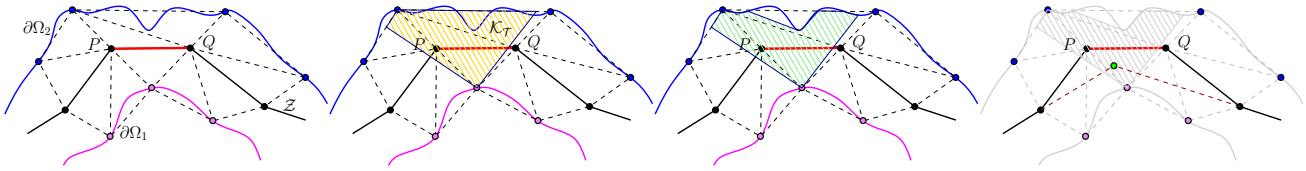


Figure 2.13: Conditions for valid edge collapse. From left to right: To collapse an edge PQ, we check for its link condition and compute the visibility kernel (middle left). Note that, all the points within this visibility kernel may not preserve the classification of sample points. The region (subset of visibility kernel) that preserves the classification of sample points is depicted in green (middle right). Since, finding such a kernel is a non-convex problem, we resort to hierarchical sampling of the visibility kernel to find the target point which preserves classification and minimizes the sum of square distances between the target vertex and the set of supporting planes of the zero-set facets located in the 2-ring of the collapsed edge as depicted in green (right).

2.4.4 All Edges

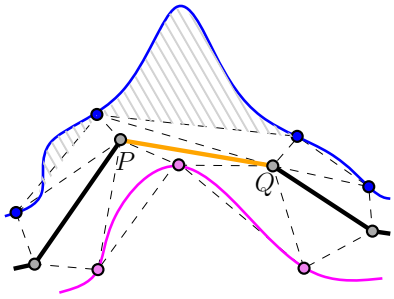


Figure 2.14: Inaccessible regions. The grayed area of Ω is inaccessible due to the restrictions imposed by $\mathcal{K}_{\mathcal{T}}(PQ)$ to the simplicial tolerance.

Due to the simplicial tolerance Γ, there may exist regions in Ω which are inaccessible (see shaded region in Figure 2.14). To make full use of the tolerance volume, we collapse edges between vertices of Γ and $\mathcal{Z}$ (Figure 2.15). It not only helps relocating the zero-set vertices to a better location with respect to the error chosen for ordering the priority queue, but also increases the size of visibility kernel and hence, helps exploring further possibilities of an edge collapse over $\mathcal{Z}$ as discussed in 2.4.3.

Figure 2.16 illustrates all steps of our algorithm on a mechanical part. The input is a raw triangle soup (20k triangles). The tolerance volume is computed as the sub-level [0-0.6] of the

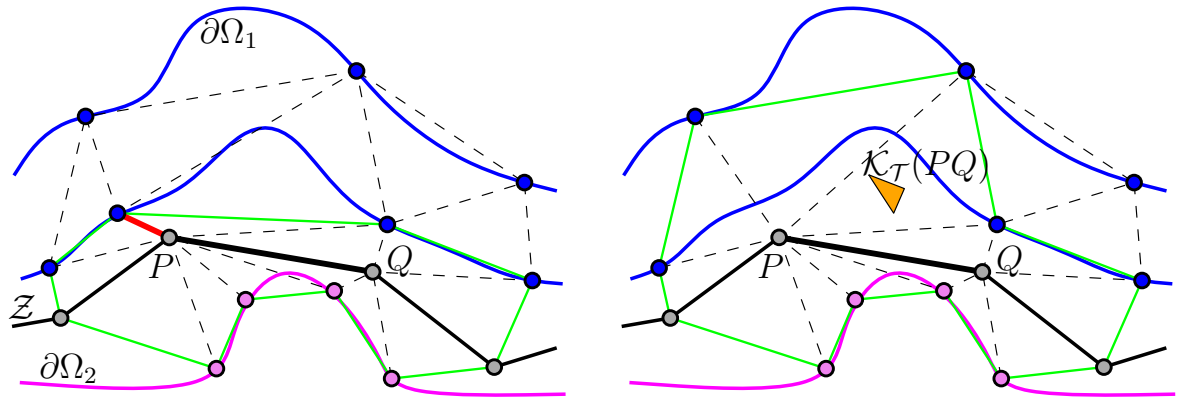


Figure 2.15: Making an edge collapsible. Left: Edge PQ is not collapsible as visibility kernel $\mathcal{K}_{\mathcal{T}}(PQ)$ is empty. Right: Kernel $\mathcal{K}_{\mathcal{T}}(PQ)$ (orange) is not empty after collapsing the red edge shown left. Collapsing an edge between a vertex of Γ and a vertex of $\mathcal{Z}$ tends to increase the area of the one-ring of PQ (green) and hence increases the probability that an edge of $\mathcal{Z}$ is collapsible.

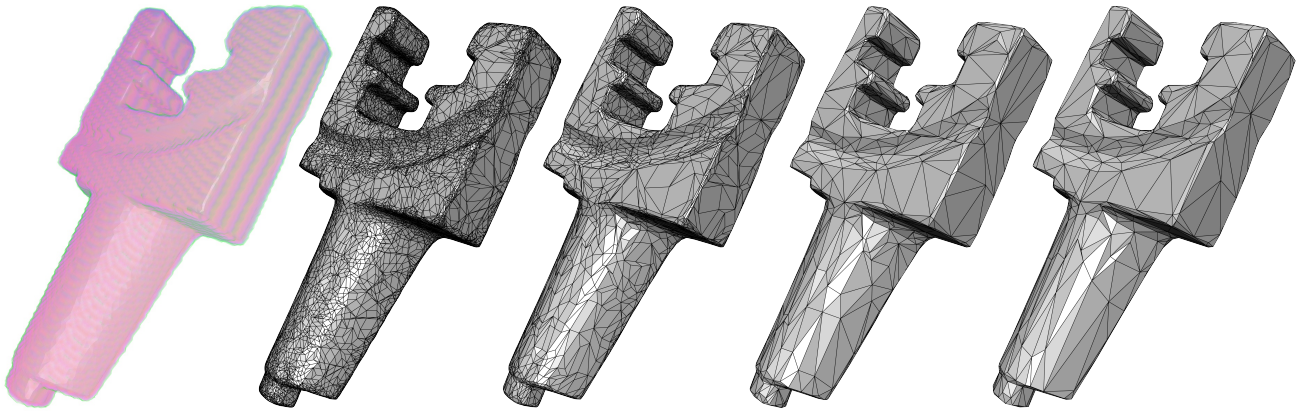


Figure 2.16: Blade. From left to right: Input tolerance ($\delta = 0.6\%$); $\mathcal{Z}$ after refinement ($20.4k$ vertices); simplification of $\partial\Gamma$ ($5.3kv$); mutual tessellation and simplification of $\mathcal{Z}$ ($1.01kv$); and the final output ($752v$).

Euclidean distance function to the input triangle soup. Note that until mutual tessellation the zero-set is made up of triangles and quadrangles before being converted into a pure triangle mesh after mutual tessellation.

2.5 Extensions

The algorithm above can be extended in order to deal with boundaries and non-manifold surfaces. While the guarantees can be extended to non-closed surfaces, for non-manifold cases, they are more difficult to formulate and are probably beyond the reach of existing tools. Still, the output is guaranteed to have the correct homotopy type in these cases also.

2.5.1 Non-closed Surfaces

Our algorithm deals with surfaces with boundaries when Σ is provided as input along with Ω. We first detect sample points corresponding to boundaries, referred to as $\mathcal{B}$, using $\gamma\delta'-$radius balls centered at Σ (Figure 2.17), where γ denotes a user defined parameter derived from the reach of input data. Parameter δ' is defined as the minimum distance between the current center of ball and $\mathcal{S}$.

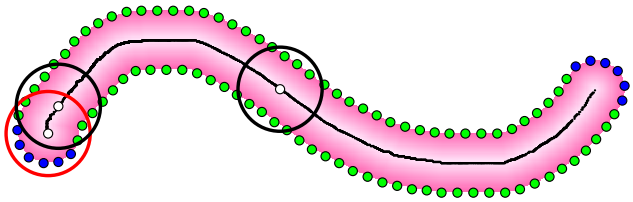


Figure 2.17: Boundary sample points. The intersection between $\mathcal{S}$ and ball located on the input surface Σ is composed of a single connected component near a boundary.

When a ball contains a single connected component (samples connected to each other by paths with a maximum step distance of 2σ) boundary surface of the tolerance volume, then, the associated samples are considered part of $\mathcal{B}$. Conversely, the sample points which correspond to a multi-component surface - with a minimum distance between the components greater or equal to $2\delta'$ - are not considered part of $\mathcal{B}$. Note that when Σ is not provided, balls centered at $\mathcal{S}$ can also be used, but this severely limits the reach size of the input data that can be dealt with. Once the boundary is detected, we use the set $\mathcal{S}$ $\mathcal{B}$ as the set of sample points in the initialization stage. In other words, we ignore the classification of $\mathcal{B}$ with respect to f. Via refinement as described in Section 2.3, we then classify all sample points of $\mathcal{S}$ $\mathcal{B}$ and clip the zero-set by Ω. We enforce during the simplification step that the two-sided Hausdorff distance between boundary of $\mathcal{Z}$ and $\mathcal{B}$ is at most δ. Furthermore, in order to preserve smoothness along the boundary, we use (in this last step besides Hausdorff distance) an extra error term defined as the sum of squared distances between the target vertex and the set of supporting boundary edges of the zero-set located in the 2−ring of the edge to be collapsed.

Figure 2.18 depicts a range scan of the *Kitten* point cloud with boundaries due to missing data. The holes are preserved by the non-closed variant of our algorithm (middle). Note that by not ignoring the parts of the zero-set outside Ω, we can also fill the large hole on the nearly flat area. Nevertheless, more work is needed to reliably deal with more complicated holes.

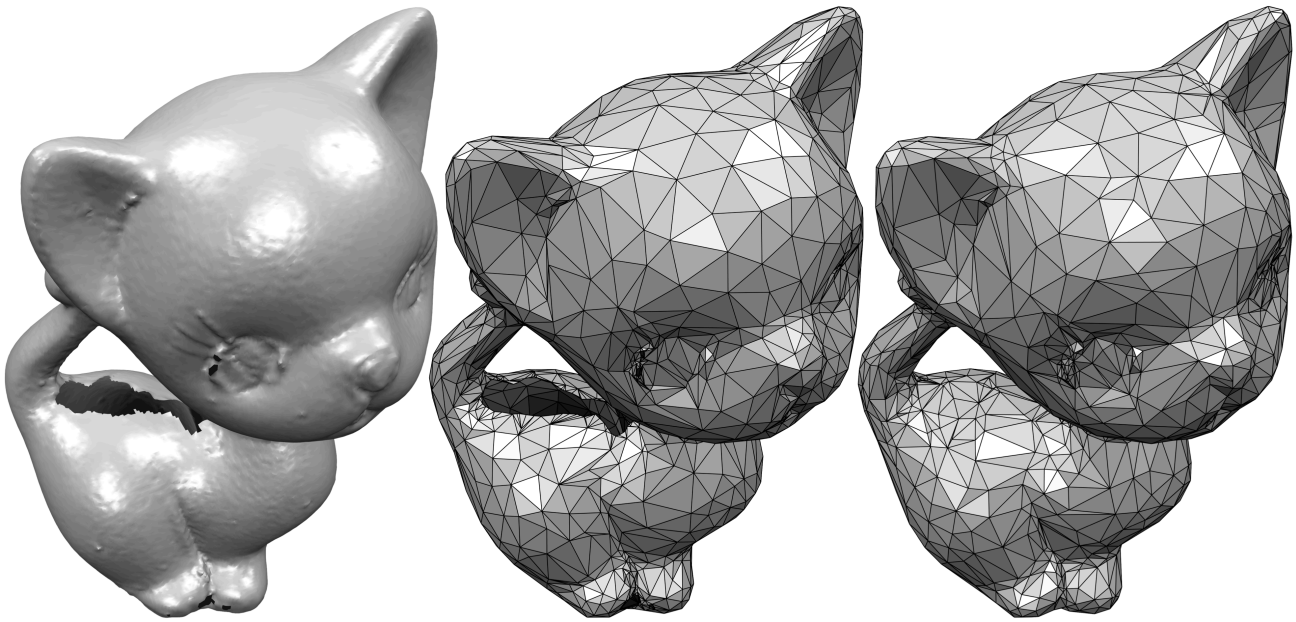


Figure 2.18: Preserving or repairing holes. Left: Range scan of a kitten. Middle: Our output as non-closed surface ($1.3k$ vertices). Right: Our output with holes filled ($1.2k$ vertices).

2.5.2 Non-manifold Surfaces

To handle non-manifold surfaces when computing the error of a sample, we evaluate f with respect to each component of $\partial\Omega$. More specifically, to evaluate the error ϵ at a sample $s \in \partial\Omega_i$ we define

$$\forall p \in \mathcal{S}, \mathcal{F}(p) = \begin{cases} +1, & p \in \partial\Omega_i \\ -1, & p \notin \partial\Omega_i. \end{cases} \tag{2.2}$$

The zero-set is ignored when its end points lie outside Ω; this configuration occurs when the input geometry is made up of several components (Figure 2.19).

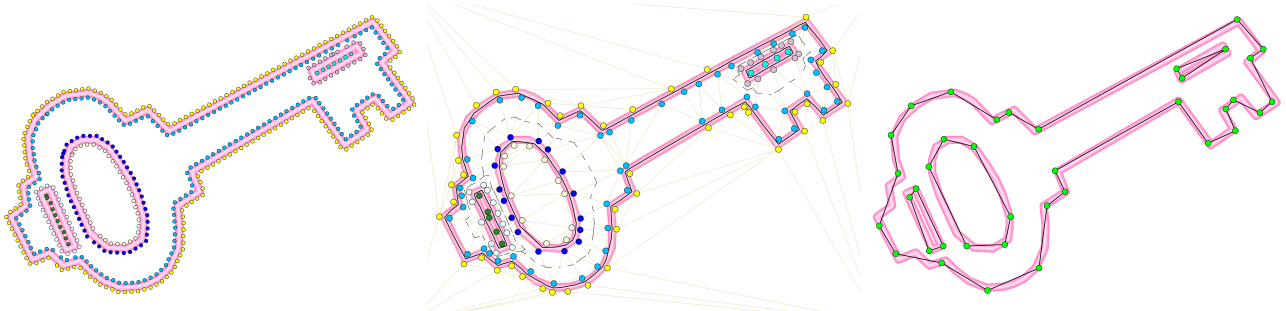


Figure 2.19: Several Components. From left to right: sampling of $\partial\Omega$, refinement until matching the topology and final output. The zero-set outside tolerance volume is depicted with dashed black lines and will be ignored.

The process described above yields a zero-set (Figure 2.20, top left) with topological artifacts

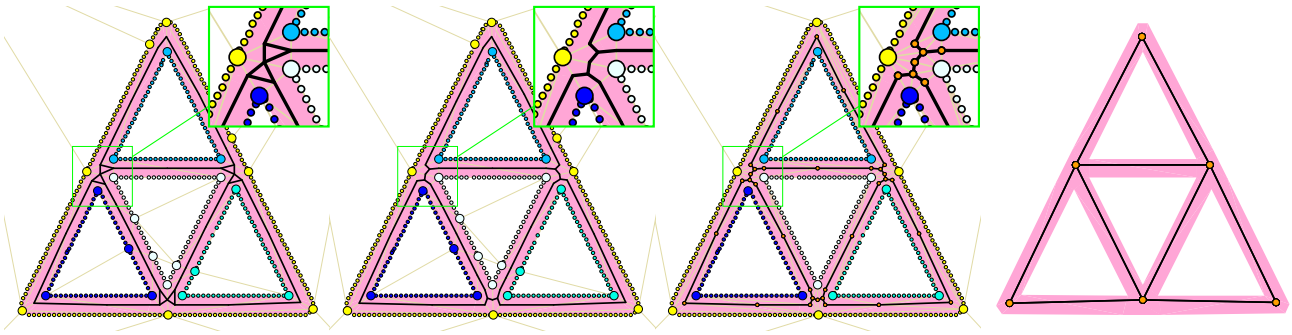


Figure 2.20: Dealing with a non-manifold geometry. From left to right: refinement until matching the topology, mutual tessellation and final output.

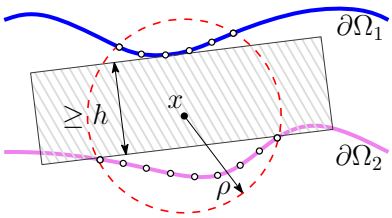
where several surfaces of $\partial\Omega$ meet. These artifacts are located inside tetrahedra containing vertices from three or more $\partial\Omega_i$. In addition, such tetrahedron may contain several zero-sets corresponding to the total number of possible permutations when assigning function values to its vertices. We remove these artifacts by joining all zero-set edges to the centroid of the zero-set vertices located on the edges of this tetrahedron. Figure 2.20 illustrates our algorithm at work on a non-manifold geometry.

2.6 Guarantees

We first derive geometric conditions under which the first three conditions of the refinement algorithm are met upon termination. Denote by ε the radius of the largest ball that can fit within Ω, and by δ the minimum separation between the two boundary components of Ω.

Condition 1 (2.3-1) is necessarily satisfied at the end of the refinement process since any sample not meeting the condition will be added to the triangulation.

Assume now that Condition 2 (2.3-2) is not satisfied upon termination. This means that there exists a heterogeneous tetrahedron t with height lower than $2\sigma/\alpha$. Denote by B its circumscribed ball and r its radius. Ball B cannot contain any sample point from $\mathcal{S}$, else that sample would have been added by the algorithm. Since $\mathcal{S}$ is a σ-sample of $\partial\Omega$, we get that the shrunk ball $B^{-\sigma}$ does not intersect $\partial\Omega$. Hence it is either empty, within Ω, or outside Ω. In the first case, $r \leq \sigma$. In the second case, we have that $r - \sigma \leq \varepsilon$. In the third case, because t is heterogeneous, B meets both boundary components of Ω, hence $\sigma \geq \delta$. As a partial conclusion, upon termination, and assuming $\sigma < \delta$, the circumradius r of a tetrahedron violating condition 2 cannot exceed $\varepsilon + \sigma$.



We now formulate our condition. Given two subsets A and B of $\mathbb{R}^3$, define the *margin* of (A, B) to be the maximum thickness of a slab separating A and B. If no such slab exists then the margin is set to zero. We say that a tolerance volume Ω is (ρ, h)-separated if for all $x \in \mathbb{R}^3$, the margin of $(\partial\Omega_1 \cap B(x, \rho), \partial\Omega_2 \cap B(x, \rho))$ is at least h.

From the above discussion, if $\sigma < \delta$, and if Ω is $(\varepsilon + \sigma, 2\sigma/\alpha)$-separated, condition 2 will be satisfied at the end of the algorithm. Similarly, condition 3 (2.3-3) will ultimately hold assuming a stronger local separation assumption on Ω. However, since this condition is not essential for the topological correctness of our algorithm, we do not elaborate further on expliciting the required separation constants.

Finally, concerning condition 4, we note for future reference that if the correct genus is not met upon termination, we can bound the circumradius r of any heterogeneous element as above. That is, assuming $\sigma < \delta$, we have that $r \leq \varepsilon + \sigma$.

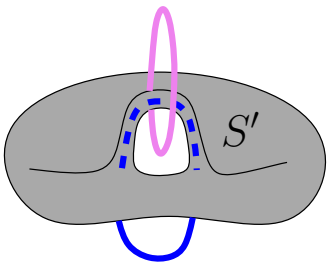
Theorem 2.6.1. *Let κ be such that for any two points x and y in $\partial\Omega_i$ at distance at most $2(\varepsilon + \sigma)$, the geodesic distance between x and y is at most κ times their Euclidean distance. Assuming Ω is a $((5 + \kappa)(\varepsilon + \sigma)/2, 2\sigma/\alpha)$-separated topological thickening of a surface S, the output of the algorithm is isotopic to S.*

Proof. Our proof for isotopy utilizes the same criterion as Chazal and Cohen-Steiner [2004]. This result states that two connected 3-dimensional surfaces S' and S are isotopic if:

1. S' is included in a topological thickening of S and separates its boundary components.
2. S' is connected and its genus does not exceed the genus of S.

In our setting, we take Ω to be a topological thickening of S. Being the zero-set of a piecewise-linear function, $\mathcal{Z}$ is a 2-manifold. Because condition 2 of the base algorithm is satisfied, all points in $\partial\Omega$ are well classified, meaning that $\mathcal{Z}$ is contained inside Ω and separates its boundary components.

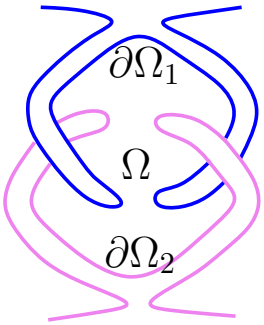
It remains to show that the second condition holds when the refinement algorithm terminates. First, because S' is a zero-set of a piecewise linear function, any component of S' must enclose a vertex of the Delaunay triangulation. As there are no vertices in the interior of Ω, these components must induce non-zero homology classes in Ω. These components are included in the simplicial tolerance, which is fibered by line segments where the piecewise linear function is monotone. This implies that S' is connected.



Now assume that the genus of S' exceeds the one of S. This means that S' contains a spurious handle. Because sub and superlevel sets of piecewise linear functions are homotopy equivalent to subcomplexes of the background triangulation, we get that the two components of $\Omega \backslash S'$ contain linked homogeneous polygonal cycles in the Delaunay triangulation. For each edge in these polygonal cycles, we may form an elementary cycle by stitching the edge with a geodesic shortest path drawn on the appropriate boundary component of Ω and joining the two endpoints of the edge. Because the two polygonal cycles are linked, there must be two linked elementary cycles with different labels.

Assume the correct genus is not met upon termination of the algorithm. Then we may assume that the Delaunay edges in the linked polygonal cycles are edges of heterogeneous tetrahedra. Hence their length is at most $2(\varepsilon + \sigma)$. Hence the length of the elementary cycles are at most $2(1+\kappa)(\varepsilon+\sigma)$. Because two of them are linked, the tolerance volume Ω cannot be $((5+\kappa)(\varepsilon+\sigma)/2, 0)$-separated (Appendix A.0.1). □

The above conditions are clearly met when the tolerance volume is a sufficiently small offset of a smooth surface, for example. Also, the algorithm can also be shown to work in situations not covered by the above theorem, as for instance tolerances bounded by convex surfaces.



The numerical constants in the theorem may be further optimized, and other types of conditions, *e.g.* based on the separation δ can also be proved to be sufficient. In particular, if $2(\varepsilon+\sigma)\sqrt{\kappa^2-1} < \delta$, the algorithm is correct. The inset figure depicts an example of tolerance volume where the algorithm would fail. It is apparent from the proof above that such configurations are essentially the only way the algorithm can fail. Note that even in such situations, the output of the algorithm will be a manifold surface, albeit possibly with a too large genus.

Finally, we note that if the only pursued goal was to provide topologically guaranteed output for tolerances that are topological thickenings, then a trivial solution would be to output one of the tolerance boundary surface. However, such methods would be limited to topological thickenings, while our approach may work in less favorable situations. In addition, algorithms inspired by the trivial idea above do not seem to allow successful subsequent simplifications, even in favorable cases.

Chapter 3

Results

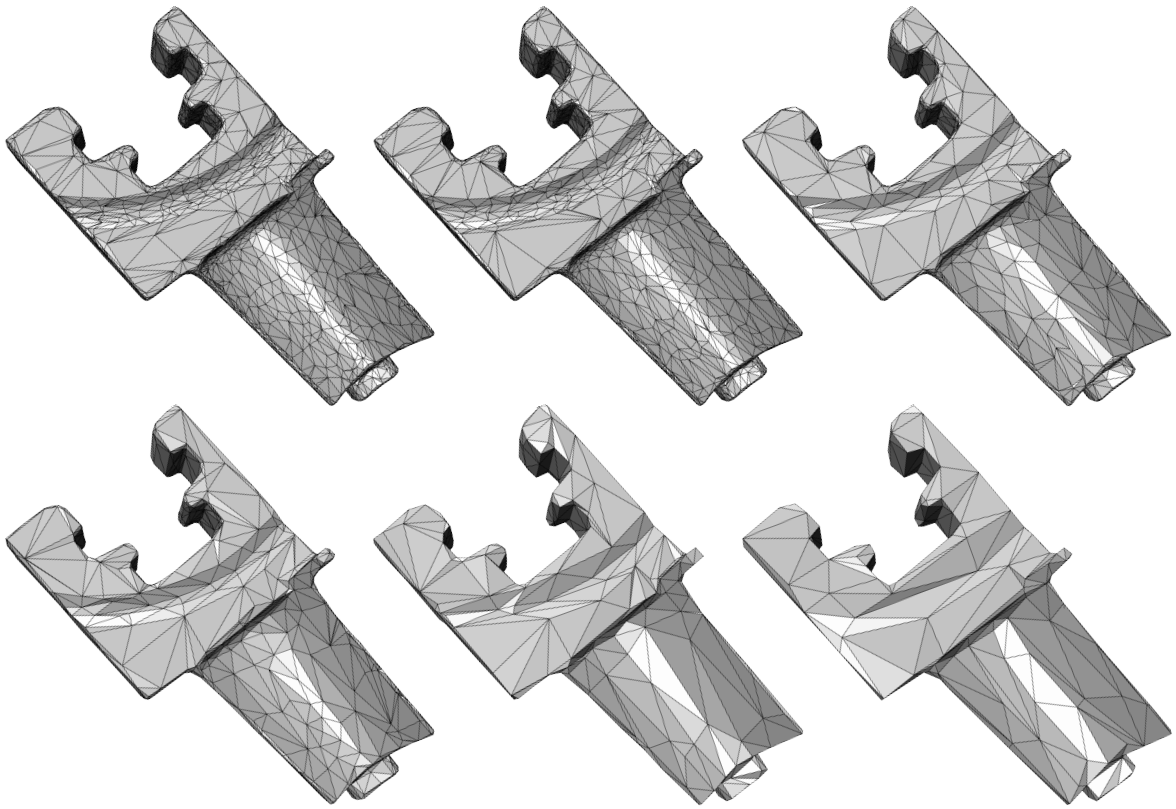


Figure 3.1: Blade. From top left to bottom right: δ is set to 0.15, 0.2, 0.35, 0.9, 1.3 and 1.5%. The final vertex count are $3,020$, 2435, $1,015$, 493, 364 and 254, respectively.

Our algorithm is implemented in C++ using the CGAL library for the triangulation data structures and the Intel Threading Building Blocks library for parallelization. 3D tolerance volumes are rendered via 3D texture mapping using pixel shaders from the NVIDIA Cg Toolkit.

All atomic operations performed over the tetrahedra or sample points are easily parallelized as they are independent. All experiments are performed on an Intel 2.4GHz 16-core machine with 128 Gb RAM. The tolerance errors are specified as a percentage of the longest edge of the bounding box of the input data. Margin α is set to 0.2 in all experiments.

Though for simplicity of exposition, we assigned $\mathcal{F}$ at bounding box vertices as that of the outer boundary. In practice, we found that by multiplying this assignment by the distance to Ω significantly reduces over-refinement. However this choice might lead to large interpolated values at samples of the outer tolerance boundary. Since this does not hinder classification, we do not further refine in such cases. One way to implement this idea, is to replace the error $\epsilon(s)$ for classification by

$$\epsilon(s) = 1 - f(s)/\mathcal{F}(s).$$

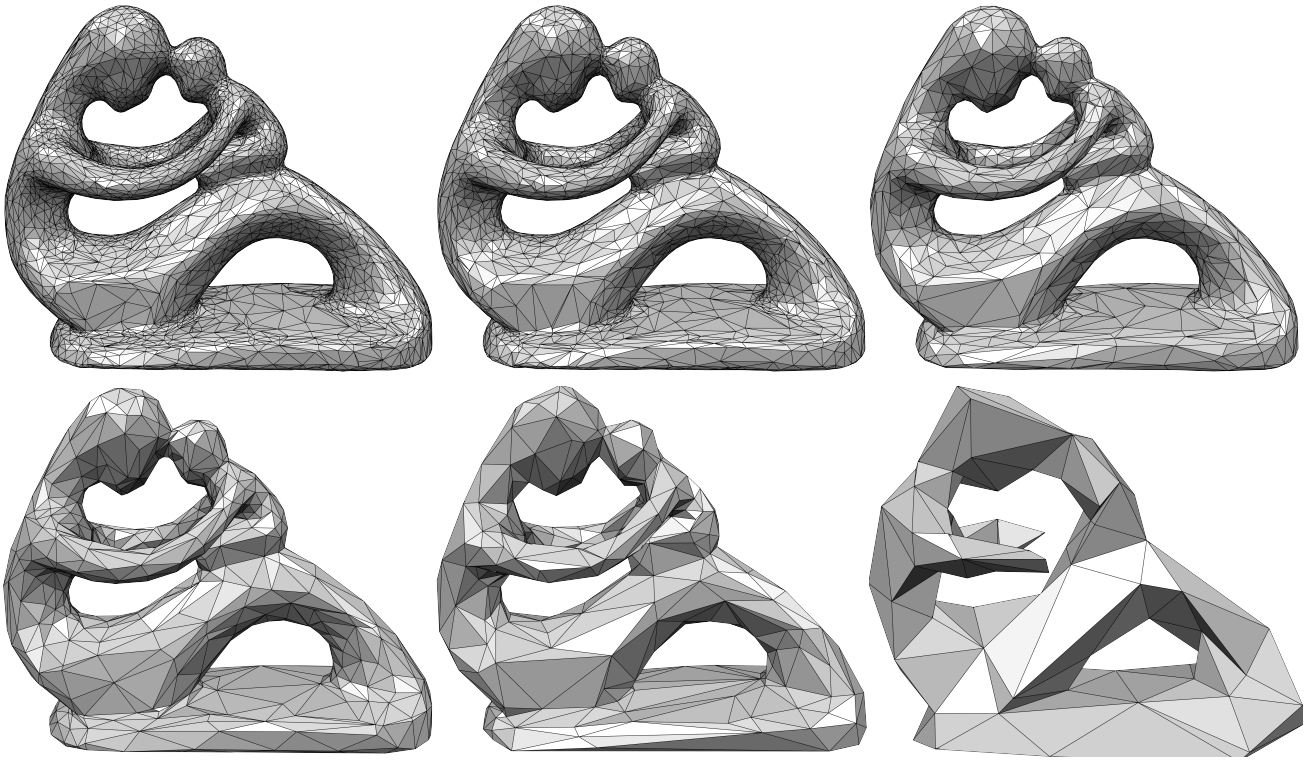


Figure 3.2: Fertility. From top left to bottom right: δ is set to 0.15, 0.25, 0.4, 0.8, 2.0 and 8.0. The final vertex count is $4,642$, $2,768$, $1,484$, 767, 417 and 120, respectively. The input model is a surface triangle mesh of the fertility model with $14k$ vertices.

Figure 3.1 illustrates our algorithm at work on a mechanical part (*blade*), for several separation distances between the boundaries of the tolerance volume. The overall time consumed by the algorithm ranges from 34 minutes for $\delta = 1.5\%$ to around 7 hours for $\delta = 0.15\%$. Figure 3.2 shows outputs of our algorithm on a smooth surface (fertility) with δ ranging from 0.15% to 8%. The one-sided Hausdorff is measured from the output to the input and is bounded in all cases.

Figure 3.3: Armadillo. δ is set to 0.1% and 0.9%. The final vertex count is $26,189$ and $1,518$. The input model is a surface triangle mesh of the armadillo model with $173k$ vertices.

Note that when δ is large (bottom right), the tolerance volume is not a topological thickening anymore as the topology of the *inner* boundary of the tolerance changes. We also run our algorithm on *Armadillo* (Figure 3.3) and *Vase Lion* (Figure 3.4) - more general meshes made of smooth and flat parts.

Figure 3.4: Vase Lion. From top left to bottom right: δ is set to 0.2, 0.3, 0.5, 2.5, and 6.0. The final vertex count is $17,723$, $11,042$, $6,127$, 637 and 109, respectively. The input model is a surface triangle mesh of the vase lion model with $200k$ vertices.

3.1 Experiments

To evaluate our algorithm, we perform several experiments and compare it with other state of the art algorithms as follows:

Vertex count over time. Figure 3.5 plots the vertex count of $\mathcal{Z}$ against time, for the fertility model and different values for δ. In all experiments, the refinement step is substantially faster than the multi-staged simplification step. The two batches of halfedge collapse operators applied to $\partial\Gamma$ and $\mathcal{Z}$ decreases the vertex count rapidly. The more general edge collapse operators are substantially slower. The time taken per operator further increases as we move from $\partial\Gamma$ to $\mathcal{Z}$, and finally to all edges. Such increase is mostly due to the transition from sampling the kernel of the edge only over $\partial\Omega$ (Figure 2.8) to pointwise probing of the whole kernel volumes in later stages. Another reason for the escalating time per operator is due to the progressive increasing of the kernel volumes when the mesh coarsens. In addition, each tetrahedron contains on average more samples and hence requires more time to verify the classification of these samples. In other words, discovering progressively the anisotropy in the input geometry, under the tolerance volume constraint, comes at an increasing cost for each edge collapse operator.

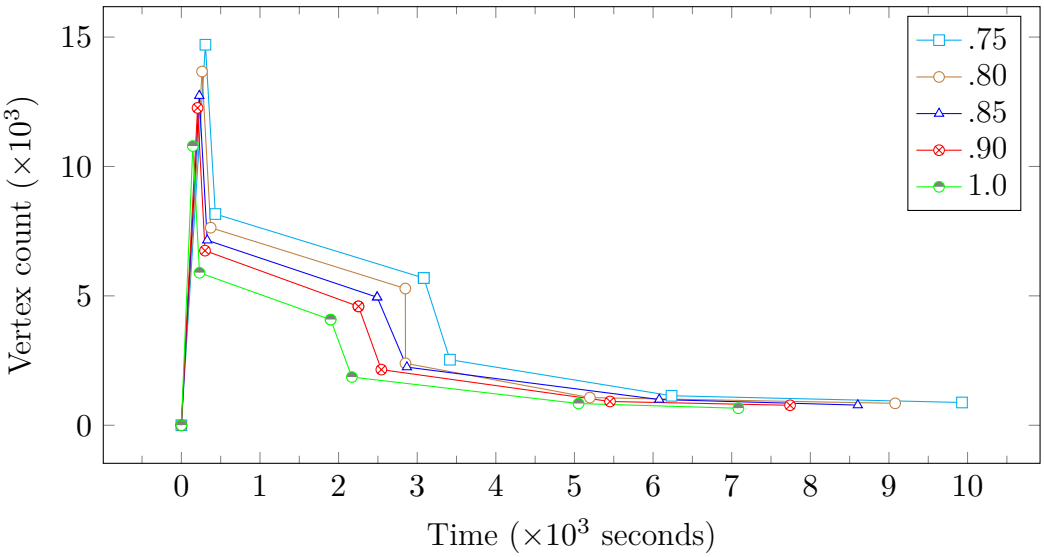


Figure 3.5: Evolution of mesh complexity over time for different $\delta(\%)$. Each mark depicts the completion of: refinement; simplification via halfedge collapses of $\partial\Gamma$; edge collapses of $\partial\Gamma$; mutual tessellation and halfedge collapses of $\mathcal{Z}$; edge collapses of $\mathcal{Z}$; and simplification of all edges. The input is a raw surface mesh of the fertility model (14kv).

Comparisons. A strict qualitative comparison with previous work is not possible as our problem statement differs. The one-sided Hausdorff distance preserved in general mesh decimation

algorithms is measured from the input to the output mesh, while we guarantee the other side of the Hausdorff distance. Nevertheless, we plot our one-sided Hausdorff distance against the number of vertices of the final output mesh, for five other mesh approximation algorithms: simplification envelopes [Cohen *et al.*, 1996], a decimation algorithm from Lindstrom-Turk [Lindstrom and Turk, 1999] (without error bounds), the MMGS remeshing algorithm [Borouchaki and Frey, 2005] (with Hausdorff error bound), MMGS with the mesh anisotropy option and a decimation algorithm with error bound implemented in OpenFlipper [Möbius and Kobbelt, 2012].

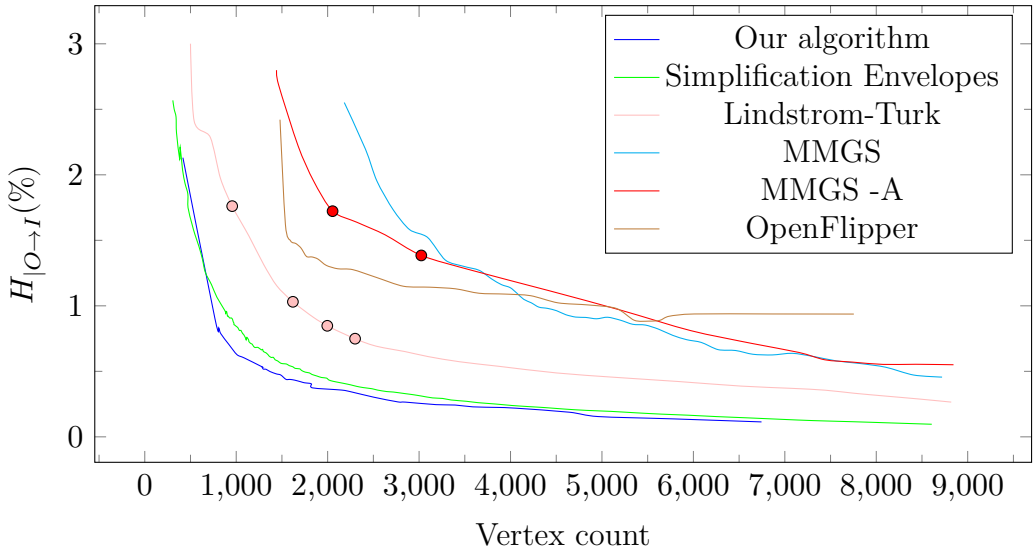


Figure 3.6: Comparisons. We plot the one-sided Hausdorff distance (output to input) ($H_{|O\to I}$) against the final number of vertices. A dot indicates a self-intersection of the output mesh. The input mesh is a clean surface triangle mesh of the fertility model (14kv).

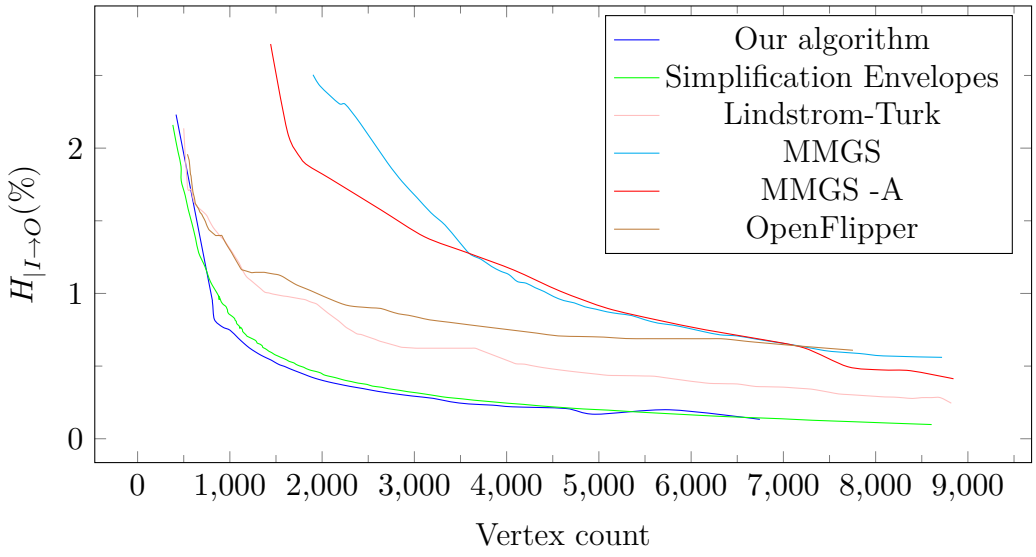


Figure 3.7: Comparisons. We plot the other-sided Hausdorff distance (input to output) ($H_{|I\to O}$) against the final number of vertices over the same data.

Albeit none of the other approaches except [Cohen *et al.*, 1996] target an intersection-free output, we indicate with a dot a self-intersection of the output mesh (Figure 3.6). For completeness we also plot the other-sided Hausdorff distance over the same input and output datasets (Figure 3.7). In both cases we achieve a lower vertex count for a given tolerance error, at the price of higher computational times.

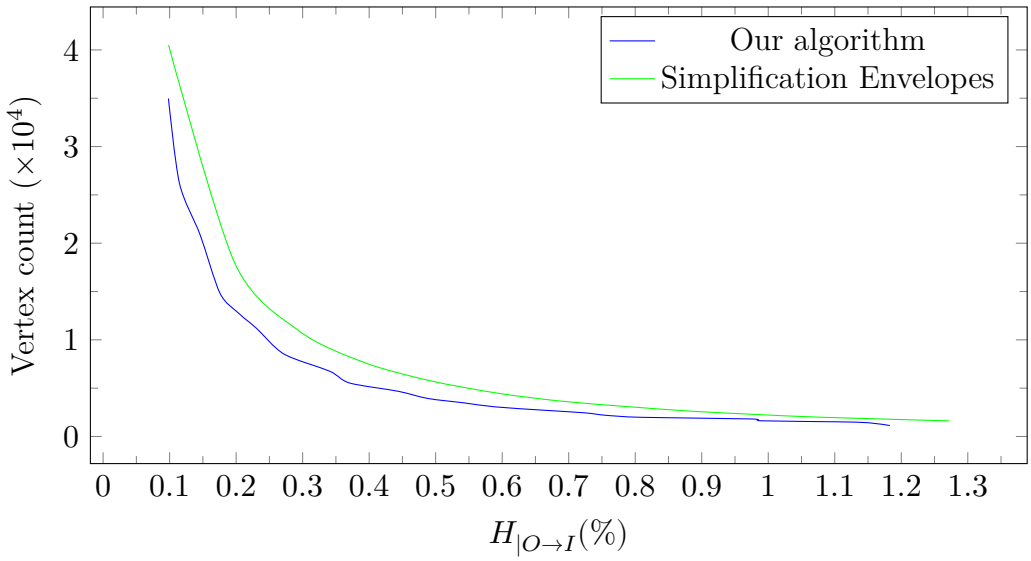


Figure 3.8: Comparison with the Simplification envelopes [Cohen et al. 1996]. We plot the final number of vertices against the one-sided Hausdorff distance (output to input) ($H_{|O \to I}$). The input is a clean surface triangle mesh of Armadillo ($173k$ vertices).

We also compare our algorithm with [Cohen *et al.*, 1996] on *Armadillo* (Figure 3.8). For a very large tolerance, the vertex count for our algorithm and simplification envelopes is comparable. However, on most part of the curve, our algorithm generates on average 10% fewer vertices, for a given tolerance error. Also note that the *simplification envelopes* require a manifold mesh as input. In addition, they cannot simplify the geometry of highly undulating surfaces beyond a certain limit, due to the specific type of tolerance volume used (Figure 3.9).

Normals. We do not provide quantitative guarantees of faithful approximation of normals in all cases. Instead, our proof (Section 2.6) yields good normals in smooth areas, when the sampling σ is dense enough. Figure 3.10 plots the distribution of normal deviation with respect to different shrinkage factors used for condition 3 for the Fertility model. Dropping this condition may cause in practice the normal deviation to exceed 90°. Figure 3.11 provides on the Lucy model a visual comparison of normals of our output to the input surface triangle mesh (left).

Dealing with sharp creases subtending small angles may require an extremely dense σ, which also translates into dense refinement. When using a too large value for σ, activating the preser-

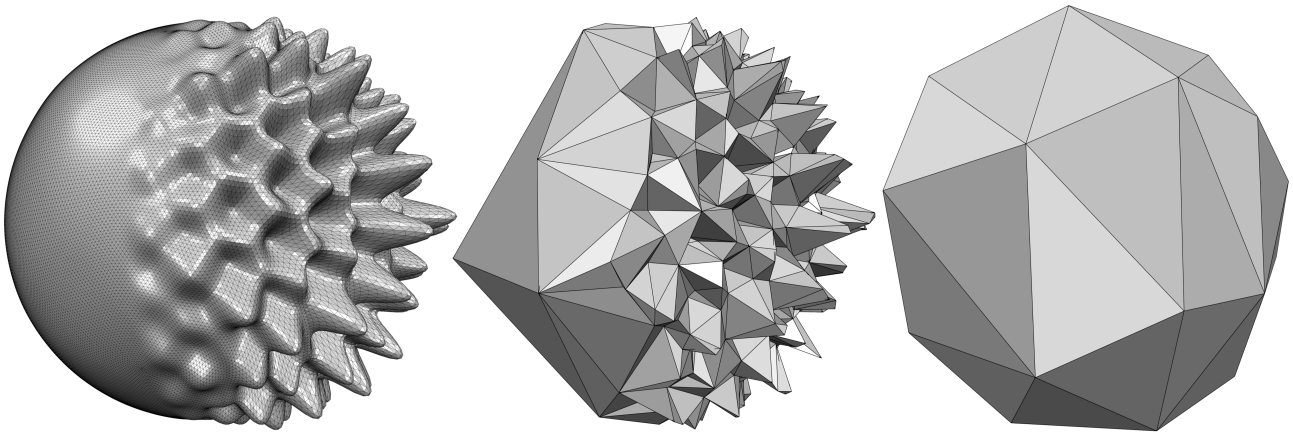


Figure 3.9: Aggressive simplification. As the tolerance in the *Simplification Envelopes* is generated by offsetting the vertices along the normals, this approach cannot simplify the geometry of highly undulating surfaces beyond a certain Hausdorff limit. Left: input mesh. Middle: output from the *Simplification Envelopes* ($H_{|O \to I} = 60\%$). Right: output from our algorithm ($H_{|O \to I} = 10\%$).

vation of normals during refinement may further translate into dense refinement as the normals are ill-defined locally. We cannot always deduce whether overly dense refinement comes from the preservation of inferred normals or from the recovery of the topology. As the decimation preserves the inferred normals we may end up with overly complex meshes. However, relaxing constraint 3 during halfedge collapses can alleviate this issue as illustrated in Fig 3.12.

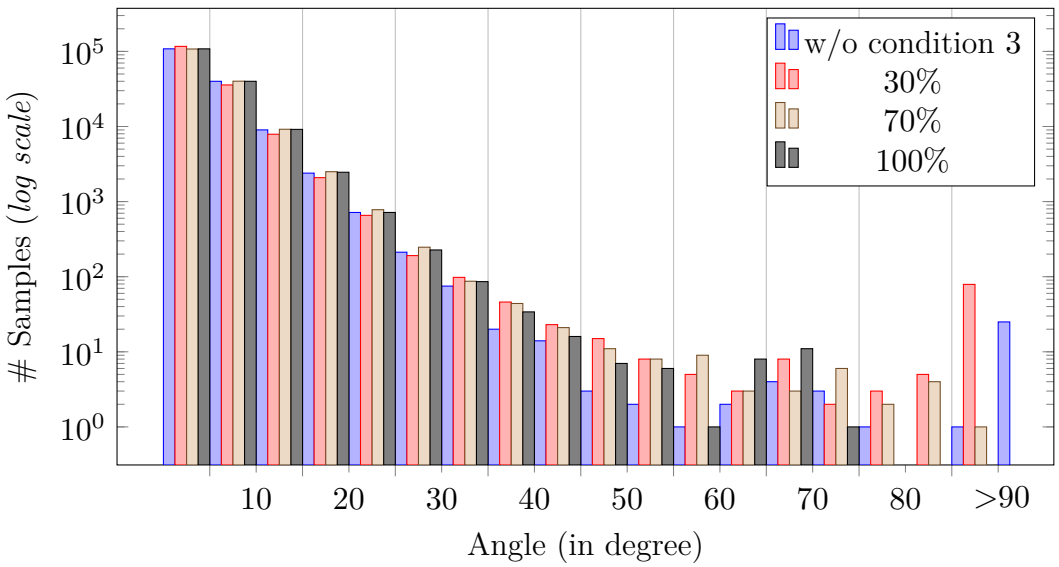


Figure 3.10: Distribution (in log scale) of normal deviation. We plot the distribution of normal deviation of our output for the fertility model ($\delta = 0.2\%$) when condition 3 is not used, and when a 30%, 70% and 100% shrinkage factor is used. The final vertex count of the output is $3,498$, $3,538$, $3,624$ and $5,715$ respectively.

Figure 3.11: Lucy. Left: Input model ($50kv$). Right: Output from our algorithm ($16.7k$ vertices, $\delta = 0.13\%$).

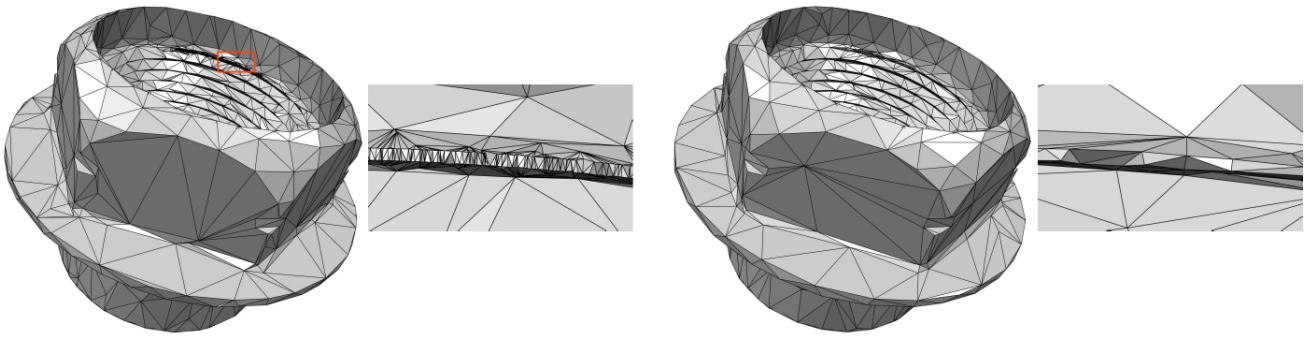


Figure 3.12: Screw. Left: when using a too large σ the decimation step terminates prematurely ($2.49kv$) due to normal constraints which are not satisfied during refinement. Right: Relaxing the normal constraint during halfedge collapse enables further decimation ($1.35kv$).

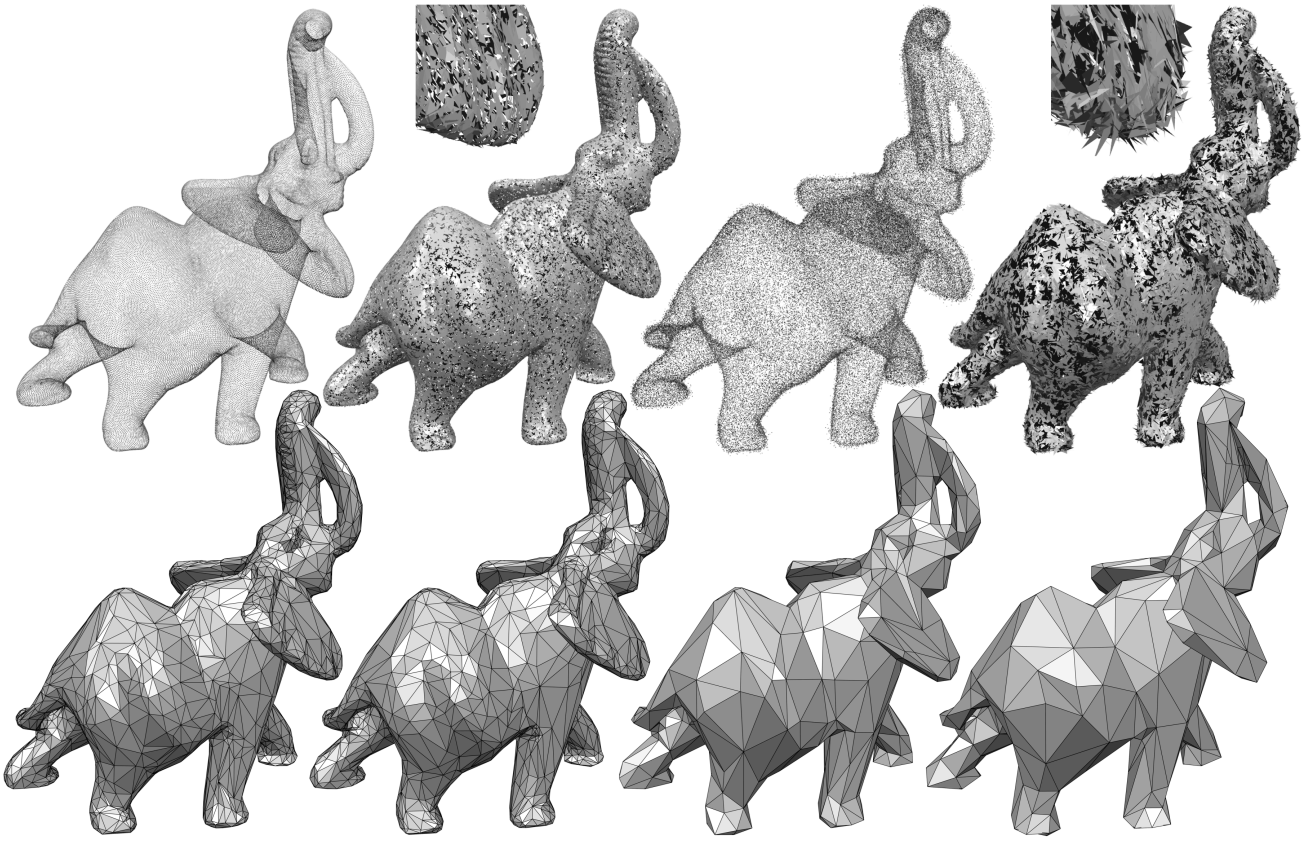


Figure 3.13: Robustness to noise and type of input datasets. From left to right: point set, triangle soup with low noise, noisy point set, and triangle soup with high noise. The corresponding δ and output vertex count are 0.9, 1.2, 2, 5% and $2,191$, $1,897$, $1,082$, 502 respectively.

Robustness. A primary virtue of our algorithm is its resilience to the type and defects of the input dataset. More specifically, and as our algorithm takes a tolerance volume as input, the robustness of our algorithm is delegated to the construction of a robust tolerance volume - then the output is guaranteed to be homotopy-equivalent to the given tolerance volume. Said

differently, our algorithm is oblivious to the dataset inside the tolerance as long as the tolerance is well-behaved. Figure 3.13 illustrates the robustness of our algorithm on point sets and defect-laden triangle soups sampled on the elephant model, with two levels of noise. We use as tolerance volume a sub-level of the robust distance function based on distances between measures [Chazal *et al.*, 2011].

Limitations. Despite its guarantees and qualitative performances, our algorithm is compute-intensive, especially when setting a small tolerance. In Figure 2.16 the tolerance is set to $\delta = 0.6\%$ and our algorithm runs for approximately 3h and consumes $2.1Gb$ of peak RAM memory. The time complexity is dominated by the simplification step. Table 3.1 lists the time taken by each step of the algorithm against the vertex count of $\mathcal{Z}$.

Table 3.1: Timing (in seconds) for each step of our algorithm for the blade model depicted in Figure 2.16.

Stage	# Vertices ($\mathcal{Z}$)	Time (s)	Time per iteration
Refinement	$20,447$	655	0.0319
halfedge - $\partial\Gamma$	$10,217$	326	0.0318
general - $\partial\Gamma$	$5,346$	$4,658$	0.956
halfedge - $\mathcal{Z}$	$2,292$	153	0.050
general - $\mathcal{Z}$	$1,015$	$1,478$	1.157
All edges	752	$4,537$	17.185
Total		$11,807$	0.2941

Most of the time spent by the algorithm is in the exhaustive search to find the best point location for an edge collapse operator, and this time escalates as the decimation proceeds. Unlike other mesh decimation algorithms, the running time of our algorithm is decreasing with parameter δ of the specified tolerance. A small tolerance requires dense sampling and hence $|\mathcal{S}|$ increases together with the time consumed to classify the samples. Another dominating factor is the sampling density used to probe a kernel when searching for the best point of a general edge collapse operator. The halfedge collapse operators are on average two orders of magnitude faster, but are not sufficient to generate coarse meshes. On the positive side, each operation performed over all sample points and tetrahedra of the triangulation is parallelizable. Figure 3.14 plots the speed up in the run-time of our algorithm versus the number of CPU cores.

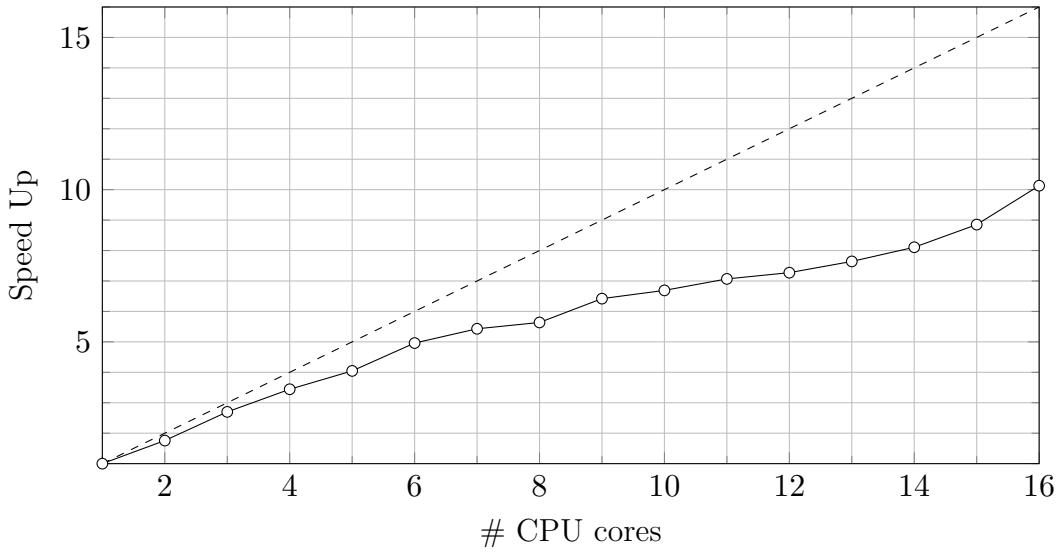


Figure 3.14: Speed up. We plot the speed up in the run-time of our algorithm versus the number of CPU cores (input: Fertility, δ is set to 0.4%).

Chapter 4

Conclusion

We introduced a novel approach to the problem of isotopic approximation within a tolerance volume. We depart from common approaches by leveraging a dense point set sampled on the boundary of the input tolerance volume. We designed a pliant meshing algorithm that first proceeds by Delaunay refinement in order to recover the correct topology through classifying the samples, and then by topology-preserving simplification in order to discover the anisotropy within the tolerance volume. A distinctive feature of our approach is its robustness to input data sets. As our approach is oblivious to the type and defects of the datasets within the tolerance volume, it can reconstruct, repair and simplify concurrently. Compared to error-driven simplification algorithms, with or without error bounds, our approach makes full use of the tolerance volume and achieves lower vertex counts for a given tolerance error, in addition to intersection-free outputs. Such lower vertex counts, however, come at a price: our current implementation is compute-intensive, and the computational times escalate when the tolerance decreases.

Future Work. The current version of the algorithm is highly parallelizable: in practice we observed that the running times are inversely proportional to the number of processors (Fig 3.14). As future work we wish to extend our approach so as to make it out-of-core. This will help us tackle the problem of both space and time complexity which are currently governed by the sampling density and simplification step. A natural direction is to cut the tolerance volume into sub-parts before stitching, but it requires another line of work to preserve the guarantees during refinement and simplification. We also observe that as the tolerance decreases, the vertex count at the end of refinement stage increases exponentially (Fig 3.5). While this is partly due to the increase in complexity (and number of features) of the shape, it is mainly due to the fact that the Delaunay triangulation is intrinsically isotropic. This not only increases the running times of the refinement step, but also has a direct impact on that of the simplification step: Over-refinement

requires more edge-collapse operations for simplification. An interesting topic for future work is to better analyze this behavior and restrict over-refinement possibly via relaxing the Delaunay property of the triangulation or via a constrained Delaunay triangulation.

Note that our algorithm assumes that a valid tolerance - which is a topological thickening - is provided as input. However, we do not provide an algorithm which is guaranteed to generate such a valid tolerance. Some previous work such as the medial axis transform [Moon *et al.*, 1997; Amenta *et al.*, 2001b] and the concept of big Delaunay balls [Dey and Sun, 2005] may provide a solution to this issue. They may also help us providing a guarantee over the two-sided Hausdorff distance which is currently not provided.

Finally, another stimulating direction is the concept of a progressive approximation algorithm, in which we could guarantee that every additional CPU cycle spent by the algorithm is making progress toward the optimal solution that matches the global minimum vertex count.

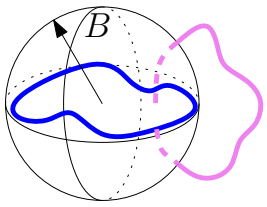
Appendices

Appendix A

Interlocked loops

Theorem A.0.1. *Assume* 2 *interlocked loops, each formed by joining a segment of length* l_e *with a continuous curve of length* l_c*. Then the continuous curves of the two loops cannot be* $(\frac{l_e+l_c}{4} + l_e, 0)$*-separated.*

Proof. Each loop is contained within a ball of radius $\frac{l_e+l_c}{4}$. Let B be such a ball for the first loop C_1. Because the two loops are linked, the part of the second loop C_2 lying in B cannot be linearly separated from the first loop. If this part only consists of the curve part, the conclusion follows. Else, let B' be the ball obtained by enlarging B by l_e. Now $B' \cap C_2$ must contain the two endpoints of the segment part of C_2. Also, $B' \cap C_2$ and C_1 are not linearly separable. Since the segment lies in the convex hull of the curve part of $B' \cap C_2$, the conclusion follows.



Part II

Low Distortion Inter-surface Mapping via Variance-Minimizing Mass Transport

Chapter 5

Introduction

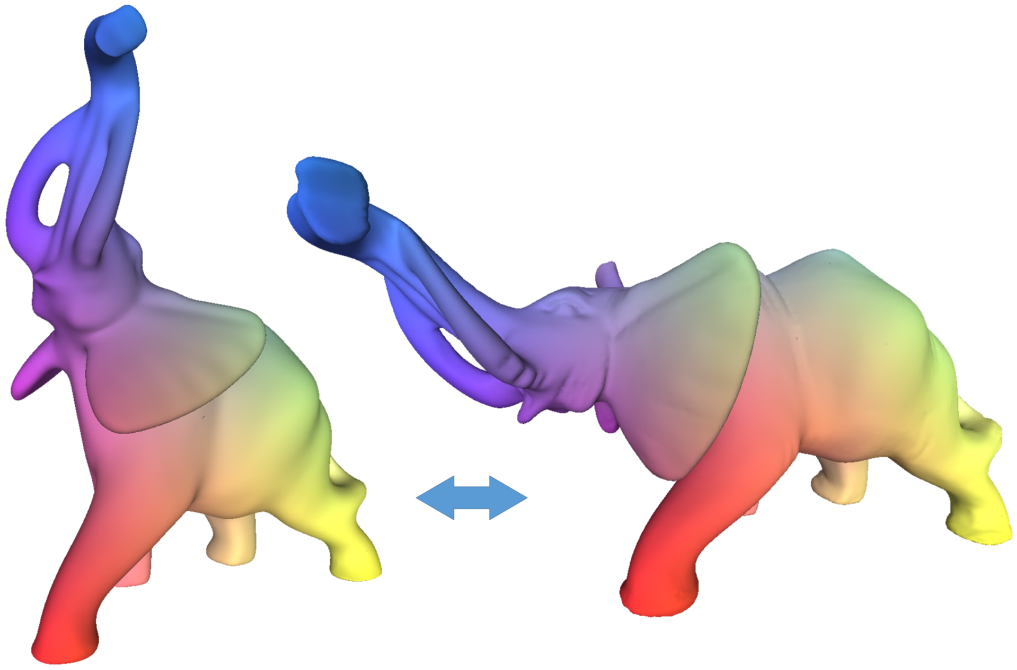


Figure 5.1: Inter-surface mapping. We generate a dense, low-distortion map between two surfaces of arbitrary genus through the computation of a transport map minimizing the local variance of geodesic neighborhoods.

Finding a mapping between two discrete surfaces is a recurring problem in geometry processing, motivated by a wide range of applications including template fitting, attribute and animation transfer, shape completion, blending, morphing, and remeshing. It consists in establishing a meaningful correspondence map between two input surfaces [van Kaick *et al.*, 2011], and has been referred to as inter-surface mapping [Schreiner *et al.*, 2004], cross-parameterization [Kraevoy and Sheffer, 2004], consistent parameterization [Asirvatham *et al.*, 2005], or shape alignment in the graphics literature.

Given two arbitrary shapes, an infinite number of mappings exist between them. If the two shapes are isometric, efficient algorithms exist to establish a good map between these shapes. However, in most practical cases where shapes may differ significantly, there is no universal consensus on what defines the *best* correspondence map, and one typically invokes a geometric prior such as conformality or area-preservation to narrow down the type of suitable maps to look for. For inter-surface maps to be useful to subsequent geometry processing, they should be *bijective* (one-to-one) and have some degree of *continuity*, i.e., small neighborhoods should map to bounded neighborhoods. Another common mapping requirement is semantic meaningfulness: semantic features in one shape should map to the corresponding features in the other shape.

5.1 Background

Before detailing the exact problem formulation and our algorithm, we mention the numerical tools that will help us to solve the problem robustly and efficiently.

5.1.1 Diffusion geometry.

For a Riemannian manifold M, its *Laplace-Beltrami* operator admits eigenvalues λ_i and corresponding eigenfunctions ϕ_i that satisfy $\triangle_M \phi_i = \lambda_i \phi_i$. This operator can be well approximated for both triangulated [Meyer *et al.*, 2003; Desbrun *et al.*, 2006] and polygon surfaces [Alexa and Wardetzky, 2011], as well as pointsets [Belkin *et al.*, 2009; Liu *et al.*, 2012] and has been extensively used in geometry processing [Sorkine, 2006]. Coifman *et al.* [2005]; Nadler *et al.* [2005] proposed the notion of *diffusion distance* which has also been lately introduced in geometry processing [Rustamov, 2007; de Goes *et al.*, 2008]. If we denote the heat kernel $K_{M,t}$ as

$$K_{M,t}(x,y) = \Phi(x)^\top \Phi(y), \tag{5.1}$$

where $\Phi(x) = (e^{-\lambda_1 t/2}\phi_1(x), e^{-\lambda_2 t/2}\phi_2(x), \dots)$ for any point $x \in M$ and t controls the diffusion time scale, then the diffusion distance between two points x and y on manifold M can be computed as

$$d_{\text{diff}}(x,y) = \|\Phi(x) - \Phi(y)\|. \tag{5.2}$$

Note that the heat kernel and diffusion distances are related via

$$d_{\text{diff}}(x,y)^2 = K_{M,t}(x,x) + K_{M,t}(y,y) - 2K_{M,t}(x,y).$$

For small t, the diffusion distance approximates the local geodesic distances well, while increasing t generates smooth long range geodesic distances [Coifman *et al.*, 2005]. Moreover, by selecting only the first n smallest eigenvalues of the Laplace-Beltrami operator (which can be solved efficiently [Vallet and Lévy, 2008]), the diffusion distances can be approximated at low computational cost. Note that the diffusion distance Eq. (5.2) isometrically embeds in (Euclidean) diffusion space, i.e. the geodesic distance between two samples x and y in $\mathbb{R}^3$ is approximated by the Euclidean distance between $\Phi(x)$ and $\Phi(y)$ in $\mathbb{R}^n$.

5.1.2 Optimal transportation

The problem of optimal transportation was originally proposed by Monge in 1781 [Monge, 1781]: given two mass distributions $\mathbf{m}$ and $\mathbf{n} \geq 0$ on $\mathbb{R}^d$, with $\int \mathbf{m} = \int \mathbf{n} = 1$, the aim is to find a map $\pi : \mathbb{R}^d \to \mathbb{R}^d$ between $\mathbf{m}$ and $\mathbf{n}$ such that it minimizes

$$\int |\pi(x) - x|\mathbf{n}(x)dx, \tag{5.3}$$

among all the maps that satisfy

$$\int_A \mathbf{m}(x)dx = \int_{\pi^{-1}(A)} \mathbf{n}(y)dy, \tag{5.4}$$

where, A denotes any Borel set in $\mathbb{R}^d$. Kantorovich also studied the problem independently and proposed a suitable framework [Kantorovich, 1942] by transforming it into a linear problem on a convex set, which later led to the proof of existence of a solution. The main idea was to look for optimal transport plans in the product space $\mathbb{R}^d \times \mathbb{R}^d$ instead of optimal transport maps. Since then, the problem has been widely generalized to general cost functions (beyond the Euclidean distance) and over general measures, and advances have been made to solve it efficiently [Smith and Knott, 1987; Cuesta-Albertos and Tuero-Díaz, 1993; Villani, 2003; Carlier *et al.*, 2010; Igbida *et al.*, 2011]. One of such extension, the Wasserstein metric, also commonly referred to as the earth mover's distance, is the measure of distance between two given probability distributions on a given metric space.

In the context of this thesis, we look at the optimal transportation problem of the form

$$\inf_{\pi \in \Pi(\mu_0, \mu_1)} \iint c(x, y)d\pi(x, y), \tag{5.5}$$

between two probability distributions μ_0 and μ_1 where, $c(x, y)$ denotes the transport cost between x and y and $\Pi(\mu_0, \mu_1)$ denotes the set of all couplings of μ_0 and μ_1, i.e., the collection of

all measures with source and target marginals as μ_0 and μ_1 respectively (Fig. 5.2).

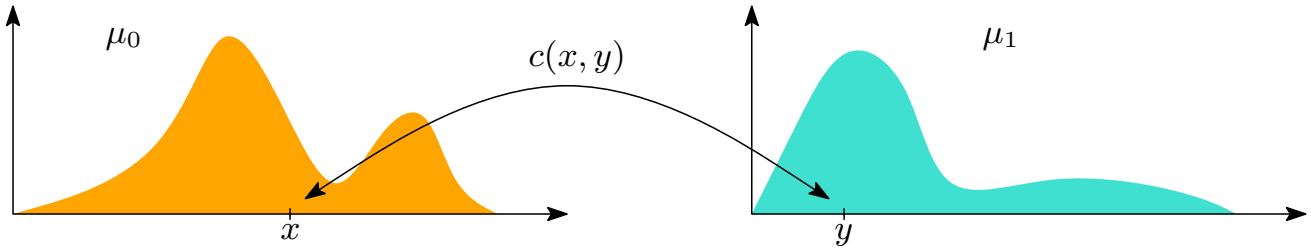


Figure 5.2: Optimal transportation problem. We search for a transport plan such that the total cost Eq. (5.5) is minimized. Note that, being a probability distribution, the total mass of both μ_0 and μ_1 is normalized to 1.

Entropic Regularization. While computing an optimal transport plan for a given linear cost functional can be achieved via linear programming, dealing with large variable counts negatively affects computation times. Recently, Cuturi [2013] proposed an entropic regularization of the optimal transport problem to smooth the conventional optimal transportation problem as

$$\inf_{\pi \in \Pi} \left[\iint c(x,y)\pi(x,y)dxdy - \gamma H(\pi) \right], \tag{5.6}$$

where $H(\pi)$ refers to the entropy of the coupling, namely,

$$H(\pi) \overset{\text{def.}}{=} - \iint \pi(x,y) \ln \pi(x,y) dxdy, \tag{5.7}$$

and γ controls the amount of regularization or smoothing. The cost function c can be associated to a kernel $\mathcal{K}_\gamma(x,y) = \exp(-c(x,y)/\gamma)$, so that we can compute the entropy-regularized transport plan from the smallest *Kullback-Leibler* (KL) divergence as

$$\gamma \left[1 + \min_{\pi \in \Pi} \mathrm{KL}(\pi | \mathcal{K}_\gamma) \right], \tag{5.8}$$

where,

$$\mathrm{KL}(\pi | \mathcal{K}_\gamma) \overset{\text{def.}}{=} \iint \pi \left[\ln \frac{\pi}{\mathcal{K}_\gamma} - 1 \right] dxdy. \tag{5.9}$$

Sinkhorn Iterations. This regularized optimal transport plan can be efficiently computed using Sinkhorn's matrix normalization algorithm [Sinkhorn, 1964; Sinkhorn and Knopp, 1967]. Following Benamou *et al.* [2015]; Solomon *et al.* [2015], given a matrix $\mathbf{H}$ discretizing the kernel $\mathcal{K}_\gamma$, the transport plan minimizing Eq. (5.8) between two discrete probability distributions $\mathbf{m}$ and $\mathbf{n}$ is of the form $\pi = \mathbf{D_m D_v H D_w D_n}$, with unique diagonal matrices $\mathbf{D_v}$ and $\mathbf{D_w}$ defined by

vectors $\mathbf{v}$, $\mathbf{w}$ satisfying:

$$\begin{cases} \mathbf{D_v H D_w n} = \mathbf{1}_{|\mathbf{m}|}, \\ \mathbf{D_w H}^\top \mathbf{D_v m} = \mathbf{1}_{|\mathbf{n}|}, \end{cases} \tag{5.10}$$

where, both π and $\mathbf{H}$ are of size $|\mathbf{m}| \times |\mathbf{n}|$, $\mathbf{D_x}$ represents the diagonal matrix with the vector $\mathbf{x}$ as its diagonal, and $\mathbf{1}_k$ is the vector of ones of size k. We find $(\mathbf{v}, \mathbf{w})$ via *Sinkhorn iterations* as detailed in Alg. 1, where $\oslash$ and $\otimes$ denote the entry-wise division and multiplication, respectively. The Sinkhorn algorithm is known to converge at a linear rate [Franklin and Lorenz, 1989; Knight, 2008] and hence, is several orders of magnitude faster than solving a linear program. Note that the result,

$$\begin{cases} \pi \mathbf{1}_{|\mathbf{n}|} = \mathbf{m}, \text{and} \\ \pi^\top \mathbf{1}_{|\mathbf{m}|} = \mathbf{n}, \end{cases} \tag{5.11}$$

leading to the two distributions can be easily deduced using Eq. (5.10).

Algorithm 1 Sinkhorn iterations to solve for optimal transport

```
function SINKHORN
    v, w ← 1
    while !converged do
        v ← 1_|m| ⊘ H(n ⊗ w)
        w ← 1_|n| ⊘ H^⊤(m ⊗ v)
    return π = D_m D_v H D_w D_n        ▷ Transport plan
```

Dykstra's Algorithm. In general, when convex sets of constraints are not affine subspaces, iterative Bregman projections [Bregman, 1967] (or Sinkhorn iterations, being a specific case) on these sets do not converge to the KL projection on the intersection. In such cases, the more general Dykstra's algorithm [Dykstra, 1983] can be used as long as the constraints are convex sets. This algorithm is proven to converge [Bauschke and Lewis, 1998].

Benamou *et al.* [2015] extended the Dykstra's algorithm to KL settings which can be used to solve such problems, for example, a relaxed transport plan. Starting with an initial solution $\pi^{(-1)} = \mathbf{H}$ and l temporary variables $q_{(k)} = \mathbf{1}$ – one corresponding to each of the l convex constraint set C_k – we iteratively solve for

$$\begin{cases} \pi^{(n)} = P^{KL}_{C_{(n \bmod l)}}(\pi^{(n-1)} \otimes q_{(n \bmod l)}), \\ q_{(n \bmod l)} = q_{(n \bmod l)} \otimes \pi^{(n-1)} \oslash \pi^{(n)}, \end{cases} \tag{5.12}$$

where, $P^{KL}_{C_k}$ denotes the projection according to the Kullback-Leibler divergence Eq. (5.9) over the constraint set C_k. The algorithm converges as $n \to \infty$.

5.2 Positioning

In our work, we adopt a mass transport formulation to seek inter-surface maps. However, instead of favoring softness and continuity as in most previous work, we promote sharpness of the map by penalizing the variance of the image of local neighborhoods. We show that our variance-based formulation favors continuity, conformality and sharpness of the map all at once. Although the problem becomes non-convex and hence substantially more challenging to minimize, we provide a formulation that turns this non-linear problem into a biconvex problem where geodesic variance is minimized in diffusion distance space, leading to a robust and efficient algorithm for inter-surface mapping. Additionally, we directly optimize a surface-to-surface map without resorting to an intermediate domain like many previous works. Our approach notably contrasts with Aflalo *et al.* [2013] which sought conformal maps with no (or bounded) area distortion: we look instead for area-preserving maps (or maps with bounded area distortion) that are as conformal as possible.

Contributions. We propose an approach to find a dense map between two arbitrary surfaces. We formulate this inter-surface mapping problem as a minimization of a Dirichlet energy for transport maps that we define in terms of the variances of the images of local neighborhoods. This energy penalizes stretching and favors conformal, point-to-point homeomorphisms. Although we cannot guarantee the existence of homeomorphisms, the idea behind our formulation, however, is to map neighborhoods to small geodesic neighborhoods, where small means low variance — hence aspiring to form a homeomorphism.

We further provide an algorithm to compute such a variance-minimizing inter-surface map via alternating convex optimizations. A coarse to fine approach based on a hierarchy built in diffusion space [Coifman *et al.*, 2005] is developed to both improve efficiency and reduce the risk of getting trapped in local minima. In addition, we modify and extend the Sinkhorn iteration algorithm [Cuturi, 2013; Solomon *et al.*, 2015] to handle low regularization and unbalanced optimal transport [Benamou *et al.*, 2015], where mass preservation can also be relaxed to enable further minimization of the map distortion.

5.3 Related Work

Over the last decade, there has been significant amount of work on conformal mapping. However, apart from some recent advancements to generate area-preserving maps, as of now, the problem of area-relaxed maps which are as conformal as possible remaines algorithmically infeasible. We discuss next these algorithms, along with other approaches in order to broaden the discussion to the general topic.

Area-preserving Mapping. Angenent *et al.* [2000] proposed an algorithm for area-preserving map with minimal distortion in the context of $3D$ medical visualization, for visualizing functional magnetic resonance imaging data of neural activity in the brain. They define a variant of the Dirichlet energy and minimize it over the space of area-preserving diffeomorphism via steepest descent. Monge-Kantorovich theory of optimal mass transportation has also been used to build area-preserving mapping due to its mass preservation property [Zhu *et al.*, 2003; Haker *et al.*, 2004]. Recently, Su *et al.* [2013] proposed an alternate method based on Monge-Brenier theory which leads to a convex energy minimization and power Voronoi diagram construction. The algorithm generates a unique solution which is invariant under isometric transformations and with an improved complexity, i.e. with a linear instead of quadratic number of variables. Mapping based on area-distortion minimization has also been studied [Desbrun *et al.*, 2002] leading to a quartic polynomial which can be efficiently solved using a simple root finder.

Apart from them, other previous approaches mainly target conformal maps and can be classified by the type of mapping (point-to-point vs. soft), its properties (bijective, locally injective), its measure of distortion (isometric, conformal) and the optimization strategy employed to find the mappings. We review the most relevant approaches next, and refer the reader to van Kaick *et al.* [2011] for a comprehensive survey on shape correspondence.

Embedding. When dealing with near isometric maps, embedding of the input manifolds into some target (Euclidean) space [Gallot *et al.*, 1994; Bronstein *et al.*, 2006; Shtern and Kimmel, 2015] is a practical way to simplify matching since corresponding points of different isometric shapes are mapped to nearby points in the target space. An extension of this family of approaches use as embedding space a common template mesh. The base domain is often chosen to be a coarse mesh obtained through mesh decimation [Schreiner *et al.*, 2004], or by defining overlapping domains bounded by consistent paths connecting feature points [Kraevoy and Sheffer, 2004]. The mapping is then constructed by optimizing a mutual tessellation of the two surfaces. While this approach can handle non-isometric transformations, decreasing the global non-linear distortion

through vertex-by-vertex updates of the map is often very slow.

Mesh Parameterization. Mesh parameterization was originally introduced in computer graphics by Bennis *et al.* [1991] for texture mapping and has been extensively used for geometry mesh processing. Since it is impossible to avoid both angle and area distortion for non-zero Gaussian curvature surfaces, Degener *et al.* [2003] proposed parameterization based on global area and angle distortion. Other approaches based on minimizing the distortion of different intrinsic measures [Desbrun *et al.*, 2002], conformal mapping via least squares approximation [Lévy *et al.*, 2002; Liu *et al.*, 2008] and globaly conformal parameterization based on gradient fields [Gu and Yau, 2003] have also been studied.

Recent approaches construct maps with bounded distortion [Lipman, 2012], use cone manifolds [Myles and Zorin, 2013] or orbifolds [Aigerman and Lipman, 2015], and relax the bijective condition of the parameterization to local injectivity [Aigerman *et al.*, 2014]. A recent approach contributed by Aflalo *et al.* [2013] seeks angle-preserving maps with as uniform as possible conformal factors, *i.e.*, as area-preserving as possible conformal maps. One main limitation of these parameterization-based approaches is that the map distortion is not directly measured between the two input surfaces, but through a common base domain, which leads to suboptimal inter-surface maps. Mapping to a common domain also often requires designing a cut-graph which can increase distortion as well (see Aigerman *et al.* [2015] for a recent improvement).

Soft Maps. While point-to-point mapping between surfaces has been favored for the past few decades, recent notions of soft maps [Solomon *et al.*, 2012] and functional maps [Ovsjanikov *et al.*, 2012] where functions or distributions are mapped between surfaces have been shown relevant to the analysis of correspondences through linear algebra tools. Using a mass transport formulation, Solomon *et al.* [2012] generated maps that are continuous, faithful to geometric descriptors and *soft*. Continuity of such maps means that nearby regions on one surface should map to nearby distributions on the other surface, where "nearby" is to be understood with respect to the geodesic earth mover distance. In addition, softness of the map is achieved by L_2 penalization, spreading the probabilities in the mapping matrix describing the mass transport plan: this quantity is convex and amenable to efficient optimization. The manner in which a soft map transports one distribution to another was also pointed as relevant to better understand and control map continuity [Solomon *et al.*, 2013]. A recent work, Solomon *et al.* [2016], further demonstrates the power of entropic regularization to render challenging problems computationally tractable, but acknowledges that limiting the induced oversmoothing is still an open problem. While soft maps are an elegant and powerful paradigm, sharper maps, from which

point-to-point correspondences can be inferred, remain a necessity for most geometry processing algorithms.

Finally, a recent trend is to adopt a machine learning approach in order to alleviate the need to define a prior on the type of deformation [Litman and Bronstein, 2014; Rodolà *et al.*, 2014].

Chapter 6

Algorithm

6.1 Optimal Transport Approach

Given two input pointsets $X = \{x_i\}$ and $Y = \{y_j\}$ sampling two surfaces, we aim to construct a map between X and Y that associates neighborhoods from one pointset with neighborhoods in the other pointset, and vice-versa. We formulate this inter-surface mapping as an optimal mass transport problem between the two input pointsets for which the transport plan minimizes the variance of the image of each small neighborhood, where the variance of the plan is efficiently measured in diffusion space.

6.1.1 Setup

In order to formulate our map optimization problem, we introduce a few key concepts that we will leverage throughout our exposition.

Surfaces as distributions. Our approach considers each pointset as a *discrete mass distribution*. More specifically, we discretize the normalized area measures of the two surfaces as weighted sums of Dirac measures centered at input points, i.e., $\mu = \sum m_i \, \delta_{x_i}$ and $\nu = \sum n_j \, \delta_{y_j}$, where the masses m_i and n_j are normalized so that each surface has total mass $\sum_i m_i = \sum_j n_j = 1$. The mass of each point can be assigned based on its estimated local Voronoi area of the surface it samples to account for irregular sampling, or simply to a uniform constant if such an estimate is unavailable.

Maps as transport plans. Given two distributions μ and ν, a transport plan π between them is by definition a matrix of size $|X| \times |Y|$ whose marginals equal μ and ν. Each entry π_{ij} represents the amount of mass transported from $x_i \in X$ to $y_j \in Y$. A transport plan π has two associated

transfer operators π_X and π_Y defined as

$$\pi_X(\delta_{x_i}) = \frac{1}{\sum_j \pi_{ij}} \sum_j \pi_{ij}\,\delta_{y_j}\,,\ \pi_Y(\delta_{y_j}) = \frac{1}{\sum_i \pi_{ij}} \sum_i \pi_{ij}\,\delta_{x_i}.$$

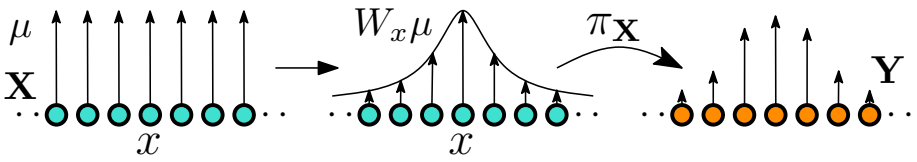
Note that the denominators are respectively equal to m_i and n_j. These two linear operators describe how mass distributions on X are mapped to mass distributions on Y with the same total mass, and vice-versa. Therefore, they contain essentially the same information as the transport plan.

Neighborhoods as weighting functions. For a given point x on a surface, we denote by W_x a weighting function centered at x. A natural choice for W_x is a truncated Gaussian function, but any near-isotropic bump function that decreases as a function of the geodesic distance from x may do. Such weighting functions will be used over the two input surfaces as test functions to measure the variance of an inter-surface map at different locations since their finite supports define geodesic neighborhoods.

6.1.2 Regularity measure for transport plans

Our approach consists in minimizing the following variance-based functional over all possible plans π between μ and ν:

$$E\,(\pi) = \int_X \text{var}\ \pi_X \left(\frac{W_x \mu}{\text{mass}(W_x \mu)} \right) d\mu(x) + \int_Y \text{var}\ \pi_Y \left(\frac{W_y \nu}{\text{mass}(W_y \nu)} \right) d\nu(y), \tag{6.1}$$



that is, we integrate the variances of images by the transfer operators of area measures modulated by local weighting functions (see inset). Note that since we consider our surfaces as discrete measures, these integrals reduce to weighted sums over data points.

Soft vs. sharp maps. The reason for choosing this cost functional is two-fold. First, penalizing the variance of the image of the map favors transport plans corresponding to actual point-to-point maps rather than soft (diffused) maps: any point mass that is mapped to a spread-out distribution incurs a significant cost. Second, these variance-minimizing maps tend to be bijective, since we also account for the inverse map in the cost evaluation. Note that bi-continuity

may also be expected for the minimizers of our cost as we penalize variances of images of small neighborhoods defined by our weighting functions.

Link to Dirichlet energy. We show in App. A that for infinitesimal weighting functions with isotropic covariance matrices tending to zero, our cost converges to the sum of the Dirichlet energies of the map and its inverse. Hence, our cost may be viewed as a symmetric Dirichlet energy for transport plans. While generalization of the Dirichlet energy to transport plans have already been proposed in the literature (see, e.g., [Solomon *et al.*, 2013]), we are not aware of any existing formulation penalizing soft maps, a property that is essential for our purpose. Moreover, we show next that our particular variance-based formulation is amenable to the use of efficient optimization methods.

6.1.3 Minimization via alternating convex optimizations

While the formulation above formalizes the notion of a good map between two surfaces, it involves a non-linear and in general non-convex functional. As such, it may seem difficult to minimize efficiently and robustly. We can, however, introduce auxiliary variables within the variance terms so that finding a variance-minimizing map amounts to solving a biconvex problem, which we achieve through *alternating convex minimization*.

Reformulation with auxiliary variables. Given a point x and a measure $\mu = \sum \mu_i \delta_{x_i}$, we denote by $\mathrm{var}(\mu, x)$ the variance of μ with respect to x:

$$\mathrm{var}(\mu, x) = \sum \mu_i d(x_i, x)^2, \tag{6.2}$$

where distances are to be understood as *geodesic* distances on the surface that μ samples and that x lies on. Using this notion of variance w.r.t. a position, we can now introduce a set of positions η_{x_i} and η_{y_j}, one for each point x_i and y_j. These auxiliary variables, which we call *centers* for reasons that will become clear soon, allow us to write our transport cost between X and Y as a function of both the transport plan π and the set of all centers η:

$$\begin{aligned}\mathcal{C}(\pi, \eta) = &\sum_i m_i \ \mathrm{var}\left[\pi_X \left(\frac{W_{x_i}\mu}{\sum_k m_k W_{x_i}(x_k)}\right), \eta_{x_i}\right] + \\ &\sum_j n_j \ \mathrm{var}\left[\pi_Y \left(\frac{W_{y_j}\nu}{\sum_k n_k W_{y_j}(y_k)}\right), \eta_{y_j}\right].\end{aligned} \tag{6.3}$$

Note that this cost functional *equals* the energy defined in Eq. (6.1) *if* each center η_{x_i} (resp., η_{y_j}) corresponds to the (weighted) *center of mass* of the image of the normalized weighting functions W_{x_i} (resp. W_{y_j}) by the transport plan π.

Minimization through alternating convex problems. This augmented cost functional $\mathcal{C}(\pi, \eta)$ is now amenable to an alternating minimization similar to Expectation-Maximization (EM): for a fixed transport plan π, minimizing $\mathcal{C}(\pi, .)$ is efficiently achieved by relocating the centers to the geodesic barycenters of the images of the normalized weighting functions by π (see Sec. 6.2.3); for fixed centers, minimizing $\mathcal{C}(., \eta)$ requires the computation of an optimal transport plan under the usual mass preservation constraints. Alternately minimizing the cost $\mathcal{C}$ over π_{ij} and η as described in Alg. 2 is a robust and efficient way to treat the original non-linear problem since both minimizations are convex problems. Fig. 6.1 shows the evolution of the transport plan over alternating minimization iterations for a curve example in the plane.

Algorithm 2 Map optimization through alternating minimization

```
function ALTERNATING MINIMIZATION
    for i = 1, 2, 3 ... do
        η ← min C(π, .)  // relocate centers to geodesic barycenters
        π ← min C(., η)  // solve optimal transport problem
    return π                                  ▷ variance-minimizing transport plan
```

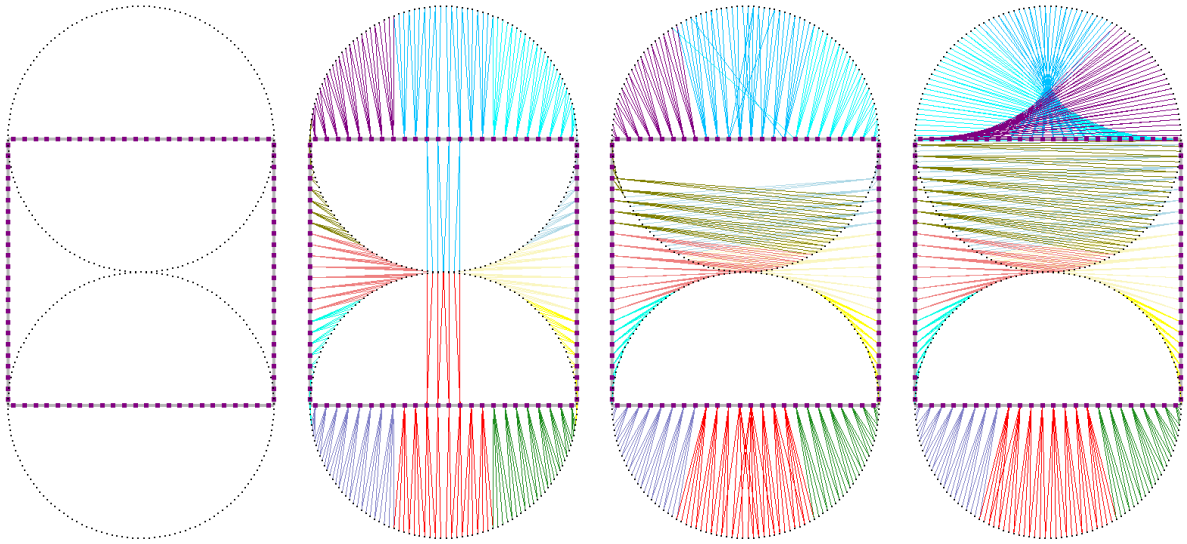


Figure 6.1: Mapping via variance minimization in 2D. An 8-shaped pointset X and a square-shaped pointset Y (left); initial and intermediate solution (middle); and convergence after 2 iterations of a map between the two input shapes (right).

Additional control over the mapping. The cost we defined in Eq. (6.3) measures the regularity of a transport plan via its local variance. However, user control over the final mapping is often desirable: the user may want to prescribe a few correspondences manually. If we wish x_i to map to y_j, we constrain the center η_{x_i} to be y_j and η_{y_j} to be x_i. Additionally, the user may want to favor mappings that match points with similar attributes such as colors or shape signatures. This can be done by adding to our cost functional an additional transport cost, with a cost matrix given by distances between feature vectors associated with each input point. Finally, adding a small amount of the usual 2-Wasserstein distance between the surfaces can help finding good initial solutions or help the solver disambiguate symmetries in the models, and thus finding better maps automatically, see Sec. 7.

6.2 Algorithms and Numerics for Scalability

We designed several additional algorithmic and numerical components to offer an efficient and scalable solver for our approach.

6.2.1 Weighting function

A natural (and consistent) choice as weighting function W is to rely upon the heat kernel from Eq. (5.1) in order to define an approximate geodesic neighborhood. While this choice is nicely isotropic for small diffusion times t, we noticed that its support becomes off-centered for large values of t (see Fig. 6.2). We use instead the following weighting function:

$$W_x(y) = \frac{2K(x,y)}{K(x,x) + K(y,y)}, \tag{6.4}$$

that remains localized and centered for all diffusion times while enforcing $W_x(x) = 1$. In addition, we truncate the weighting function below $\varepsilon = 0.5$ (Fig. 6.2) in order to further localize its associated geodesic neighborhood, which will reduce computational times required to evaluate the geodesic variance.

6.2.2 Fast and robust optimal transport

Instead of solving Step 4 of Alg. 2 via linear programming, we use Sinkhorn iterations (Alg. 1) to compute the optimal transport plan efficiently. While the original approach requires assembling the kernel matrix $\mathbf{H}$ through its elements $H_{ij} = \exp(-\text{coef}(\pi_{ij})/\gamma)$, where $\text{coef}(\pi_{ij})$ refers to the coefficient of π_{ij} in the cost function (6.3), we proceed in log space instead to improve robustness.

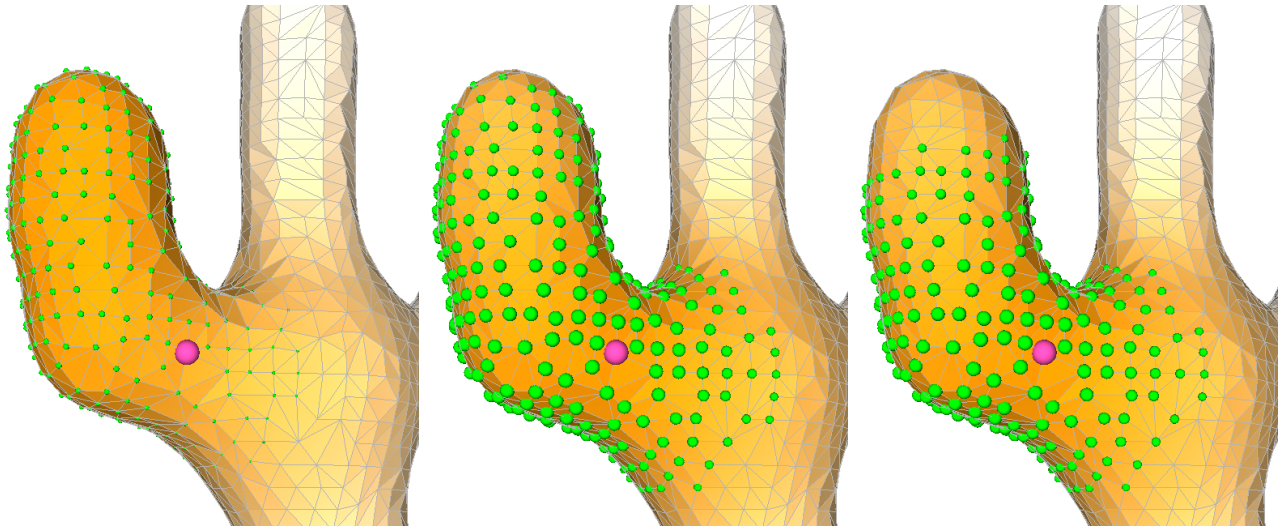


Figure 6.2: Weighting functions. From left to right are depicted $K(x,y)$, $K(x,y)/\sqrt{K(x,x)K(y,y)}$ and $2K(x,y)/(K(x,x)+K(y,y))$. The neighborhood of the pink vertex corresponding to $\{y : W_x(y) > \varepsilon\}$ is shown with green vertices.

Improving robustness. The regularization coefficient γ used to compute the optimal transport is recommended to be set such that $\gamma > \max\{\text{coef}(\pi_{ij})\}/200$ [Cuturi, 2013] in practice, to prevent numerical blowups due to limited floating-point dynamic range. However, this choice of coefficient usually creates quite a diffuse transport plan, conflicting with our objective to construct variance-minimizing plans. We bypass this numerical limitation by storing the coefficients of the kernel matrix in log scale and compute the Sinkhorn iterations in log scale directly. This implies that we now store $H_{ij} = -\text{coef}(\pi_{ij})/\gamma$, and perform the inner products $a^\top b$ in Steps 4-6 of Alg. 1 (involving columns or rows of H) by first subtracting the maximum element $M = \max_i\{a[i]+b[i]\}$ from each element, and evaluating

$$a^\top b := M + \log \sum_j \exp(a[j] + b[j] - M).$$

Working in log scale along with the subtraction of the maximum element drastically reduces the negative effects of a limited floating-point dynamic range, allowing us to safely set to $\gamma = \max\{\text{coef}(\pi_{ij})\}/10^4$, a 50-fold decrease in regularization.

As $\gamma \to 0$, Sinkhorn's algorithm can require many iterations to converge. In practice we thus perform only 100 iterations between η updates. Once the η updates are small enough, we perform annealing by first increasing γ by 50%, followed by 5 extra η updates with 1,000 Sinkhorn iterations each. Parameter γ is then reverted back to its original value. This scheduling experimentally improves the convergence of Sinkhorn iterations, and we observe that mass preserving constraints are nearly satisfied upon completion (Fig. 6.3). In order to further improve efficiency,

we also use the previous values of v and w instead of initializing them to 1 in each successive relaxation iterations.

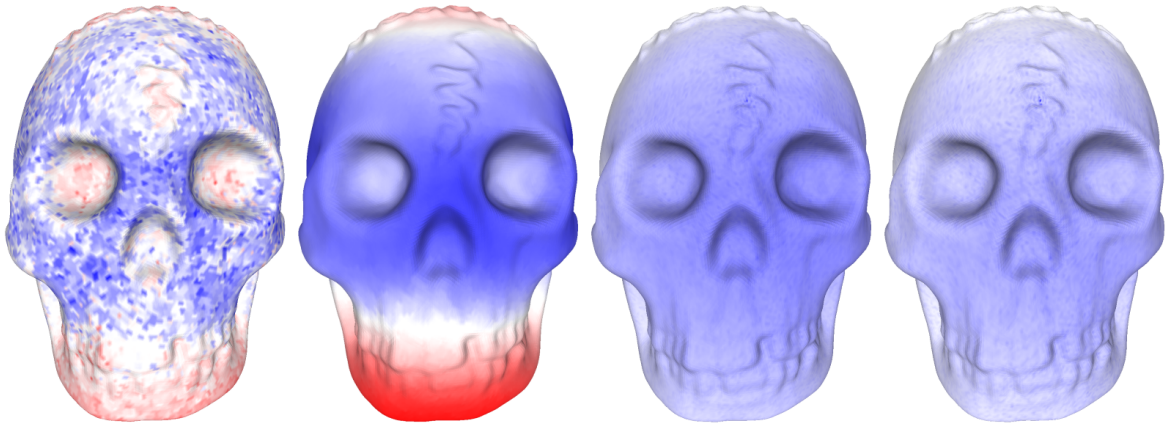


Figure 6.3: Annealing. Percentages of mass densities outside the interval [0.999, 1.001], minimum (blue) and maximum (red) densities. From left to right: After η stabilizes, 95%, [0.415, 2.41]; First step of annealing: 90%, [0.54, 2.1]; Second step: 90%, [0.97, 1.2]; Final map: 0.01%, [0.9902, 1.009].

6.2.3 Barycenters in diffusion space

Computing the geodesic barycenters η described in Sec. 6.1.3 is not tractable (Alg. 2 Step 3). Observe however, that we do not need to compute them on the surfaces: we only require an estimate of the (geodesic) variance. We thus resort to a surrogate version through diffusion distances using the first few eigenvectors of the *Laplace-Beltrami* operator precomputed on the input datasets. Since diffusion distances behave like Euclidean distances in diffusion space, we can explicitly solve for η in diffuson space by minimizing Eq. (6.3), i.e.,

$$\arg\min_{\eta_x} \left[\sum_{x_i \in X} W_x(x_i) \sum_{y_j \in Y} \left[\pi_{ij} \|\Phi(y_j) - \eta_x\|^2 \right] \right],$$

where $W_x(x_i)$ denotes the value of the weighting function W_x at x_i. We thus use the following closed form formula as an update rule for η once a map π is known:

$$\eta_x = \frac{\sum_{i,j} W_x(x_i)\, \pi_{ij} \Phi(y_j)}{\sum_{i,j} W_x(x_i)\, \pi_{ij}}. \tag{6.5}$$

6.2.4 Coarse to Fine Solver

Despite our use of a fast method to compute optimal transport plans, the number of variables $|X|+|Y|$ can be quite large, slowing down convergence of our alternating minimization. A common approach to both accelerate convergence and prevent the alternating minimization from getting stuck in local minima of the energy is to use a coarse to fine strategy. By starting the map optimization on a coarse sampling of the surfaces (and thus, a small number of variables) and progressively increasing resolution to refine the map, we significantly reduce the total computation time required to find a variance-minimizing map. Our approach consists in first establishing a hierarchical sampling where the cardinality of a neighbourhood, defined as $\mathcal{N}_x = \{x_i \in X : W_x(x_i) \geq \varepsilon\}$ as mentioned in Sec. 6.2.1, remains constant as we go to finer levels in the hierarchy, so that the number of variables needed at each level remains small.

Hierarchical Sampling. We leverage the smoothing properties of the time scale t in diffusion distance to construct a hierarchy of dyadic levels, one for each time scale $t_\ell = 2^{-\ell} t_0$. Since our weighting function is defined according to Eq. (6.4), reducing the time scale shrinks the support of W_x to a smaller neighborhood. Starting from a large diffusion time scale t_0, we build a sampling $\mathcal{S}_0$ of the input dataset via farthest point insertion until the farthest point p satisfies $W_x(p) > \tau \ \forall x \in \mathcal{S}_0$ for some $\tau > \varepsilon$ (we take $\tau = 0.9$ in our experiments), indicating a dense enough sampling for that level. In practice, this approach ensures that the area contributing to the neighborhood of any point x in the level (namely, the set of points z satisfying $\varepsilon < W_x(z) < \tau$) is covered by at most a constant number of Voronoi cells, leading to a constant neighborhood cardinality. As the time scale decreases, the levels $\mathcal{S}_\ell$ become more dense until, eventually, no additional points can be inserted: since only a limited number of eigenvectors are used to compute diffusion distances, our weighting functions will stop being "discerning" past a certain scale. Further levels can be computed using Euclidean distances instead: locally, geodesic and Euclidean distances behave the same at such scales.

In these Euclidean stages of the hierarchy, we thus revert to a plain (unnormalized) Gaussian function for the weighting function: we use

$$W_x(x_i) = \exp\left(-\frac{(x - x_i)^2}{2\sigma^2}\right), \tag{6.6}$$

where σ is initialized using the average radius r_{avg} of the neighborhoods in the last diffusion stage such that $\varepsilon = \exp(-r_{\text{avg}}^2 / 2\sigma^2)$. To avoid any ill-effects when σ tends to input data discretization limits, we always have a lower bound on the cardinality of the neighbourhood. Furthermore, for each new hierarchy level ℓ in Euclidean space, σ is directly updated to $\sigma/2$. Note that

the Euclidean hierarchy stages further improve results: using only the first few eigenvectors of the Laplace-Beltrami operator for diffusion distances would fail to capture local surface details corresponding to high frequencies.

Local Variable Instantiation. At the coarsest level of the hierarchy, we can afford to solve the full optimal transportation problem. In order to maintain a reasonable computational cost at higher resolutions, we limit the number of variables in the Sinkhorn iterations as follows. Consider the transport plan found so far; since our functional favors homeomorphisms, we can expect that the support of the transport plan is sparse since each point is mapped to only a small number of points as a consequence of variance minimization. Now, we consider a slightly enlarged support containing all pairs (x', y') such that there exists a pair (x, y) in the current support with $x' \in \mathcal{N}_x$ and $y' \in \mathcal{N}_y$. For the next Sinkhorn iterations, we limit our transport plan to this enlarged support, which limits the number of variables to instantiate (Fig. 6.4). Note that this process is repeated within each level of the hierarchy several times during our alternating minimization steps, so the support can gradually move to an optimal location during center relocation—thus eliminating the artifacts that a naive reduction of variables in the transport plan could induce. Once the map stops evolving, we move to the next hierarchy level.

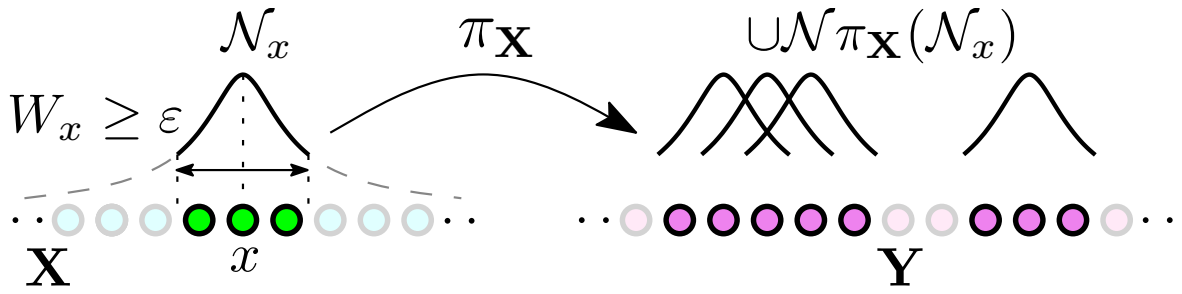


Figure 6.4: Local variable instantiation. For a point x, we instantiate variables only in the neighborhood of the mapping of its neighborhood. By localizing the samples it may be mapped to, we dramatically reduce the search space, without adverse effects.

Efficiency of Sinkhorn iterations. We leverage the locality of the transport map we are solving for to further accelerate the computation involved in Sinkhorn iterations. At a given level ℓ of the hierarchy, a naive implementation of Sinkhorn iterations would use a matrix $\mathbf{H}$ of size $|\mathcal{S}_\ell| \times |\mathcal{S}_\ell|$, and each iteration would have a complexity of $|\mathcal{S}_\ell| \cdot |\mathcal{S}_\ell|$. Instead, we exploit our hierarchical framework and the local instantiation of variables used by Sinkhorn iterations for storing $\mathbf{H}$ as a *sparse* matrix that stores only the instantiated variables. All other matrix coefficients are set to $10 \max\{\text{coef}\,(\pi_{ij})\}$ to strongly penalize transport.

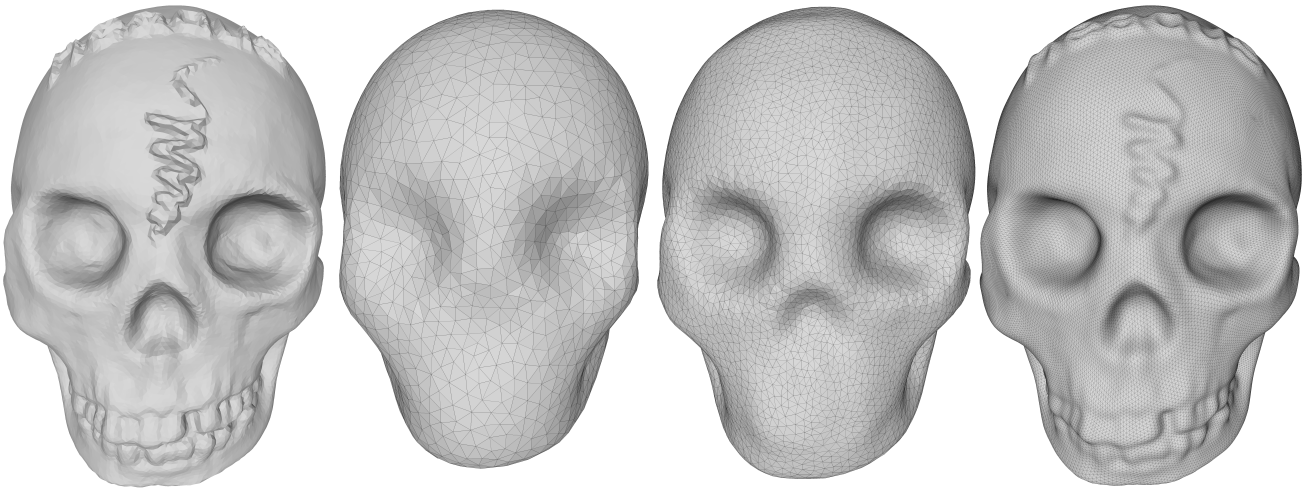


Figure 6.5: Hierarchical solving. An isotropic sphere mesh ($65kv$) is mapped to the input skull mesh ($35kv$) (left) by relocating the vertices of the sphere mesh onto the skull using η, across 3 Euclidean steps of the hierarchical solve (diffusion steps are not depicted as the corresponding η's live in diffusion space).

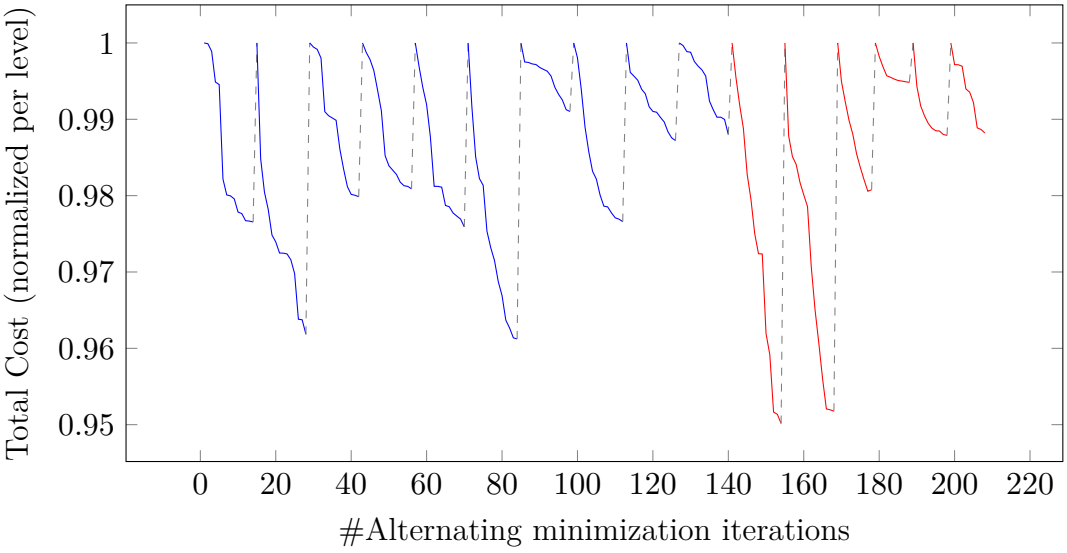


Figure 6.6: Progress of algorithm. We plot the relative decrease of the total cost (6.3) (normalized per level) at every alternating minimization iteration while optimizing the map between two dogs (Fig. 7.2). Blue depicts the diffusion distance phase (10 levels) and red depicts the Euclidean phase (6 levels). The dashed lines represent a change of hierarchy level.

Sinkhorn iterations are then computed efficiently by first precomputing $\sum_i \mathbf{m} \otimes \mathbf{v}[i]$ (resp. $\sum_j \mathbf{n} \otimes \mathbf{w}[j]$) in Alg. 1 and then accordingly adjusting the value of $\mathbf{H}(\mathbf{m} \otimes \mathbf{v})$ computed using the sparse matrix $\mathbf{H}$. As we keep the cardinality of neighborhoods constant during the construction of the hierarchy, both storage and computation involved in the optimal transport solver become in practice *linear* in the number of samples. Fig. 6.5 shows the result obtained via hierarchical

sampling, while Fig. 6.6 plots the total cost (normalized per level) over iterations while solving for a map. The total cost was normalized per level as it is hard to compare the cost between two levels as sample count, diffusion time and neighborhood evolve. Note that while each level corresponds to a decrease of only 4% of the total energy, each reduction has exponential impact across the hierarchy.

6.2.5 Flips and twists

We achieve dramatic speedup by approximating geodesic distances via diffusion distances, reducing the neighborhood over the hierarchy, and limiting the instantiation of variables to a small neighborhood. However, these algorithmic and numerical shortcuts lead to a practical drawback on surfaces with thin protrusions: a few, localized discontinuities of the map may appear in the form of twists (180 degree rotations of the map) and/or flips (mirrored versions of the map), see Fig. 6.7. While increasing the number of eigenvectors used for diffusion distances and increasing mesh density prevent these issues, it comes at the cost of increased running times.

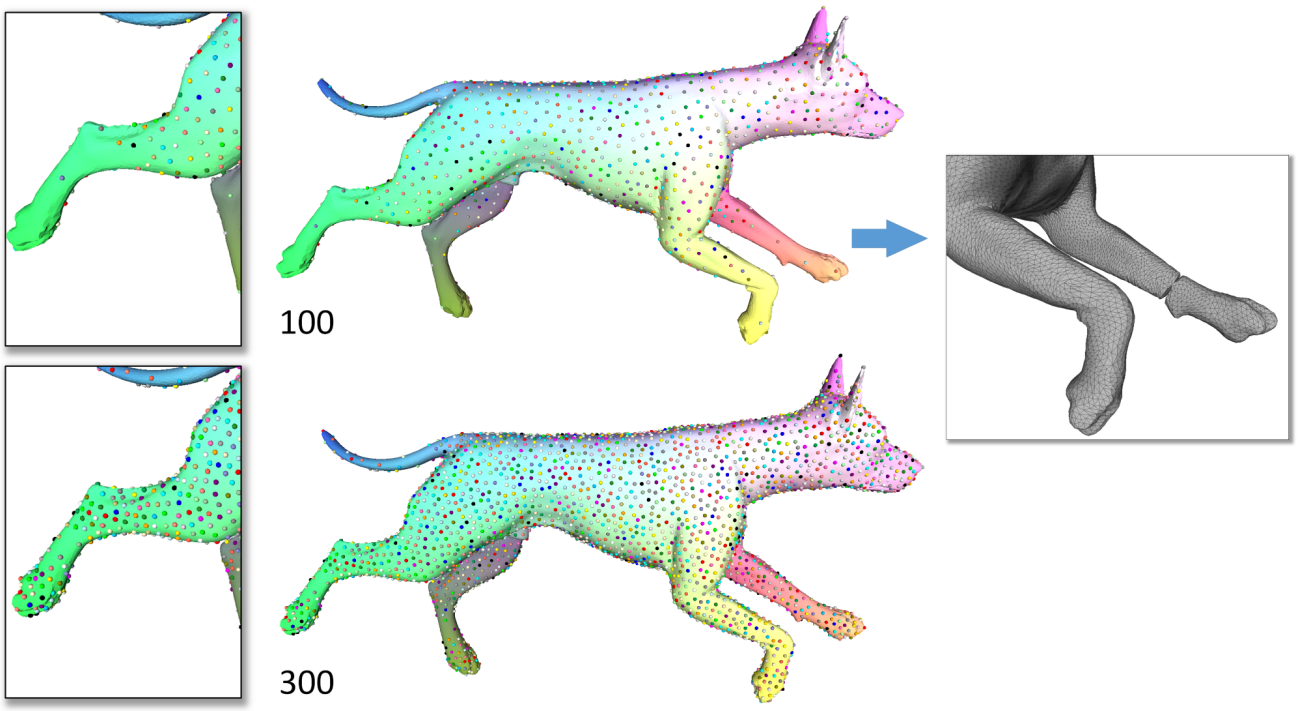


Figure 6.7: Diffusion distance estimates on thin parts. Colored spheres on the dog depict the final sampling during the last, finest diffusion stage. Top: with 100 eigenvectors, sampling is insufficient to get satisfactory precision on thin parts of the dog (feet, tail, ears). This may translate into twists of the mapping (right). Bottom: using 300 eigenvectors significantly improves precision.

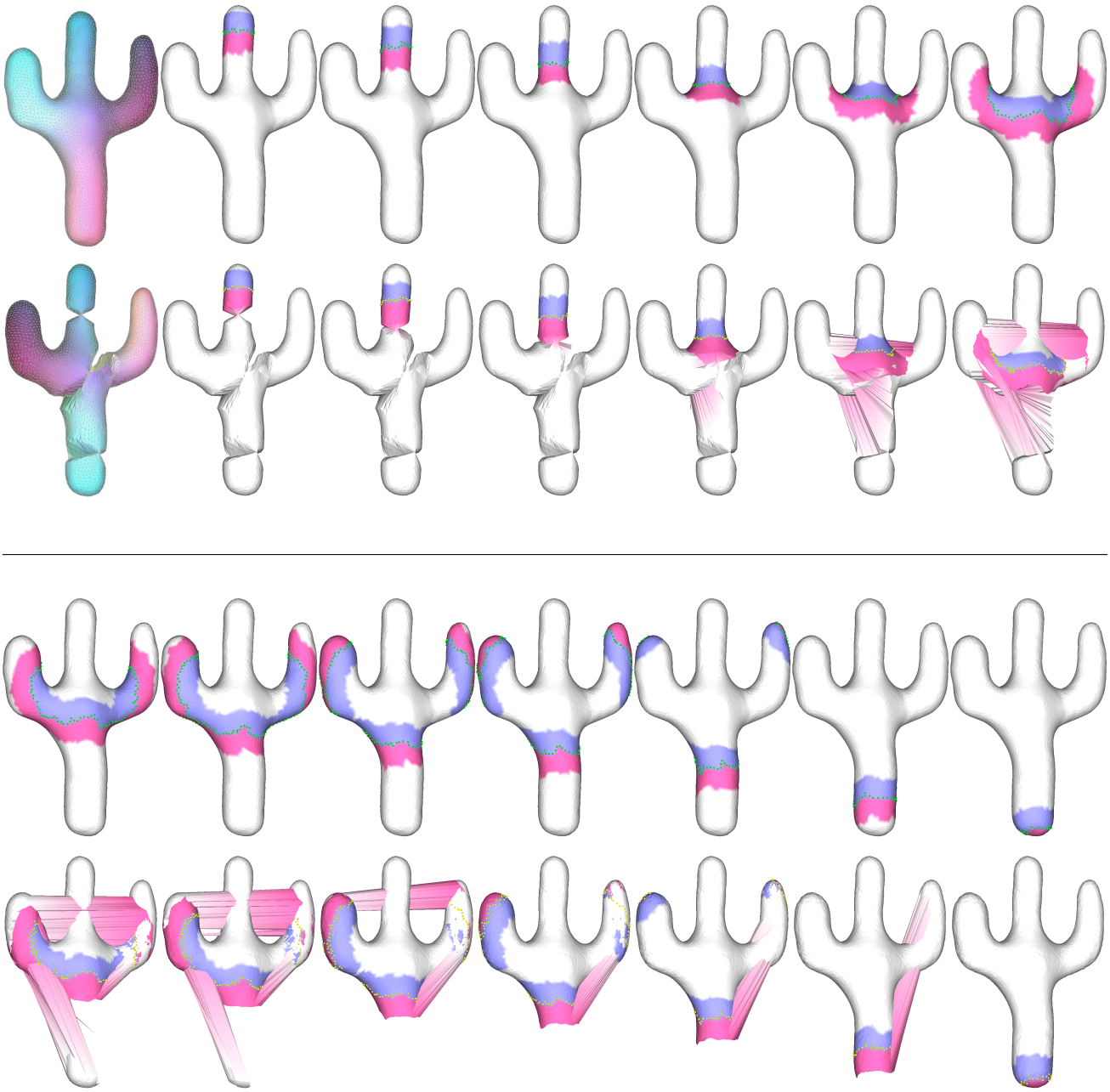


Figure 6.8: Fixing flips and twists. Starting from a suboptimal map having discontinuity (left, cactus (top) to itself (bottom) built by intentionally placing bad constraints), ironing proceeds through front propagation to remove all artifacts. For each strip, blue indicates the visited parts while the non visited parts are in pink. In case of multiple fronts, each front is considered independently. Note that the main idea behind whole approach is to slightly distort the neighborhood and favor already visited parts. Using high value for ω (close to 1) cannot unfold the mapping, while low values (close to 0) seriously pull η's towards already-visited parts.

Instead, we found more efficient to run the algorithm as is, and simply fix these small and local map artifacts at the end through what can be best described as "ironing": we correct problem areas by increasing the weights of correct neighborhoods during the η update, effectively

overpowering these twists and flips and bringing the solution out of the local minimum at which it was. More specifically, starting from a user defined point or a patch on X with a correct map in its neighborhood, we build a strip around a *front* (i.e., the delineation between already visited and non-visited points) by uniformly and geodesically thickening it. The corresponding strip is built on Y by geodesically thickening the image of the front on X proportional to the mass of the visited and non-visited parts. We now consider these two strips as two new (small) datasets, and solve for a variance minimizing transport plan satisfying Eq. (6.1) while keeping η for the visited part constant. The update η in non-visited part is achieved using an asymmetric weighting function:

$$W_x^*(y) = \begin{cases} W_x(y), & y \in \text{visited}, \\ \omega W_x(y), & \text{otherwise}, \end{cases} \tag{6.7}$$

where, ω set to 0.5 in all our experiments. Once these η stabilize, the front is moved outward and the process is repeated until covering the whole surface, eventually removing flips/twists as shown in Fig. 6.8 and the video in the additional material. This final ironing process is very efficient, and gracefully removes all remaining artifacts of the map and its inverse, even if we intentionally flip the map significantly as in Fig. 6.8.

6.3 Relaxing Area Preservation

When dealing with non isometric shapes, enforcing strict mass preservation leads to mappings with anisotropic differentials. Depending on the shapes, this anisotropy may become significant. Relaxing the mass preservation constraint helps remove this issue.

6.3.1 Sinkhorn Iterations for Relaxed Optimal Transport

A way to relax the mass preservation constraint in the framework of entropy-regularized optimal transport is to solve the following optimization problem:

$$\min_{\pi \in \Pi} \mathrm{KL}(\pi | \mathcal{K}_\gamma)$$

where Π denotes the set of matrices satisfying:

$$\begin{cases} \pi_{ij} \geq 0, \\ \sum_{i,j} \pi_{ij} = 1, \\ \alpha_i m_i \leq \sum_j \pi_{ij} \leq \beta_i m_i, \\ \alpha'_j n_j \leq \sum_i \pi_{ij} \leq \beta'_j n_j, \end{cases} \tag{6.8}$$

where, $0 \leq \alpha_k^{(\prime)} \leq 1 \leq \beta_k^{(\prime)}$ are the prescribed lower and upper bounds on the allowed mass distortion. In practice we use $\alpha_k = \beta_k = 1$ and $\alpha'_k = 1/\beta'_k$, which is a relaxation of the set of diffeomorphisms whose Jacobian and inverse Jacobian are bounded by a constant. We may write the convex set of allowed solutions as the intersection of two convex sets, namely the set Π_1 of matrices satisfying the first, second and third condition above, and the set Π_2 of matrices satisfying the first, second and fourth condition. Optimizing over their intersection can be done using *Dykstra's algorithm* as adapted to KL divergence in [Bauschke and Lewis, 1998; Benamou *et al.*, 2015]. This algorithm only requires projections of distributions on each convex set (in the sense of KL divergence), which we derive in App. B.

It turns out that the KL projections on Π_1 and Π_2 respectively boil down to row and column rescaling; because of these particular forms, the auxiliary variables used in the original Dykstra's algorithm do not play any role, and the overall algorithm reduces to iterated KL projections on Π_1 and Π_2. Also, since only row and column rescaling is used, we may represent the transport plan π at each iteration as a product of matrices associated to vectors $\mathbf{v}$ and $\mathbf{w}$ as in the usual Sinkhorn algorithm. A pseudocode for the algorithm is given below (Algo. 3).

Algorithm 3 Relaxed Sinkhorn iterations

function RELAXED_SINKHORN
  while !converged **do**
    $\mathbf{lb} \leftarrow \alpha \oslash \mathbf{H}(\mathbf{n} \otimes \mathbf{w})$ ▷ lower bound
    $\mathbf{ub} \leftarrow \beta \oslash \mathbf{H}(\mathbf{n} \otimes \mathbf{w})$ ▷ upper bound
    $r \leftarrow$ solution to GLOBAL_MASS_V($\mathbf{lb}, \mathbf{ub}$)
    $\mathbf{v} \leftarrow$ UPDATE_VECTOR_GLOBAL_MASS($r, \mathbf{lb}, \mathbf{ub}$)
    $\mathbf{lb} \leftarrow \alpha' \oslash \mathbf{H}^\top(\mathbf{m} \otimes \mathbf{v})$
    $\mathbf{ub} \leftarrow \beta' \oslash \mathbf{H}^\top(\mathbf{m} \otimes \mathbf{v})$
    $r \leftarrow$ solution to GLOBAL_MASS_W($\mathbf{lb}, \mathbf{ub}$)
    $\mathbf{w} \leftarrow$ UPDATE_VECTOR_GLOBAL_MASS($r, \mathbf{lb}, \mathbf{ub}$)
  return $\pi = \mathbf{D_m D_v H D_w D_n}$ ▷ Relaxed transport plan

Note that each KL projection requires solving an equation for determining parameter r, which must be chosen so that the resulting transport plan indeed has global mass equal to 1. Fortunately, the global mass is piecewise linear and non decreasing as a function of r, with

a number of nodes equal to twice the number of points in each dataset. Hence, solving these equations can be done by sorting these nodes, in time linear in the dataset size up to a logarithmic factor. Also, it may happen that the equations have no solution, in which case we set r to a large magnitude, with the appropriate sign.

Algorithm 4 Relaxed Sinkhorn subroutines

function $\mathbf{x}$ =UPDATE_VECTOR_GLOBAL_MASS($r, \mathbf{lb}, \mathbf{ub}$)
  for all i **do**
    if $\mathbf{lb}[i] \leq r \leq \mathbf{ub}[i]$ **then**
      $\mathbf{x}[i] \leftarrow r$
    else if $\mathbf{ub}[i] < r$ **then**
      $\mathbf{x}[i] \leftarrow \mathbf{ub}[i]$
    else
      $\mathbf{x}[i] \leftarrow \mathbf{lb}[i]$

function r = GLOBAL_MASS_V($\mathbf{lb}, \mathbf{ub}$)
  $\mathbf{x}(r) \leftarrow$UPDATE_VECTOR_GLOBAL_MASS($r, \mathbf{lb}, \mathbf{ub}$)
  $r \leftarrow$ solution to $\mathbf{m}^\top \mathbf{D_x H D_w n} = 1$

function r = GLOBAL_MASS_W($\mathbf{lb}, \mathbf{ub}$)
  $\mathbf{x}(r) \leftarrow$UPDATE_VECTOR_GLOBAL_MASS($r, \mathbf{lb}, \mathbf{ub}$)
  $r \leftarrow$ solution to $\mathbf{m}^\top \mathbf{D_v H D_x n} = 1$

6.3.2 Three-fold Alternating Minimizations

Our alternating minimization algorithm cannot be directly extended to the case where we relax the mass preservation constraint. Indeed, the denominator in our definition of transfer operators makes the transport plan optimization step more difficult, and seemingly not reducible to a convex problem. To solve this issue, we introduce additional variables m'_i and n'_j, which ideally would correspond respectively to $\sum_j \pi_{ij}$ and $\sum_i \pi_{ij}$. Rather than enforcing strict equality, we allow some slack and set the constraints

$$\begin{cases} C^{-1} m'_i \leq \sum_j \pi_{ij} \leq C m'_i, \\ C^{-1} n'_j \leq \sum_i \pi_{ij} \leq C n'_j. \end{cases} \tag{6.9}$$

for some constant C close to 1 (set to 2 and gradually decreased to 1.1 in our experiments). Additionally, we impose that $D^{-1} m_i \leq m'_i \leq D m_i$ and $D^{-1} n_j \leq n'_j \leq D n_j$, where D is a user-chosen bound on the allowed mass distortion. We can then form the following modified transfer operators

$$\pi'_X(\delta_{x_i}) = \frac{1}{m'_i} \sum_j \pi_{ij} \delta_{y_j}, \text{ and, } \pi'_Y(\delta_{y_j}) = \frac{1}{n'_j} \sum_i \pi_{ij} \delta_{x_i},$$

and optimize the functional $\mathcal{C}'(\pi, \eta)$ obtained from $\mathcal{C}(\pi, \eta)$ in (6.3) by replacing π_X by π'_X and π_Y by π'_Y.

We then proceed by alternately optimizing over the three variables π, (m', n'), and η. To solve for π_{ij}, we use the relaxed Sinkhorn iterations described above. Solving for m' and n' requires the optimization of a positive linear combination of inverses of their coefficients over a hypercube. This convex optimization problem is solved in near linear time using an algorithm similar to the one used in the previous paragraph for enforcing global mass preservation. Finally, the update of η remains a simple weighted average as before. Algorithm 5 provides a pseudocode for the overall algorithm. A result of this approach is depicted in Fig. 6.9.

Algorithm 5 Overall Algorithm

```
function OVERALL_ALGORITHM
    for all S_0, S_1 ... S_ℓ do // traverse hierarchy
        while !converged do
            π ← min C(., η, (m', n')) // solve relaxed optimal transport problem
            (m', n') ← min C(π, η, .) // update local mass relaxation
            η ← min C(π, ., (m', n')) // relocate centers to geodesic barycenters
    return [π, η, (m', n')]                          ▷ Area-relaxed Mapping
```

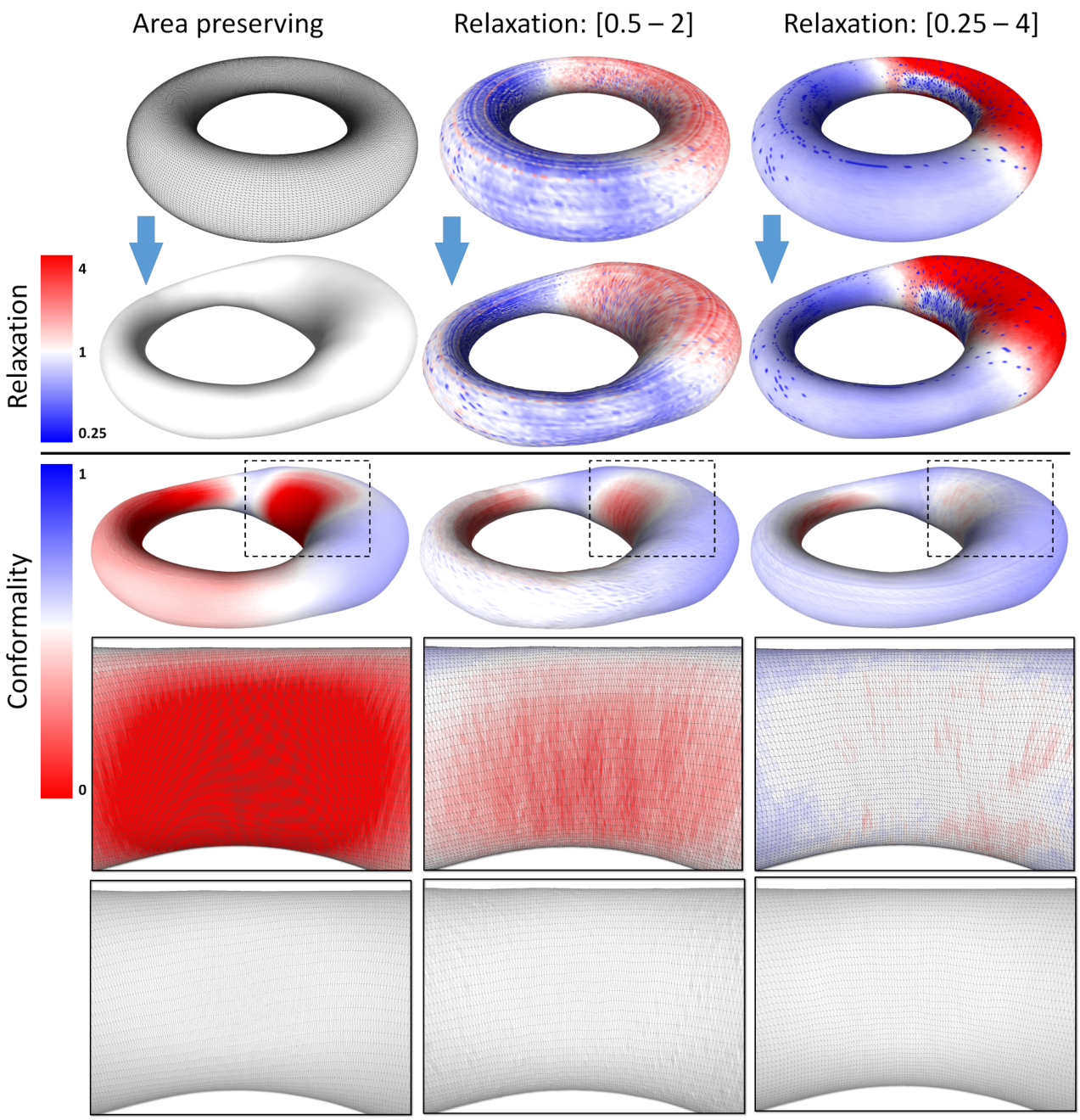


Figure 6.9: Relaxing area preservation. We compute a map between a canonical torus and a swollen torus using three different relaxation factors (from left to right : 1, 2 and 4). Top: colors indicate the relaxed mass transported to this location (blue=small, red=large). As expected the part mapping to the swollen region gets more mass and vice-versa. Middle: we depict the quasi-conformal distortion for each triangle, ranging from 0.0 (degenerated) to 1.0 (perfect). Larger relaxation factors get better conformality. When using no relaxation, the area is exactly preserved, hence triangles from the thin part of the torus get stretched due to tangential transport. Conversely, triangles are compressed in the opposite direction on the swollen part of the torus. Bottom: closeups on the triangle meshes to highlight improved conformality. Note that although the mass distribution is locally non smooth, the resulting map is smooth nonetheless since our transport stencils are several rings wide.

Chapter 7

Results

We implemented our algorithm in C++, using the *spectra* library [Qiu *et al.*, 2016] to compute the eigenvectors of the heat diffusion operator, the *Eigen* library for linear algebra operations, CGAL [The CGAL Project, 2016] for geometric computations. Parallelization was acheived using Intel Threading Building Blocks library. Unless otherwise indicated, we used the first 100 eigenvectors of the heat diffusion operator to approximate geodesic distances. For visual evaluation of the maps we use one color per vertex on the triangle meshes. Each vertex color is set based on a smooth 3D function, sometimes modulated by a 3D axis-aligned checkerboard (called grid coloring hereafter) to improve legibility.

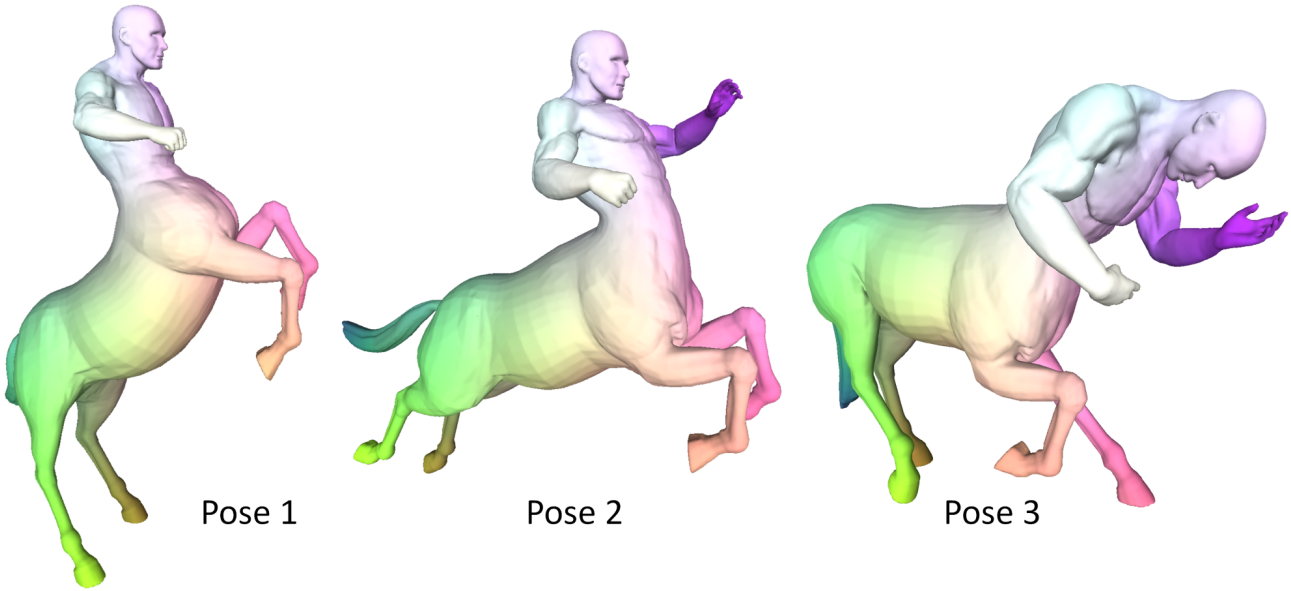


Figure 7.1: Centaur in three isometric poses. A dense map is successfully established between three isometric poses of the centaur model. No user interaction was required.

Initialization. When the input models are properly aligned in space (up to a translation), minimizing only the Wasserstein-2 transport distance is sufficient to initialize η for the first level of the hierarchy. Unless otherwise specified a very small additional Wasserstein-2 distance term (coefficient $1e-5$) is then added to our cost function in order to help tunneling out of local minima. When the input models are not aligned, we initialize with $\eta = 0$ and use no Wasserstein-2 distance term. A few (two to four) user-defined point constraints are often sufficient to obtain meaningful maps in these cases.

7.1 Experiments

Comparison against most of the methods which target conformality will show that they are more conformal while our method is able to produce area-relaxed maps which are as conformal as possible. Although such a direct comparison might not be meaningful in most of the cases, nevertheless, we evaluate our algorithm with the state of the art algorithms as follows:

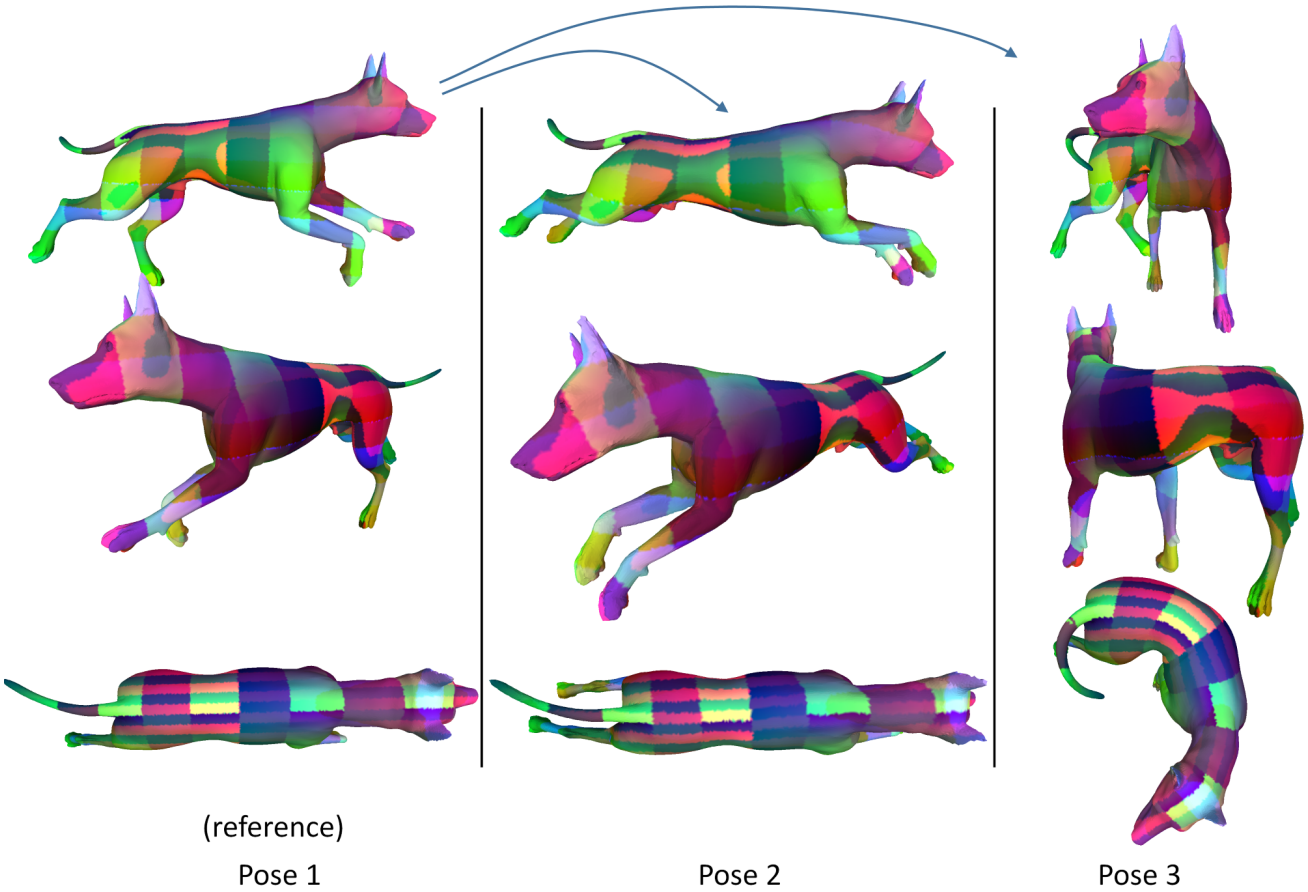


Figure 7.2: Isometric deformation of a dog. Our approach successfully establishes a proper dense map between three isometric poses of a dog; the left pose is chosen as reference for coloring.

Isometric Case. We begin our experiments on isometric models from the TOSCA dataset [Bronstein *et al.*, 2008]. Fig. 7.1 shows a centaur in three isometric poses. All maps are obtained without any user interaction in these cases. Fig. 7.2 shows a dog in three isometric poses. Except for the initialization stage we solve for mappings that minimize only the variance of the transport plans. The mapping to the middle dog is obtained without any user interaction, the Wasserstein-2 term used during initialization being sufficient to disambiguate the intrinsic left/right symmetry. The mapping to the right dog however requires 4 user-specified constraints, one per leg, to avoid left/right flipping. Grid coloring of the source dog (left) is mapped to the two target dogs (middle, right) using the variance-minimizing transport plan computed by our algorithm. For both the dog and centaur models the isometric maps are found without any area relaxation.

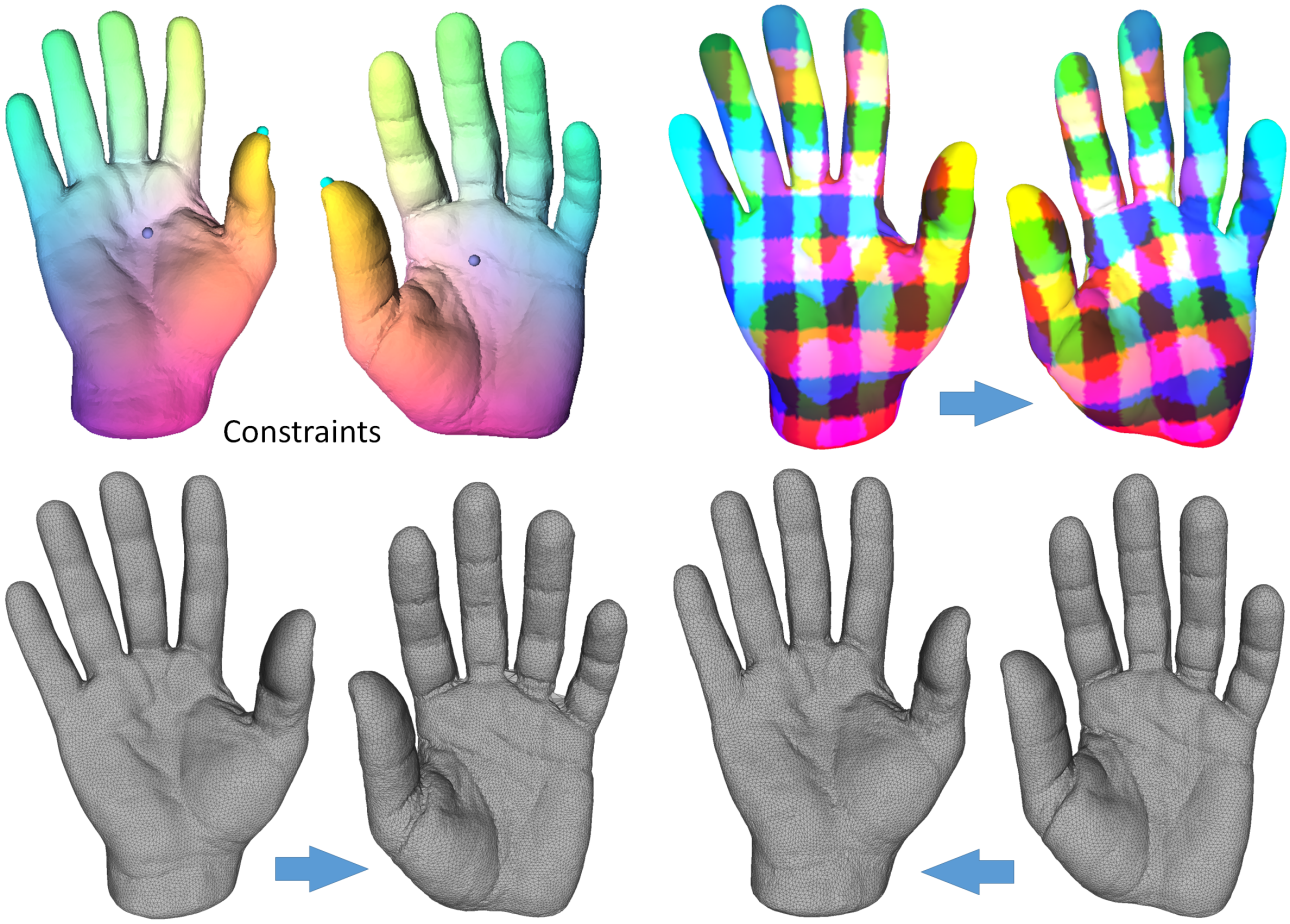


Figure 7.3: Mapping with user interaction. Top left: a source right hand is mapped to a target left hand using two user-defined constraints. Top right: the map is depicted through a color grid. Bottom left: source mapped onto the target mesh. Bottom right: target mapped onto the source mesh.

Non-isometric Case. We also ran our algorithm on models that are increasingly non isometric. Fig. 7.3 illustrates the mapping between two human hands. The hands differ in types (left vs.

right) and are non-isometric. In addition, one hand extends to the wrist. Two user-specified points (one to guide thumb to thumb and the other to disambiguate between front and back) are used as constraints to guide the minimization algorithm toward a proper mapping that turns one hand inside-out. We further illustrate the mapping through cross-remeshing, i.e., by relocating the mesh vertices onto the other model (and vice versa) to the center of the images (via the optimal mass transport plan) of each vertex neighborhood.

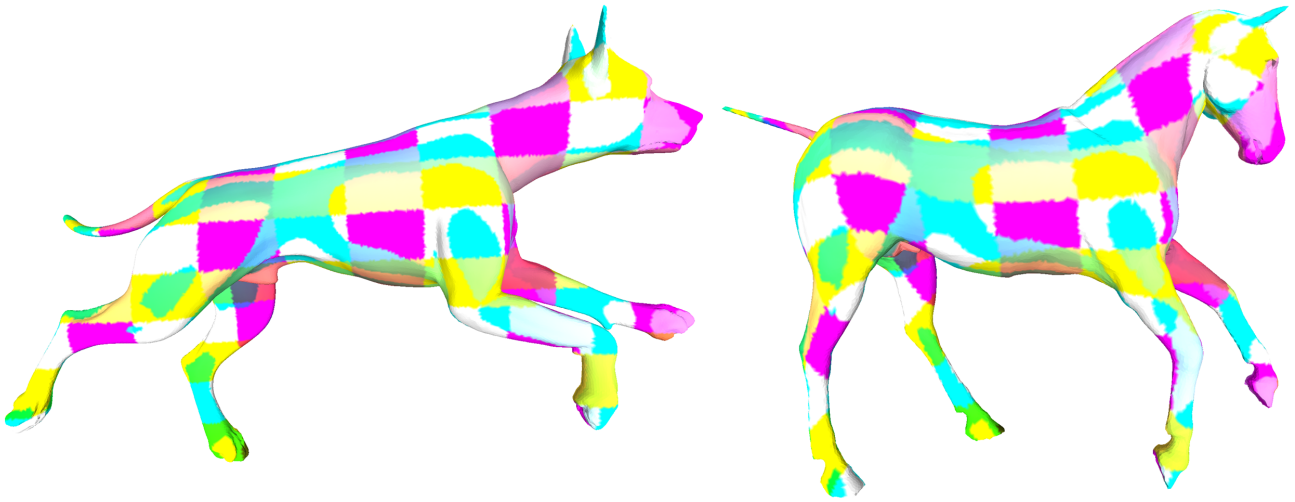


Figure 7.4: Non-isometric mapping. The dog model is mapped to the horse model without any user-specified constraints.

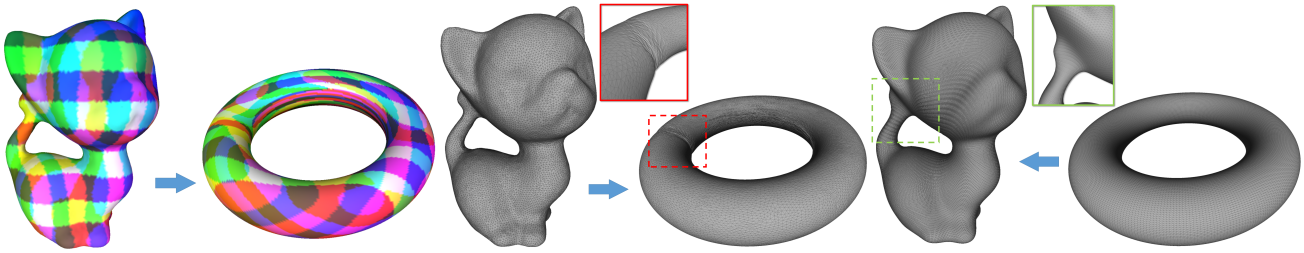


Figure 7.5: Highly non-isometric mapping. A kitten is mapped to a torus. Left: grid on kitten mapped to the torus. Middle: kitten mesh mapped to the torus. Right: torus mesh mapped to the kitten. Closeups highlight the angle distortion of the mapping.

Fig. 7.4 illustrates a fully automatic mapping between a dog and a horse. No point constraints were used for this example. Ears map to ears, legs to legs, and so on, solely due to our computation of a variance-minimizing map without any mass relaxation. We used 300 eigenvectors to get a reliable estimate of diffusion distances on the thin features - legs and tail - of the dog model: a smaller number of eigenvectors would lead to unreliable geodesic distance estimates, slowly translating into twists of the map along thin protrusions in the Euclidean stages (see Fig. 6.7) which would then rely totally on ironing.

Figure 7.6: Non-isometric mapping. An area relaxation (factor of 10) is used, along with 4 constraints (one at the extremity of each limb) to map the two very different shapes of a gorilla and a man. Requiring more relaxation to better improve the map further would face very slow convergence.

We challenge our algorithm in Fig. 7.5 by mapping between two highly non-isometric models of genus 1, a torus and a kitten, with no area relaxation. As expected, the map gets highly non-conformal in this example, but the resulting map is nice nonetheless.

Fig. 7.6 depicts a mapping between two non-isometric models, a man and a gorilla, with a maximum area relaxation factor set to 10 (i.e., area distortion allowed from 0.1 to 10).

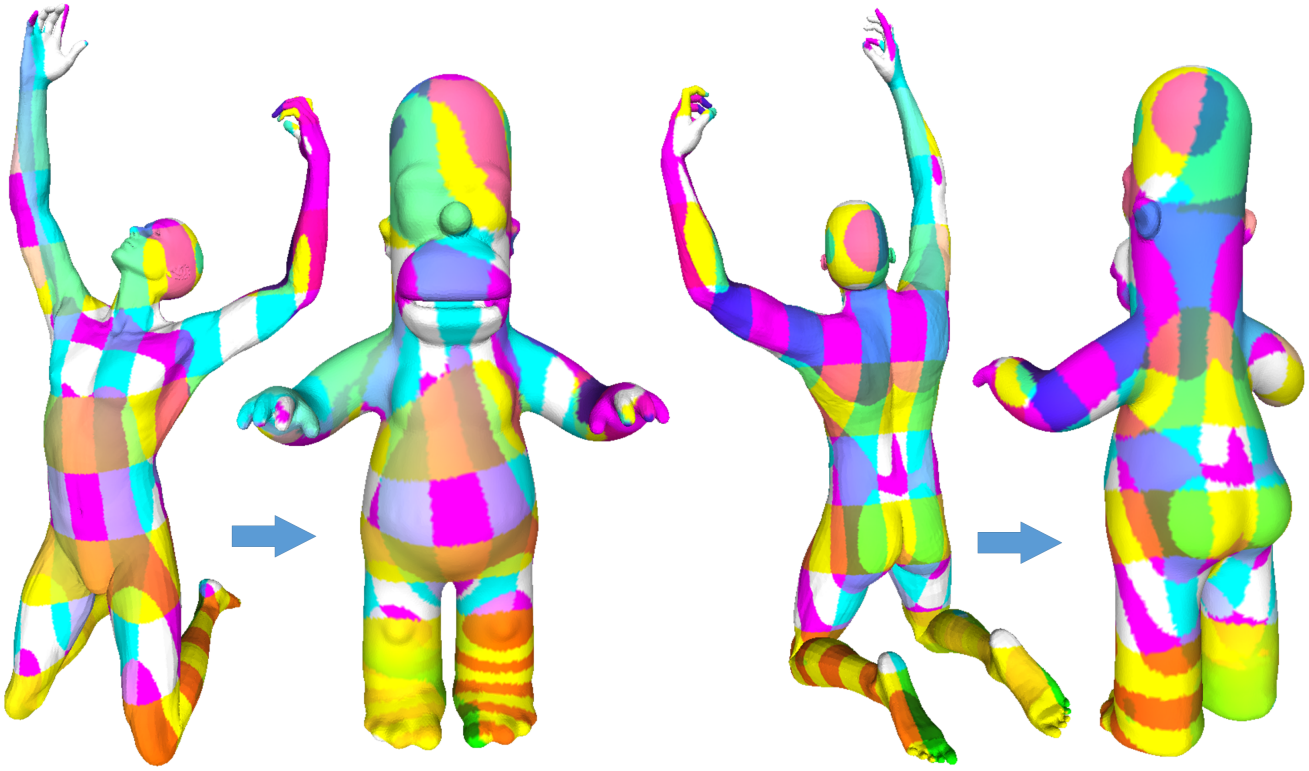


Figure 7.7: Non-isometric mapping. For this challenging example, no mass relaxation was used. Four constraints, one at the extremity of each limb were used to guide the map.

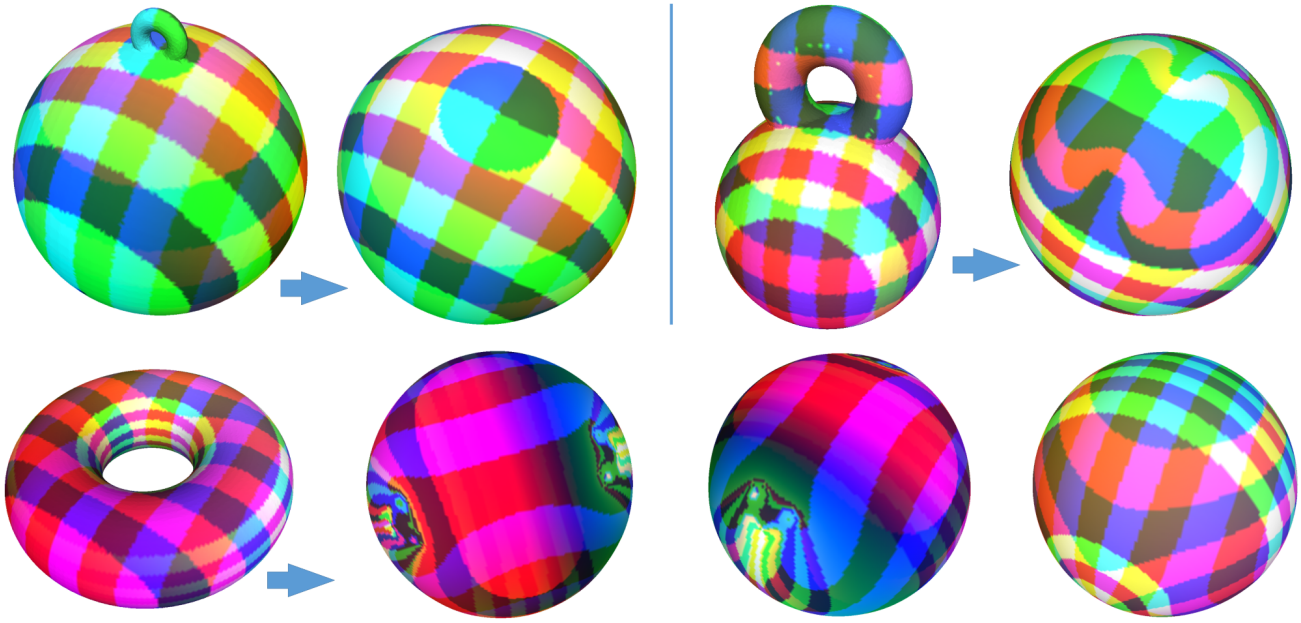


Figure 7.8: Mapping between different genus. Top left: a sphere with a small handle is mapped to a sphere. The mapping is discontinuous, but the discontinuity is local. Top right: same with a medium-size handle. Bottom: a regular torus is mapped to a sphere. The discontinuity is no longer local, but the map is continuous over half of the sphere.

Four constraints have been used (two hands and two feet). The resulting map is nice despite large differences in the proportion of the limbs.

Fig. 7.7 depicts a mapping between a man onto Homer, with no area relaxation. Four constraints are specified: two on the hands and two on the feet. As expected, the distortion is very large due to vast differences in shape, especially on the feet and head. However, semantic correspondences are well captured almost everywhere.

We further challenge our algorithm by mapping a sphere onto three genus-1 surfaces (Fig. 7.8), in order to evaluate the robustness to topological noise and analyze the mappings when the topology differs. We thus modeled (1) a sphere with a small handle, (2) a sphere with a larger handle and (3) a canonical torus. As expected the maps are discontinuous but the discontinuities are only local. No mass relaxation was used in these examples. Fig. 7.9 depicts a mapping between two poses of a kid. While on one pose the arm merges with his chest, the map is only little distorted.

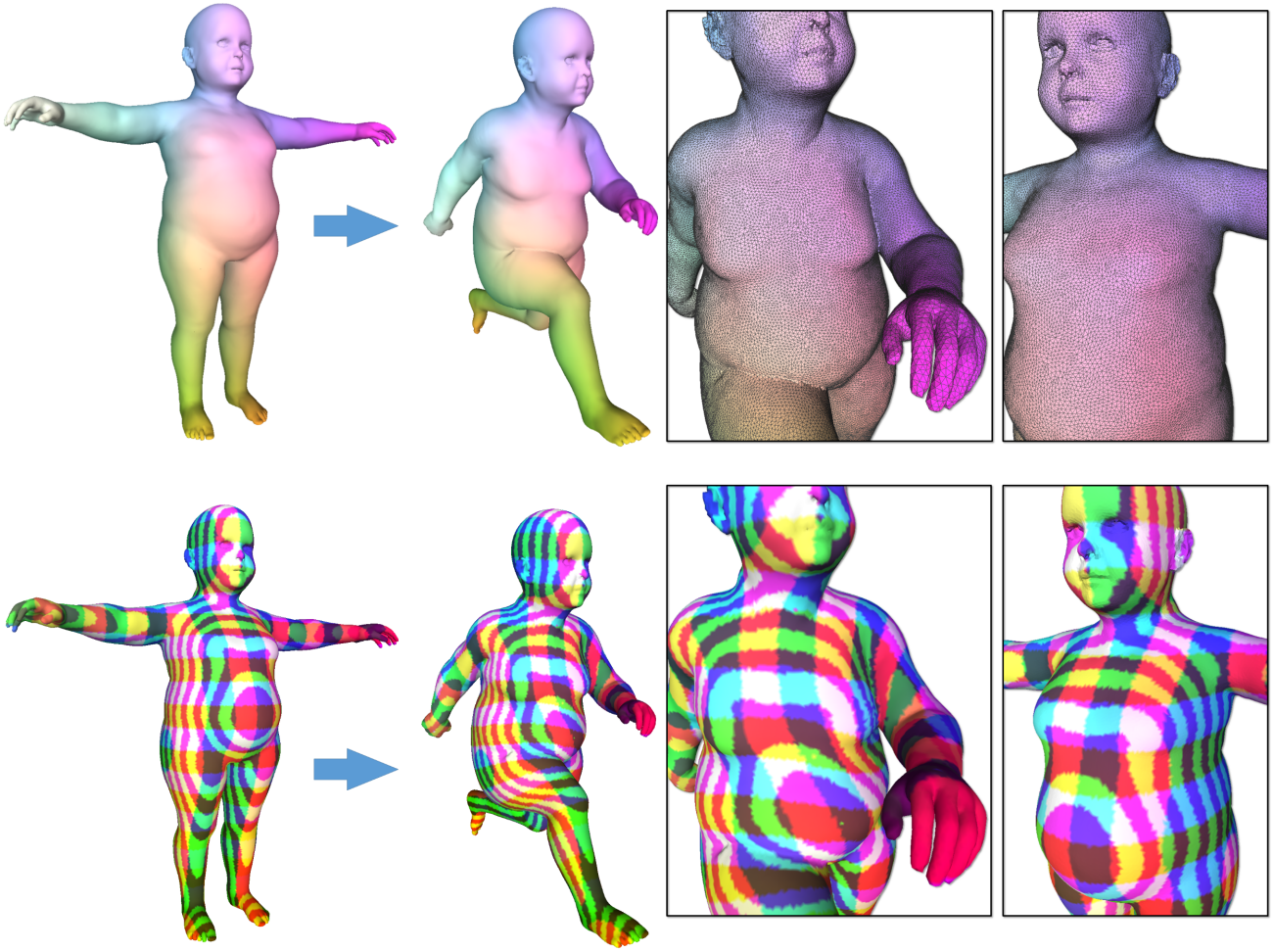


Figure 7.9: Mapping with local contact. While the arm of the baby touches (i.e., merges with) his chest on the second pose, we find a good map with 4 constraints (one at the extremity of each limb).

Comparisons. We first compare our approach on an isometric model from the TOSCA dataset, with the blended intrinsic maps (BIM) [Kim *et al.*, 2011] and a recent continuous matching approach (referred to as CM) contributed by Corman *et al.* [2015]. In Fig. 7.10 we plot the quality of correspondences for our approach, with or without ironing. Our method outperforms BIM and underperforms CM.

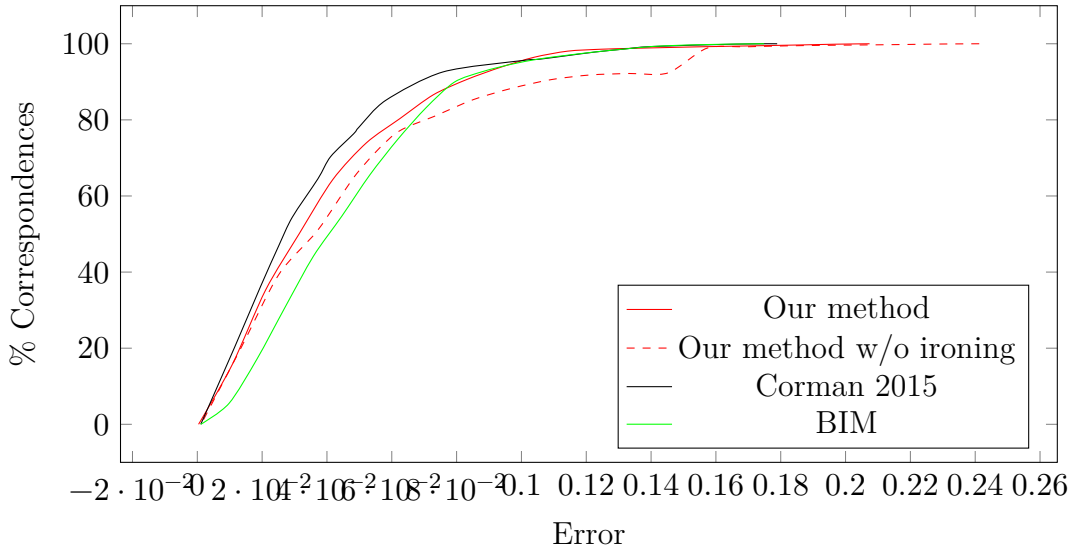


Figure 7.10: Comparisons. We plot the correspondence (in %) *vs* error on the dog model from the TOSCA dataset.

In Fig. 7.11 we evaluate our approach to our main target problem, i.e., non-isometric mappings, on six models from the body model dataset [Yang *et al.*, 2014], with area relaxation (factor 5). One man model is chosen as reference. We depict for each model the xyz mapping function as well the relaxed area. All maps are point-to-point and smooth, and the relaxed area reflects the differences in proportions of the limbs and improves the quality of the maps under the area relaxation bound. We also evaluate the correspondence quality and compare it to BIM, while taking as ground truth the semantic-based correspondences available from the dataset. Our method performs on average similarly to, or slightly better than, BIM. Note however that our approach is neither driven by semantic nor guided by feature points, hence we are not really competing on semantic-based correspondences. Our approach is instead devised to generate maps that are as conformal as possible within area preserving or area bounding constraints.

Timings. Computing the eigenvectors using *spectra* as a preprocessing step takes up to 2.5 seconds for 100 eigenvectors on a mesh with 5k vertices and 236.4 seconds for 200 eigenvectors on mesh with 150k vertices. Assuming k be the average neighborhood cardinality and n the total number of samples, the computation cost of each Sinkhorn iteration (including building/storing

H) is $\mathcal{O}(ck^2n)$, where c represents the transport plan spreading (average number of points to which a point is mapped) that depends on γ. Furthermore, we parallelize computations, requiring 2 to 5 mins for small (20kV) meshes, and up to 40 mins for 200kV. Note that without hierarchy and local variable instantiation, each Sinkhorn iteration would have a time and space complexity of $\mathcal{O}(n^2)$, with an additional $\mathcal{O}(ckn^2)$ time for building **H**. Despite its overall linear complexity, our approach is compute intensive due to three nested loops: the traversal of the hierarchy, the convergence of the geodesic centers of mass and the Sinkhorn iterations.

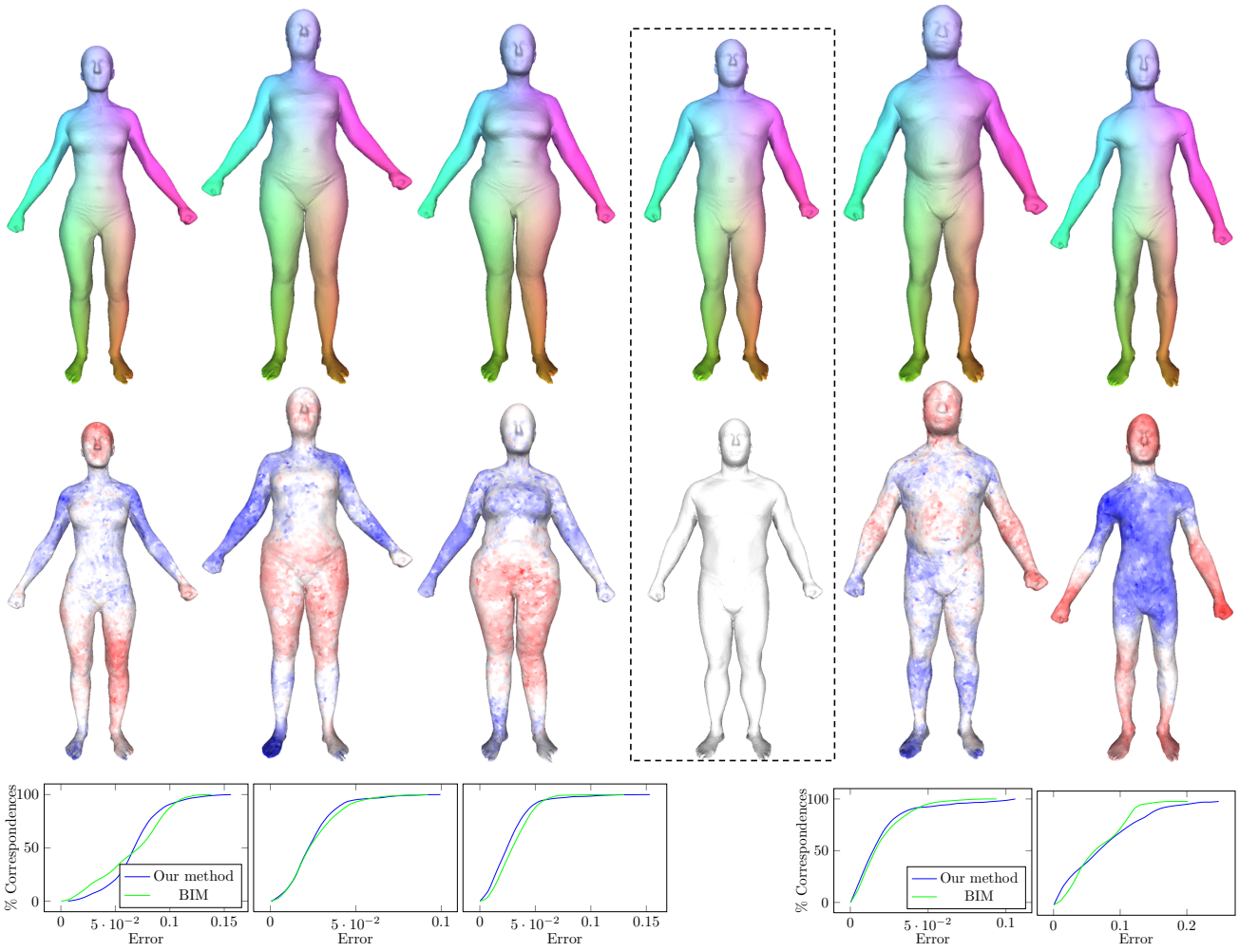


Figure 7.11: Mapping between human bodies. The reference model is outlined. We depict the mappings (xyz coloring) and the relaxed area. The curves depict the quality of the correspondences and offer a comparison to BIM.

Chapter 8

Conclusion

Recent advances in improving the efficiency of optimal transport solvers is providing an opportunity to derive inter-surface mapping through direct energy minimization. Our paper proposes a first variational model of the sort, where transport plans are optimized to be locally "sharp" so as to yield homeomorphisms. We showed that our approach finds a least-stretched map with exact area preservation or bounded area distortion, and demonstrated its performance on a range of challenging models, with possibly different or non-trivial topology.

Limitations. Although our algorithm provides a novel formulation, there is no direct practical application where it can be applied. It cannot be used directly for shape matching as our cost function does not target semantically correct map. For other applications involving finding a balance between conformality and area preservation, several methods which outperform ours (in terms of running times) already exist.

Furthermore, the algorithm is not robust to thin features and depends on ironing (Section 6.2.5) for the continuity of the map. However, it does not guarantee continuity in all the cases and may fail in practice. Note that the strip in ironing is generated by isotropically extending the front in the both directions proportional to the area. Hence, it might not work when area-relaxation is involved. Moreover, the ironing step mainly works only in cases where the topology of the front or the strip on both meshes is identical: appearance of any significant enough extra feature (thin protrusions) in one mesh will break it.

The numerics of our approach works best for uniform isotropic meshes since σ used for weighting function W is set proportionally to the vertex density, thus ensuring that the neighborhood cardinality remains nearly constant. As a consequence, we locally refine meshes with anisotropic vertex distribution as a preprocessing stage to ensure robustness. As relaxing the area preservation constraints through unbalanced mass transport increases the number of variables, we

cannot allow too large of a mass relaxation. In addition, if the mass is relaxed to the extent that a neighborhood maps to a single point, our algorithm is no longer able to favor conformality locally. An obvious solution is to increase the neighborhood (by adjusting σ), but this comes again at the cost of increased computation times.

Future Work. Along with Sinkhorn iterations, other compute-intensive parts can be transferred to GPU. This will significantly improve the running time of our algorithm. An obvious direction will be to incorporate shape signatures in our algorithm. We tried Heat Kernel Signature [Sun *et al.*, 2009], but it does not significantly improve the results because it is also based on the surface metric which is already captured by our objective function. One weakness of our approach is that convergence may be slow, especially on areas with continuous symmetries such as cylinders, and on models with a wide range of feature size. This is also probably relating to the convergence properties of Sinkhorn iterations for different sparse matrices. It will be interesting to better analyze and understand this behavior. One direction left for future work is to use a Newton iteration within the nested loops to accelerate convergence of the centroids. Recall that the centroids are just an approximation in Euclidean stages and do not lie on the surface. Another stimulating direction is implementing the concept of variable neighborhood size. This will allow us to directly proceed with our algorithm on anisotropic meshes (to some extent) without the need for isotropic refinement.

As future work we wish to explore the enduring challenge of partial matching which is currently not achievable through area relaxation. Exploring other geometric priors or bounds is also a stimulating avenue that we plan to tackle.

Appendices

Appendix A

Convergence to Dirichlet Energy

Let $f : S \mapsto S'$ be a diffeomorphism between two surfaces. Take for the weighting function at $x \in S$ a Gaussian with covariance $\sigma^2 I$, where σ is a small positive number. Let v be a random vector drawn accordingly. Then in the cost we define, the integrand at x is by definition the variance of $f(v)$. For small σ, we may approximate locally the two manifolds by their tangent spaces. Under this approximation $f(v)$ is a Gaussian with covariance $\sigma^2 DfDf^*$, hence $\text{var}(f(v)) = 2\sigma^2 ||Df||_F^2$. As a consequence, the total energy is proportional to the sum of the Dirichlet energy of f and the Dirichlet energy of its inverse.

The same conclusion holds for non Gaussian weighting functions, as long as they have isotropic covariances with total variance independent of x, and tending to 0.

Appendix B

KL projections

Let us derive the KL projection of a matrix ρ on Π_1, assuming the entries of ρ are positive. The gradient of the KL divergence to ρ at π is given by $\log \pi \oslash \rho$. By the optimality conditions, the projection π^* can be written as:

$$\log \pi^*_{ij}/\rho_{ij} = \lambda + \lambda_i + \xi_{ij}$$

where λ comes from the total mass constraint, λ_i from the row mass constraints, and ξ_{ij} from the nonnegativity condition. Parameter λ_i is zero when the mass bound is not attained, non negative if the lower mass bound is attained, and non positive if the upper mass bound is attained. Parameter ξ_{ij} is zero when π^*_{ij} is positive, and non negative otherwise. Hence $\pi^*_{ij} = \rho_{ij} e^{\lambda} e^{\lambda_i} e^{\xi_{ij}}$, from which we see that π^*_{ij} is positive, implying that $\xi = 0$. Let now $\phi_i = e^{\lambda} e^{\lambda_i}$ and $\phi = e^{\lambda}$. Using these variables, the optimality conditions read:

$$\phi_i \begin{cases} = \phi & \text{if } \sum_j \rho_{ij}\phi_i \in (\alpha_i m_i, \beta_i m_i) \\ \leq \phi & \text{if } \sum_j \rho_{ij}\phi_i = \beta_i m_i \\ \geq \phi & \text{if } \sum_j \rho_{ij}\phi_i = \alpha_i m_i \end{cases}$$

In particular, ϕ_i is determined by ϕ. More specifically, $\phi_i = f_i(\phi)$, where f_i is the closest point projection on interval $[\alpha_i m_i / \sum_j \rho_{ij}, \beta_i m_i / \sum_j \rho_{ij}]$, which is a non decreasing piecewise linear function with two nodes. In order to determine the correct value for ϕ, it suffices to solve the equation $\sum_{ij} \rho_{ij} f_i(\phi) = 1$, so that the global mass constraint is satisfied. Taking into account the change of variables due to the normalized masses $\mathbf{m}$ and $\mathbf{n}$ yields the algorithm described in Section 6.3.

Bibliography

Adamy, U., Giesen, J., and John, M., 2002. Surface reconstruction using umbrella filters. *Comput. Geom. Theory Appl.*, 21(1-2):63–86. ISSN 0925-7721.

Aflalo, Y., Kimmel, R., and Zibulevsky, M., 2013. Conformal mapping with as uniform as possible conformal factor. *SIAM J. Imaging Sciences*, 6(1):78–101.

Agarwal, P.K. and Suri, S., 1998. Surface approximation and geometric partitions. *Journal of Computing*, 27(4):1016–1035.

Aggarwal, A., Booth, H., O'Rourke, J., Suri, S., and Yap, C.K., 1985. Finding minimal convex nested polygons. In Proceedings of ACM Symposium on Computational Geometry, 296–304. ISBN 0-89791-163-6.

Aigerman, N. and Lipman, Y., 2015. Orbifold Tutte embeddings. *ACM Trans. Graph.*, 34(6):Art. 190.

Aigerman, N., Poranne, R., and Lipman, Y., 2014. Lifted bijections for low distortion surface mappings. *ACM Trans. Graph*, 33(4):Art. 69.

Aigerman, N., Poranne, R., and Lipman, Y., 2015. Seamless surface mappings. *ACM Trans. Graph.*, 34(4):Art. 72.

Alexa, M. and Wardetzky, M., 2011. Discrete laplacians on general polygonal meshes. *ACM Trans. Graph.*, 30(4):102:1–102:10. ISSN 0730-0301.

Alliez, P., Cohen-Steiner, D., Tong, Y., and Desbrun, M., 2007. Voronoi-based variational reconstruction of unoriented point sets. In Proceedings of the Fifth Eurographics Symposium on Geometry Processing, SGP '07, 39–48. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland. ISBN 978-3-905673-46-3.

Amenta, N. and Bern, M., 1998. Surface reconstruction by voronoi filtering. *Discrete and Computational Geometry*, 22:481–504.

Amenta, N., Bern, M., and Kamvysselis, M., 1998. A new voronoi-based surface reconstruction algorithm. In Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '98, 415–421. ACM, New York, NY, USA. ISBN 0-89791-999-8.

Amenta, N., Choi, S., Dey, T.K., and Leekha, N., 2000. A simple algorithm for homeomorphic surface reconstruction. In Proceedings of ACM Symposium on Computational Geometry, 213–222.

Amenta, N., Choi, S., and Kolluri, R.K., 2001a. The power crust. In Proceedings of the Sixth ACM Symposium on Solid Modeling and Applications, SMA '01, 249–266. ACM, New York, NY, USA. ISBN 1-58113-366-9.

Amenta, N., Choi, S., and Kolluri, R.K., 2001b. The power crust, unions of balls, and the medial axis transform. *Comput. Geom. Theory Appl.*, 19(2-3):127–153. ISSN 0925-7721.

Amenta, N., Peters, T.J., and Russell, A.C., 2003. Computational topology: Ambient isotopic approximation of 2-manifolds. *Theor. Comput. Sci.*, 305(1-3):3–15. ISSN 0304-3975.

Angenent, S., Haker, S., Tannenbaum, A., and Kikinis, R., 2000. On Area Preserving Mappings of Minimal Distortion, 275–286. Springer US, Boston, MA. ISBN 978-1-4615-5223-9.

Argudo, O., Brunet, P., Chica, A., and Vinacua, A., 2015. Biharmonic fields and mesh completion. *Graph. Models*, 82(C):137–148. ISSN 1524-0703.

Asirvatham, A., Praun, E., and Hoppe, H., 2005. Consistent spherical parameterization. In Int. Conf. Computational Science, 265–272. Springer.

Attali, D., Lieutier, A., and Salinas, D., 2011. Vietoris-rips complexes also provide topologically correct reconstructions of sampled shapes. In Proceedings of the Twenty-seventh Annual Symposium on Computational Geometry, SoCG '11, 491–500. ACM, New York, NY, USA. ISBN 978-1-4503-0682-9.

Attene, M., 2010. A lightweight approach to repairing digitized polygon meshes. *Vis. Comput.*, 26(11):1393–1406. ISSN 0178-2789.

Attene, M., Campen, M., and Kobbelt, L., 2013. Polygon mesh repairing: An application perspective. *ACM Comput. Surv.*, 45(2):15:1–15:33. ISSN 0360-0300.

Azernikov, S. and Fischer, A., 2005. Anisotropic meshing of implicit surfaces. In Proceedings of the International Conference on Shape Modeling and Applications 2005, SMI '05, 94–103. IEEE Computer Society, Washington, DC, USA. ISBN 0-7695-2379-X.

Bajaj, C.L. and Schikore, D.R., 1996. Error-bounded reduction of triangle meshes with multivariate data. In In Proceedings of SPIE Symposium on Visual Data Exploration and Analysis III, 34–45.

Bauschke, H.H. and Lewis, A.S., 1998. Dykstra's algorithm with bregman projections: a convergence proof. *Optimization*, 48:409–427.

Belkin, M., Sun, J., and Wang, Y., 2009. Constructing laplace operator from point clouds in rd. In Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA '09, 1031–1040. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA.

Benamou, J.D., Carlier, G., Cuturi, M., Nenna, L., and Peyré, G., 2015. Iterative bregman projections for regularized transportation problems. *SIAM J. Scient. Comp.*, 37(2).

Bennis, C., Vézien, J.M., and Iglésias, G., 1991. Piecewise surface flattening for non-distorted texture mapping. *SIGGRAPH Comput. Graph.*, 25(4):237–246. ISSN 0097-8930.

Bischoff, S., Pavic, D., and Kobbelt, L., 2005. Automatic restoration of polygon models. *ACM Transactions on Graphics*, 24:1332–1352.

Bohn, J.H., 1993. Automatic CAD-model Repair. Ph.D. thesis, Troy, NY, USA. UMI Order No. GAX94-05656.

Boissonnat, J.D. and Cazals, F., 2000. Smooth surface reconstruction via natural neighbour interpolation of distance functions. In Proceedings of ACM Symposium on Computational Geometry, 223–232. ISBN 1-58113-224-7.

Boissonnat, J.D. and Oudot, S., 2005. Provably good sampling and meshing of surfaces. *Graph. Models*, 67(5):405–451. ISSN 1524-0703.

Boissonnat, J.D., Wormser, C., and Yvinec, M., 2008. Locally uniform anisotropic meshing. In Proceedings of the Twenty-fourth Annual Symposium on Computational Geometry, SCG '08, 270–277. ACM, New York, NY, USA. ISBN 978-1-60558-071-5.

Boissonnat, J.D., Wormser, C., and Yvinec, M., 2014. Anisotropic Delaunay Mesh Generation. Research Report RR-7712, Inria Sophia Antipolis ; INRIA.

Borouchaki, H. and Frey, P., 2005. Simplification of surface mesh using hausdorff envelope. *Computer Methods in Applied Mechanics and Engineering*, 194(48-49):4864 – 4884.

Borouchaki, H., George, P.L., Hecht, F., Laug, P., and Saltel, E., 1997. Delaunay mesh generation governed by metric specifications. part i algorithms. *Finite Elem. Anal. Des.*, 25(1-2):61–83. ISSN 0168-874X.

Bossen, F. and Heckbert, P.S., 1996. A pliant method for anisotropic mesh generation. In INTERNATIONAL MESHING ROUNDTABLE, 63–74.

Botsch, M., Bommes, D., Vogel, C., and Kobbelt, L., 2004. GPU-based tolerance volumes for mesh processing. In Pacific Conference on Computer Graphics and Applications, 237–243. IEEE Computer Society.

Bregman, L., 1967. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Mathematical Physics*, 7(3):200 – 217. ISSN 0041-5553.

Bronstein, A.M., Bronstein, M.M., and Kimmel, R., 2006. Generalized multidimensional scaling: A framework for isometry-invariant partial surface matching. *Proceedings of the National Academy of Sciences*, 103(5):1168–1172.

Bronstein, A.M., Bronstein, M.M., and Kimmel, R., 2008. Numerical Geometry of Non-Rigid Shapes. Springer.

Burr, M., Choi, S.W., Galehouse, B., and Yap, C.K., 2012. Complete subdivision algorithms, ii: Isotopic meshing of singular algebraic curves. *J. Symb. Comput.*, 47(2):131–152. ISSN 0747-7171.

Carlier, G., Galichon, A., and Santambrogio, F., 2010. From knothe's transport to brenier's map and a continuation method for optimal transport. *SIAM Journal on Mathematical Analysis*, 41(6):2554–2576.

Chazal, F. and Cohen-Steiner, D., 2004. A condition for isotopic approximation. In Proceedings of ACM Symposium on Solid Modeling and Applications, 93–99. ISBN 3-905673-55-X.

Chazal, F., Cohen-Steiner, D., and Mérigot, Q., 2011. Geometric Inference for Measures based on Distance Functions. *Foundations of Computational Mathematics*, 11(6):733–751.

Chazal, F. and Lieutier, A., 2008. Smooth manifold reconstruction from noisy and non-uniform approximation with guarantees. *Comput. Geom. Theory Appl.*, 40(2):156–170. ISSN 0925-7721.

Chen, L., Sun, P., and Xu, J., 2007. Optimal anisotropic meshes for minimizing interpolation errors in l^p-norm. *Math. Comp*, 76:179–204.

Ciampalini, A., Cignoni, P., Montani, C., and Scopigno, R., 1997. Multiresolution decimation based on global error. *The Visual Computer*, 13(5):228–246.

Cignoni, P., Constanza, D., Montani, C., Rocchini, C., and Scopigno, R., 2000. Simplification of tetrahedral meshes with accurate error evaluation. In Proceedings of the Conference on Visualization '00, VIS '00, 85–92. IEEE Computer Society Press, Los Alamitos, CA, USA. ISBN 1-58113-309-X.

Cignoni, P., Montani, C., and Scopigno, R., 1998. A comparison of mesh simplification algorithms. *Computers and Graphics*, 22(1):37 – 54. ISSN 0097-8493.

Cignoni, P., Rocchini, C., and Scopigno, R., 1996. Metro: Measuring error on simplified surfaces. Technical report, Paris, France, France.

Cohen, J., Manocha, D., and Olano, M., 2003. Successive mappings: An approach to polygonal mesh simplification with guaranteed error bounds. *International Journal of Computational Geometry and Applications*, 13(1):61–96.

Cohen, J., Varshney, A., Manocha, D., Turk, G., Weber, H., Agarwal, P., Brooks, F., and Wright, W., 1996. Simplification envelopes. In Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '96, 119–128. ACM, New York, NY, USA. ISBN 0-89791-746-4.

Cohen-Steiner, D. and Da, F., 2004. A greedy delaunay-based surface reconstruction algorithm. *Vis. Comput.*, 20(1):4–16. ISSN 0178-2789.

Coifman, R.R., Lafon, S., Lee, A.B., Maggioni, M., Nadler, B., Warner, F., and Zucker, S.W., 2005. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *Proceedings of the National Academy of Sciences of the United States of America*, 102(21):7426–7431. ISSN 0027-8424.

Coll, N., Guerrieri, M., Rivara, M.C., and Sellarès, J.A., 2011. Adaptive simplification of huge sets of terrain grid data for geosciences applications. *Journal of Computational and Applied Mathematics*, 236(6):1410 – 1422. ISSN 0377-0427.

Corman, E., Ovsjanikov, M., and Chambolle, A., 2015. Continuous matching via vector field flow. *Comput. Graph. Forum*, 34(5):129–139.

Cuesta-Albertos, J. and Tuero-Díaz, A., 1993. A characterization for the solution of the monge—kantorovich mass transference problem. *Statistics and Probability Letters*, 16(2):147 – 152. ISSN 0167-7152.

Cuturi, M., 2013. Sinkhorn distances: Lightspeed computation of optimal transport. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, Advances in Neural Information Processing Systems, 26, 2292–2300. Curran Associates, Inc.

de Goes, F., Goldenstein, S., and Velho, L., 2008. A hierarchical segmentation of articulated bodies. In Proceedings of the Symposium on Geometry Processing, SGP '08, 1349–1356. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland.

Degener, P., Meseth, J., and Klein, R., 2003. An adaptable surface parameterization method. In J. Shepherd, editor, IMR, 201–213.

Desbrun, M., Kanso, E., and Tong, Y., 2006. Discrete differential forms for computational modeling. In ACM SIGGRAPH 2006 Courses, SIGGRAPH '06, 39–54. ACM, New York, NY, USA. ISBN 1-59593-364-6.

Desbrun, M., Meyer, M., and Alliez, P., 2002. Intrinsic parameterizations of surface meshes. *Computer Graphics Forum*. ISSN 1467-8659.

Dey, T.K., 2006. Curve and Surface Reconstruction: Algorithms with Mathematical Analysis (Cambridge Monographs on Applied and Computational Mathematics). Cambridge University Press. ISBN 0521863708.

Dey, T.K., Edelsbrunner, H., Guha, S., and Nekhayev, D.V., 1998. Topology preserving edge contraction. *Publ. Inst. Math. (Beograd) (N.S*, 66:23–45.

Dey, T.K. and Goswami, S., 2003. Tight cocone: A water-tight surface reconstructor. In Proceedings of ACM Symposium on Solid Modeling and Applications, 127–134. ISBN 1-58113-706-0.

Dey, T.K., Li, K., Ramos, E.A., and Wenger, R., 2009. Isotopic reconstruction of surfaces with boundaries. *Computer Graphics Forum*, 28(5):1371–1382.

Dey, T.K. and Sun, J., 2005. An adaptive mls surface for reconstruction with guarantees. In Proceedings of EUROGRAPHICS Symposium on Geometry Processing.

Digne, J., Cohen-Steiner, D., Alliez, P., Goes, F., and Desbrun, M., 2014. Feature-preserving surface reconstruction and simplification from defect-laden point sets. *J. Math. Imaging Vis.*, 48(2):369–382. ISSN 0924-9907.

Du, Q. and Wang, D., 2005. Anisotropic centroidal voronoi tessellations and their applications. *SIAM J. Sci. Comput.*, 26(3):737–761. ISSN 1064-8275.

Dykstra, R., 1983. An algorithm for restricted least squares regression. *J. Amer. Statist. Assoc.*, 78(384):837–842.

Edelsbrunner, H., 2003. Surface Reconstruction by Wrapping Finite Sets in Space, 379–404. Springer Berlin Heidelberg, Berlin, Heidelberg. ISBN 978-3-642-55566-4.

Fiorin, V., Cignoni, P., and Scopigno, R., 2007. Out-of-core mls reconstruction. In Proceedings of the Ninth IASTED International Conference on Computer Graphics and Imaging, CGIM '07, 27–34. ACTA Press, Anaheim, CA, USA. ISBN 978-0-88986-645-4.

Franklin, J. and Lorenz, J., 1989. On the scaling of multidimensional matrices. *Linear Algebra and its Applications*, 114/115:717–735. ISSN 0024-3795 (print), 1873-1856 (electronic).

Fu, X.M., Liu, Y., Snyder, J., and Guo, B., 2014. Anisotropic simplicial meshing using local convex functions. *ACM Trans. Graph.*, 33(6):182:1–182:11. ISSN 0730-0301.

Gallot, S., Besson, G., and Bérard, P., 1994. Embedding Riemannian manifolds by their heat kernel. *Geom. Funct. Anal.*, 4(4):373–398.

Garland, M. and Heckbert, P.S., 1997. Surface simplification using quadric error metrics. In Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '97, 209–216. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA. ISBN 0-89791-896-7.

Giraudot, S., Cohen-Steiner, D., and Alliez, P., 2013. Noise-adaptive shape reconstruction from raw point sets. In Proceedings of the Eleventh Eurographics/ACMSIGGRAPH Symposium on Geometry Processing, SGP '13, 229–238. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland.

Goes, F.d., Cohen-Steiner, D., Alliez, P., and Desbrun, M., 2011. An optimal transport approach to robust reconstruction and simplification of 2d shapes. *Computer Graphics Forum*. ISSN 1467-8659.

Gu, X. and Yau, S.T., 2003. Global conformal surface parameterization. In Proceedings of the 2003 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing, SGP '03, 127–137. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland. ISBN 1-58113-687-0.

Guéziec, A., 1996. Surface simplification inside a tolerance volume. Technical Report 20440. IBM Research Report RC 20440.

Guéziec, A., Taubin, G., Lazarus, F., and Horn, W., 1998. Converting sets of polygons to manifold surfaces by cutting and stitching. In Proceedings of the Conference on Visualization '98, VIS '98, 383–390. IEEE Computer Society Press, Los Alamitos, CA, USA. ISBN 1-58113-106-2.

Gumhold, S., Borodin, P., and Klein, R., 2003. Intersection free simplification. *International Journal of Shape Modeling*, 9(2):155–176.

Guskov, I. and Wood, Z.J., 2001. Topological noise removal. In Proceedings of Graphics Interface 2001, GI '01, 19–26. Canadian Information Processing Society, Toronto, Ont., Canada, Canada. ISBN 0-9688808-0-0.

Haker, S., Zhu, L., Tannenbaum, A., and Angenent, S., 2004. Optimal mass transport for registration and warping. *International Journal of Computer Vision*, 60(3):225–240. ISSN 1573-1405.

Hoppe, H., 1999. New quadric metric for simplifying meshes with appearance attributes. In Proceedings of the 10th IEEE Visualization 1999 Conference (VIS '99), VISUALIZATION '99, –. IEEE Computer Society, Washington, DC, USA. ISBN 0-7803-5897-X.

Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., and Stuetzle, W., 1992. Surface reconstruction from unorganized points. *SIGGRAPH Comput. Graph.*, 26(2):71–78. ISSN 0097-8930.

Hornung, A. and Kobbelt, L., 2006. Robust reconstruction of watertight 3d models from non-uniformly sampled point clouds without normal information. In Proceedings of EUROGRAPHICS Symposium on Geometry Processing, 41–50. ISBN 3-905673-36-3.

Igbida, N., Mazón, J., Rossi, J., and Toledo, J., 2011. A monge–kantorovich mass transport problem for a discrete distance. *Journal of Functional Analysis*, 260(12):3494 – 3534. ISSN 0022-1236.

Ju, T., 2004. Robust repair of polygonal models. *ACM Transactions on Graphics*, 23(3):888–895.

Kalvin, A.D. and Taylor, R.H., 1996. Superfaces: Polygonal mesh simplification with bounded error. *IEEE Computer Graphics and Applications*, 16(3).

Kantorovich, L.V., 1942. On the translocation of masses. *C. R. (Doklady) Acad. Sci. USSR*, 321:199–201.

Kazhdan, M., Bolitho, M., and Hoppe, H., 2006. Poisson surface reconstruction. In Proceedings of EUROGRAPHICS Symposium on Geometry Processing, 61–70.

Kim, S., Kim, C., and Levin, D., 2002. Surface simplification using a discrete curvature norm. *Computers and Graphics*, 26(5):657–663. ISSN 0097-8493.

Kim, V.G., Lipman, Y., and Funkhouser, T.A., 2011. Blended intrinsic maps. *ACM Trans. Graph*, 30(4):79.

Klein, R., Liebich, G., and Strasser, W., 1996. Mesh reduction with error control. In Proceedings of the 7th Conference on Visualization '96, VIS '96, 311–318. IEEE Computer Society Press, Los Alamitos, CA, USA. ISBN 0-89791-864-9.

Knight, P.A., 2008. The sinkhorn-knopp algorithm: Convergence and applications. *SIAM J. Matrix Anal. Appl.*, 30(1):261–275. ISSN 0895-4798.

Kobbelt, L., Campagna, S., and peter Seidel, H., 1998. A general framework for mesh decimation. In in Proceedings of Graphics Interface, 43–50.

Kraevoy, V. and Sheffer, A., 2004. Cross-parameterization and compatible remeshing of 3D models. In Proceedings of ACM SIGGRAPH, volume 23(3), 861–869.

Leibon, G. and Letscher, D., 2000. Delaunay triangulations and voronoi diagrams for riemannian manifolds. In Proceedings of the Sixteenth Annual Symposium on Computational Geometry, SCG '00, 341–349. ACM, New York, NY, USA. ISBN 1-58113-224-7.

Lévy, B. and Bonneel, N., 2013. Variational Anisotropic Surface Meshing with Voronoi Parallel Linear Enumeration, 349–366. Springer Berlin Heidelberg, Berlin, Heidelberg. ISBN 978-3-642-33573-0.

Lévy, B., Petitjean, S., Ray, N., and Maillot, J., 2002. Least squares conformal maps for automatic texture atlas generation. *ACM Trans. Graph.*, 21(3):362–371. ISSN 0730-0301.

Liepa, P., 2003. Filling holes in meshes. In Proceedings of the 2003 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing, SGP '03, 200–205. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland. ISBN 1-58113-687-0.

Lin, L. and Yap, C.K., 2009. Adaptive isotopic approximation of nonsingular curves: The parametrizability and nonlocal isotopy approach. In Proceedings of the Twenty-fifth Annual Symposium on Computational Geometry, SCG '09, 351–360. ACM, New York, NY, USA. ISBN 978-1-60558-501-7.

Lindstrom, P. and Turk, G., 1999. Evaluation of memoryless simplification. *IEEE Transactions on Visualization and Computer Graphics*, 5(2):98–115.

Lipman, Y., 2012. Bounded distortion mapping spaces for triangular meshes. *ACM Trans. Graph.*, 31(4):Art. 108.

Litman, R. and Bronstein, A.M., 2014. Learning spectral descriptors for deformable shape correspondence. *IEEE Trans. Pattern Anal. Mach. Intell*, 36(1):171–180.

Liu, L., Zhang, L., Xu, Y., Gotsman, C., and Gortler, S.J., 2008. A local/global approach to mesh parameterization. In Proceedings of the Symposium on Geometry Processing, SGP '08, 1495–1504. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland.

Liu, Y., Prabhakaran, B., and Guo, X., 2012. Point-based manifold harmonics. *IEEE Transactions on Visualization and Computer Graphics*, 18(10):1693–1703. ISSN 1077-2626.

Long, C. and Jin-chao, X., 2004. Optimal delaunay triangulations. *Journal of Computational Mathematics*, 22(2):299–308.

Luebke, D.P., 2001. A developer's survey of polygonal simplification algorithms. *IEEE Comput. Graph. Appl.*, 21(3):24–35. ISSN 0272-1716.

Meyer, M., Desbrun, M., Schröder, P., and Barr, A.H., 2003. Discrete Differential-Geometry Operators for Triangulated 2-Manifolds, 35–57. Springer Berlin Heidelberg, Berlin, Heidelberg. ISBN 978-3-662-05105-4.

Möbius, J. and Kobbelt, L., 2012. Openflipper: An open source geometry processing and rendering framework. In Proceedings of International Conference on Curves and Surfaces, 488–500. Springer-Verlag.

Monge, G., 1781. Mémoire sur la théorie des déblais et des remblais. *Histoire de l'Académie Royale des Sciences*, 666–704.

Moon, H.P., Choi, H.I., and Choi, S.W., 1997. Mathematical theory of medial axis transform.

Morigi, S. and Rucci, M., 2014. Multilevel mesh simplification. *The Visual Computer*, 30(5):479–492. ISSN 1432-2315.

Mullen, P., De Goes, F., Desbrun, M., Cohen-Steiner, D., and Alliez, P., 2010. Signing the unsigned: Robust surface reconstruction from raw pointsets. *Computer Graphics Forum*, 29(5):1733–1741. ISSN 1467-8659.

Myles, A. and Zorin, D., 2013. Controlled-distortion constrained global parametrization. *ACM Trans. Graph*, 32(4):105:1–105:14.

Nadler, B., Lafon, S., Kevrekidis, I., and Coifman, R.R., 2005. Diffusion maps, spectral clustering and eigenfunctions of Fokker-Planck operators. *Appl. Comput. Harmon. Anal.*, 21(1):113 – 127.

Ovreiu, E., Riveros Reyes, J.G., Valette, S., and Prost, R., 2012. Mesh simplification using a two-sided error minimization. In Proceedings of International Conference on Image, Vision and Computing, 26–30.

Ovsjanikov, M., Ben-Chen, M., Solomon, J., Butscher, A., and Guibas, L.J., 2012. Functional maps: a flexible representation of maps between shapes. *ACM Trans. Graph.*, 31(4):Art. 30.

Plantinga, S. and Vegter, G., 2004. Isotopic approximation of implicit curves and surfaces. In Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing, SGP '04, 245–254. ACM, New York, NY, USA. ISBN 3-905673-13-4.

Qiu, Y., Guennebaud, G., and Niesen, J., 2016. Spectra: A header-only c++ library for large scale eigenvalue problems.

Richter, R. and Alexa, M., 2015. Mahalanobis centroidal voronoi tessellations. *Comput. Graph.*, 46(C):48–54. ISSN 0097-8493.

Rodolà, E., Bulò, S.R., Windheuser, T., Vestner, M., and Cremers, D., 2014. Dense non-rigid shape correspondence using random forests. In Proceedings of CVPR, 4177–4184. IEEE.

Ronfard, R. and Rossignac, J., 1996. Full-range approximation of triangulated polyhedra. *Computer Graphics Forum*, 15(3):67–76.

Rustamov, R.M., 2007. Laplace-Beltrami eigenfunctions for deformation invariant shape representation. In Proceedings of the Fifth Eurographics Symposium on Geometry Processing, SGP '07, 225–233. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland. ISBN 978-3-905673-46-3.

Sakkalis, T. and Peters, T.J., 2003. Ambient isotopic approximations for surface reconstruction and interval solids. In Proceedings of the Eighth ACM Symposium on Solid Modeling and Applications, SM '03, 176–184. ACM, New York, NY, USA. ISBN 1-58113-706-0.

Schnabel, R., Degener, P., and Klein, R., 2009. Completion and reconstruction with primitive shapes. *Computer Graphics Forum (Proc. of Eurographics)*, 28(2):503–512.

Schreiner, J., Asirvatham, A., Praun, E., and Hoppe, H., 2004. Inter-surface mapping. In Proceedings of ACM SIGGRAPH, 870–877.

Shalom, S., Shamir, A., Zhang, H., and Cohen-Or, D., 2010. Cone carving for surface reconstruction. In ACM SIGGRAPH Asia 2010 Papers, SIGGRAPH ASIA '10, 150:1–150:10. ACM, New York, NY, USA. ISBN 978-1-4503-0439-9.

Shen, C., O'Brien, J.F., and Shewchuk, J.R., 2004. Interpolating and approximating implicit surfaces from polygon soup. *ACM Transactions on Graphics*, 23(3):896–904.

Shepard, D., 1968. A two-dimensional interpolation function for irregularly-spaced data. In Proceedings of the 1968 23rd ACM National Conference, ACM '68, 517–524. ACM, New York, NY, USA.

Shtern, A. and Kimmel, R., 2015. Spectral gradient fields embedding for nonrigid shape matching. *Computer Vision and Image Understanding*, 140:21–29.

Sinkhorn, R., 1964. A relationship between arbitrary positive matrices and doubly stochastic matrices. *Annals Math. Statist.*, 35.

Sinkhorn, R. and Knopp, P., 1967. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific J. Math.*, 21(2):343–348.

Smith, C.S. and Knott, M., 1987. Note on the optimal transportation of distributions. *Journal of Optimization Theory and Applications*, 52(2):323–329. ISSN 1573-2878.

Snyder, J.M., 1992. Interval analysis for computer graphics. *SIGGRAPH Comput. Graph.*, 26(2):121–130. ISSN 0097-8930.

Solomon, J., de Goes, F., Peyré, G., Cuturi, M., Butscher, A., Nguyen, A., Du, T., and Guibas, L., 2015. Convolutional wasserstein distances: Efficient optimal transportation on geometric domains. *ACM Trans. Graph.*, 34(4):Art. 66.

Solomon, J., Guibas, L.J., and Butscher, A., 2013. Dirichlet energy for analysis and synthesis of soft maps. *Comp. Graph. Forum*, 32(5):197–206.

Solomon, J., Nguyen, A., Butscher, A., Ben-Chen, M., and Guibas, L.J., 2012. Soft maps between surfaces. *Comp. Graph. Forum*, 31(5):1617–1626.

Solomon, J., Peyré, G., Kim, V., and Sra, S., 2016. Entropic metric alignment for correspondence problems. *ACM Trans. Graph.*

Sorkine, O., 2006. Differential representations for mesh processing. *Computer Graphics Forum*, 25(4):789–807.

Su, Z., Zeng, W., Shi, R., Wang, Y., Sun, J., and Gu, X., 2013. Area preserving brain mapping. In Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition, CVPR '13, 2235–2242. IEEE Computer Society, Washington, DC, USA. ISBN 978-0-7695-4989-7.

Sun, F., Choi, Y.K., Wang, W., Yan, D.M., Liu, Y., and Lévy, B., 2011. Obtuse triangle suppression in anisotropic meshes. *Comput. Aided Geom. Des.*, 28(9):537–548. ISSN 0167-8396.

Sun, J., Ovsjanikov, M., and Guibas, L., 2009. A concise and provably informative multi-scale signature based on heat diffusion. In Proceedings of the Symposium on Geometry Processing, SGP '09, 1383–1392. Eurographics Association.

The CGAL Project, 2016. CGAL User and Reference Manual. CGAL Editorial Board, 4.8 edition.

Vallet, B. and Lévy, B., 2008. Spectral geometry processing with manifold harmonics. *Computer Graphics Forum*, 27(2):251–260. ISSN 1467-8659.

van Kaick, O., Zhang, H., Hamarneh, G., and Cohen-Or, D., 2011. A survey on shape correspondence. *Comp. Graph. Forum*, 30(6):1681–1707.

Villani, C., 2003. Topics in Optimal Transportation. Graduate studies in mathematics. American Mathematical Society. ISBN 9780821833124.

Yang, Y., Yu, Y., Zhou, Y., Du, S., Davis, J., and Yang, R., 2014. Semantic parametric reshaping of human body models. In 3DV Workshop, 41–48.

Zelinka, S. and Garland, M., 2002. Permission grids: Practical, error-bounded simplification. *ACM Transactions on Graphics*, 21(2):207–229.

Zhong, Z., Guo, X., Wang, W., Lévy, B., Sun, F., Liu, Y., and Mao, W., 2013. Particle-based anisotropic surface meshing. *ACM Trans. Graph.*, 32(4):99:1–99:14. ISSN 0730-0301.

Zhong, Z., Shuai, L., Jin, M., and Guo, X., 2014. Anisotropic surface meshing with conformal embedding. *Graph. Models*, 76(5):468–483. ISSN 1524-0703.

Zhu, L., Haker, S., and Tannenbaum, A., 2003. Area-Preserving Mappings for the Visualization of Medical Structures, 277–284. Springer Berlin Heidelberg, Berlin, Heidelberg. ISBN 978-3-540-39903-2.