Surface Maps via Adaptive Triangulations
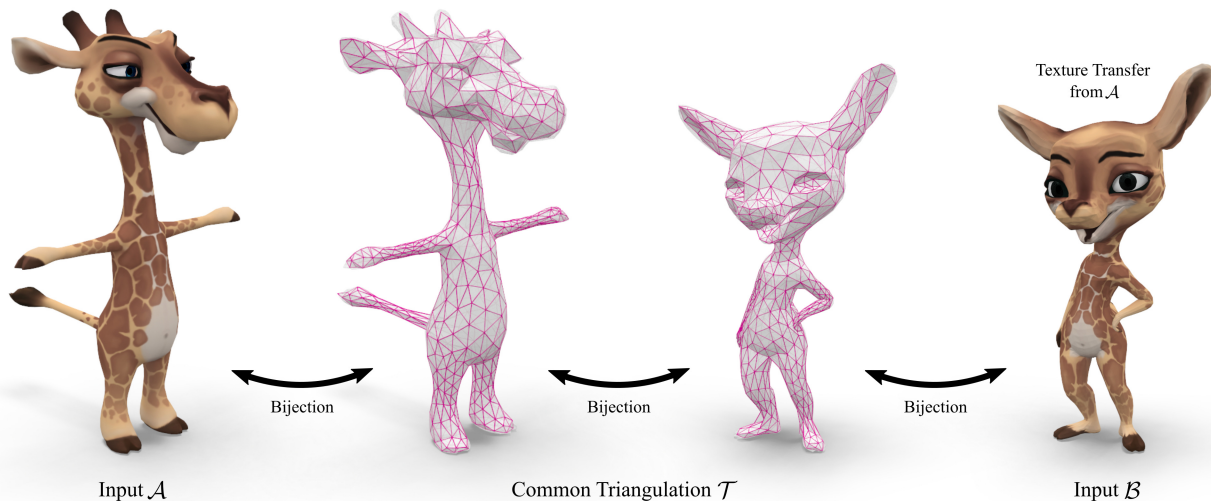
P. Schmidt[1] [iD] D. Pieper[1] [iD] L. Kobbelt[1] [iD]

[1]RWTH Aachen University, Germany

Figure 1: *Bijective map between genus-0 models, visualized via texture transfer. The map is represented by an approximating (rather than exact) common triangulation, which remains in bijective correspondence to the input surfaces via spherical parametrizations. In a discrete-continuous optimization, we treat both the connectivity and the geometric embeddings of the triangulation as degrees of freedom. This allows optimizing genus-0 surface homeomorphisms at adaptive resolutions, independently of the input mesh complexity, which can be simpler, faster, and more robust than existing overlay-based methods.*

Abstract

We present a new method to compute continuous and bijective maps (surface homeomorphisms) between two or more genus-0 triangle meshes. In contrast to previous approaches, we decouple the resolution at which a map is represented from the resolution of the input meshes. We discretize maps via common triangulations that approximate the input meshes while remaining in bijective correspondence to them. Both the geometry and the connectivity of these triangulations are optimized with respect to a single objective function that simultaneously controls mapping distortion, triangulation quality, and approximation error. A discrete-continuous optimization algorithm performs both energy-based remeshing as well as global second-order optimization of vertex positions, parametrized via the sphere. With this, we combine the disciplines of compatible remeshing and surface map optimization in a unified formulation and make a contribution in both fields. While existing compatible remeshing algorithms often operate on a fixed pre-computed surface map, we can now globally update this correspondence during remeshing. On the other hand, bijective surface-to-surface map optimization previously required computing costly overlay meshes that are inherently tied to the input mesh resolution. We achieve significant complexity reduction by instead assessing distortion between the approximating triangulations. This new map representation is inherently more robust than previous overlay-based approaches, is less intricate to implement, and naturally supports mapping between more than two surfaces. Moreover, it enables adaptive multi-resolution schemes that, e.g., first align corresponding surface regions at coarse resolutions before refining the map where needed. We demonstrate significant speedups and increased flexibility over state-of-the art mapping algorithms at similar map quality, and also provide a reference implementation of the method.

CCS Concepts

• *Computing methodologies* → *Computer graphics;* *Mesh models; Mesh geometry models;*
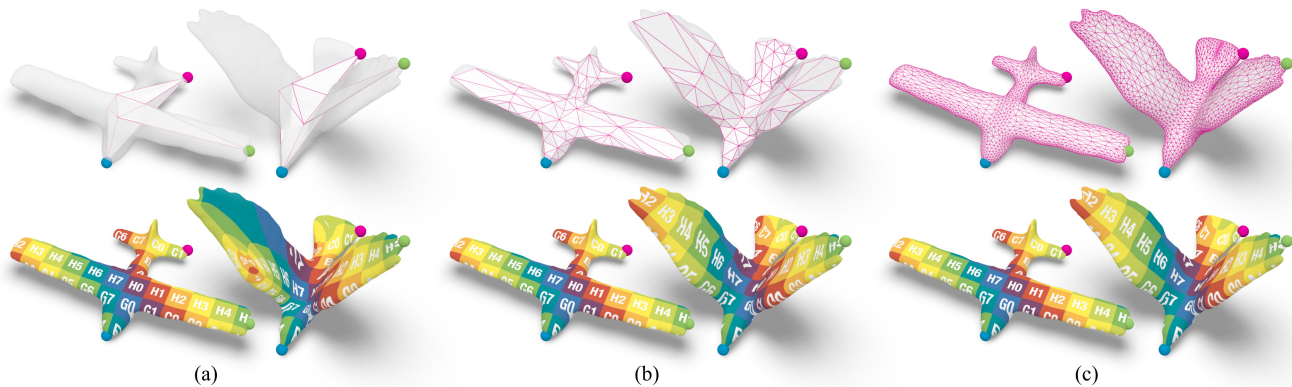
Figure 2: *Coarse-to-fine optimization of a common triangulation (top), which defines a bijective map between the genus-0 input surfaces (bottom). (a) An extremely coarse triangulation is optimized to initially satisfy three landmark constraints. The implied homeomorphism between the original surfaces is already of high quality near the landmarks, but lacks geometric feature alignment in other areas. (b) While only slightly refining the triangulation, the map is optimized for low distortion, which properly aligns similarly curved regions. (c) Finally, the common triangulation is refined to a target resolution, allowing for fine-level map adjustments. All three snapshots are part of a discrete-continuous optimization sequence (taking 40 sec). The change in mesh resolution is driven solely by adjusting parameters of the objective function between the stages (a), (b), and (c). A video of the full sequence is provided in the supplementary material.*

1. Introduction

Maps between the surfaces of two or more 3D models are an important ingredient in many different geometry processing tasks. They are required when, e.g., transferring data between surfaces, processing multiple surfaces in correspondence, interpolating between shapes, etc. Of particular interest is the class of *surface homeomorphisms*: continuous and bijective maps that are defined at every surface point and are guaranteed to be free of tearing or foldover artifacts. Most desirable is optimizing such homeomorphisms for low mapping distortion.

Despite vast literature on relaxed versions of this problem, optimizing strict homeomorphisms between discrete surfaces remains extremely challenging. Only a handful of methods [SAPH04, LDRS05, SBCK19, SCBK20, Tak22] are known to combine both bijectivity guarantees and optimization of surface-to-surface distortion (as opposed to distortion into an intermediate domain). While being the first to operate in this difficult problem setting at all, the above approaches result in complex algorithms that are not easy to implement, full of robustness challenges, and slow at run time (easily up to hours for a single pair of moderately-sized input meshes). A key source of this complexity is the computation of common tessellations that *exactly* represent a pair of input meshes via a set of planar elements, which are necessary for distortion assessment. These tessellations contain the vertices of both input meshes and can come in form of *overlay meshes* [SAPH04, SBCK19, SCBK20] (additionally containing all edge-edge intersections as vertices), or compatible intrinsic triangulations [Tak22] (whose construction currently remains heuristic). They usually need to be re-computed [SBCK19, SCBK20, Tak22] or updated [SAPH04] in each step of an optimization algorithm. Besides the mere cost of computing these exact common tessellations, their use as integration domain inevitably ties the size of the optimization problem to the input mesh resolution. Moreover, it is difficult to achieve full numerical robustness using overlay meshes, for example, due to the inherent

Figure 3: *Approximating common triangulation with 581 vertices vs. overlay mesh with 105 677 vertices as map representation.*

presence of near-degenerate elements (Figure 3) or because of the ill-conditioned problem of intersecting almost parallel edges.

We avoid computing overlay meshes altogether and instead measure distortion via *common triangulations* that only *approximate* the input surfaces rather than representing them exactly. This improves robustness (due to full control over element quality), increases performance (due to independence from input resolution), and eventually leads to simpler algorithms. That is, our approach trades the robustness and complexity issues of overlay meshes for having to control the approximation quality of common triangulations. We find this trade-off to be extremely worthwhile: We are now able to compute low-resolution homeomorphisms between the original-resolution input meshes within seconds (rather than minutes or hours), making iterative workflows a lot more practical.

A related problem setting is *compatible remeshing*, where two (or more) surfaces are re-triangulated in correspondence [KS04, YFC*18, YZL*20]. This, however, requires an underlying surface map, which is often computed in a pre-process. Such a two-step approach requires solving the surface mapping problem—with all its challenges—before even starting a remeshing procedure. Exact distortion assessment via overlay meshes in the first step can be a superfluous effort, especially when switching to approximating triangulations in the second step anyways. Moreover, the geometric quality of a particular common triangulation in the second step can often be improved by adjusting the underlying surface map; a degree of freedom that is not always fully exploited (e.g. only by local vertex re-locations [KS04, YZL*20]).
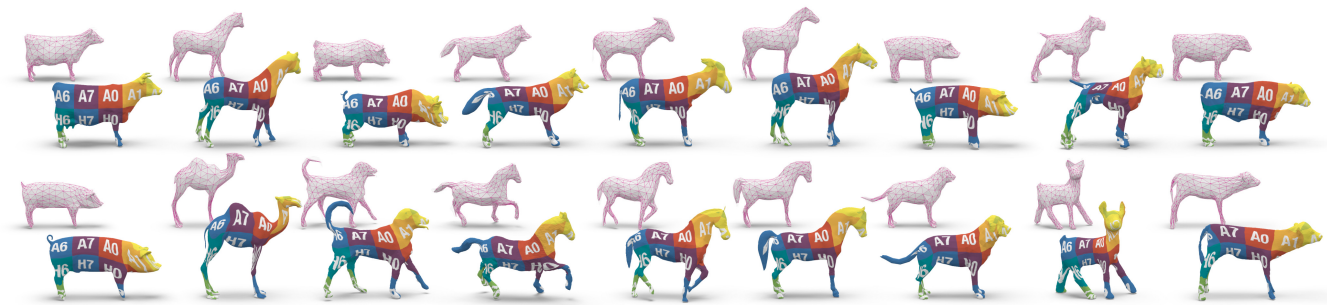
Figure 4: *Cycle-consistent maps between 18 meshes, simultaneously optimized in 5 minutes via a common triangulation of only 513 vertices.*

1.1. Contribution

We combine the tasks of surface map optimization and compatible remeshing in a single problem formulation. Our approach is based on a new map representation that defines

- continuous and bijective maps (surface homeomorphisms) between two or more genus-0 triangle meshes
- via common approximating triangulations (with guaranteed bijective correspondence to the input surfaces).

Depending on the application, either the homeomorphism or the common triangulation can be viewed as the output of our method. We present a discrete-continuous optimization algorithm that

- leverages both the connectivity of the common triangulation as well as its geometric embeddings as degrees of freedom,
- allows flexible control over mapping distortion, mesh quality, and surface approximation via a single objective function, and
- enables adaptive multi-resolution schemes that, e.g., perform alignment of geometric features at coarse resolutions before refining the map to target quality.

This new approach is a step towards making bijective surface map optimization more practically available. It is inherently more robust than previous overlay-based mapping approaches [SAPH04, SBCK19, SCBK20], as it avoids their most intricate challenges. We demonstrate significant speedups while attaining similar map quality. Compared to state-of-the-art compatible remeshing algorithms [YZL*20] we are able to fulfill the same kinds of error bounds while, in addition, directly optimizing surface-to-surface distortion. Our formulation naturally allows mapping between more than two surfaces and can be adapted to different scenarios by adding custom objective terms. We argue that it is also simpler to implement than many overlay-based mapping algorithms and provide a reference implementation[1].

To demonstrate the practical applicability of our method, we repeat a surface mapping and compatible remeshing task from developmental biology [ESD*22], where a time-continuous model of heart tube formation in mice is estimated by interpolating within a data set of 51 surfaces. We reduce the total computation time of the experiment from originally 36 hours via [SCBK20] to 38 minutes using our method.

---

[1] github.com/patr-schm/surface-maps-via-adaptive-triangulations

1.2. Method Overview

We start by introducing our approach for a pair of genus-0 input surfaces and later generalize it to more than two shapes (Section 7).

First, we independently map both input surfaces to instances of the unit sphere and keep these maps fixed throughout our algorithm. We then construct a common triangulation which we also bijectively embed on both spheres, establishing a correspondence between them. This defines a surface homeomorphism as the composition of three maps: (1) from the first input surface to the sphere, (2) from one sphere to the other sphere, (3) from the sphere to the second input surface (Section 4). Only the intermediate map is considered variable.

Lifting the vertices of the common triangulation to the two input surfaces and connecting them via flat triangles in 3D defines the geometry of the common triangulation, based on which we formulate an objective function (Section 5). In a continuous optimization step, we differentiate the objective with respect to the vertex positions of the common triangulation on the sphere and perform global second-order updates via a projected-Newton method (Section 6.2).

To update the connectivity of the common triangulation, we perform incremental remeshing via edge splits, collapses, and flips (Section 6.1). In contrast to classical algorithms [BK04, DVBB13], we replace the criterion for these operations by our objective function: we perform local connectivity updates if they improve the current objective value. Instead of a subsequent smoothing step, we alternate remeshing with continuous optimization iterations.

2. Related Work

We give a brief overview of adjacent literature in surface mapping and compatible remeshing.

2.1. Strict Surface Homeomorphisms

Via Intermediate Domains. A variety of methods represents surface homeomorphisms by composing bijective maps into an intermediate domain such as the plane [KSK97, LDRS05, TDIN*11, WZ14, APL14, APL15, AL15, SBCK19, SCBK20, Liv21, YZL*20], the sphere [Ale00, APH05, PT16, AKL17, BCK18, SCBK20], the hyperbolic plane [LBG*08, TFV*13, AL16, SZS*16, SCBK20], or a piecewise flat base mesh [LDSS99, PSS01, KS04]. The supported surface topology (e.g. disk or closed surfaces of a specific genus) is often restricted by the total curvature of the intermediate domain.

Some approaches achieve more flexibility by introducing cone singularities [TFV*13, AL15, AL16, AKL17], which allow extra curvature at specific points, but usually dictate a fixed correspondence at these points.

Without Intermediate Domains. Direct representations avoid the extra step through an intermediate domain, but are faced with additional consistency challenges. Notably, [SAPH04] need to carefully maintain an overlay connectivity throughout their algorithm (instead of re-computing it from an intermediate domain in each step). A recent work [Tak22] defines homeomorphisms via compatible intrinsic triangulations, constructed from a pair of vertex-to-surface maps in each step. While the representation is conceptually very attractive, current algorithms for its construction are not guaranteed to succeed. In a different context, [JSZP20] define homeomorphisms between a family of surfaces within a volume around the input surface via a bijective projection operator in ambient space.

Map Topology. Choosing a map's homotopy class (cf. [FM11]), i.e., how it twists around surface handles, tunnels, and fixed correspondences is an ongoing research effort [KS04, SAPH04, LGQ08, BSK21, BSCK21]. In this paper, we separate our problem setting from these research questions by focusing on the topologically simpler case of mapping between genus-0 surfaces via the sphere.

Distortion Optimization and Overlay Meshes. Optimizing distortion from a surface to an intermediate domain, e.g., the plane, is well-studied [KGL16, RPPSH17, CBSS17, SPSH*17, LYNF18, ZBK18, NZZ20, SYLF20, BN21], but does not in general yield low distortion of the composed, surface-to-surface map. An exception are conformal maps, whose composition is conformal again [LBG*08, APL15, BCK18]. However, isometric distortion measures are often of higher practical interest. The main challenge in optimizing surface-to-surface distortion arises from different mesh connectivities of source and target surface. Methods evaluate distortion either inexactly via sampling [LDRS05], or exactly via overlay meshes [SAPH04, SBCK19, SCBK20] or intrinsic triangulations [Tak22]. Overlay meshes as (piecewise-linear) map representation, but not necessarily for distortion optimization, are also used in [KSK97, LDSS99, Ale00]. They also appear (implicitly or explicitly) in works that consider different (intrinsic) triangulations of the same surface [FSSB07, SSC19, CCS*21, GSC21b, GSC21a].

2.2. Relaxed Map Representations

Distortion assessment can become easier when relaxing the strict continuity and bijectivity constraints. One option is to extrinsically deform the source mesh towards the target surface [SBSCO06, ZLJW06, WPYM07, HAWG08, LSP08, TCL*13] and sometimes also vice versa [ESBC19, EHA*19]. This allows working with a fixed mesh connectivity, but comes with the additional challenge of constraining the deformation to the target surface. Methods often rely on extrinsic projection operators, which usually cannot guarantee continuity and bijectivity, although some methods can optimize towards these goals [ESBC19, EHA*19]. Furthermore, a number of recent works use neural networks to represent such deformations, either optimized for each pair of input shapes [MAKM21, HPG*22] or trained on shape collections [AGK*22].

Other methods further relax the problem of mapping between surface points to mapping between soft distributions [SNB*12, MCSK*17] or mapping between surface functions expressed in a spectral basis [OBCS*12, RMC15, GBKS18, MRR*19]. The functional settings is attractive, because maps can be represented in a linear and low-dimensional manner, however no guaranteed conversion to strict homeomorphisms is known.
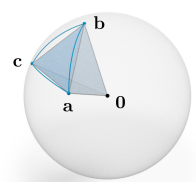
2.3. (Compatible) Remeshing

Incremental remeshing algorithms on single shapes typically generate approximating surface meshes via a series of local mesh connectivity modifications and geometric update steps. While classical methods are driven solely by a target edge length criterion [BK04, DVBB13], a different approach is to formulate the task as an energy minimization problem [ZWG*22]. Besides soft optimization goals, many methods are able to achieve hard quality bounds, e.g., in terms of surface approximation error [MCSA15, HYB*16, CFZC19, JSZP20, ZWG*22] or mesh quality [YW15, YLH18, ZWG*22]. Yet, only very few of these parametrization-free approaches offer a bijection to the input surface [JSZP20].

In compatible remeshing, a single mesh connectivity with geometric embeddings on two (or more) target surfaces is generated. To this end, most methods initially pre-compute a surface map, which is then kept fixed [MKFC01, PT16] or only updated in a very limited manner during remeshing, e.g., via local vertex relocation [KS04, YZL*20]. Distortion optimization via global updates is addressed in [YFC*18], however, at the expense of having to solve a volumetric map construction problem in 3D and losing bijectivity to the target surface.

In a 2D setting, [Liv20] and [ZPBK17] also address a mapping problem via mesh generation. Similarly to our method, the latter performs discrete-continuous optimization based on a single objective that combines mapping distortion and triangulation quality.

3. Preliminaries: Triangulations on the Sphere

Given a closed 2-manifold triangle mesh $\mathcal{A} = (\mathcal{V}_{\mathcal{A}}, \mathcal{E}_{\mathcal{A}}, \mathcal{F}_{\mathcal{A}})$ of genus 0, a *spherical embedding* is defined by assigning to each vertex a position on the unit sphere $\mathcal{S}^2 \subset \mathbb{R}^3$. Spherical triangles are formed by connecting adjacent vertices via shortest geodesics (great-circle arcs). A triangle is positively oriented if the signed volume of the tetrahedron spanned by the origin $\mathbf{0}$ and the vertices $\mathbf{a}, \mathbf{b}, \mathbf{c} \in \mathcal{S}^2$ is positive (see inset). Note that this condition also excludes adjacent vertices from being antipodal and triangles from spanning a full hemisphere. We choose to parametrize the interior of spherical triangles via barycentric interpolation between $\mathbf{a}, \mathbf{b}, \mathbf{c}$ in the ambient space $\mathbb{R}^3$, followed by a central projection to the sphere (cf. [GGS03]):

$$\mathbf{p}' = \alpha\mathbf{a} + \beta\mathbf{b} + (1 - \alpha - \beta)\mathbf{c}, \qquad \mathbf{p} = \frac{\mathbf{p}'}{\|\mathbf{p}'\|}. \qquad (1)$$

Equipped with this interpolation operator, a spherical embedding establishes a piecewise map $f_{\mathcal{A}} : \mathcal{A} \to \mathcal{S}^2$ from all points on the surface $\mathcal{A} \subset \mathbb{R}^3$ to the sphere. As interpolation in adjacent triangles

agrees along their common edge, $f_{\mathcal{A}}$ is continuous. If all spherical triangles are positively oriented (via the above condition) and the sphere is covered exactly once, the map $f_{\mathcal{A}}$ is bijective. Its inverse map $f_{\mathcal{A}}^{-1}$ can be evaluated at $\mathbf{p} \in \mathcal{S}^2$ by intersecting the ray from $\mathbf{0}$ through $\mathbf{p}$ with the flat triangle spanned by $\mathbf{a}, \mathbf{b}, \mathbf{c}$ in $\mathbb{R}^3$ and expressing the intersection point $\mathbf{p}'$ in barycentric coordinates [SCBK20]:

$$\alpha = \frac{\det[\mathbf{p}, \mathbf{c}, \mathbf{c} - \mathbf{b}]}{\det[\mathbf{p}, \mathbf{c} - \mathbf{a}, \mathbf{c} - \mathbf{b}]} \qquad \beta = \frac{\det[\mathbf{p}, \mathbf{c} - \mathbf{a}, \mathbf{c}]}{\det[\mathbf{p}, \mathbf{c} - \mathbf{a}, \mathbf{c} - \mathbf{b}]}. \qquad (2)$$

4. Map Representation

The input to our method is a pair of genus-0 triangle meshes $\mathcal{A} = (\mathcal{V}_{\mathcal{A}}, \mathcal{E}_{\mathcal{A}}, \mathcal{F}_{\mathcal{A}})$ and $\mathcal{B} = (\mathcal{V}_{\mathcal{B}}, \mathcal{E}_{\mathcal{B}}, \mathcal{F}_{\mathcal{B}})$, embedded in $\mathbb{R}^3$ with no degenerate triangles. Optionally, a sparse set of corresponding landmark vertices can be supplied. The output is a continuous and bijective map $\Phi : \mathcal{A} \rightarrow \mathcal{B}$ assigning to every point on the surface of $\mathcal{A}$ a point on the surface of $\mathcal{B}$. A generalization to maps between more than two surfaces is straightforward and described in Section 7.

Map Composition. We consider bijective sphere embeddings $f_{\mathcal{A}} : \mathcal{A} \rightarrow \mathcal{S}^2$ and $f_{\mathcal{B}} : \mathcal{B} \rightarrow \mathcal{S}^2$ of the input meshes. In contrast to other approaches, these do not directly define a map between $\mathcal{A}$ and $\mathcal{B}$. Instead, they are brought into correspondence via a third triangulation $\mathcal{T} = (\mathcal{V}_{\mathcal{T}}, \mathcal{E}_{\mathcal{T}}, \mathcal{F}_{\mathcal{T}})$. This abstract triangulation has two different spherical embeddings $g_{\mathcal{A}}$ and $g_{\mathcal{B}}$ that define a piecewise map $\psi : \mathcal{S}^2 \rightarrow \mathcal{S}^2$ from the sphere to the sphere (see Figure 5). The composition of these three maps, each of which is continuous and bijective and uses the interpolation operator from Section 3, yields the surface homeomorphism

$$\Phi := f_{\mathcal{B}}^{-1} \circ \psi \circ f_{\mathcal{A}}.$$

Degrees of Freedom. By introducing the additional intermediate map ψ, we are able to keep the input mesh embeddings $f_{\mathcal{A}}$ and $f_{\mathcal{B}}$ fixed, while only optimizing ψ. The geometry of the map ψ is parametrized via the two sets of vertex positions of $\mathcal{T}$ on the sphere, i.e., via $g_{\mathcal{A}}$ and $g_{\mathcal{B}}$. In addition, we treat the mesh connectivity of $\mathcal{T}$ as a degree of freedom. That is, we can adaptively refine $\mathcal{T}$ where more map resolution is needed and coarsen it in other regions to reduce computational complexity. Depending on the application, $\mathcal{T}$ can be significantly coarser than the input meshes $\mathcal{A}$ and $\mathcal{B}$. At any time, the state of our algorithm is fully represented by the meshes $\mathcal{A}, \mathcal{B}, \mathcal{T}$ and the sphere embeddings $f_{\mathcal{A}}, f_{\mathcal{B}}, g_{\mathcal{A}}, g_{\mathcal{B}}$. A state is valid if all sphere embeddings are bijective.

Approximating Common Triangulation. Lifting the vertices of $\mathcal{T}$ to the input surfaces $\mathcal{A}$ and $\mathcal{B}$ (via $f_{\mathcal{A}}^{-1} \circ g_{\mathcal{A}}$ and $f_{\mathcal{B}}^{-1} \circ g_{\mathcal{B}}$) and connecting them via flat triangles in $\mathbb{R}^3$ yields meshes $\mathcal{T}_{\mathcal{A}}$ and $\mathcal{T}_{\mathcal{B}}$ that approximate the input surfaces. Since these share the same connectivity $\mathcal{T}$, a piecewise-linear homeomorphism $\bar{\Phi} : \mathcal{T}_{\mathcal{A}} \rightarrow \mathcal{T}_{\mathcal{B}}$ is trivially defined per triangle. A key choice of our approach is optimizing the quality of this approximating map $\bar{\Phi}$ (which is a lot easier to assess) as a proxy for the quality of the exact map Φ (which would require costly mesh overlay computations). In applications where we are eventually interested in the full map Φ, this approximation can be seen as a speedup. However, in some applications
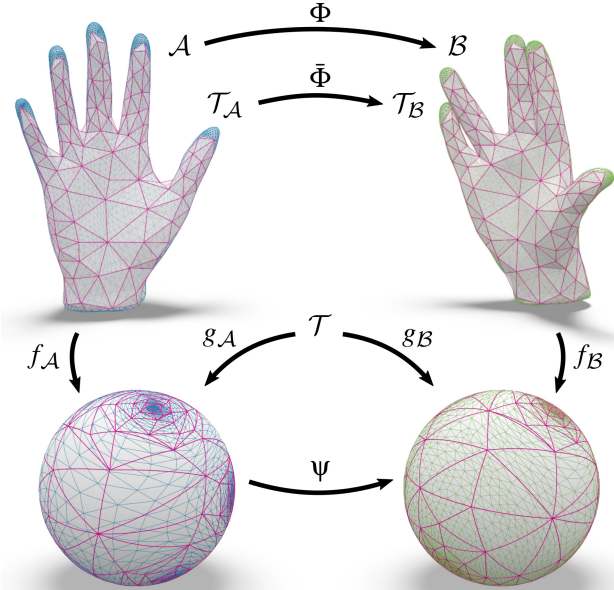

Figure 5: *A homeomorphism Φ between the input surfaces $\mathcal{A}$ (blue) and $\mathcal{B}$ (green) is represented by composing three maps: $f_{\mathcal{A}}$ from $\mathcal{A}$ to the sphere, ψ from one sphere to the other, and $f_{\mathcal{B}}^{-1}$ from the sphere to $\mathcal{B}$. The intermediate map ψ is defined via the common triangulation $\mathcal{T}$ (pink) on both spheres. Lifting the vertices of $\mathcal{T}$ to the input surfaces defines approximating triangulations $\mathcal{T}_{\mathcal{A}}$ and $\mathcal{T}_{\mathcal{B}}$, connected by the piecewise-linear homeomorphism $\bar{\Phi}$.*

(e.g. Section 8.5) the approximating meshes themselves are the desired output and we benefit from direct control over their resolution and geometry.

Initialization. The fixed sphere embeddings $f_{\mathcal{A}}, f_{\mathcal{B}}$ can be precomputed independently of each other via standard methods (Section 8.1). Mesh $\mathcal{T}$ and its embeddings can be initialized as any triangulation of the sphere. We chose a copy of mesh $\mathcal{A}$ in our experiments, i.e., $\mathcal{T} := \mathcal{A}$ and $g_{\mathcal{A}}, g_{\mathcal{B}} := f_{\mathcal{A}}$. This means that the intermediate map ψ is initialized as the identity. Satisfaction of landmark constraints is discussed in Section 8.2.

5. Objective Function

Both the continuous optimization and the remeshing algorithm are driven by the same objective function $E(\mathcal{T}, \mathbf{x}) \in \mathbb{R}$. It depends on the mesh connectivity $\mathcal{T}$ and its two sets of vertex positions on the sphere, parametrized via a variable vector $\mathbf{x} \in \mathbb{R}^n$. E is a sum of multiple terms controlling map distortion (Section 5.2), mesh quality (Section 5.3), surface approximation (Section 5.4), and maintaining strict bijectivity (Section 5.1). However, this formulation is flexible and custom objective terms can be added or removed depending on the application (see e.g. Section 8.4).

All terms are carefully designed to be (1) differentiable with respect to $\mathbf{x}$ and (2) at least C^0 continuous in the vertex positions on the sphere. This allows us to practically perform second-order optimization in the style of Newton's method, moving all vertices of $\mathcal{T}$
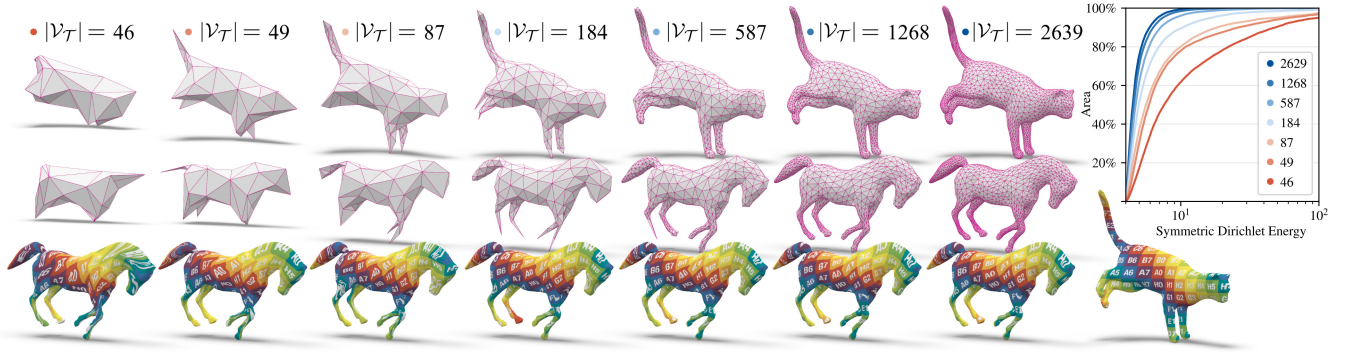
Figure 6: *Quality of the induced surface homeomorphism depending on the resolution of the common triangulation $\mathcal{T}$. Each column shows a converged result for a specific target resolution. While extremely coarse triangulations do not provide the necessary degrees of freedom to effectively reduce distortion, the map quality converges already at a moderate resolution of less than 3000 vertices. The plot shows, for each resolution, the amount of surface area below a distortion threshold.*

at once (Section 6.2). Note that a higher degree of continuity cannot easily be achieved without switching to higher-order surface representations: already the map $f_{\mathcal{A}}^{-1}$ is C^1 discontinuous across the edges of $\mathcal{A}$ due to the piecewise linear nature of the input mesh (as elaborated by e.g. [SBCK19, SCBK20]).

5.1. Bijectivity Barrier

The validity of our map representation depends on maintaining bijectivity of both sphere embeddings of $\mathcal{T}$. For $g_{\mathcal{A}}(\mathcal{T})$ (analogously for $g_{\mathcal{B}}(\mathcal{T})$) we impose a barrier function implementing the bijectivity condition from Section 3:

$$E_{\text{bij}}^{\mathcal{A}}(\mathcal{T}, \mathbf{x}) = \begin{cases} \infty & \text{any } \text{vol}(t) \leq 0 & \text{(3a)} \\ \infty & \sum_t \text{area}(t) \neq 4\pi & \text{(3b)} \\ -\sum_t \log(\text{vol}(t)) & \text{otherwise.} & \text{(3c)} \end{cases}$$

Cases 3a and 3c form a smooth barrier for positive triangle orientation by preventing the tetrahedral volume $\text{vol}(t) = \frac{1}{6} \det[\mathbf{a}, \mathbf{b}, \mathbf{c}]$, spanned by each triangle t with the origin, from degenerating. Additionally, Case 3b rejects jumps to multi-covers of the sphere, by checking the total oriented spherical area of the embedding.

5.2. Map Distortion

We measure distortion of the map $\bar{\Phi}$ between the two surface approximations via the symmetric Dirichlet energy. Because $\mathcal{T}_{\mathcal{A}}$ and $\mathcal{T}_{\mathcal{B}}$ share the same connectivity, the map is linear between each pair of corresponding triangles $t_{\mathcal{A}}$ and $t_{\mathcal{B}}$. Their vertex positions on the sphere are first lifted to $\mathbb{R}^3$ via the fixed maps $f_{\mathcal{A}}^{-1}$ and $f_{\mathcal{B}}^{-1}$ and then expressed in orthonormal 2D coordinate systems of $t_{\mathcal{A}}$ and $t_{\mathcal{B}}$ as $\mathbf{a}_{\mathcal{A}}, \mathbf{b}_{\mathcal{A}}, \mathbf{c}_{\mathcal{A}}$ and $\mathbf{a}_{\mathcal{B}}, \mathbf{b}_{\mathcal{B}}, \mathbf{c}_{\mathcal{B}} \in \mathbb{R}^2$. Based on the Jacobian

$$\mathbf{J}_t = [\mathbf{b}_{\mathcal{B}} - \mathbf{a}_{\mathcal{B}}, \mathbf{c}_{\mathcal{B}} - \mathbf{a}_{\mathcal{B}}][\mathbf{b}_{\mathcal{A}} - \mathbf{a}_{\mathcal{A}}, \mathbf{c}_{\mathcal{A}} - \mathbf{a}_{\mathcal{A}}]^{-1} \in \mathbb{R}^{2 \times 2},$$

we measure the Symmetric Dirichlet energy [SAPH04, SS15] of $\bar{\Phi}$:

$$E_{\text{map}}^{\mathcal{AB}}(\mathcal{T}, \mathbf{x}) = \frac{1}{4} \sum_{t \in \mathcal{F}_{\mathcal{T}}} \left(\text{area}(t_{\mathcal{B}}) \|\mathbf{J}_t\|_F^2 + \text{area}(t_{\mathcal{A}}) \|\mathbf{J}_t^{-1}\|_F^2 \right), \quad (4)$$

which we divide by 4 such that its minimum is 1 in case of unit surface area. Note that any other distortion energy based on the 3D

vertex positions of $\mathcal{T}_{\mathcal{A}}$ and $\mathcal{T}_{\mathcal{B}}$ could be used as well. In particular, it is not necessary to choose a flip-preventing energy, as map bijectivity is already guaranteed by the barrier term in Section 5.1.

5.3. Mesh Quality

Besides map distortion, we also control the resolution and element quality of the approximating triangulations $\mathcal{T}_{\mathcal{A}}$ and $\mathcal{T}_{\mathcal{B}}$. This is of particular importance in applications where the common triangulation, rather than the induced homeomorphism between the input surfaces, is considered the output of our method.

For each triangle $t_{\mathcal{A}}$ (and analogously for $t_{\mathcal{B}}$), we choose as its target shape $t_{\mathcal{A}}^*$ an equilateral triangle whose size $s(t_{\mathcal{A}}^*)$ is adaptively determined depending on the location on the input surface. The term $E_{\text{mesh}}^{\mathcal{A}}(\mathcal{T}, \mathbf{x})$ then measures the deviation of each triangle from its target shape, again via the symmetric Dirichlet energy as in Equation 4.

5.3.1. Adaptive Sizing

Consider a pair of sizing fields $s_{\mathcal{A}} : \mathcal{A} \to \mathbb{R}^{>0}$ and $s_{\mathcal{B}} : \mathcal{B} \to \mathbb{R}^{>0}$ on the input surfaces (defined at their vertices and linearly interpolated inside triangles). These fields control the adaptive resolution of $\mathcal{T}$ in different surface regions and also drive the multi-resolution aspect of our coarse-to-fine optimization strategy (Section 8.2). At corresponding locations on $\mathcal{A}$ and $\mathcal{B}$ we respect the finer of the two target resolutions, i.e., we treat undersampling as more severe than
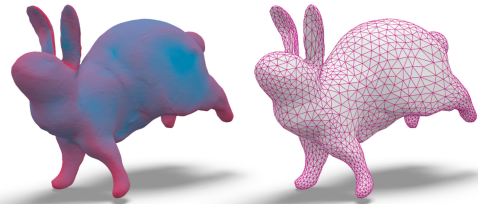

Figure 7: *(left) Curvature-based sizing field. (right) Our algorithm optimizes a symmetric Dirichlet energy between each triangle and an equilateral triangle whose size depends on the underlying field.*

oversampling (following [YZL*20]). Specifically, for a pair of corresponding triangles $t_\mathcal{A}$ and $t_\mathcal{B}$, we pick the values of $s_\mathcal{A}$ and $s_\mathcal{B}$ at their triangle midpoints and choose the smaller one as the common target size

$$s(t^*) = \min(s_\mathcal{A}(\mathbf{m}_\mathcal{A}), s_\mathcal{B}(\mathbf{m}_\mathcal{B}))),$$

where the triangle midpoints $\mathbf{m}_\mathcal{A} \in \mathcal{A}$ and $\mathbf{m}_\mathcal{B} \in \mathcal{B}$ are computed via the spherical interpolation operator in Equation 1.

Curvature-Based Sizing. By default, we compute the sizing fields $s_\mathcal{A}$ and $s_\mathcal{B}$ based on the maximum principal curvature and a desired surface approximation error $\tilde{\varepsilon}$, as described in [DVBB13]. That is, we choose finer resolutions in regions of high curvature and coarser resolutions in regions of low curvature (Figure 7). Our multi-resolution schedule (Section 8.2) is driven by adjusting the approximation parameter $\tilde{\varepsilon}$.

User-Supplied Sizing. Alternatively, one or both sizing fields can be provided as input to our method, which gives users manual resolution control (see Figure 8).

In Figure 6, we evaluate the effect of mesh resolution in $\mathcal{T}$ on the quality of the induced homeomorphism.

5.4. Surface Approximation

While the vertices of the approximating triangulation $\mathcal{T}_\mathcal{A}$ always lie on the input surface $\mathcal{A}$ by construction, the reverse statement is not true. Vertices of $\mathcal{A}$ do not lie exactly on $\mathcal{T}_\mathcal{A}$, because the flat triangles of $\mathcal{T}_\mathcal{A}$ in $\mathbb{R}^3$ leave the input surface. In the worst case, this allows thin geometric features of the input surface to poke through approximating triangles, even if the resolution is relatively fine (see inset, left).

To control the approximation error of $\mathcal{T}_\mathcal{A}$ (similarly for $\mathcal{T}_\mathcal{B}$), we measure the Euclidean distance from each input vertex position $\mathbf{v}$ of $\mathcal{A}$ to its corresponding base point $\bar{\mathbf{v}}$ on $\mathcal{T}_\mathcal{A}$. Instead of choosing the closest point on $\mathcal{T}_\mathcal{A}$, we use the base point $\bar{\mathbf{v}}$ defined by our bijective correspondence (see inset). This overestimates the shortest distance to the surface, but is better suited for optimization because $\bar{\mathbf{v}}$ slides over $\mathcal{T}_\mathcal{A}$ continuously as $\mathcal{T}_\mathcal{A}$ moves in $\mathbb{R}^3$.

The base point $\bar{\mathbf{v}}$ is obtained from $\mathbf{v}$ by computing its barycentric coordinates α, β in a triangle $\mathbf{a}, \mathbf{b}, \mathbf{c}$ of $\mathcal{T}$ on the sphere (Equation 2) and evaluating them again in $\mathbb{R}^3$:

$$\bar{\mathbf{v}} = \alpha f_\mathcal{A}^{-1}(\mathbf{a}) + \beta f_\mathcal{A}^{-1}(\mathbf{b}) + (1 - \alpha - \beta) f_\mathcal{A}^{-1}(\mathbf{c}).$$

We set up a quadratic penalty term measuring the approximation error per vertex of $\mathcal{A}$ (similarly for $\mathcal{B}$):

$$E_{\text{approx}}^{\mathcal{A}}(\mathcal{T}, \mathbf{x}) = \frac{1}{\tilde{\varepsilon}^2} \sum_{v \in \mathcal{V}_\mathcal{A}} \text{area}(v) \cdot \|\mathbf{v} - \bar{\mathbf{v}}(\mathbf{x})\|^2,$$

Figure 8: *(left) Mesh $\mathcal{A}$ with user-designed sizing field as input. (center, right) Common triangulation $\mathcal{T}$ embedded on $\mathcal{A}$ and $\mathcal{B}$, respecting the specified sizing.*

where area(v) is the normalized Voronoi area of vertex v. We divide by $\tilde{\varepsilon}^2$ to help balancing with other objective terms: $E_{\text{approx}}^{\mathcal{A}}$ integrates to 1 if the approximation error matches the target error $\tilde{\varepsilon}$ (that determines the mesh resolution) everywhere. In Section 8.4, we demonstrate replacing this soft penalty by a hard error bound.

The surface approximation term is the only one in our objective function that sums over the input vertices instead of over the elements of $\mathcal{T}$, implying a linear dependency on the input mesh complexity. For too high-resolution inputs, we break this dependency by only summing over a constant-sized subset of vertices.

5.5. Landmarks

A sparse set $\mathcal{L} = \{(v_\mathcal{A}, v_\mathcal{B}), ...\}$ of landmark vertices can be supplied as input. Since $v_\mathcal{A}$ and $v_\mathcal{B}$ might not initially correspond under Φ, we select a vertex $v_\mathcal{T}$ of $\mathcal{T}$ to represent this pair and encourage its two instances to move towards $v_\mathcal{A}$ and $v_\mathcal{B}$ via penalty terms

$$E_{\text{land}}^{\mathcal{A}} = \sum_{v_\mathcal{A} \in \mathcal{L}} \|f_\mathcal{A}(v_\mathcal{A}) - g_\mathcal{A}(v_\mathcal{T})\|^2$$

and $E_{\text{land}}^{\mathcal{B}}$, respectively. These terms compare vertex positions on the sphere, instead of on the surfaces, to avoid local minima and use the Euclidean distance for simplicity. For each pair, the representing vertex $v_\mathcal{T}$ is chosen as the one minimizing $E_{\text{land}}^{\mathcal{A}} + E_{\text{land}}^{\mathcal{B}}$ at initialization.

Once a landmark correspondence is fulfilled, i.e., the two instances of $v_\mathcal{T}$ coincide with $v_\mathcal{A}$ and $v_\mathcal{B}$ up to some tolerance, we can lock them in place by eliminating $g_\mathcal{A}(v_\mathcal{T})$ and $g_\mathcal{B}(v_\mathcal{T})$ from the optimization problem in Section 6.2.

5.6. Parameters

Finally, the combined objective function $E(\mathcal{T}, \mathbf{x})$ is a weighted sum of the above terms for bijectivity, mesh quality, surface approximation, landmark constraints, and map distortion:

$$
\begin{aligned}
E = \ &\omega_{\text{bij}} \ \frac{1}{2}\left(E_{\text{bij}}^{\mathcal{A}} + E_{\text{bij}}^{\mathcal{B}}\right) &&+ \omega_{\text{mesh}} \ \frac{1}{2}\left(E_{\text{mesh}}^{\mathcal{A}} + E_{\text{mesh}}^{\mathcal{B}}\right) \\
&+ \omega_{\text{approx}} \ \frac{1}{2}\left(E_{\text{approx}}^{\mathcal{A}} + E_{\text{approx}}^{\mathcal{B}}\right) &&+ \omega_{\text{land}} \ \frac{1}{2}\left(E_{\text{land}}^{\mathcal{A}} + E_{\text{land}}^{\mathcal{B}}\right) \\
&+ \omega_{\text{map}} \quad E_{\text{map}}^{\mathcal{AB}}.
\end{aligned}
\tag{5}
$$

Fortunately, we can choose reasonable default values for all weights: We set ω_{bij} to a small number (e.g. 10^{-6}), as we are only interested in the term's barrier character. Conversely, ω_{land} can always be a large number (e.g. 10^6), because we expect the landmark penalty term to vanish. Furthermore, we choose $\omega_{mesh} = 1$ and $\omega_{map} = 1$ to weigh mesh quality and mapping distortion equally. Both are comparable because they are symmetric Dirichlet energies, integrated over $\mathcal{T}$, with minimum 1. We normalize the size of all integration domains by initially scaling the input meshes $\mathcal{A}$ and $\mathcal{B}$ to unit surface area. The remaining weight ω_{approx}, does pose a trade-off between surface approximation and the ability of mesh $\mathcal{T}$ to slide over the input surfaces unobstructedly. However, since we normalized the approximation term to 1 for the case that the target error $\tilde{\varepsilon}$ is attained everywhere, we can also choose $\omega_{approx} = 1$. Intuitively, the approximation term does not dominate if the target error is respected.

This only leaves the target approximation error $\tilde{\varepsilon}$ to be chosen. We adapt this parameter to drive the different stages of our coarse-to-fine optimization schedule (Section 8.2), but eventually its final value is a user-parameter that defines the target resolution.

6. Discrete-Continuous Optimization

The optimization algorithm (Figure 9) alternates between discrete steps (updating the mesh connectivity of the common triangulation $\mathcal{T}$) and continuous steps (moving the vertices of $\mathcal{T}$ on the sphere), both such that the objective $E(\mathcal{T}, \mathbf{x})$ decreases monotonously.

6.1. Remeshing

The discrete step performs a series of edge splits, edge collapses, and edge flips. In contrast to classical incremental remeshing algorithms [BK04, DVBB13, YZL*20] with specialized criteria for each operation, we simply apply the ones that decrease the objective E. In addition, we disallow removal of landmark vertices.

We use a simple parallelization strategy: Per type of operation (split, collapse, flip), we first evaluate all possible candidates locally and in parallel. We then sort the ones that yield an improvement in E in descending order and apply them in serial (to avoid parallel mesh modifications). Because operations with overlapping area of effect influence each other, we check for improvement in E again before applying an operation.

For edge splits (and collapses), we choose the position of the new (remaining) vertex as the respective edge midpoint on the sphere. Instead of subsequent Laplacian smoothing, as e.g. in [BK04], we continue with one iteration of continuous optimization.

6.2. Continuous Optimization

We simultaneously move the vertex positions $g_{\mathcal{A}}(\mathcal{T})$ and $g_{\mathcal{B}}(\mathcal{T})$ on the sphere via projected-Newton updates. Following the instructions in [SBB*22, Sec. 4.4], we parametrize vertex trajectories as tangent vectors of the sphere under a retraction operator: We begin an iteration by choosing local 2D coordinate systems, tangent to the sphere, centered at the current vertex positions in $g_{\mathcal{A}}(\mathcal{T})$ and $g_{\mathcal{B}}(\mathcal{T})$. The variable vector $\mathbf{x} \in \mathbb{R}^n$ then encodes trajectories in

---

Algorithm 1: Discrete-Continuous Optimization

repeat
    1. Split edges of $\mathcal{T}$ if they improve E
    2. Collapse edges of $\mathcal{T}$ if they improve E
    3. Flip edges of $\mathcal{T}$ if they improve E
    4. Move vertices of $\mathcal{T}$ on the sphere (Newton step)
until converged or max iters reached

Figure 9: *Driven by the objective $E(\mathcal{T}, \mathbf{x})$, we alternate between updating the mesh connectivity of the common triangulation $\mathcal{T}$ and continuously moving its vertices over all input surfaces at once.*

these tangent spaces, i.e. $n = 4|V_{\mathcal{T}}|$, and we initially have $\mathbf{x} = \mathbf{0}$ in each iteration. The objective $E(\mathbf{x})$ first applies the trajectories $\mathbf{x}$ in 3D ambient space and then retracts them to the unit sphere via normalization, before evaluating Equation 5. That is, $E(\mathbf{x})$ is defined via composition with a retraction operator.

Based on the current choice of tangent spaces, we compute the derivatives of E with respect to $\mathbf{x}$ via automatic differentiation using TinyAD [SBB*22]. These are the gradient $\mathbf{g} \in \mathbb{R}^n$ and sparse Hessian $\mathbf{H}_+ \in \mathbb{R}^{n \times n}$ (projected to a positive definite matrix [TSIF05]). We then compute the Newton direction $\mathbf{d} \in \mathbb{R}^n$ by solving the linear system $\mathbf{H}_+ \mathbf{d} = -\mathbf{g}$, find a step size $s \in \mathbb{R}$ via backtracking line search, and set $\mathbf{x} := s\mathbf{d}$. Finally, we apply $\mathbf{x}$ (again via the retraction operator), to obtain the new vertex positions of $g_{\mathcal{A}}(\mathcal{T})$ and $g_{\mathcal{B}}(\mathcal{T})$. The next iteration begins by choosing a new set of local coordinate systems at the updated vertex positions and resetting $\mathbf{x} := \mathbf{0}$.

We consider the discrete-continuous algorithm to be converged when no more edge splits, collapses, or flips can yield improvement in E and the Newton decrement $\sqrt{-\mathbf{d}^\top \mathbf{g}}$ falls below a threshold (10^{-4} in our experiments).

7. Maps Between Multiple Surfaces

The map representation and algorithm introduced so far naturally extend to mapping between $N \geq 1$ surfaces. For $N = 1$ the method reduces to a remeshing algorithm for single surfaces, that maintains a bijection between input and output surface.

Map Representation. For N input meshes $\mathcal{M}_1, \ldots, \mathcal{M}_N$, we independently pre-compute sphere embeddings $f_1, \ldots, f_N$. The common triangulation is still defined via a single mesh connectivity $\mathcal{T}$, that now has N sphere embeddings $g_1, \ldots, g_N$. This representation yields pairwise homeomorphisms Φ_{ij} between all input surfaces. These homeomorphisms are cycle-consistent by construction, i.e., any composition of maps starting and ending at the same surface yields the identity map: $\Phi_{i,\bullet} \circ \cdots \circ \Phi_{\bullet,i} = \mathrm{Id}$. This is because any point on any surface is uniquely represented by its barycentric coordinates in a triangle of $\mathcal{T}$. See Figure 4 for an example with $N = 18$.

Objective Function. The terms for bijectivity (Section 5.1), mesh quality (Section 5.3), and surface approximation (Section 5.4) are defined individually per mesh $\mathcal{T}_i$. Only mapping distortion (Section 5.2) is measured between pairs $\mathcal{T}_i, \mathcal{T}_j$. While considering all pairwise maps $\bar{\Phi}_{ij}$ for distortion assessment seems most natural,

this leads to a number of terms quadratic in N, which is only practical for relatively small N. However, depending on the application scenario, measuring distortion only between a subset of pairs can be sufficient: e.g. sequentially (between surfaces with consecutive index), cyclic (additionally between first and last), or with star topology (compare a reference surface to all others), etc. The objective E is again a weighted sum as in Equation 5, where all per-surface terms are now normalized by N and the distortion term by the number of pairs. In the remeshing algorithm, E is still evaluated locally around a split/collapse/flip in $\mathcal{T}$, but now considering all N embeddings. In the continuous step, we differentiate E w.r.t. all $2N|\mathcal{V}_{\mathcal{T}}|$ variables and move the vertices of all N sphere embeddings at once.

8. Results & Applications

After giving some details on the setup of our experiments, we compare our method to current surface mapping algorithms, where we find large improvements in run time and convergence radius at comparable mapping distortion (Section 8.3). We then conduct an experiment in which we achieve the same bounds on surface approximation as a state-of-the art compatible remeshing algorithm (Section 8.4). Finally, we demonstrate an application to shape averaging and interpolation that benefits from mapping between multiple shapes at once (Section 8.5).

8.1. Initialization

Our algorithm initially requires bijective spherical embeddings of all input meshes. Conceptually, many different methods can be used for this step [KSBC12, CPS13, BCK18]. However, strict bijectivity is not always guaranteed and conformal methods may yield extreme scale distortion in the intermediate domain. For increased robustness, we instead construct embeddings via the hierarchical strategy of [PH03], while encouraging isometry: We decimate the genus-0 input mesh until only a tetrahedron remains (cf. [Hop96]), canonically embed the tetrahedron on the unit sphere, and then undo the decimation sequence while bijectively inserting vertices inside their one-ring on the sphere (see also [HFL17, SJZP19]). We locally optimize each inserted vertex and interleave the sequence with a few iterations of global optimization w.r.t. the angle and area-based objective function in [SCBK20, Sec. 6.6], using the same algorithm as in Section 6.2. Finally, we rotationally align all sphere embeddings to each other with respect to the supplied landmark correspondences.

8.2. Coarse-to-Fine Optimization

Depending on the application scenario our general algorithm can be applied in different ways. In our experiments, we use a simple coarse-to-fine schedule comprised of the following three phases.

Landmark Satisfaction. The above initialization procedure (with rigid spherical alignment) does not fulfill landmark constraints. In a first phase, we switch to an extremely coarse triangulation $\mathcal{T}$ to quickly optimize for constraint satisfaction (i.e. move landmark vertices of $\mathcal{T}$ towards fixed target positions on the spheres). We disable all objective terms except for the bijectivity barrier (Section 5.1) and landmark penalty (Section 5.5). Since the bijectivity
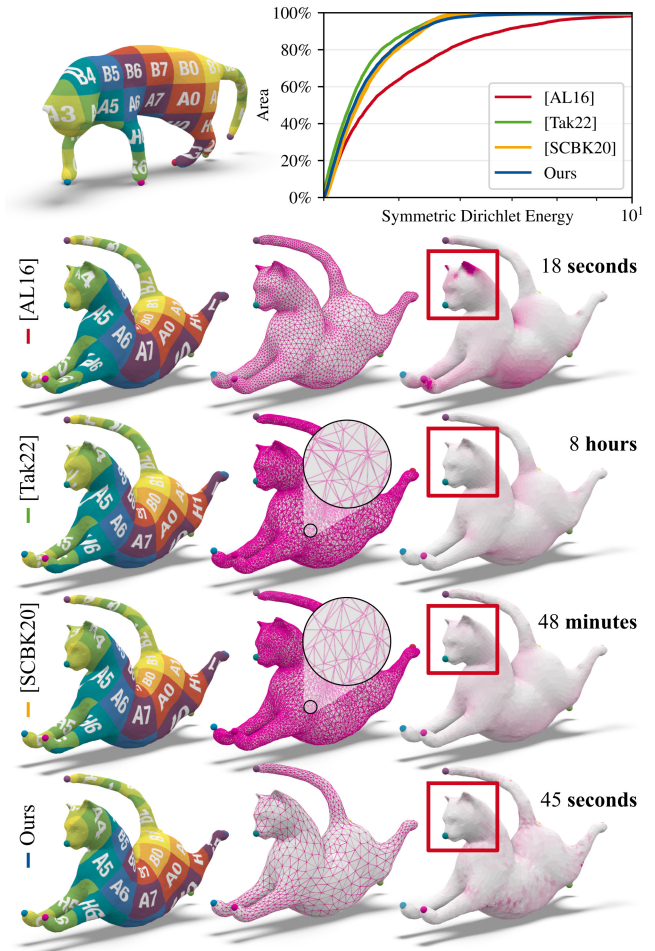

Figure 10: *Maps between two cat models with no landmarks at the ears. On this input, all three distortion-minimizing methods find the desired minimum (see heatmap). However, [Tak22] and [SCBK20], which are tied to the input mesh resolution, take 8 hours and 48 minutes, respectively. Our method completes the task in 45 seconds.*

barrier regularizes towards large triangles on the sphere, the algorithm automatically coarsens mesh $\mathcal{T}$ until almost only landmark vertices remain. We perform a maximum of 100 iterations or stop early when when all landmarks are within a threshold of their target position. We then lock the landmarks in place for all following phases (or release them if only supplied to initially guide the map towards the desired distortion minimum). Note that this procedure for initial landmark satisfaction is a simple heuristic to a difficult problem that involves choosing a map homotopy class (between the genus-0 input surfaces punctured at the landmark positions, cf. [BSK21]). While it is not in general guaranteed that our penalty term satisfies all landmark constraints, it succeeded within a few seconds in all examples shown here. In principle, if a landmark cannot be satisfied, the rest of our algorithm is still able to proceed without this constraint.

Coarse Map Optimization. We then perform the main map optimization workload, e.g., aligning similarly curved regions, at a rel-
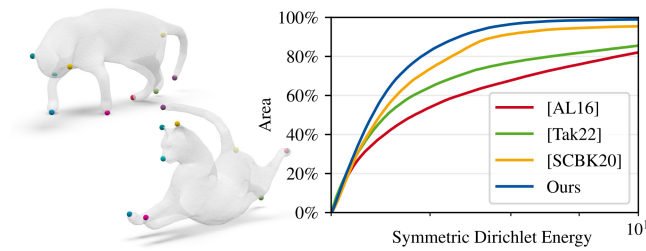
Figure 11: *Cumulative distortion over 8 maps. Each time, we disabled one of the input landmarks to test if the algorithm still correctly aligns geometric features and finds the global distortion minimum. [Tak22] and [SCBK20] get stuck more often than our algorithm does. [AL16] does not directly optimize mapping distortion.*

Figure 12: *Experiment from [SCBK20] testing resilience w.r.t. challenging initializations. After satisfying different sets of landmark constraints (top row), we release all landmarks and continue our optimization (bottom row). We find the global distortion minimum in cases where the initial map is reasonably close.*

atively coarse resolution. That is, we enable all objective terms as described in Section 5.6 and choose a target approximation error of $\tilde{\epsilon} = 0.01$ in all examples, which causes the algorithm to refine mesh $\mathcal{T}$ from the extremely coarse result of the landmark phase within a few iterations. We apply a total of 50 iterations in this coarse map optimization phase by default.

Map Refinement. Finally, we reduce the target approximation error to the desired value, depending on the application. We choose $\tilde{\epsilon} = 0.001$ as default and perform another 50 iterations. As a consequence, additional map refinement is performed, giving the distortion minimization the necessary degrees of freedom to make fine-level map improvements. Note again, that the parameter $\tilde{\epsilon}$ specifies a soft approximation goal which is translated into adaptive target edge lengths fields. We experiment with hard approximation bounds in Section 8.4. An example of the result after each algorithm phase is shown in Figure 2.

8.3. Comparison: Homeomorphic Map Optimization

We compare against two recent methods for distortion minimization of continuous and bijective maps: Inter-Surface Maps via Constant-Curvature Metrics [SCBK20] and Compatible Intrinsic Triangulations [Tak22]. For reference, we also include Hyperbolic Orbifold Tutte Embeddings [AL16], which is used as initialization for [Tak22]. We use the author's implementations with their default settings and evaluate run time, mapping distortion, and robustness with respect to different input landmark configurations. For comparability across different map representations, we measure distortion by deforming input mesh $\mathcal{A}$ according to its mapped vertex positions on $\mathcal{B}$. Throughout our evaluation, we show cumulative distortion plots stating the percentage of surface area that is below a certain distortion threshold, i.e., curves closer to the top left corner of the plot represent less distortion.

In Figure 10, we map between two near-isometric shapes with about 10k vertices combined and landmarks placed at all geometric extremities except for the cat's ears. [AL16], which does not optimize surface-to-surface distortion, quickly produces a smooth map, but slightly misaligns the ears. Starting from this input, [Tak22] correctly aligns the ears and also reduces distortion in other areas as much as possible, but takes 8 hours to terminate. [SCBK20] finds a similar low-distortion map within 48 minutes. We speculate that the

marginally higher overall distortion of [SCBK20] (see plot) could be due to regularization terms in the intermediate domain, which are absent in [Tak22]. Our method also successfully aligns the ears with comparable overall distortion. The distortion heatmap appears slightly less smooth due to the much lower resolution of our map representation (1665 vertices in our compatible triangulation vs. roughly 50k vertices in the overlay mesh of [SCBK20]). Our map computation took a total of 45 seconds (15 of which are spent on the initial sphere embeddings). This is a 64-fold improvement over [SCBK20] and 640-fold improvement over [Tak22]. Qualitative results of our method on less isometric inputs can be found in Figures 2,4,6,8,13, and 15.

To further test robustness and convergence radius, we again map between the same pair of shapes multiple times, each time releasing a different landmark (out of 9, see Figure 11). Missing landmarks at geometric extremities cause all methods to initially produce highly distorted maps (e.g. due to a leg mapped to a flat part of the surface). As all three distortion-minimizing methods ([Tak22], [SCBK20], ours) then attempt to solve a non-convex C^1-discontinuous optimization problem, it is possible for them to converge to local optima or get stuck in other ways. In one case (no landmark at tail) all but our method failed without producing a result (presumably due to too high distortion in the intermediate domains). [Tak22] failed or made no progress in 4 more cases (no landmarks at legs), but produced the desired map the other 4 cases. [SCBK20] did produce results in all 8 remaining cases but converged to local optima when landmarks at the legs were missing. Due to our coarse-to-fine approach, our method reached the desired map in all but one case (where it did not fully extend one of the legs). We report the cumulative distortion over all cases (except the one where all others failed) in Figure 11. When [Tak22] failed, we report the distortion of its initialization.

In Figure 12, we repeat an experiment from [SCBK20] where we compute initial maps between two hands from increasingly bad landmark correspondences. We then release these landmarks to see if the algorithm still converges to the desired distortion minimum, correctly aligning all fingers. Our method reaches the global optimum in 4 out of 5 cases; one more than [SCBK20]. However, on adversarial inputs, e.g., when the initial map provokes two fingers being mapped to one, our method converges to a local optimum as well.
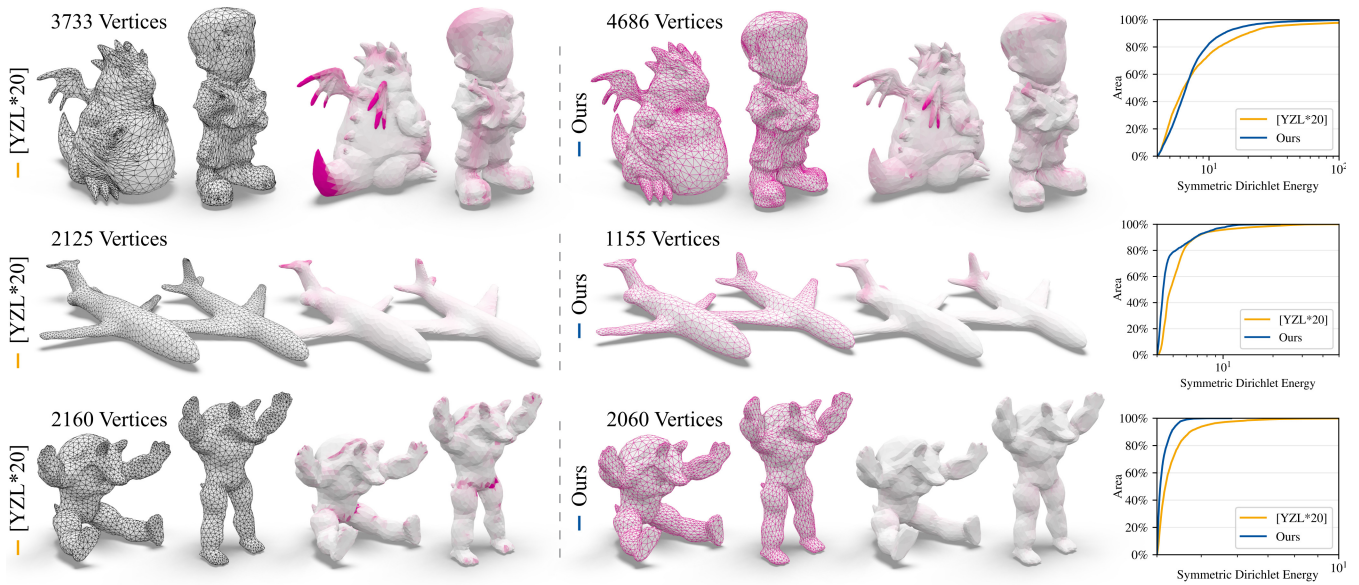
Figure 13: *Common triangulation with bounded surface approximation error. Similarly to [YZL*20] (left) we are able to satisfy error bounds via adaptive refinement. We then maintain these bounds using barrier functions, while reducing distortion via global map optimization (right).*

8.4. Comparison: Compatible Remeshing

To demonstrate a setting in which our common triangulation, instead of the homeomorphism, is considered the output of our method, we compare against a state-of-the art compatible remeshing algorithm [YZL*20]. The method pre-computes a map without surface-to-surface distortion optimization and can afterwards only deviate from it via local vertex re-locations. Hard error bounds on surface approximation are initially fulfilled via adaptive refinement and subsequently maintained by disallowing violating operations.

In an experiment we adapt our formulation to practically fulfill the same bounds as [YZL*20], while additionally maintaining access to global map optimization. After applying our default coarse-to-fine schedule, we satisfy an error bound ε on the symmetric Hausdorff distance (sampled at the input mesh vertices) by locally and iteratively decreasing the sizing fields $s_\mathcal{A}$ and $s_\mathcal{B}$ in areas where the bound is violated (following [CFZC19]). After bound satisfaction, we continue our optimization, but replace the soft surface approximation term $E_{\text{approx}}^{\mathcal{A}}$ by a barrier

$$E_{\text{bound}}^{\mathcal{A}}(\mathcal{T}, \mathbf{x}) = \begin{cases} \infty & \text{any } d(v) \geq \varepsilon \\ \sum_{v \in \mathcal{V}_{\mathcal{A}}} \text{area}(v) \cdot b_{\varepsilon}\left(d(v)\right) & \text{otherwise,} \end{cases}$$

where $d(v)$ is the Euclidean distance from an input mesh vertex to its base point on the approximating mesh (cf. Section 5.4). Inspired by [SKPSH13, Eq. 6], we use the barrier function $b_{\varepsilon}(d(v)) = \frac{d(v)^3}{\varepsilon^3 - d(v)^3}$, which diverges to infinity when $d(v)$ approaches ε and smoothly reaches 0 when $d(v)$ goes to 0 (see inset). Similarly to our bijectivity barrier, this disallows mesh modifications that violate the approximation bound and pushes the continuous optimization away from violating configurations.
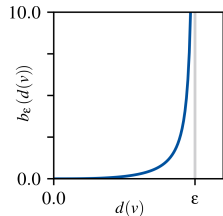
Figure 13 shows three examples in which we satisfy the same bound as [YZL*20] (0.3% of the bounding box diagonal) while reducing mapping distortion. In a highly non-isometric case we distribute the inevitable distortion more evenly while using slightly more vertices, whereas in simpler cases, we reduce both total distortion and vertex count compared to [YZL*20].

8.5. Application: Shape Averaging & Interpolation

Two operations that often appear in applications are shape averaging and interpolation between keyframe surfaces (e.g. via [HRS*16]). Both require establishing a common tessellation. To demonstrate the practical applicability of our method, we repeat an experiment from developmental biology [ESD*22], where an average growth process of the early mouse heart is estimated from a dataset of 51 surface meshes, obtained via confocal microscopy. The surfaces are clustered into 10 groups of similar developmental stage and the tasks are (1) computing an average surface per group and (2) sequentially interpolating between these average surfaces via a common triangulation. The original experiment in [ESD*22] was performed via [SCBK20] (initialized with [BSK21]), which only allows computing pairwise maps. Within each group, one surface was mapped to all others to allow averaging, and sequential maps were computed between consecutive pairs of average surfaces. This required computing a total of 50 pairwise maps, for each of which [SCBK20] took between 27 and 58 minutes on input meshes with 5k vertices each. In a post process, a remeshing algorithm [DVBB13, YZL*20] was run to improve the common triangulation of the sequence, based on fixed surface maps.

In contrast, since we are able to map between more than two surfaces, we can compute the average shapes with a single run of our method per group, followed by another run for the sequential mapping. Despite mapping between 3 to 10 surfaces at a time,
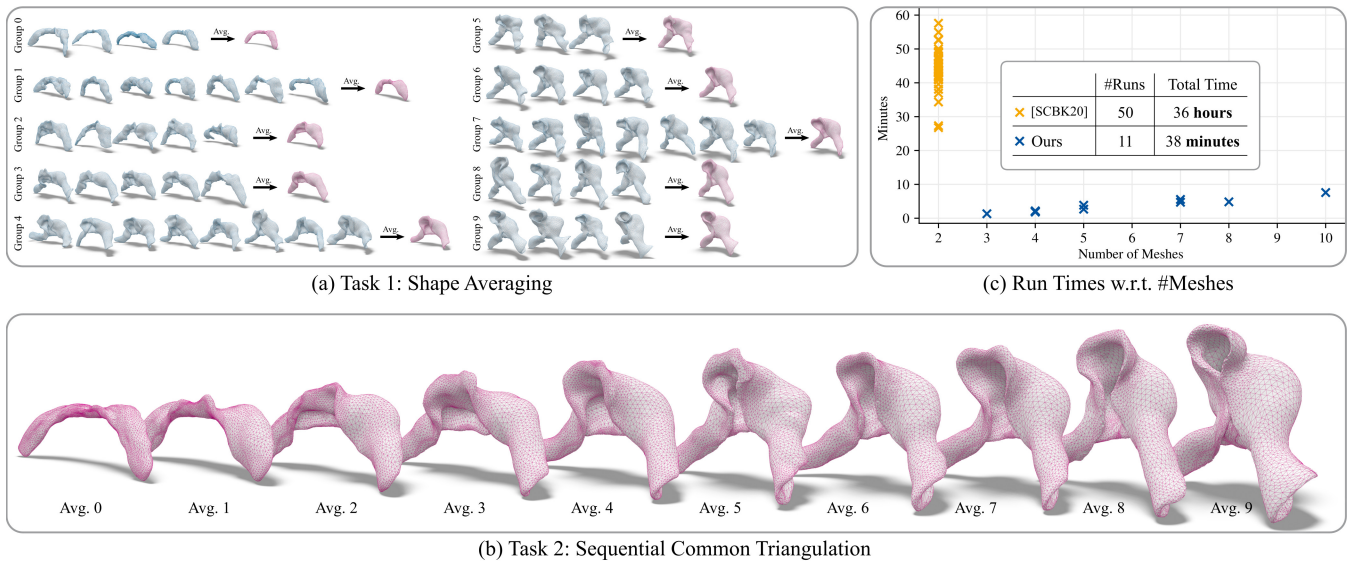
(a) Task 1: Shape Averaging

(c) Run Times w.r.t. #Meshes

(b) Task 2: Sequential Common Triangulation

Figure 14: *(a) Dataset of 51 embryonic heart shapes of the mouse, organized in 10 groups of similar developmental stage [ESD*22]. Per group, we run our algorithm once to compute an average shape. (b) We then sequentially map between the 10 average shapes to establish a common triangulation describing the entire growth process. (c) Executing our algorithm 11 times (blue crosses) took a total of 38 minutes, with run time approximately linear in the number of input meshes. The reference solution via [SCBK20] required computing 50 pairwise maps (yellow crosses), which took a total of 36 hours.*

each run of our algorithm is significantly faster than a pairwise run via [SCBK20], reducing the total computation time from 36 hours to 38 minutes. In Figure 14 we show the resulting compatible triangulation of the animation sequence as well as a run time plot revealing an approximately linear dependency on the number of input meshes per run.

9. Limitations & Future Work

Our method offers great flexibility to express different optimization goals in the objective function. However, this also means that a number of weights and parameters have to be chosen, in particular as we add a term for surface approximation to the mapping problem. While application-dependent adjustment of parameters or input constraints is, in one form or another, necessary using any of the known approaches, we argue that our improved turnaround times make such iterative surface mapping workflows more practical. Still, finding fully automatic algorithms with fewer approximations and fewer parameters remains an important goal.

Mapping via the sphere without cutting inevitably causes high distortion in the intermediate domain. Numerically, this poses a limit on handling long geometric features, which undergo extreme shrinking (a limitation shared with [SCBK20] and others). Our method sometimes struggles with untwisting a map around thin features (e.g. because the bijectivity barrier can easily dominate in such regions). Alternative map representations with cone singularities (e.g. [AL16]) offer a possible remedy, but have other disadvantages such as fixed correspondence at these points. Direct mapping methods such as [Tak22] open an interesting new direction.

Bijective surface-to-surface map optimization on piecewise linear triangle meshes, in its currently available representations, inevitably amounts to optimization problems that are only C^0 continuous but C^1 discontinuous. These do not meet the theoretical pre-conditions necessary to safely employ Newton-style methods. Some heuristic remedies are used in [SBCK19, SCBK20, Tak22], but a fully satisfying solution is yet to be found. While we are practically able to obtain high-quality maps, all of the above algorithms, as well as our method, can potentially get stuck at discontinuities.

An obvious next step is extending the method to higher genus cases, for example, via tilings of the (hyperbolic) plane as in [SCBK20]. Besides practical challenges (handling the interplay between embeddings of $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{T}$ with different cuts and transition functions), this problem also requires an explicit choice and representation of map homotopy. While [BSCK21] gives an answer to this question in terms of homology, a robust conversion to homotopy remains open.
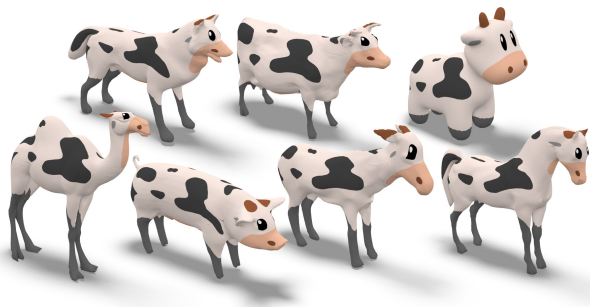
Acknowledgments

Figure 15: *Spot and friends. The map computed by our method can be used to transfer any kind of surface data, e.g., textures, from a source mesh (top right) to multiple target meshes.*

References

[AGK*22] AIGERMAN N., GUPTA K., KIM V. G., CHAUDHURI S., SAITO J., GROUEIX T.: Neural jacobian fields: Learning intrinsic mappings of arbitrary meshes. *ACM Transactions on Graphics 41*, 4 (2022). 4

[AKL17] AIGERMAN N., KOVALSKY S. Z., LIPMAN Y.: Spherical orbifold Tutte embeddings. *ACM Transactions on Graphics 36*, 4 (2017). 3, 4

[AL15] AIGERMAN N., LIPMAN Y.: Orbifold tutte embeddings. *ACM Transactions on Graphics 34*, 6 (2015). 3, 4

[AL16] AIGERMAN N., LIPMAN Y.: Hyperbolic orbifold Tutte embeddings. *ACM Transactions on Graphics 35*, 6 (2016). 3, 4, 10, 12

[Ale00] ALEXA M.: Merging polyhedral shapes with scattered features. *The Visual Computer 16*, 1 (2000). 3, 4

[APH05] ASIRVATHAM A., PRAUN E., HOPPE H.: Consistent spherical parameterization. In *International Conference on Computational Science* (2005). 3

[APL14] AIGERMAN N., PORANNE R., LIPMAN Y.: Lifted bijections for low distortion surface mappings. *ACM Transactions on Graphics 33*, 4 (2014). 3

[APL15] AIGERMAN N., PORANNE R., LIPMAN Y.: Seamless surface mappings. *ACM Transactions on Graphics 34*, 4 (2015). 3, 4

[BCK18] BADEN A., CRANE K., KAZHDAN M.: Möbius registration. *Computer Graphics Forum 37*, 5 (2018). 3, 4, 9

[BK04] BOTSCH M., KOBBELT L.: A remeshing approach to multiresolution modeling. In *Proceedings of the Eurographics Symposium on Geometry Processing* (2004). 3, 4, 8

[BN21] BROWN G. E., NARAIN R.: Wrapd: weighted rotation-aware admm for parameterization and deformation. *ACM Transactions on Graphics 40*, 4 (2021). 4

[BSCK21] BORN J., SCHMIDT P., CAMPEN M., KOBBELT L.: Surface map homology inference. *Computer Graphics Forum 40*, 5 (2021). 4, 12

[BSK21] BORN J., SCHMIDT P., KOBBELT L.: Layout embedding via combinatorial optimization. *Computer Graphics Forum 40*, 2 (2021). 4, 9, 11

[CBSS17] CLAICI S., BESSMELTSEV M., SCHAEFER S., SOLOMON J.: Isometry-aware preconditioning for mesh parameterization. *Computer Graphics Forum 36*, 5 (2017). 4

[CCS*21] CAMPEN M., CAPOUELLEZ R., SHEN H., ZHU L., PANOZZO D., ZORIN D.: Efficient and robust discrete conformal equivalence with boundary. *ACM Transactions on Graphics 40*, 6 (2021). 4

[CFZC19] CHENG X.-X., FU X.-M., ZHANG C., CHAI S.: Practical error-bounded remeshing by adaptive refinement. *Computers & Graphics 82* (2019). 4, 11

[CPS13] CRANE K., PINKALL U., SCHRÖDER P.: Robust fairing via conformal curvature flow. *ACM Transactions on Graphics 32*, 4 (2013). 9

[DVBB13] DUNYACH M., VANDERHAEGHE D., BARTHE L., BOTSCH M.: Adaptive remeshing for real-time mesh deformation. In *Proceedings of Eurographics* (2013). 3, 4, 7, 8, 11

[EHA*19] EZUZ D., HEEREN B., AZENCOT O., RUMPF M., BEN-CHEN M.: Elastic correspondence between triangle meshes. *Computer Graphics Forum 38*, 2 (2019). 4

[ESBC19] EZUZ D., SOLOMON J., BEN-CHEN M.: Reversible harmonic maps between discrete surfaces. *ACM Transactions on Graphics 38*, 2 (2019). 4

[ESD*22] ESTEBAN I., SCHMIDT P., DESGRANGE A., RAIOLA M., TEMIÑO S., MEILHAC S. M., KOBBELT L., TORRES M.: Pseudo-dynamic analysis of heart tube formation in the mouse reveals strong regional variability and early left-right asymmetry. *Nature Cardiovascular Research 1*, 5 (2022). 3, 11, 12

[FM11] FARB B., MARGALIT D.: *A Primer on Mapping Class Groups.* Princeton University Press, 2011. 4

[FSSB07] FISHER M., SPRINGBORN B., SCHRÖDER P., BOBENKO A. I.: An algorithm for the construction of intrinsic delaunay triangulations with applications to digital geometry processing. *Computing 81*, 2 (2007). 4

[GBKS18] GEHRE A., BRONSTEIN M., KOBBELT L., SOLOMON J.: Interactive curve constrained functional maps. *Computer Graphics Forum 37*, 5 (2018). 4

[GGS03] GOTSMAN C., GU X., SHEFFER A.: Fundamentals of spherical parameterization for 3d meshes. *ACM Transactions on Graphics 22*, 3 (2003). 4

[GSC21a] GILLESPIE M., SHARP N., CRANE K.: Integer coordinates for intrinsic geometry processing. *ACM Transactions on Graphics 40*, 6 (2021), 1–13. 4

[GSC21b] GILLESPIE M., SPRINGBORN B., CRANE K.: Discrete conformal equivalence of polyhedral surfaces. *ACM Transactions on Graphics 40*, 4 (2021). 4

[HAWG08] HUANG Q.-X., ADAMS B., WICKE M., GUIBAS L. J.: Non-rigid registration under isometric deformations. *Computer Graphics Forum 27*, 5 (2008). 4

[HFL17] HU X., FU X.-M., LIU L.: Advanced hierarchical spherical parameterizations. *IEEE Transactions on Visualization and Computer Graphics 24*, 6 (2017). 9

[Hop96] HOPPE H.: Progressive meshes. In *Proceedings of SIGGRAPH* (1996). 9

[HPG*22] HERTZ A., PEREL O., GIRYES R., SORKINE-HORNUNG O., COHEN-OR D.: Mesh draping: Parametrization-free neural mesh transfer. *Computer Graphics Forum* (2022). 4

[HRS*16] HEEREN B., RUMPF M., SCHRÖDER P., WARDETZKY M., WIRTH B.: Splines in the space of shells. *Computer Graphics Forum 35*, 5 (2016). 11

[HYB*16] HU K., YAN D.-M., BOMMES D., ALLIEZ P., BENES B.: Error-bounded and feature preserving surface remeshing with minimal angle improvement. *IEEE Transactions on Visualization and Computer Graphics 23*, 12 (2016). 4

[JSZP20] JIANG Z., SCHNEIDER T., ZORIN D., PANOZZO D.: Bijective projection in a shell. *ACM Transactions on Graphics 39*, 6 (2020). 4

[KGL16] KOVALSKY S. Z., GALUN M., LIPMAN Y.: Accelerated quadratic proxy for geometric optimization. *ACM Transactions on Graphics 35*, 4 (2016). 4

[KS04] KRAEVOY V., SHEFFER A.: Cross-parameterization and compatible remeshing of 3d models. *ACM Transactions on Graphics 23*, 3 (2004). 2, 3, 4

[KSBC12] KAZHDAN M., SOLOMON J., BEN-CHEN M.: Can mean-curvature flow be modified to be non-singular? *Computer Graphics Forum 31*, 5 (2012). 9

[KSK97] KANAI T., SUZUKI H., KIMURA F.: 3d geometric metamorphosis based on harmonic map. In *Proceedings of Pacific Graphics* (1997). 3, 4

[LBG*08] LI X., BAO Y., GUO X., JIN M., GU X., QIN H.: Globally optimal surface mapping for surfaces with arbitrary topology. *IEEE Transactions on Visualization and Computer Graphics 14*, 4 (2008). 3, 4

[LDRS05] LITKE N., DROSKE M., RUMPF M., SCHRÖDER P.: An image processing approach to surface matching. In *Symposium on Geometry Processing* (2005), vol. 255. 2, 3, 4

[LDSS99] LEE A. W. F., DOBKIN D., SWELDENS W., SCHRÖDER P.: Multiresolution mesh morphing. In *Proceedings of SIGGRAPH* (1999). 3, 4

[LGQ08] LI X., GU X., QIN H.: Surface matching using consistent pants decomposition. In *Proceedings of the ACM Symposium on Solid and Physical Modeling* (2008). 4

[Liv20] LIVESU M.: A mesh generation perspective on robust mappings. In *STAG: Smart Tools and Applications in Graphics* (2020). 4

[Liv21] LIVESU M.: Scalable mesh refinement for canonical polygonal schemas of extremely high genus shapes. *IEEE Transactions on Visualization and Computer Graphics 27*, 1 (2021). 3

[LSP08] LI H., SUMNER R. W., PAULY M.: Global correspondence optimization for non-rigid registration of depth scans. *Computer Graphics Forum 27*, 5 (2008). 4

[LYNF18] LIU L., YE C., NI R., FU X.-M.: Progressive parameterizations. *ACM Transactions on Graphics 37*, 4 (2018). 4

[MAKM21] MORREALE L., AIGERMAN N., KIM V. G., MITRA N. J.: Neural surface maps. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2021). 4

[MCSA15] MANDAD M., COHEN-STEINER D., ALLIEZ P.: Isotopic approximation within a tolerance volume. *ACM Transactions on Graphics 34*, 4 (2015). 4

[MCSK*17] MANDAD M., COHEN-STEINER D., KOBBELT L., ALLIEZ P., DESBRUN M.: Variance-minimizing transport plans for inter-surface mapping. *ACM Transactions on Graphics 36*, 4 (2017). 4

[MKFC01] MICHIKAWA T., KANAI T., FUJITA M., CHIYOKURA H.: Multiresolution interpolation meshes. In *Proceedings of Pacific Graphics 2001* (2001). 4

[MRR*19] MELZI S., REN J., RODOLA E., SHARMA A., WONKA P., OVSJANIKOV M.: Zoomout: Spectral upsampling for efficient shape correspondence. *ACM Transactions on Graphics 38*, 6 (2019). 4

[NZZ20] NAITSAT A., ZHU Y., ZEEVI Y. Y.: Adaptive block coordinate descent for distortion optimization. *Computer Graphics Forum 39*, 6 (2020). 4

[OBCS*12] OVSJANIKOV M., BEN-CHEN M., SOLOMON J., BUTSCHER A., GUIBAS L.: Functional maps: a flexible representation of maps between shapes. *ACM Transactions on Graphics 31*, 4 (2012). 4

[PH03] PRAUN E., HOPPE H.: Spherical parametrization and remeshing. *ACM Transactions on Graphics 22*, 3 (2003). 9

[PSS01] PRAUN E., SWELDENS W., SCHRÖDER P.: Consistent mesh parameterizations. In *Proceedings of SIGGRAPH* (2001). 3

[PT16] PENG C., TIMALSENA S.: Fast mapping and morphing for genus-zero meshes with cross spherical parameterization. *Computers & Graphics 59* (2016). 3, 4

[RMC15] RODOLÀ E., MOELLER M., CREMERS D.: Point-wise Map Recovery and Refinement from Functional Correspondence. In *Vision, Modeling & Visualization* (2015). 4

[RPPSH17] RABINOVICH M., PORANNE R., PANOZZO D., SORKINE-HORNUNG O.: Scalable locally injective mappings. *ACM Transactions on Graphics 36*, 4 (2017). 4

[SAPH04] SCHREINER J., ASIRVATHAM A., PRAUN E., HOPPE H.: Inter-surface mapping. *ACM Transactions on Graphics 23*, 3 (2004). 2, 3, 4, 6

[SBB*22] SCHMIDT P., BORN J., BOMMES D., CAMPEN M., KOBBELT L.: TinyAD: Automatic differentiation in geometry processing made simple. *Computer Graphics Forum 41*, 5 (2022). 8

[SBCK19] SCHMIDT P., BORN J., CAMPEN M., KOBBELT L.: Distortion-minimizing injective maps between surfaces. *ACM Transactions on Graphics 38*, 6 (2019). 2, 3, 4, 6, 12

[SBSCO06] SHARF A., BLUMENKRANTS M., SHAMIR A., COHEN-OR D.: Snappaste: An interactive technique for easy mesh composition. *The Visual Computer 22*, 9 (2006). 4

[SCBK20] SCHMIDT P., CAMPEN M., BORN J., KOBBELT L.: Inter-surface maps via constant-curvature metrics. *ACM Transactions on Graphics 39*, 4 (2020). 2, 3, 4, 5, 6, 9, 10, 11, 12

[SJZP19] SHEN H., JIANG Z., ZORIN D., PANOZZO D.: Progressive embedding. *ACM Transactions on Graphics 38*, 4 (2019). 9

[SKPSH13] SCHÜLLER C., KAVAN L., PANOZZO D., SORKINE-HORNUNG O.: Locally injective mappings. *Computer Graphics Forum 32*, 5 (2013). 11

[SNB*12] SOLOMON J., NGUYEN A., BUTSCHER A., BEN-CHEN M., GUIBAS L.: Soft maps between surfaces. *Computer Graphics Forum 31*, 5 (2012). 4

[SPSH*17] SHTENGEL A., PORANNE R., SORKINE-HORNUNG O., KOVALSKY S. Z., LIPMAN Y.: Geometric optimization via composite majorization. *ACM Transactions on Graphics 36*, 4 (2017). 4

[SS15] SMITH J., SCHAEFER S.: Bijective parameterization with free boundaries. *ACM Transactions on Graphics 34*, 4 (2015). 6

[SSC19] SHARP N., SOLIMAN Y., CRANE K.: Navigating intrinsic triangulations. *ACM Transactions on Graphics 38*, 4 (2019). 4

[SYLF20] SU J.-P., YE C., LIU L., FU X.-M.: Efficient bijective parameterizations. *ACM Transactions on Graphics 39*, 4 (2020). 4

[SZS*16] SHI R., ZENG W., SU Z., JIANG J., DAMASIO H., LU Z., WANG Y., YAU S.-T., GU X.: Hyperbolic harmonic mapping for surface registration. *IEEE transactions on pattern analysis and machine intelligence 39*, 5 (2016). 3

[Tak22] TAKAYAMA K.: Compatible intrinsic triangulations. *ACM Transactions on Graphics 41*, 4 (2022). 2, 4, 9, 10, 12

[TCL*13] TAM G. K., CHENG Z.-Q., LAI Y.-K., LANGBEIN F., LIU Y., MARSHALL A. D., MARTIN R., SUN X., ROSIN P.: Registration of 3d point clouds and meshes: A survey from rigid to nonrigid. *IEEE Transactions on Visualization and Computer Graphics 19*, 7 (2013). 4

[TDIN*11] TIERNY J., DANIELS II J., NONATO L. G., PASCUCCI V., SILVA C. T.: Inspired quadrangulation. *Computer-Aided Design 43*, 11 (2011). 3

[TFV*13] TSUI A., FENTON D., VUONG P., HASS J., KOEHL P., AMENTA N., COEURJOLLY D., DECARLI C., CARMICHAEL O.: Globally optimal cortical surface matching with exact landmark correspondence. In *International Conference on Information Processing in Medical Imaging* (2013). 3, 4

[TSIF05] TERAN J., SIFAKIS E., IRVING G., FEDKIW R.: Robust quasistatic finite elements and flesh simulation. In *Proceedings of the ACM SIGGRAPH/Eurographics symposium on Computer animation* (2005). 8

[WPYM07] WU H.-Y., PAN C., YANG Q., MA S.: Consistent correspondence between arbitrary manifold surfaces. In *IEEE International Conference on Computer Vision* (2007). 4

[WZ14] WEBER O., ZORIN D.: Locally injective parametrization with arbitrary fixed boundaries. *ACM Transactions on Graphics 33*, 4 (2014). 3

[YFC*18] YANG Y., FU X.-M., CHAI S., XIAO S.-W., LIU L.: Volume-enhanced compatible remeshing of 3d models. *IEEE Transactions on Visualization and Computer Graphics 25*, 10 (2018). 2, 4

[YLH18] YI R., LIU Y.-J., HE Y.: Delaunay mesh simplification with differential evolution. *ACM Transactions on Graphics 37*, 6 (2018). 4

[YW15] YAN D.-M., WONKA P.: Non-obtuse remeshing with centroidal voronoi tessellation. *IEEE Transactions on Visualization and Computer Graphics 22*, 9 (2015). 4

[YZL*20] YANG Y., ZHANG W.-X., LIU Y., LIU L., FU X.-M.: Error-bounded compatible remeshing. *ACM Transactions on Graphics 39*, 4 (2020). 2, 3, 4, 7, 8, 11

[ZBK18] ZHU Y., BRIDSON R., KAUFMAN D. M.: Blended cured quasi-newton for distortion optimization. *ACM Transactions on Graphics 37*, 4 (2018). 4

[ZLJW06] ZHANG L., LIU L., JI Z., WANG G.: Manifold parameterization. In *Advances in Computer Graphics*. Springer, 2006. 4

[ZPBK17] ZHU Y., POPOVIĆ J., BRIDSON R., KAUFMAN D. M.: Planar interpolation with extreme deformation, topology change and dynamics. *ACM Transactions on Graphics 36*, 6 (2017). 4

[ZWG*22] ZHANG W.-X., WANG Q., GUO J.-P., CHAI S., LIU L., FU X.-M.: Constrained remeshing using evolutionary vertex optimization. *Computer Graphics Forum 41*, 2 (2022). 4