Seamless Surface Mappings

Noam Aigerman Roi Poranne Yaron Lipman
Weizmann Institute of Science

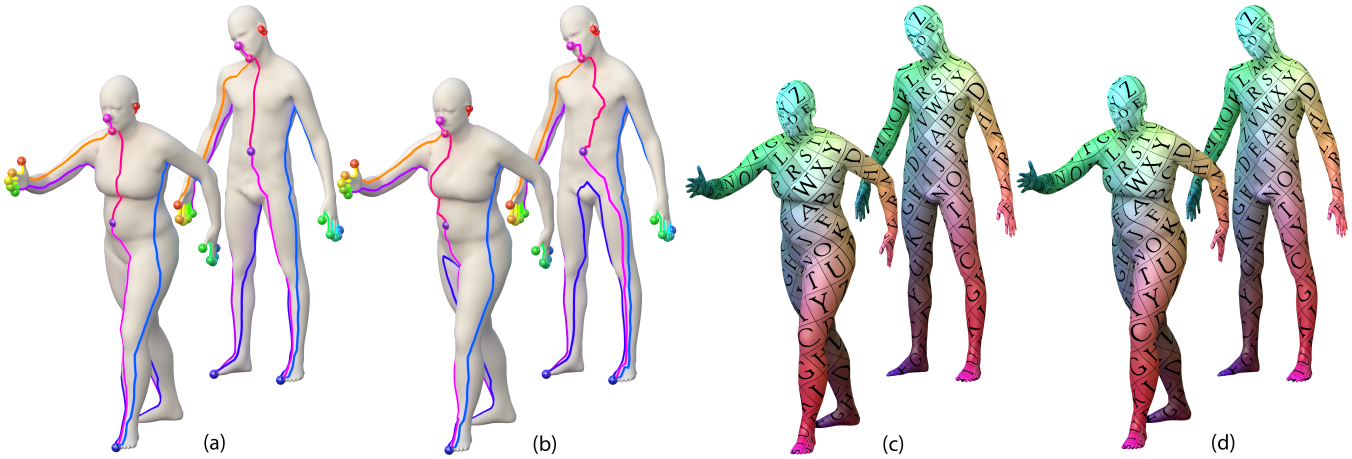


Figure 1: *Two bijective seamless mappings between models of two humans are shown in (c),(d), generated by our algorithm from the two different cut-placements in (a),(b) (respectively), cuts visualized as colored curves. The two maps interpolate the same set of user-given landmarks, shown as colored spheres. The maps are visualized by texturing the male model and transferring the texture to the female model using the mappings. The algorithm is not affected by the choice of cuts: the maps do not exhibit any artifacts near the cut nor does the poor cut-correspondence (e.g.the torso in (b)) affect them, and in fact for the two different cut-placements, the produced maps are identical.*

Abstract

We introduce a method for computing seamless bijective mappings between two surface-meshes that interpolates a given set of correspondences.
A common approach for computing a map between surfaces is to cut the surfaces to disks, flatten them to the plane, and extract the mapping from the flattenings by composing one flattening with the inverse of the other. So far, a significant drawback in this class of techniques is that the choice of cuts introduces a bias in the computation of the map that often causes visible artifacts and wrong correspondences.
In this paper we develop a surface mapping technique that is indifferent to the particular cut choice. This is achieved by a novel type of surface flattenings that encodes this cut-invariance, and when optimized with a suitable energy functional results in a seamless surface-to-surface map.
We show the algorithm enables producing high-quality seamless bijective maps for pairs of surfaces with a wide range of shape variability and from a small number of prescribed correspondences. We also used this framework to produce three-way, consistent and seamless mappings for triplets of surfaces.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling

Keywords: surface mesh, seamless, conformal distortion, bijective simplicial mappings

1 Introduction

Computation of mappings between surfaces is a core problem in computer graphics and vision, medical imaging and related fields of research. Many applications require the map to (i) be bijective, so as to establish point-to-point correspondences between the surfaces, and (ii) posses low-distortion to enable transferring of surface attributes with as little corruption as possible.

The basic task tackled in this paper is the computation of a low-distortion piecewise-linear bijective map $f : \mathbf{A} \to \mathbf{B}$ between two surfaces $\mathbf{A}$, $\mathbf{B}$, while interpolating a set of corresponding landmark points provided on the two surfaces, $p_i \in \mathbf{A}$, $q_i \in \mathbf{B}$.

A common approach for constructing such a map is to first cut the two surfaces consistently into topological disks (one disk, or many) and then map the disks to a common planar domain via some optimization process. The surface mapping is then induced according to the overlay of the flattened disks of one mesh over the flattened disks of the other mesh. This method often requires prescribing correspondences also along the cuts [Aigerman et al. 2014] or alternatively facing the challenging problem of optimizing also the disk-boundary correspondence and/or the cuts directly over the surfaces [Schreiner et al. 2004; Kraevoy and Sheffer 2004]. In most cases the choice of cuts affects the resulting mapping, and quite often the cut area will exhibit visible artifacts such as non-smoothness, distortion bias or wrong correspondences.

The goal of this paper is to develop an algorithm for constructing low-distortion bijective mappings of surfaces that are *seamless*, that is, indifferent to the choice of cuts, thereby relieving the user from the need to worry about cut placement. The key new insight is that there exist a set of rather simple conditions that, when enforced on the flattenings of the surfaces $\mathbf{A}, \mathbf{B}$, assures the resulting map $f : \mathbf{A} \to \mathbf{B}$ between the surfaces is seamless. We name flattenings which satisfy these conditions *G-flattenings*, where G stands for a group of transformations which relates planar copies of the mesh cut curves.

Figure 1 shows an example of two different mappings, (c) and (d), between two human models from the FAUST dataset [Bogo et al. 2014], computed by the algorithm when provided with two different cut placements, (a) and (b), respectively. Although the cuts are placed in different locations in (a) and (b), the two computed mappings are practically identical, and seamless, as there are no artifacts near the seams and the wrong correspondence of the seams has no effect on the produced map. See also Figure 6, for an experiment in which we completely change the chosen cuts.

We demonstrate that the algorithm developed in this paper can be used to produce high-quality bijective and seamless mappings between surface-meshes, and show, both theoretically and practically that the resulting maps are indifferent to homotopic changes in the cuts of the surfaces.

2 Previous work

Surface maps via base domains. Our work builds upon the common approach of computing surface maps using mesh-parameterizations ([Lee et al. 1999; Praun et al. 2001; Michikawa et al. 2001; Lin et al. 2003; Sheffer et al. 2006; Hormann et al. 2007]). In this approach, one computes mappings of the two meshes into a simpler base-domain, and then defines the surface map as the composition of the inverse of one parameterization over the other parameterization. In [Kraevoy and Sheffer 2004; Bradley et al. 2008; Schreiner et al. 2004] coarse meshes are used as the base domain. In [Aigerman et al. 2014] the mappings to the common planar domain are not required to be injective but rather only locally-injective, however they also specify the correspondences along geodesics connecting the input landmarks, thereby prescribing the image of the seams. [Steiner and Fischer 2005] generate seamless parameterizations of a mesh. [Tsui et al. 2013] compute a mapping between two spherical meshes interpolating a set of prescribed correspondences by first establishing a hyperbolic cone metric over the two surfaces and then minimize the Dirichlet energy of an initial plausible mapping between them.

Generalized mappings. Several works extend the notion of mappings, see [van Kaick et al. 2011] for a detailed survey. A common approach is to use ideas from metric-geometry, such as the Gromov-Hausdorff distance [Bronstein et al. 2006; Mémoli and Sapiro 2005]. Others use spectral embeddings [Jain et al. 2007; Ovsjanikov et al. 2010], or represent maps in the functional linear space [Ovsjanikov et al. 2012]. [Panozzo et al. 2013] represent maps using generalized weighted averages. Although able to produce good correspondences, these methods solve a slightly different problem as they do not produce a continuous bijective map.

Quadrangulation algorithms. Our work is also related to previous work dealing with quad-meshing (for a recent survey see [Bommes et al. 2013]). The general quadrangulation approach is to cut the mesh, parameterize it, and then overlay an integer-grid over the parameterization to define the quads. A main issue is ensuring the grid connects across the cuts in a seamless manner. This is guaranteed by requiring that planar copies of the same cut are related by automorphisms of the planar grid, *i.e.*, rotations by integer multiplication of $\pi/2$ and integer translations. Cross fields are used to guide the parameterization [Ray et al. 2006; Kälberer et al. 2007; Ray et al. 2008; Knöppel et al. 2013]. [Tong et al. 2006] use discrete harmonic forms to compute two scalar fields which define the quadrangulation. Bommes *et al.*, [2009] apply a mixed integer solver to ensure the grid connects across the seam while minimizing a smoothness energy defined via a cross-field. [Myles and Zorin 2012; Myles and Zorin 2013] compute a seamless metric while placing cone singularities to reduce curvature at other locations. Our method shares quite a few similar components with quadrangulation methods, in particular the fact that a transformation group relates copies of the cuts in the plane. The crucial difference is that in our case we apply this procedure to two surfaces simultaneously. Since the two parameterizations of the two surfaces are composed one with the inverse of the other to yield the final map, we only need to require that the transformation group G is invariant to the energy minimized on the flattenings. This allows using continuous groups like planar similarities which are tractable and are easy to optimize over, in strong contrast to the discrete group of planar grid automorphisms used in quadrangulations.

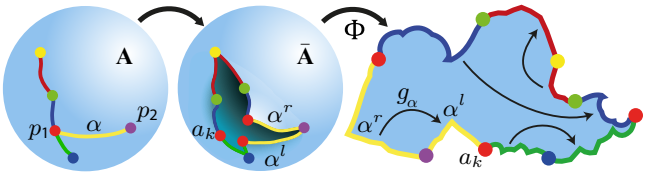


Figure 2: *G-flattening of a sphere. First the surface is cut to a disk (middle), and then flattened to the plane (right) so that twin seams (visualized using the same color) are related by transformations from a group of affine motions $g \in G$. Here G was chosen to be the group of similarity transformations, thus each seam is a scaled and rotated copy of its twin.*

3 Problem statement and overview

We assume to be given two surface-meshes, $\mathbf{A}$ and $\mathbf{B}$, with vertices, edges and faces $(\mathbf{V_A}, \mathbf{E_A}, \mathbf{T_A}), (\mathbf{V_B}, \mathbf{E_B}, \mathbf{T_B})$ (respectively). We are also given as input a set of corresponding landmarks $\mathcal{P} = \{p_i\} \subset \mathbf{V_A}$, $\mathcal{Q} = \{q_i\} \subset \mathbf{V_B}$. The output of our algorithm is a bijective surface-to-surface mapping $f : \mathbf{A} \to \mathbf{B}$ that maps the landmarks correctly, i.e. $f(p_i) = q_i$.

The algorithm consists of three stages: 1) The two surfaces $\mathbf{A}, \mathbf{B}$ are cut into topological disks $\bar{\mathbf{A}}, \bar{\mathbf{B}}$; 2) The two disks are mapped to the plane, each via a *G-flattening*, where G denotes a subgroup of the affine transformations. The G-flattenings of $\bar{\mathbf{A}}$ and $\bar{\mathbf{B}}$ are optimized to minimize some distortion energy E which is invariant to the group G; and 3) The final map $f : \mathbf{A} \to \mathbf{B}$ is computed from the flattenings.

The key ingredient in the algorithm above, and the reason the resulting map f is seamless is the G-flattening and the way it is used to define f. We proceed by laying out the foundations of our algorithm by first introducing the definition for G-flattenings of a single surface-mesh in Section 4. We then discuss how two G-flattenings can be used simultaneously to define a bijective mapping between surface-meshes in Section 5. In Section 6 we define the concept of *seamless surface maps* and characterize the G-flattenings which yield them. In Section 7 we present the details of our algorithm to compute seamless G-flattenings in practice and extract the final map. Section 8 presents an evaluation of the algorithm.

4 G - Flattening

In this section we define the concept of *G-flattening*, a specific type of flattening of disk-type meshes which is instrumental in our construction of seamless mappings. For this definition we may focus on a single surface-mesh, $\mathbf{A}$.

cut-graph. Before flattening is possible, the surface needs to be cut to a topological disk, as illustrated in Figure 2. In the first step of the algorithm, a *cut-graph* $\mathcal{G}_\mathbf{A} \subset \mathbf{E_A}$ spanning the landmarks $\mathcal{P}$ is computed, and the surface is cut along it into a topological disk. The cut-graph consists of a set of disjoint *seams* $\mathcal{G}_\mathbf{A} = \{\alpha_\ell\}$, where a seam, denoted henceforth without a subindex as $\alpha \in \mathcal{G}_\mathbf{A}$, is a

simple path[1] of edges, $\alpha \subset \mathbf{E_A}$, connecting a pair of landmarks $p_i, p_j \in \mathcal{P}$, *e.g.*, in Figure 2 (left), the seam α connects p_1 to p_2. Cutting $\mathbf{A}$ along $\mathcal{G}_\mathbf{A}$ transforms the mesh into a topological disk, as illustrated in Figure 2 (middle). We denote the cut mesh by $\bar{\mathbf{A}}$; $\bar{\mathbf{A}}$ is constructed by duplicating all vertices along the seams, and disconnecting triangles on opposite sides of a seam. The process of cutting the meshes creates two "twin" copies of each seam: the *left side* and the *right side* of the seam, which are denoted for a seam α as α^l, α^r, as depicted in Figure 2 (middle). We visualize the correspondence between twin seams by coloring them with the same color. Similarly, the cutting duplicates each landmark $p_i \in \mathcal{P}$ several times, where the number of copies of each landmark is the number of seams connected to it, for example in Figure 2 the number of copies of the red landmark p_1 is three, one of which is a_k. The landmark-copies are colored according to the color of the original landmark. To simplify notation we will denote the duplicated landmarks by $\bar{\mathcal{P}} = \{a_k\}$, where we assume we have logged for each duplicate landmark a_k its original landmark p_i.

Definition of G-flattenings. The flattenings considered in this paper are *simplicial mappings* of the cut surface-meshes, $\Phi : \bar{\mathbf{A}} \to \mathbb{R}^2$, *i.e.*, affine when restricted to any face of the mesh and globally continuous. Simplicial maps are defined by prescribing the image $\Phi(v) \in \mathbb{R}^2$ of every vertex $v \in \mathbf{V}_{\bar{\mathbf{A}}}$, and extending linearly to every face $t \in \mathbf{T}_{\bar{\mathbf{A}}}$.

A *G-flattening* is a specific type of a simplicial map: Given a subgroup $G = \{g\}$ of planar affine transformations, $\Phi : \bar{\mathbf{A}} \to \mathbb{R}^2$ is a G-flattening if it satisfies the following two properties:

1. *Local-injectivity*. That is, Φ is injective when restricted to every 1-ring of $\bar{\mathbf{A}}$, except possibly at the landmarks.
2. *G seam-transformations.* The planar images of both copies of each seam are related by a *seam-transformation* in the group G. That is, for every seam $\alpha \in \mathcal{G}_\mathbf{A}$ which the cutting has split into α^l, α^r, there exists $g_\alpha \in G$ such that

$$\Phi(\alpha^r) = g_\alpha \circ \Phi(\alpha^l). \tag{1}$$

A specific type of G-flattenings is used, for example, in surface quadrangulation [Bommes et al. 2013] where the particular choice of G is the group of integer-grid automorphisms in the plane, namely rotations by integer multiplications of $\pi/2$ and integer translations. For seamless surface mappings, the choice of G is related to the type of energy one wishes to optimize. As we will mainly optimize conformal distortion, we focus on the similarity-transformation group, that is the group of scaled rotations, which can be parameterized as the linear space,

$$G = \left\{ x \mapsto Tx + \eta \;\middle|\; T = \begin{pmatrix} a & -b \\ b & a \end{pmatrix} \in \mathbb{R}^{2\times 2}, \eta \in \mathbb{R}^{2\times 1} \right\} \tag{2}$$

See Figure 2 (right) for an illustration of a G-flattening with this choice of G, where a black arrow marks seams that are related by a seam-transformation from G. The theory we develop, however, is also applicable to other subgroups of affine transformations.

5 Surface maps induced by G-flattenings

Having defined G-flattenings on a single surface-mesh, in this section we use G-Flattenings of a pair of surface meshes to define a family of bijective surface maps $f : \mathbf{A} \to \mathbf{B}$. A main benefit in this representation is that the seams are not necessarily mapped to seams. Later, in Section 6, we will explain how to choose a *seamless* representative from this family.

[1] a path is simple if it is intersection-free.

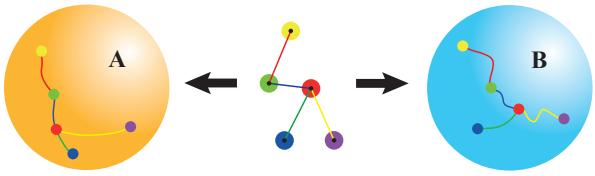


Figure 3: *The two surfaces (left, right) are cut according to the same cut-logic (middle).*

Similarly to the surface $\mathbf{A}$ in Section 4, we assume to have also cut the second surface-mesh, $\mathbf{B}$, to a topological disk $\bar{\mathbf{B}}$ using a cut-graph $\mathcal{G}_\mathbf{B} = \{\beta_\ell\}$ which is consistent with $\mathcal{G}_\mathbf{A}$, *i.e.*, they both share the same *cut-logic*, as depicted in Figure 3, middle. Namely, there is a cut $\alpha \in \mathcal{G}_\mathbf{A}$ connecting the landmarks p_i, p_j if and only if there is a cut $\beta \in \mathcal{G}_\mathbf{B}$ connecting q_i, q_j. We denote by $\bar{\mathcal{Q}} = \{b_k\}$ the copies of the landmarks $\mathcal{Q}$ created in the cutting process of $\mathbf{B}$, and note that the cut-graphs induce a bijective correspondence $a_k \leftrightarrow b_k$ between the duplicated landmarks $\bar{\mathcal{P}}$ and $\bar{\mathcal{Q}}$ in $\bar{\mathbf{A}}$ and $\bar{\mathbf{B}}$, respectively.

Given two G-Flattenings, $\Phi : \bar{\mathbf{A}} \to \mathbb{R}^2$, $\Psi : \bar{\mathbf{B}} \to \mathbb{R}^2$, we now consider the question: when do Φ, Ψ define a bijection $f : \mathbf{A} \to \mathbf{B}$ between the uncut meshes? If Φ, Ψ were both bijective mappings into the same planar domain, we could have defined $f = \Psi^{-1} \circ \Phi$. In that case corresponding seams α, β are mapped to one another as boundaries of the cut meshes.

As it turns out, for G-flattenings, the condition that seams are mapped to seams is not necessary for defining a bijective mapping between the surfaces. Leaving aside rather pathological cases for now, to ensure Φ, Ψ define a bijective $f : \mathbf{A} \to \mathbf{B}$ it is enough to merely add the requirement that Φ and Ψ map copies of matching landmarks to the same planar location, and that corresponding seams have the same seam-transformation. We refer to these requirements as the *G-mapping-conditions*:

Definition 1. *$\Phi : \bar{\boldsymbol{A}} \to \mathbb{R}^2, \Psi : \bar{\boldsymbol{B}} \to \mathbb{R}^2$ satisfy the G-mapping-conditions if*

1. *Φ, Ψ are G-flattenings.*
2. *Corresponding seams α, β share the same seam-transformation, that is $g_\alpha = g_\beta$.*
3. *All corresponding copies of landmarks a_k, b_k satisfy*

$$\Phi(a_k) = \Psi(b_k).$$

For the choice of the similarity-transformations group for G, the second condition is implied from the third one, as a similarity is uniquely defined by prescribing the image of two points.

To see how the G-mapping-conditions define a bijective map $f : \mathbf{A} \to \mathbf{B}$ consider the example depicted in Figure 4. In (a), two flattenings Φ, Ψ that satisfy the G-mapping-conditions are shown. By using the seam-transformations we can cut pieces from the flattening of $\bar{\mathbf{A}}$, that is from $\Phi(\bar{\mathbf{A}})$, and move them to match the flattening of $\bar{\mathbf{B}}$, as explained next: due to the G-mapping-conditions, two corresponding seams α, β share the same seam-transformation, $g_\alpha = g_\beta$. This entails that the area confined between the flattenings of α^l and β^l is in fact identical, up-to applying the seam-transformation g_α, to the area confined between the flattenings of α^r and β^r. Hence, cutting these areas from the G-flattening of $\bar{\mathbf{A}}$ accordingly and transforming them with g_α, as shown for one area in (b), makes the image of Φ identical to that of Ψ, as shown in (c). These operations are equivalent to moving the seams of $\mathcal{G}_\mathbf{A}$ on

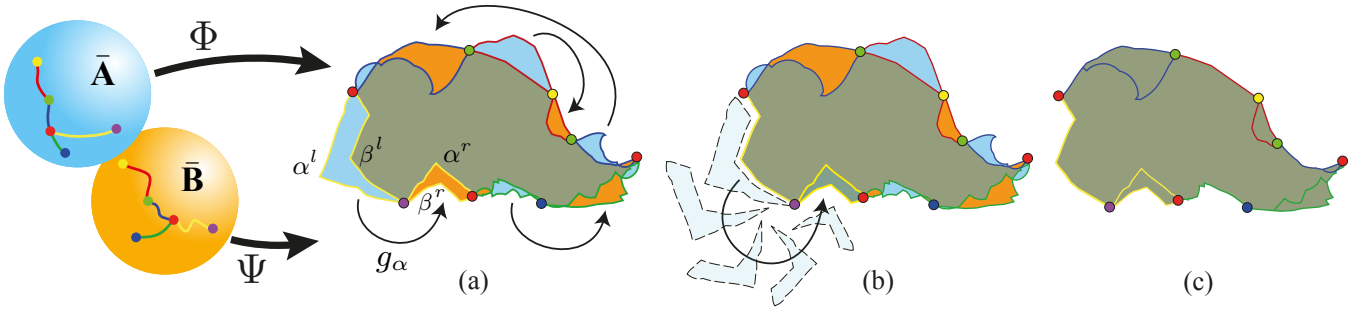


Figure 4: *Φ, Ψ which satisfy the G-mapping-conditions define a bijective map between **A** and **B**. In (a), two G-flattenings are shown. Although the images of the mappings Φ, Ψ do not coincide, the mutually-exclusive parts in orange and blue are perfect similarities of one another, as indicated with black arrows. As shown in (b), blue pieces can be cut and matched using the seam-transformation. Repeating this process leads to (c) where the planar images of Φ, Ψ are identical, allowing the definition of the map $f = \Psi^{-1} \circ \Phi$. These cut-and-paste operations are equivalent to moving the seams on the mesh **A**.*

the surface **A**. More precisely, each such operation replaces a single seam $\alpha \in \mathcal{G}_{\mathbf{A}}$ with a new one α'. Note that after each such operation the flattenings Φ, Ψ continue to satisfy the G-mapping-conditions.

Upon reaching the situation where the images of Φ, Ψ coincide, the mapping $f : \mathbf{A} \to \mathbf{B}$ can be inferred. In cases in which the mappings are injective, like the case depicted in Figure 4, one can simply define $f = \Psi^{-1} \circ \Phi$. More generally, the map f can be defined also in cases where Φ, Ψ are locally, but not globally injective. In these cases, after applying the cut-and-paste operations and reaching the situation where the images of Φ, Ψ coincide, the map f can be defined via the implicit relation $\Phi = \Psi \circ f$, and can be computed using a lifting algorithm, similar to the one detailed in Aigerman *et al.*, [2014]. That algorithm sequentially computes the map piece by piece by using the fact that locally the flattenings are injective and can be inverted. In practice, it is not required to actually perform the cut-and-paste operations described above, and the lifting algorithm can be directly adapted to our case by "jumping across the seams" using the seam-transformations. This will be detailed in Section 7.

As noted above, there are some pathological cases where local injectivity of the flattenings, and the fact that their images coincide, is not enough to guarantee the existence of a bijective map $f : \mathbf{A} \to \mathbf{B}$. As will be explained in Subsection 7.4, the algorithm we use to calculate Φ, Ψ is nevertheless guaranteed to avoid these cases.

6 Seamless mappings

In this section we detail how to compute a *seamless* mapping from the family of mappings defined in the previous section. Intuitively, a map is seamless if it presents no bias towards the cuts. The key observation is that a pair of flattenings $\Phi : \bar{\mathbf{A}} \to \mathbb{R}^2, \Psi : \bar{\mathbf{B}} \to \mathbb{R}^2$ which satisfy the G-mapping-conditions will define a seamless mapping between the surfaces $f : \mathbf{A} \to \mathbf{B}$ if they minimize a G-invariant energy functional. Specifically, we define the joint energy $\mathcal{E}$ of Φ and Ψ as

$$\mathcal{E}(\Phi, \Psi) = E(\Phi) + E(\Psi),$$

and ask that E is G-invariant, as explained next.

G-invariant energy. For achieving seamless mappings the energy functional E, and the group of transformations G should be related. More specifically, we require E to be invariant to compositions with transformations from the group G, that is, for all simplicial flattenings Φ we ask that

$$E(g \circ \Phi) = E(\Phi), \qquad \forall g \in G. \tag{3}$$

In this paper we choose E to be the L_2 average of the conformal distortion, which is invariant to composition with similarity transformations, our choice for the group G. We define,

$$E(\Phi) = \sum_{t_j \in \mathbf{T}_{\bar{\mathbf{A}}}} D_{\text{conf}}\left(A_j\right)^2 |t_j| \,, \tag{4}$$

where $A_j \in \mathbb{R}^{2\times 2}$ is the linear part (differential) of the affine map $\Phi|_{t_j}$ mapping the face t_j to the plane, $D_{\text{conf}}(A) = \Sigma(A)/\sigma(A)$ is the conformal distortion of the matrix A, namely the ratio of the maximal ($\Sigma(A)$) and minimal ($\sigma(A)$) singular values of A, and $|t_j|$ is the area of the face t_j. Although $D_{\text{conf}}(A)$ is not a convex energy and poses a certain challenge to optimize, we are unaware of existing simpler formulations of energies which measure other notions of discrete conformality of simplicial maps that are guaranteed to be similarity-invariant, produce locally-injective mappings, and allow easy incorporation of positional constraints. For instance, Least-Squares-Conformal-Maps [Lévy et al. 2002] measures the L_2 deviation of the mapping from satisfying the Cauchy-Riemann equations, and is convex and quadratic but is not scale invariant and in general does not guarantee local-injectivity as the mapping may triangles .

We are now ready to formulate the optimization problem that leads to a seamless mapping f,

$$\begin{aligned} &\min_{\Phi,\Psi} \quad \mathcal{E}(\Phi, \Psi) \\ &\text{s.t.} \quad \Phi, \Psi \ \text{ satisfy the } G\text{-mapping-conditions.} \end{aligned} \tag{5}$$

The algorithmic details of how we solve problem (5) in practice are discussed in Section 7. We next explain in what sense do the local minimizers Φ^*, Ψ^* of (5) define a seamless mapping $f : \mathbf{A} \to \mathbf{B}$. We first formally define what is a seamless mapping, and then show that f constructed from Φ^*, Ψ^* satisfies this definition.

Definition of seamless mappings. To properly define which mappings $\{f\}$ that are defined via a pair of G-flattenings $\Phi : \bar{\mathbf{A}} \to \mathbb{R}^2, \Psi : \bar{\mathbf{B}} \to \mathbb{R}^2$ are seamless, we need to introduce the notion of homotopies of seams and cut-graphs. A *seam homotopy* is a modification of a seam $\alpha \in \mathcal{G}_{\mathbf{A}}$ on the mesh **A** by repeatedly applying the atomic operation of continuously moving faces from one side of the seam to the other side while maintaining that the seam is simple.

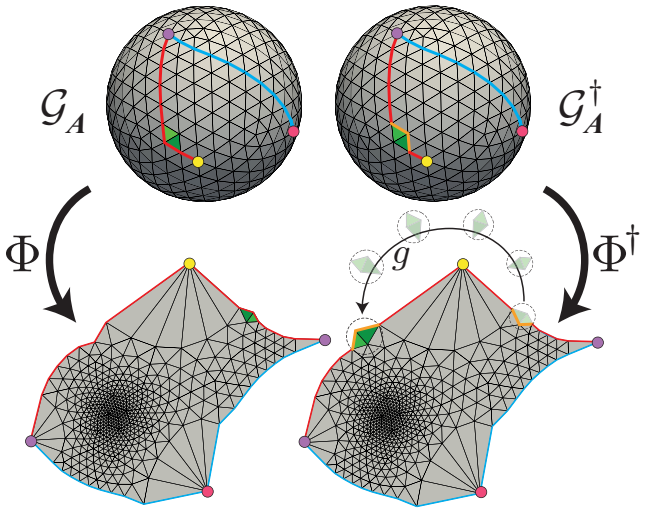


Figure 5: *For a G-flattening, one may homotopically modify a seam while modifying the flattening's boundary using the seam-transformation.*

For example, Figure 5 shows a homotopy of the red seam (top-left) which moves two faces (in green) from one side of the seam to another (top-right). A *homotopy of the cut-graph* is a homotopy of the seams such that seams do not intersect each-other and their end points remain at the landmarks.

Now, consider a mapping $f : \mathbf{A} \to \mathbf{B}$ defined by a pair of flattenings $\Phi^* : \bar{\mathbf{A}} \to \mathbb{R}^2$, $\Psi^* : \bar{\mathbf{B}} \to \mathbb{R}^2$ locally minimizing (5).

Definition 2. *We say that f is* seamless *if for an arbitrary homotopic deformation of the cut-graphs, $\mathcal{G}_{\boldsymbol{A}}^\dagger, \mathcal{G}_{\boldsymbol{B}}^\dagger$, there exist a pair of flattenings $\Phi^\dagger, \Psi^\dagger$ of these differently-cut meshes such that $\Phi^\dagger$, $\Psi^\dagger$*

1. *Satisfy the G-mapping-conditions,*
2. *Define the same surface map $f : \boldsymbol{A} \to \boldsymbol{B}$ as Φ^*, Ψ^*, and*
3. *Are also local minimizers of $\mathcal{E}$, satisfying $\mathcal{E}(\Phi^\dagger, \Psi^\dagger) = \mathcal{E}(\Phi^*, \Psi^*)$.*

Intuitively, the definition says that f constructed from (Φ^*, Ψ^*) is seamless if one can choose any other arbitrary (but homotopic) cut-graphs, and construct the same map f by minimizing the *same* energy functional.

We now show that in the case $\mathcal{E}$ is G-invariant, it follows that a local minimum (Φ^*, Ψ^*) of (5) indeed defines a seamless map in the above sense. Consider some arbitrary homotopic cut-graphs $\mathcal{G}_{\mathbf{A}}^\dagger, \mathcal{G}_{\mathbf{B}}^\dagger$, and let us show the existence of $\Phi^\dagger, \Psi^\dagger$. Consider the cut-graph $\mathcal{G}_{\mathbf{A}}$ and its homotopic deformation to $\mathcal{G}_{\mathbf{A}}^\dagger$. Each step of the homotopy moves a seam by one face t_j at a time. Each such operation induces a corresponding change to the G-flattening Φ, moving the face t_j from one copy of a seam to to its twin copy, using the relevant seam-transformation from G. Two operations of this sort (moving two faces) are shown in Figure 5, bottom row. These operations can be applied repeatedly and independently both to Φ^* and Ψ^* while maintaining that the G-mapping-conditions are satisfied. We thus achieve $\Phi^\dagger, \Psi^\dagger$ that satisfy the G-mapping-conditions and correspond to the new cut-graphs $\mathcal{G}_{\mathbf{A}}^\dagger, \mathcal{G}_{\mathbf{B}}^\dagger$. By construction $\Phi^\dagger, \Psi^\dagger$ define the same map f as Φ^*, Ψ^*. Lastly, since E is invariant to compositions with $g \in G$ we have that $E(\Phi^\dagger) = E(\Phi^*)$ and $E(\Psi^\dagger) = E(\Psi^*)$ and hence $\mathcal{E}(\Phi^*, \Psi^*) = \mathcal{E}(\Phi^\dagger, \Psi^\dagger)$. As the procedure above is reversible, Φ^*, Ψ^* is a local minimum of $\mathcal{E}$ if and only if $\Phi^\dagger, \Psi^\dagger$ is a local minimum. Thus we've shown,

Theorem 1. *If E is G-invariant, and the flattenings (Φ^*, Ψ^*) are local minimum of problem (5), then the bijective mapping $f : \boldsymbol{A} \to \boldsymbol{B}$ defined via (Φ^*, Ψ^*) is seamless.*

We summarize this section in Algorithm 1, detailing the framework for computing bijective seamless mappings between two surfaces, given a set of corresponding landmark points. In the next section we complete the picture by providing the implementation details for the algorithm.

Algorithm 1: Computation of seamless surface-maps

input : Meshes $\mathbf{A}, \mathbf{B}$
Landmarks $\mathcal{P} \subset \mathbf{V_A}$, $\mathcal{Q} \subset \mathbf{V_B}$,
Transformation group G, and G-invariant energy E.
output: Bijective mapping $f : \mathbf{A} \to \mathbf{B}$

1 Cut the two meshes $\mathbf{A}, \mathbf{B}$ into the disks $\bar{\mathbf{A}}, \bar{\mathbf{B}}$.
2 Compute $\Phi : \bar{\mathbf{A}} \to \mathbb{R}^2, \Psi : \bar{\mathbf{B}} \to \mathbb{R}^2$ by optimizing problem (5).
3 Extract the surface-map $f : \mathbf{A} \to \mathbf{B}$ from the flattenings.

7 Algorithm and implementation details

We now complete the implementation details of the algorithm for computing seamless bijective mappings between surfaces as outlined in Algorithm 1. We will first elaborate on the cutting process, move to the optimization of problem (5), and finish with the extraction of the map from the flattenings. Each of these three steps is self-contained and does not depend on the implementation of the other two.

7.1 Cutting the meshes

The goal of this step is to produce two consistent cut-graphs $\mathcal{G}_{\mathbf{A}}, \mathcal{G}_{\mathbf{B}}$ using the input of the corresponding landmark points $\mathcal{P} \subset \mathbf{V_A}, \mathcal{Q} \subset \mathbf{V_B}$ on the two surfaces $\mathbf{A}, \mathbf{B}$. For simplicity we assume in the following that the surface is of genus zero, however the algorithm can be readily applied to any genus, as exemplified in Figure 11.

The algorithm is initialized with $\bar{\mathbf{A}} = \mathbf{A}$, $\bar{\mathbf{B}} = \mathbf{B}$ and empty cut-graphs $\mathcal{G}_{\mathbf{A}} = \emptyset, \mathcal{G}_{\mathbf{B}} = \emptyset$. In each iteration, while not all landmarks participate in the cut-graphs, a pair of seams, $\alpha \subset \mathbf{E}_{\bar{\mathbf{A}}}$ connecting $p_i \notin \mathcal{G}_{\mathbf{A}}$, and $p_j \in \mathcal{G}_{\mathbf{A}}$, and $\beta \subset \mathbf{E}_{\bar{\mathbf{B}}}$ connecting $q_i \notin \mathcal{G}_{\mathbf{B}}$, $q_j \in \mathcal{G}_{\mathbf{B}}$, are added to the cut-graphs $\mathcal{G}_{\mathbf{A}}$ and $\mathcal{G}_{\mathbf{B}}$, respectively. The meshes are then cut accordingly and $\bar{\mathbf{A}}, \bar{\mathbf{B}}$ are updated.

As the computed maps are invariant to homotopic changes of the cut-graphs (as explained in Section 6), the particular choice of cuts in a fixed homotopy class does not affect the resulting map. Therefore, we use the convenient choice of Dijkstra's algorithm for producing the seams and cutting the meshes. The particular choice of seams α, β to be added at each iteration of the above algorithm is decided as follows. We iteratively take the next seams α, β to be the ones with minimal sum of distances $\mathrm{dist}_{\mathbf{A}}(p_i, p_j) + \mathrm{dist}_{\mathbf{B}}(q_i, q_j)$ among all pairs $p_i \notin \mathcal{G}_{\mathbf{A}}$, $p_j \in \mathcal{G}_{\mathbf{A}}$, $q_i \notin \mathcal{G}_{\mathbf{B}}$, $q_j \in \mathcal{G}_{\mathbf{B}}$. The distance functions $\mathrm{dist}_{\mathbf{A}}$, $\mathrm{dist}_{\mathbf{B}}$ in $\mathbf{A}$ and $\mathbf{B}$ are measured using a flat metric produced by mapping the cut meshes $\bar{\mathbf{A}}$, $\bar{\mathbf{B}}$, after the first cut is performed, to the unit disk using Mean Value Coordinates ([Floater 2003]). This is a heuristic aimed at producing cut-graphs that induce mappings in the natural homotopy class of mappings. Although not guaranteed, it produced natural homotopic class in all examples tested in this paper. We discuss the affect of choosing cut-graphs of different homotopy classes in Section 8.

The above algorithm is guaranteed to produce topologically *consistent* cut meshes $\bar{\mathbf{A}}, \bar{\mathbf{B}}$, in the sense that it is always possible to construct a homeomorphism $f_0 : \mathbf{A} \to \mathbf{B}$ by prescribing two flattenings $\Phi_0 : \bar{\mathbf{A}} \to \mathbb{R}^2$, $\Psi_0 : \bar{\mathbf{B}} \to \mathbb{R}^2$ that satisfy the G-mapping-conditions. Note that previous works ([Kraevoy and Sheffer 2004; Schreiner et al. 2004]) map the surface to a coarser version and therefore cut it into patches rather than along a tree. This requires care to assure no blocking of paths, or inconsistent orientation of paths occurs.

7.2 Optimization

We now elaborate on the optimization of problem (5). This task poses two challenges: first, the energy $\mathcal{E}$ is non-convex (since the conformal energy $D_{\text{conf}}(A)$ is non-convex), and second, the constraints of the G-mapping-conditions on Φ, Ψ are bilinear. We first explain how the optimization of the energy is performed and then how to incorporate the constraints.

Setting. At this stage of the algorithm we are provided with two disk-type surface-meshes $\bar{\mathbf{A}}$, $\bar{\mathbf{B}}$. Our goal is to compute two simplicial flattenings $\Phi : \bar{\mathbf{A}} \to \mathbb{R}^2$, $\Psi : \bar{\mathbf{B}} \to \mathbb{R}^2$ that satisfy the G-mapping-conditions and locally minimize the L_2 average conformal distortion. The free variables in the optimization problem are the planar vertex images of $\bar{\mathbf{A}}$ and $\bar{\mathbf{B}}$ under Φ and Ψ (respectively), that is $\Phi(v)$, $\forall v \in \mathbf{V}_{\bar{\mathbf{A}}}$ and $\Psi(v)$, $\forall v \in \mathbf{V}_{\bar{\mathbf{B}}}$. The differentials A_j of the simplicial flattenings are expressed, as usual, as linear combinations of the variables, as described in Appendix B.

Optimization of the energy. The energy $\mathcal{E}$ is a sum of two energies, $E(\Phi)$ and $E(\Psi)$, that measure the L_2 average conformal distortion of the simplicial flattenings Φ and Ψ, as defined in Equation (4). We concentrate on the optimization of only $E(\Phi)$ (the combination of both $E(\Phi), E(\Psi)$ follows naturally). We suggest a novel way to optimize $E(\Phi)$ using a re-weighting scheme of the isometric distortion energy,

$$E'(\Phi, \mathbf{c}) = \frac{1}{4} \sum_{t_j \in \mathbf{T}_{\bar{\mathbf{A}}}} \left[D_{\text{iso}}(c_j A_j) \right]^4 |t_j| \,, \tag{6}$$

where

$$D_{\text{iso}}(A) = \left(\Sigma(A)^2 + \frac{1}{\sigma(A)^2} \right)^{1/2}, \tag{7}$$

$A_j \in \mathbb{R}^{2\times 2}$ is the differential of Φ restricted to face t_j, and $\mathbf{c} = \left(c_1, ..., c_{|\mathbf{T}_{\bar{\mathbf{A}}}|}\right) \in \mathbb{R}^{|\mathbf{T}_{\bar{\mathbf{A}}}|}$ is a vector of weights representing the area-distortion of each face.
The motivation in casting the conformal energy as in Eq. (6) stems from the fact that the isometric distortion (7) has an efficient convex relaxation. The following proposition, proven in Appendix B, asserts that minimizing this alternative energy will yield minimizers of the original L_2 conformal energy:

Proposition 2. *For a given non-degenerate simplicial map Φ,*

$$E(\Phi) = \min_{\boldsymbol{c}>0} E'(\Phi, \boldsymbol{c}),$$

where the optimal $\mathbf{c}$ *is given by*

$$c_j^* = \left| \det\left(A_j\right) \right|^{-1/2}. \tag{8}$$

Note that Eq. (8) prescribes c_j to be one over the area-distortion of face j; in case A is a similarity then c_j, as defined by Eq. (8) is the reciprocal of the conformal factor of the face t_j.

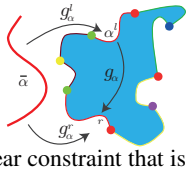
This leads to the following scheme for optimizing $E(\Phi)$: Start from an initial feasible Φ_0, set $\mathbf{c} = \mathbf{1}$ (a vector of all ones), and alternate between optimizing $E'(\Phi; \mathbf{c})$ as a function of Φ while $\mathbf{c}$ is held fixed, and updating $\mathbf{c}$ according to Eq. (8) while Φ is fixed. For the optimization of $E'(\Phi; \mathbf{c})$ as a function of Φ we employ the convexification method of [Aigerman et al. 2014]. For completeness, in Appendix B we provide all the necessary details of the optimization of $E'(\Phi; \mathbf{c})$. This algorithm is guaranteed to reduce the energy $E(\Phi)$ monotonically, and maintain local injectivity of Φ, except possibly at the landmarks.

G-flattening constraints. For incorporating the G-mapping-conditions on Φ, Ψ, we first note that local injectivity of Φ, Ψ is already assured by the optimization described above. Furthermore, the constraints $\Phi(a_k) = \Psi(b_k)$ for all corresponding landmark copies $a_k \leftrightarrow b_k$ are linear in the optimization variables (remember that the landmarks are vertices in the original meshes). Lastly, the G-mapping-conditions require that for each seam α, there exists a similarity-transformation $g_\alpha \in G$ which satisfies

$$\Phi(\alpha^r) = T_\alpha \cdot \Phi(\alpha^l) + \eta_\alpha,$$

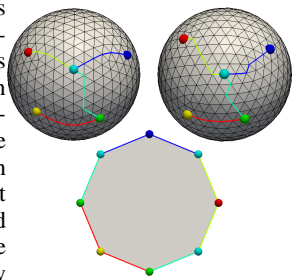
where $T_\alpha \in \mathbb{R}^{2\times 2}$, $\eta_\alpha \in \mathbb{R}^{2\times 1}$ are respectively the linear-transformation part and the translational part which make up g_α as defined in Eq. (2). These constraints are bilinear in the variables, and we optimize them by interleaving. We use an equivalent symmetric formulation to this constraint and introduce an auxiliary copy $\bar{\alpha}$ of the seam α and require

$$T_\alpha^l \bar{\alpha} + \eta_\alpha^l = \Phi(\alpha^l), \quad T_\alpha^r \bar{\alpha} + \eta_\alpha^r = \Phi(\alpha^r), \tag{9}$$



where $T_\alpha^l, \eta_\alpha^l$ and $T_\alpha^r, \eta_\alpha^r$ are unknowns defining two similarity transformations, g_α^l, g_α^r which relate the seam copy $\bar{\alpha}$ with its realizations α^l, α^r (see inset). We alternate between fixing T_α^l, T_α^r for all α and fixing $\bar{\alpha}$ in the optimization. In both cases Eq. (9) becomes a linear constraint that is added to the optimization. We eliminate the translation degree of freedom of $\bar{\alpha}$ by fixing one end point of $\bar{\alpha}$. The global rigid-motion invariance of the energy $\mathbf{E}'$ is eliminated by adding a constraint fixing one landmark to its current place and a second landmark to lie on the infinite ray spanned between it and the first landmark. Lastly, for numerical stability we limit the area scale of all faces from above and below $100^{-1} \leq c_j \leq 100$.

Initialization. The optimization algorithm requires a pair of feasible initial mappings Φ_0, Ψ_0. These are constructed as follows. We use Mean Value Coordinates [Floater 2003] to produce G-flattenings by prescribing the images of the boundaries of $\bar{\mathbf{A}}, \bar{\mathbf{B}}$ to lie on the same convex regular planar polygon where copies of landmarks are mapped to vertices of the polygon and the seams are mapped to straight lines, with vertices equally-spaced across each seam, as depicted in the inset. This ensures that a similarity $g \in G$ (in-fact, a rigid motion) exists between every pair of matching seams. Convex combination maps to a convex domain are guaranteed to be globally injective, and by construction corresponding copies of landmarks are mapped to the same vertex of the regular planar polygon. It follows that Φ_0, Ψ_0 satisfy the G-mapping-conditions. In this case the mapping $f_0 : \mathbf{A} \to \mathbf{B}$ is defined via $f_0 = \Psi_0^{-1} \circ \Phi_0$. Note this map is not seamless as Φ_0, Ψ_0 are not local minimizer of $\mathcal{E}$, however it serves well as an initial feasible guess.



We summarize the optimization of problem (5) in Algorithm 2. The algorithm monotonically decreases the energy $\mathcal{E}$ until converging to

the two flattenings Φ^*, Ψ^* which are a local minimum of Problem (5), hence defining a seamless surface map $f : \mathbf{A} \to \mathbf{B}$, as defined in Section 6.

Algorithm 2: Optimization of flattenings

input : Cut meshes $\bar{\mathbf{A}}, \bar{\mathbf{B}}$
output: $\Phi^* : \bar{\mathbf{A}} \to \mathbb{R}^2$, $\Psi^* : \bar{\mathbf{B}} \to \mathbb{R}^2$ with G-mapping-conditions

Compute Φ_0, Ψ_0 via convex combination maps
while Φ_n, Ψ_n *have not converged* **do**
  Update $\mathbf{c}_\Phi, \mathbf{c}_\Psi$ according to Φ_{n-1}, Ψ_{n-1} using Eq. (8)
  if *current iteration is odd-numbered* **then**
    Fix all seam-copies $\bar{\alpha}$, $\bar{\beta}$
  else
    Fix all transformations $T^l_\alpha, T^r_\alpha, T^l_\beta, T^r_\beta$
  Compute Φ_n, Ψ_n by optimizing the following problem (as detailed in Appendix B)

$$\begin{aligned} \min_{\Phi,\Psi} \quad & E'(\Phi, \mathbf{c}_\Phi) + E'(\Psi, \mathbf{c}_\Psi) \\ \text{s.t.} \quad & \Phi(a_k) = \Psi(b_k), && \forall k \\ & T^s_\alpha \bar{\alpha} + \eta^s_\alpha = \Phi(\alpha^s), && \forall \alpha \in \mathcal{G}_\mathbf{A}, s \in \{l, r\} \\ & T^s_\beta \bar{\beta} + \eta^s_\beta = \Phi(\beta^s), && \forall \beta \in \mathcal{G}_\mathbf{B}, s \in \{l, r\} \end{aligned}$$

7.3 Computing the map

The last step in the algorithm is to recover the surface bijection $f : \mathbf{A} \to \mathbf{B}$ from the G-flattenings $\Phi = \Phi^*$, $\Psi = \Psi^*$. Towards this end we adapt the lifting algorithm from [Aigerman et al. 2014] to our settings, as detailed next. To map a point z on $\bar{\mathbf{A}}$, we choose a landmark-copy $a_k \in \mathbf{V}_{\bar{\mathbf{A}}}$ and its corresponding landmark-copy $b_k \in \mathbf{V}_{\bar{\mathbf{B}}}$. We connect a_k to z by constructing a simple polygonal curve Γ (*e.g.*, using Dijkstra's algorithm). We then map Γ to the plane via Φ, to get the planar polygonal curve $\gamma = \Phi(\Gamma)$. Next, γ is lifted to $\mathbf{B}$ by tracing it, as follows. We traverse γ and at every step restrict our attention to a 1-ring (denoted $\mathcal{R}$) in the target domain $\bar{\mathbf{B}}$. Restricted to this 1-ring, the map $\Psi|_\mathcal{R}$ is injective and thus can be inverted to map the next piece of the curve γ to $\bar{\mathbf{B}}$. The 1-ring $\mathcal{R}$ is then updated according to the end point of the lifted piece of the curve and the algorithm continues. The only change we need to apply to this lifting algorithm is to handle the case in which γ crosses a seam β of the flattening $\Psi(\bar{\mathbf{B}})$. In which case we apply the relevant seam-transformation, g_β, to make the transition to the 'other side' of the seam. This process of lifting γ is described in Algorithm 3 using the path γ and b_k (the corresponding landmark to a_k) as the initial input. In practice, for efficiency, we compute the image under f of all the vertices in $\mathbf{A}$ (or any other point on $\mathbf{A}$ we wish to map) by constructing a tree rooted at some landmark a_k and spanning all vertices $\mathbf{V}_\mathbf{A}$. The tree can be lifted to $\mathbf{B}$ similarly to the algorithm described above for curves.

7.4 Avoiding pathological cases

There exist pathological examples of flattenings Φ, Ψ which satisfy the G-mapping-conditions but do not define a homeomorphism $f : \mathbf{A} \to \mathbf{B}$. One such example can be created using a construction attributed to Milnor, shown in the inset (we illustrate the cut meshes as 2-dimensional discs). Φ, Ψ satisfy the G-mapping-conditions, but there is no possible homeomorphism $f : \mathbf{A} \to \mathbf{B}$ that can be defined in this case: first, the images

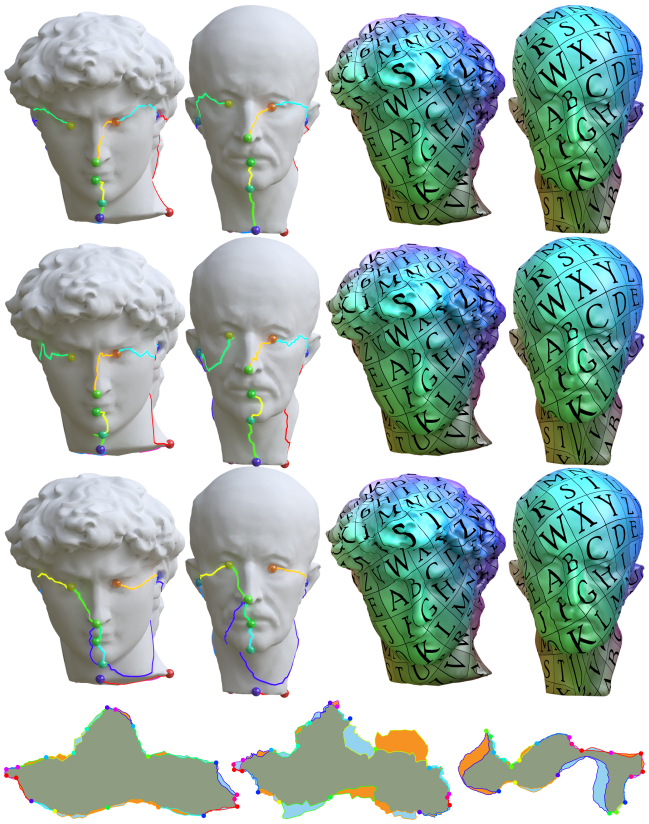
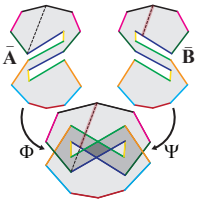


Figure 6: *Three mappings computed for different cut-graphs, exhibiting our algorithm's invariance to cut placement. All three maps are identical, although the cuts vary at each row: the middle row's cuts are a perturbed variation of the cuts of the top row, and the bottom row has a completely different cut-graph. At the bottom we show the three G-flattenings that produced these maps.*



of Φ, Ψ coincide, and hence no cut-and-paste operations are required. However, considering the image of the dashed curve under Φ, it has no preimage under Ψ which yields a single connected curve, hence a homeomorphism cannot exist. Our optimization algorithm avoids these cases, as it solves a feasible convex problem at each iteration, and starts from a feasible configuration Φ_0, Ψ_0 for which a homeomorphism f_0 exists. This entails that the flattenings are continuously deformed while preserving their local-injectivity during the optimization, ensuring such pathological examples (which require a non-homotopic or non-locally-injective deformation of the flattenings) cannot happen in practice.

8 Results

We now present the evaluation of the algorithm described in the previous sections. We have tested its different properties, compared it to relevant previous work and experimented with a wide range of surfaces.

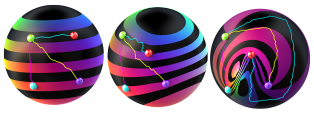
Visualization. We visualize the maps by first transferring an RGB color function from the left surface to the right one using the computed map. This provides a good visualization of the global structure of the map. Additionally, we compute texture coordinates

Figure 7: *Mappings produced by our algorithm on pairs of models from the SHREC07 [Giorgi et al. 2007] dataset.*

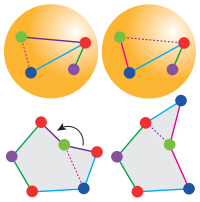
for the visible part of the left mesh (to get a low distortion texture) and then transfer a letter-grid texture from the left model to the right, again using the computed map. This provides more detailed, local information about the map. Note that the second approach might leave un-textured areas on the target mesh in some cases, although the map is always surjective, as exhibited by the coloring.

Effect of the cut-graph on the resulting mapping. We illustrate the invariance of our algorithm to cut placement in Figures 1 and 6. In Figure 6 we compute the map for one seam placement (top row), a perturbation of it (middle row), and a completely different cut-logic (bottom). In all three cases the produced mappings are identical.

Homotopy classes. As discussed in Section 6 and shown in Figures 1, 6, the algorithm is invariant to the particular choice of the cuts as long they are chosen in the same homotopy class. The question is what happens when the cut-graphs are changed non-homotopically? In this case the algorithm may produce a homeomorphism from a different homotopy-class of homeomorphisms (note that we consider here the surfaces as punctured at the landmarks $\mathcal{P}$, $\mathcal{Q}$). For example, the inset shows two non-homotopic homeomorphisms produced by choosing non-homotopic cuts. On the left, the source mesh is shown, with the prescribed four landmarks and cut-graph. In the middle, the first map is shown, computed for a choice of a cut-graph in the "natural" homotopy class, which yields the expected homeomorphism. On the right, the second map is shown, where this time the cut-graph was chosen from a different homotopy class. Note that the cut-graph in the source surface is the same for both maps. As discussed in Section 7.1, our algorithm uses a heuristic which produced cuts in the correct homotopy class for all examples we tested.



The class of cut-graphs for which the algorithm is invariant to is actually bigger than homotopic cut-graphs; it is possible to perform other modifications to the cut-graphs and still maintain the same homotopy class for the homeomorphism. Figure 6 shows such an example: the cut-graph in the bottom row has a different connectivity compared to the rows above it (*e.g.*, the nose), yet it produces the same map (right-column). One possible atomic operation that can be applied to both cut-graphs of the surfaces without changing the homeomorphism's homotopy class is depicted in the inset, for one of the surfaces: create a new seam connecting the blue and green landmarks (bottom-row shows the G-flattenings) and stitch back the seam between the green and red landmarks. On the surface (top-row) this is equivalent to replacing a seam in the cut-graph with another. Doing this operation to both cut-graphs of the two surfaces keeps the homeomorphism they define intact and in particular in the same homotopy class. Furthermore, this statement also shows that there is nothing to be gained by adding more cuts to the mesh than is needed to cut it to a topological disk.



Computation of maps in SHREC07 We computed maps between different models in the SHREC07 dataset [Giorgi et al. 2007], as depicted in Figure 7. Note the low-distortion seamless mapping of the humans from a sparse set of landmarks at the extremities and on the head of the models. The algorithm also produces high quality maps for articulated non-isometric pairs such as the 4-legged animals. Note the cuts pass on arbitrary and different positions on the pig's and bull's bodies, however the produced mapping still maps that area correctly, from a sparse set of landmarks which are rather far from the main part of the body. The algorithm also nicely handles the twisted and elongated parts of the octopus, although the cuts are rather arbitrary in this case. On the pair of hands, each model is of a different length, and the landmarks give no cue to the algorithm as to where the palm of the hand starts,

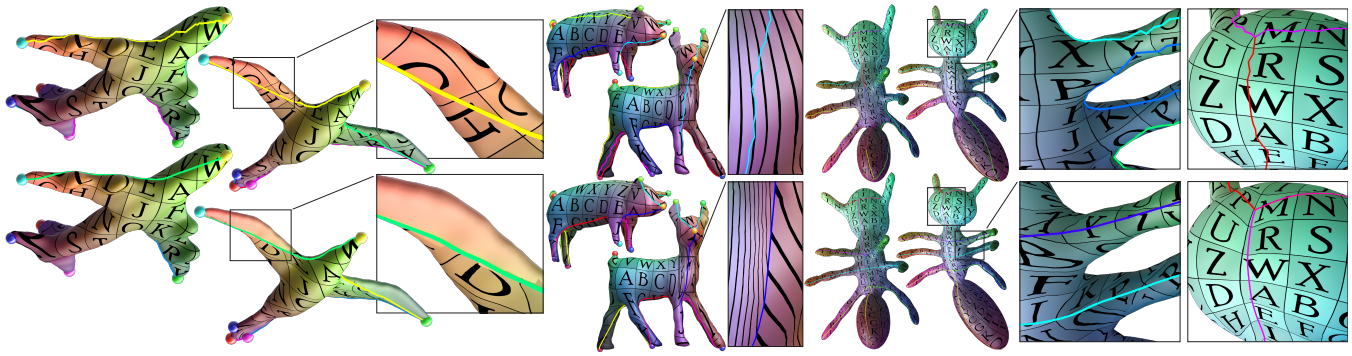
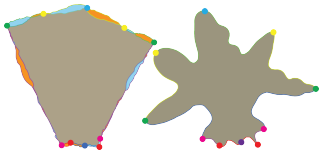


Figure 8: *Comparison of the method presented in this paper (top-row) to the method of [Aigerman et al. 2014] (bottom-row). The left example exhibits our method's invariance to the cut placement, while the fixed correspondences of the cuts in [Aigerman et al. 2014] cause unnatural mapping in the wing area. The middle example exhibits the seamlessness of our method as demonstrated using the flow-lines. On the right is one of the results used in [Aigerman et al. 2014], for which our method also compares favorably due to its seamless property.*

however the map still correctly maps the palms with high accuracy. Also note how the map gracefully handles the fact the middle-finger and ring-finger are not separated on the left model. Lastly, note the smooth mapping of the sharp features of the mechanical parts. We note that in case one wishes to exactly map feature lines to feature lines, it is possible to add seam constraints to force that in the mapping, however this is out-of-scope for this paper.

Comparison to previous work We have compared our algorithm to the method of Aigerman *et al.*[2014] which seems to be the closest to this work. Results are shown in Figure 8. On the left example, due to the sparse set of correspondences, the cuts' positioning is rather arbitrary as the green and yellow cuts pass through the bottom of the bird's wings, but on top of the plane's wings. This does not affect our algorithm, which maps the wings correctly. The map produced by [Aigerman et al. 2014], on the other hand, requires that the flattenings have the same image (as shown in the inset, our G-flattenings on the left, Aigerman's on the right). Hence, it is forced to map cuts to cuts, which results in mapping the top of the plane's wing to the bottom of the bird's wing (the lack of texture stems from the mapping of the untextured part of the top of the plane's wing). In the middle example, the two non-isometric animals require a map which possesses relatively high distortion levels. In this case the map of [Aigerman et al. 2014] presents a visible "jump" across the seam as shown in the blowup using the flow-lines. In contrast, our map does not exhibit any special behavior in the vicinity of the cut. On the right example, we used data from [Aigerman et al. 2014], with the same landmarks they used. Our map ignores the placement of the seams and produces a more natural correspondence, without introducing jumps across the cuts as is visible in the result from [Aigerman et al. 2014].



Cycle-consistent maps between collections Our method can be readily applied to a collection of more than two meshes, by simply extending the G-mapping-conditions, defined in Section 5, to a collection of k meshes, as illustrated in Figures 9a and 9b for $k = 3$. Given k meshes $\mathbf{M}_1, \ldots, \mathbf{M}_k$ and corresponding landmarks prescribed on each mesh, we first cut the meshes consistently as described in Subsection 7.1 and then map each mesh $\mathbf{M}_j$ with a G-flattening Φ_j so that corresponding landmark-copies are mapped to the same position for all meshes. We then optimize the energy $\mathcal{E}(\Phi_1, \ldots, \Phi_k) = \sum_{i=1}^{k} E(\Phi_i)$. Considering any pair of meshes $\mathbf{M}_i, \mathbf{M}_j$ in the collection, and their flattenings Φ_i, Φ_j, this process defines a bijection between the meshes, as all pair of flattenings satisfy the G-mapping-conditions. Furthermore, the joint flattening of all meshes actually entails a stronger claim, that the maps are cycle-consistent in the sense of [Nguyen et al. 2011; Huang and Guibas 2013]: if $f_{i\to j}$ is the map between $\mathbf{M}_i$ and $\mathbf{M}_j$ then $f_{\ell\to i} \circ f_{j\to \ell} \circ f_{i\to j} \equiv I_{i\to i}$, where $I_{i\to i}$ is the identity mapping on

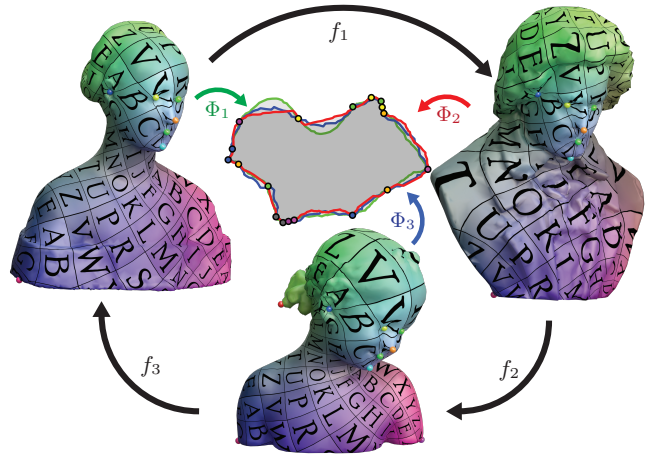
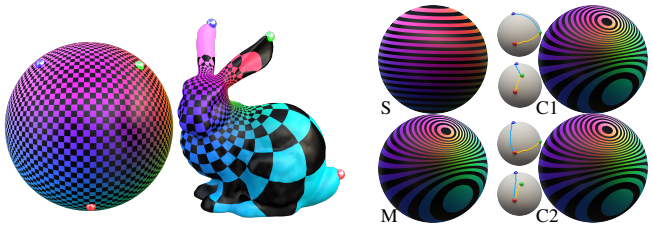


Figure 9: *Our method can be applied to any number of meshes (in this example, three) by extending the G-mapping-conditions to a set of k flattenings, as shown in (a). In the middle, the G-flattenings Φ_1, Φ_2, Φ_3 of the three meshes $\mathbf{M}_1, \mathbf{M}_2, \mathbf{M}_3$ are shown. This process defines the seamless bijective mappings f_1, f_2, f_3 between any pair of the collection. The maps are guaranteed to be cycle-consistent. (Differently from the rest of the paper, here for each flattening we color all its seams with one color). In (b) we show another example of a cycle-consistent mapping of three surfaces.*

M_i. For this application, the ability to ignore the existence of the seams is crucial, as other mapping methods would need to *simultaneously* plan the seam placement for k meshes, or optimize the seams' position on all meshes.

Approximation of conformal mappings Our method strives to minimize the average conformal distortion of the resulting map. To test its effectiveness, we have run experiments in which we prescribe three landmarks on genus-zero surfaces, for which case the theory guarantees the existence of a conformal map mapping the landmarks correctly. As depicted in Figure 10(a), our algorithm produces a discrete approximation of a conformal map between the Stanford Bunny and the sphere. In Figure 10(b) we were able to reproduce the unique Möbius transformation interpolating three landmark between two spheres. The Möbius map is shown at the bottom-left of (b), while results of our algorithm from two different cuts are shown on the right column of (b).



(a) (b)

Figure 10: *Given 3 landmarks, our method approximates a conformal map, such as the mapping of the sphere to the Stanford Bunny shown in (a). In (b), the algorithm reproduces the unique conformal Möbius transformation which maps the landmarks (the analytic mapping shown at bottom-left). Textured source mesh is shown on the top-left. To the right, we show our result for two different cut choices. The choice of cuts is shown to the left of each map, source mesh at the top and target at the bottom.*

Higher genus. Topologically equivalent meshes of higher genus can also be mapped using our method, as illustrated for a simple torus in Figure 11 in which we map the torus to itself using a single landmark (in yellow). The only change to the algorithm is in the cutting step, to ensure the cut-graphs cut the meshes into disks.

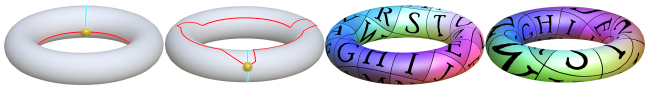


Figure 11: *Our method is applicable to meshes of any genus. In this case a mapping of the torus was generated from a single landmark (in yellow).*

Implementation and timings. We implemented the algorithm in Matlab using MOSEK's Second-Order-Cone solver [Andersen and Andersen 1999] and the YALMIP environment [Löfberg 2004]. The experiments were run on a 3.50GHz Intel i7. For a pair of meshes with a total of 11k vertices and 30k faces, the optimization ran for 23 minutes, and setup took 26 seconds. For a collection of three meshes, with a total of 14k vertices and 28k faces, the optimization ran for half an hour and setup took half a minute. We note that our code is not optimized and we believe significant speedups are possible.

9 Limitations and failure cases

Our algorithm has two main limitations, discussed next.

Distortion near landmarks. Although the produced maps are seamless, in some cases they still present stretching near the landmarks. In Figure 12 we show 4 of the maps from Figure 7 with a dense checkerboard pattern to better illustrate the behaviour near landmarks. Unlike the seams, the landmarks are part of the original problem. Producing a map without bias to the landmarks requires taking a different approach and we mark this goal as an interesting future research direction.

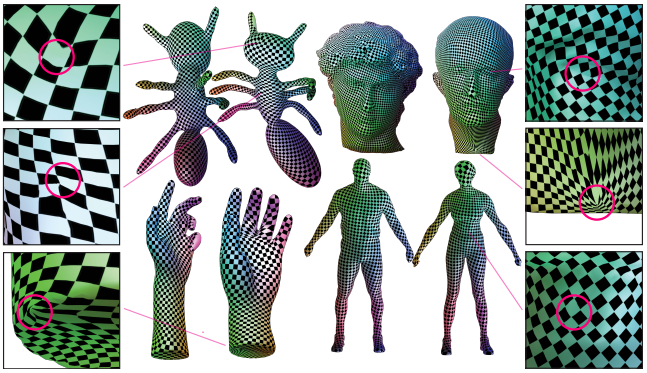


Figure 12: *Maps from Figure 7 visualized with a dense checkerboard, highlighting the distortion next to the landmarks. Although pinching is noticeable in some cases (the busts' neck; the bottom of the hand) it is mild in others (ants) or non-existent (humans; face of busts).*

Numerical issues. The optimization problem we solve includes about five second-order cone constraints per face of the meshes. This results in rather large-scale SOCP that causes slow running times and limits the size of meshes for which our algorithm is applicable. As the size of the problem (*i.e.*, meshes) grows we noticed convergence issues with our solver of choice (MOSEK). We have conducted an experiment in which we have computed a mapping between two spheres with the same input landmarks and cut-graphs for different mesh resolutions and observed that above 80K faces the solver failed to properly converge. We believe these aforementioned numerical issues can be solved by a tailor-made solver for this type of Geometry-Processing optimization problems, and we consider this to be an important future research venue.

10 Conclusion

We have presented a method to produce high-quality seamless bijective mappings between surface-meshes interpolating a given set of correspondences. The invariance of the algorithm to cut placement allows the user to place the desired landmarks without concerning about the placement of the cuts, enabling further automation of the map-computation than was previously possible for similar methods. We have demonstrated the method's ability to produce state-of-the-art mappings between pairs of surfaces, as well as being able to produce seamless cycle-consistent mappings of triplets of surfaces.

We believe our algorithm can be useful in many applications, such as collection-mapping and shape matching, where the ability to produce high quality bijections fully-automatically is crucial. A possible interesting future venue of research is combining our method in a collection-analysis framework.

11 Acknowledgements

This work was funded by the European Research Council (ERC Starting Grant "SurfComp"), the Israel Science Foundation (grant No. 1284/12) and the I-CORE program of the Israel PBC and ISF (Grant No. 4/11). The authors would like to thank the anonymous reviewers for their comments and suggestions.

References

AIGERMAN, N., PORANNE, R., AND LIPMAN, Y. 2014. Lifted bijections for low distortion surface mappings. *ACM Trans. Graph. 33*, 4 (July), 69:1–69:12.

ANDERSEN, E. D., AND ANDERSEN, K. D. 1999. *The MOSEK interior point optimization for linear programming: an implementation of the homogeneous algorithm*. Kluwer Academic Publishers, 197–232.

BOGO, F., ROMERO, J., LOPER, M., AND BLACK, M. J. 2014. FAUST: Dataset and evaluation for 3D mesh registration. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Piscataway, NJ, USA.

BOMMES, D., ZIMMER, H., AND KOBBELT, L. 2009. Mixed-integer quadrangulation. *ACM Trans. Graph. 28*, 3 (July), 77:1–77:10.

BOMMES, D., LÉVY, B., PIETRONI, N., PUPPO, E., SILVA, C., TARINI, M., AND ZORIN, D. 2013. Quad-Mesh Generation and Processing: A Survey. *Computer Graphics Forum 32*, 6, 51–76.

BRADLEY, D., POPA, T., SHEFFER, A., HEIDRICH, W., AND BOUBEKEUR, T. 2008. Markerless garment capture. *ACM Trans. Graph. 27*, 3 (Aug.), 99:1–99:9.

BRONSTEIN, A. M., BRONSTEIN, M. M., AND KIMMEL, R. 2006. Generalized multidimensional scaling: a framework for isometry-invariant partial surface matching. *Proc. National Academy of Sciences (PNAS)*.

FLOATER, M. S. 2003. Mean value coordinates. *Computer Aided Geometric Design 20*, 1, 19–27.

GIORGI, D., BIASOTTI, S., AND PARABOSCHI, L. 2007. SHREC: SHape REtrieval Contest: Watertight models track. *http://watertight.ge.imati.cnr.it/*.

HORMANN, K., LÉVY, B., AND SHEFFER, A. 2007. Mesh parameterization: Theory and practice video files associated with this course are available from the citation page. In *ACM SIGGRAPH 2007 Courses*, ACM, New York, NY, USA, SIGGRAPH '07.

HUANG, Q.-X., AND GUIBAS, L. 2013. Consistent shape maps via semidefinite programming. In *Proceedings of the Eleventh Eurographics/ACMSIGGRAPH Symposium on Geometry Processing*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, SGP '13, 177–186.

JAIN, V., ZHANG, H., AND VAN KAICK, O. 2007. Non-rigid spectral correspondence of triangle meshes. *International Journal on Shape Modeling 13*, 1, 101–124.

KÄLBERER, F., NIESER, M., AND POLTHIER, K. 2007. Quadcover - surface parameterization using branched coverings. 375–384.

KNÖPPEL, F., CRANE, K., PINKALL, U., AND SCHRÖDER, P. 2013. Globally optimal direction fields. *ACM Trans. Graph. 32*, 4.

KRAEVOY, V., AND SHEFFER, A. 2004. Cross-parameterization and compatible remeshing of 3d models. *ACM Trans. Graph. 23*, 3 (Aug.), 861–869.

LEE, A. W. F., DOBKIN, D., SWELDENS, W., AND SCHRÖDER, P. 1999. Multiresolution mesh morphing. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, SIGGRAPH '99, 343–350.

LÉVY, B., PETITJEAN, S., RAY, N., AND MAILLOT, J. 2002. Least squares conformal maps for automatic texture atlas generation. *ACM Trans. Graph. 21*, 3 (July), 362–371.

LIN, J. L., CHUANG, J. H., LIN, C. C., AND CHEN, C. C. 2003. Consistent parametrization by quinary subdivision for remeshing and mesh metamorphosis. In *Proceedings of the 1st International Conference on Computer Graphics and Interactive Techniques in Australasia and South East Asia*, ACM, New York, NY, USA, GRAPHITE '03, 151–158.

LIPMAN, Y. 2012. Bounded distortion mapping spaces for triangular meshes. *ACM Trans. Graph. 31*, 4 (July), 108:1–108:13.

LÖFBERG, J. 2004. Yalmip : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*.

MÉMOLI, F., AND SAPIRO, G. 2005. A theoretical and computational framework for isometry invariant recognition of point cloud data. *Foundations of Computational Mathematics 5*, 3, 313–347.

MICHIKAWA, T., KANAI, T., FUJITA, M., AND CHIYOKURA, H. 2001. Multiresolution interpolation meshes. In *Computer Graphics and Applications, 2001. Proceedings. Ninth Pacific Conference on*, 60–69.

MYLES, A., AND ZORIN, D. 2012. Global parametrization by incremental flattening. *ACM Trans. Graph. 31*, 4 (July), 109:1–109:11.

MYLES, A., AND ZORIN, D. 2013. Controlled-distortion constrained global parametrization. *ACM Trans. Graph. 32*, 4 (July), 105:1–105:14.

NGUYEN, A., BEN-CHEN, M., WELNICKA, K., YE, Y., AND GUIBAS, L. 2011. An optimization approach to improving collections of shape maps. *Computer Graphics Forum 30*, 5, 1481–1491.

OVSJANIKOV, M., MÉRIGOT, Q., MÉMOLI, F., AND GUIBAS, L. 2010. One point isometric matching with the heat kernel. In *Computer Graphics Forum (Proc. of SGP)*.

OVSJANIKOV, M., BEN-CHEN, M., SOLOMON, J., BUTSCHER, A., AND GUIBAS, L. 2012. Functional maps: A flexible representation of maps between shapes. *ACM Trans. Graph. 31*, 4 (July), 30:1–30:11.

PANOZZO, D., BARAN, I., DIAMANTI, O., AND SORKINE-HORNUNG, O. 2013. Weighted averages on surfaces. *ACM Trans. Graph. 32*, 4 (July), 60:1–60:12.

PRAUN, E., SWELDENS, W., AND SCHRÖDER, P. 2001. Consistent mesh parameterizations. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ACM, 179–184.

RAY, N., LI, W. C., LÉVY, B., SHEFFER, A., AND ALLIEZ, P. 2006. Periodic global parameterization. *ACM Trans. Graph. 25*, 4 (Oct.), 1460–1485.

RAY, N., VALLET, B., LI, W. C., AND LÉVY, B. 2008. N-symmetry direction field design. *ACM Trans. Graph. 27*, 2 (May), 10:1–10:13.

SCHREINER, J., ASIRVATHAM, A., PRAUN, E., AND HOPPE, H. 2004. Inter-surface mapping. *ACM Trans. Graph. 23*, 3 (Aug.), 870–877.

SHEFFER, A., PRAUN, E., AND ROSE, K. 2006. Mesh parameterization methods and their applications. *Found. Trends. Comput. Graph. Vis. 2*, 2 (Jan.), 105–171.

STEINER, D., AND FISCHER, A. 2005. Planar parameterization for closed manifold genus-g meshes using any type of positive weights. *Journal of Computing and Information Science in Engineering 5*, 2, 118–125.

TONG, Y., ALLIEZ, P., COHEN-STEINER, D., AND DESBRUN, M. 2006. Designing quadrangulations with discrete harmonic forms. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, SGP '06, 201–210.

TSUI, A., FENTON, D., VUONG, P., HASS, J., KOEHL, P., AMENTA, N., COEURJOLLY, D., DECARLI, C., AND CARMICHAEL, O. 2013. Globally optimal cortical surface matching with exact landmark correspondence. In *Proceedings of the 23rd International Conference on Information Processing in Medical Imaging*, Springer-Verlag, Berlin, Heidelberg, IPMI'13, 487–498.

VAN KAICK, O., ZHANG, H., HAMARNEH, G., AND COHEN-OR, D. 2011. A survey on shape correspondence. *Computer Graphics Forum 30*, 6, 1681–1707.

Appendix A

We prove Proposition 2. For a given Φ, each summand of $E'(\Phi, \mathbf{c})$ depends on a different c_j and therefore the problem is separable and we can minimize each summand independently. Let $\Sigma \geq \sigma$ be the singular values of $A \in \mathbb{R}^{2\times 2}$. Let $h(c) = D_{\text{iso}}(cA)^4 = \left[(c\Sigma)^2 + (c\sigma)^{-2}\right]^2$. If $\sigma > 0$ (we assume the simplicial map is non-degenerate) the only critical point of h in the domain $c > 0$ is

$$c^* = (\Sigma\sigma)^{-1/2}.$$

Since h is differentiable and h goes to $+\infty$ as c approaches zero or infinity this critical point is the global minimum. Plugging the minimizer back into h gives

$$h(c^*) = 4\frac{\Sigma^2}{\sigma^2} = 4D_{\text{conf}}(A)^2.$$

Using this for each of the summands by plugging $c = c_j$, $A = A_j$ provides the desired result. Lastly, we note that for a 2×2 matrix A, $|\det(A)| = \Sigma\sigma$, and hence $c_j^* = |\det(A_j)|^{-1/2}$. □

Appendix B Optimization of E'

In this appendix we explain how to approach the optimization of $E'(\Phi, \mathbf{c}_\Phi) + E'(\Psi, \mathbf{c}_\Psi)$ in Algorithm 2. It is enough to consider one of the two terms, *e.g.*, $E'(\Phi, \mathbf{c}_\Phi)$. Let us first set some notation and definitions. We represent Φ as $\{\mathbf{u}_1, \ldots, \mathbf{u}_n\}$, where n is the number of vertices in the cut mesh $\bar{\mathbf{A}}$, and $\mathbf{u}_i \in \mathbb{R}^{2\times 1}$ represents the optimization variables $\Phi(v_i)$. Prescribing all $\mathbf{u}_i$ uniquely defines the simplicial map Φ. For each face $t_j \in \mathbf{T}_{\bar{\mathbf{A}}}$ we choose an arbitrary orthonormal frame and denote the affine map of Φ restricted to t_j in this frame by $\Phi|_{t_j}(\mathbf{x}) = A_j\mathbf{x} + \boldsymbol{\delta}$. We denote by $\mathbf{v}_{j_1}, \mathbf{v}_{j_2}, \mathbf{v}_{j_3} \in \mathbb{R}^{2\times 1}$ the vertices of the face t_j in the local frame. Given the prescribed vertices' positions, A_j can be obtained by solving the linear equation

$$A_j \left[\mathbf{v}_{j_1}, \mathbf{v}_{j_2}, \mathbf{v}_{j_3}\right] D = \left[\mathbf{u}_{j_1}, \mathbf{u}_{j_2}, \mathbf{u}_{j_3}\right] D,$$

where $D = \left(I - \frac{1}{3}\mathbf{1}\mathbf{1}^T\right)$ translates the three vertices so their centroid is at the origin. Let us focus on one such matrix A (we omit the subscript). To represent the two components of D_{iso}, namely Σ and σ^{-1}, in a convex framework, we follow the relaxation suggested in [Aigerman et al. 2014; Lipman 2012]: we decompose $A = B + C$, where $B = \frac{1}{2}\left(A - A^T + \text{tr}(A)I\right)$ is a similarity matrix, and $C = \frac{1}{2}\left(A + A^T - \text{tr}(A)I\right)$ is an anti-similarity matrix. The singular values of A can be written as

$$\Sigma = 2^{-1/2}\left(\|B\|_F + \|C\|_F\right),\ \sigma = 2^{-1/2}\left|\,\|B\|_F - \|C\|_F\,\right|.$$

Thus to bound the larger singular value from above by r, we use the following convex constraint which can be expressed using second-order cones (SOC):

$$\|B\|_F + \|C\|_F \leq \sqrt{2}\,r. \tag{10a}$$

To bound the smaller singular value from below by u, we convexify the non-convex constraint $u \leq 2^{-1/2}\left|\,\|B\|_F - \|C\|_F\,\right|$ by replacing $\|B\|$ with a linear underestimate of it, to get the following SOC which guarantees a bound on the smaller singular value,

$$\|C\|_F \leq \text{tr}(R^T B) - \sqrt{2}u, \tag{10b}$$

where $R \in SO(2)$ is a rotation matrix that determines which linear underestimate of $\|B\|$ will be chosen.
The convex constraint $1/u \leq s$ is formulated by the SOC:

$$\sqrt{(u-s)^2 + 4} \leq u + s. \tag{10c}$$

From these three equations we get for a scalar constant $c > 0$,

$$(c\Sigma)^2 + (c\sigma)^{-2} \leq (cr)^2 + (c^{-1}s)^2.$$

The constraint $(cr)^2 + (c^{-1}s)^2 \leq S^2$ is achieved via the SOC

$$\sqrt{(c^{-1}s)^2 + (cr)^2} \leq S \tag{10d}$$

And lastly the constraint $S^2 \leq U$ is equivalent to the SOC

$$\sqrt{S^2 + (U - 1/4)^2} \leq U + 1/4. \tag{10e}$$

Imposing the constraints (10a)-(10e) for every differential A_j of every face t_j, we get

$$\left(D_{\text{iso}}(c_j A_j)\right)^2 = \left((c_j\Sigma_j)^2 + (c_j\sigma_j)^{-2}\right) \leq U_j,$$

where we denote $\Sigma_j = \Sigma(A_j)$, and $\sigma_j = \sigma(A_j)$. Hence, minimizing the convex quadratic objective

$$\sum_{t_j \in \mathbf{T}_{\bar{\mathbf{A}}}} U_j^2 \tag{11}$$

minimizes the energy $E'(\Phi, \mathbf{c})$ as a function of Φ. Lastly, given Φ_{n-1} from a previous iteration, the optimal choice for the rotation matrix R_j (Eq. (10b)) for each differential A_j is to choose R_j to be the rotation part of the polar decomposition of A_j, that is $A_j = R_j P_j$.

To summarize, in each iteration in Algorithm 2 we replace the functional $E'(\Phi, \mathbf{c}_\Phi) + E'(\Psi, \mathbf{c}_\Psi)$ with

$$\sum_{t_j \in \mathbf{T}_{\bar{\mathbf{A}}}} \left(U_j^\Phi\right)^2 + \sum_{t_j \in \mathbf{T}_{\bar{\mathbf{B}}}} \left(U_j^\Psi\right)^2,$$

and add the constraints in Eqs. (10a)-(10e) for all differentials of Φ, Ψ. As each iteration ends, we update the rotations of Eq. (10b) for all differentials using their polar decomposition as explained above.

Algorithm 3: Path Lifting

Input: Polygonal path $\gamma = [\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_n, \mathbf{x}]$ in the plane
A point $\mathbf{y}_0 \in \bar{\mathbf{B}}$ known to be corresponding to $\mathbf{x}_0$
G-flattening $\Psi : \bar{\mathbf{B}} \to \mathbb{R}^2$
Output: A lift $\tau = [\mathbf{y}_0, \mathbf{y}_1, ..., \mathbf{y}] \subset \bar{\mathbf{B}}$ such that $\mathbf{y} = f(\mathbf{x})$

if γ *contains only one vertex,* $\boldsymbol{x}_0$ **then**
  return $\mathbf{y}_0$
Let $e = (\mathbf{x}_0, \mathbf{x}_1]$ be the first edge in γ
Let $\mathcal{R} := \Psi(\mathcal{R}_{\bar{\mathbf{B}}}(\mathbf{y}_0))$ the image of the one-ring of $\mathbf{y}_0$
if e *is contained in* $\mathcal{R}$ **then**
  Compute $\mathbf{y}_1 \in \bar{\mathbf{B}}$ by solving $\Psi|_{\mathcal{R}_{\bar{\mathbf{B}}}(\mathbf{y}_0)}(\mathbf{y}_1) = \mathbf{x}_1$
  run Path Lifting on $\gamma' = [\mathbf{x}_1, ..., \mathbf{x}_n, \mathbf{x}]$, and $\mathbf{y}_1$
else
  find the edge e' of $\mathcal{R}$ which e crosses and the point of intersection $\mathbf{x}'$.
  if e' *is part of a seam,* β^s *of* $\bar{\boldsymbol{B}}$*,* $s \in r, l$ **then**
    Compute $\mathbf{y}' \in \bar{\mathbf{B}}$ by solving $\Psi|_{\mathcal{R}_{\bar{\mathbf{B}}}(\mathbf{y}_0)}(\mathbf{y}') = \mathbf{x}'$
    let β^t be the twin seam of β^s
    let g_β be the seam-transformation taking β^s to β^t
    set $\mathbf{y}_1$ to be the corresponding point on β^t to $\mathbf{y}'$
    set $\gamma' = g_\beta([\mathbf{x}', \mathbf{x}_1, ..., \mathbf{x}_n, \mathbf{x}])$
    run path lifting on $\gamma', \mathbf{y}_1$
  else
    Run Path Lifting on $\gamma' = [\mathbf{x}_0, \mathbf{x}', \mathbf{x}_1, ..., \mathbf{x}_n, \mathbf{x}]$ and $\mathbf{y}_0$

Appendix C Lifting

We next detail the algorithm we use to compute the mapping f from the flattenings, as discussed in Subsection 7.3. As mentioned there, the algorithm in initiated with a polygonal curve $\gamma = \Phi(\Gamma)$. We denote by $\gamma = [\mathbf{x}_0, ..., \mathbf{x}_n]$ the polygonal curve where each $\mathbf{x}_i, \mathbf{x}_{i+1}$ is an edge, and by $\mathcal{R}(\mathbf{y})$ we denote the 1-ring of $\mathbf{y}$ in the mesh $\bar{\mathbf{B}}$, namely the union of all faces containing it (even if $\mathbf{y}$ is not a vertex).