Computing Inversion-Free Mappings by Simplex Assembly

Xiao-Ming Fu*† Yang Liu†
*University of Science and Technology of China †Microsoft Research Asia

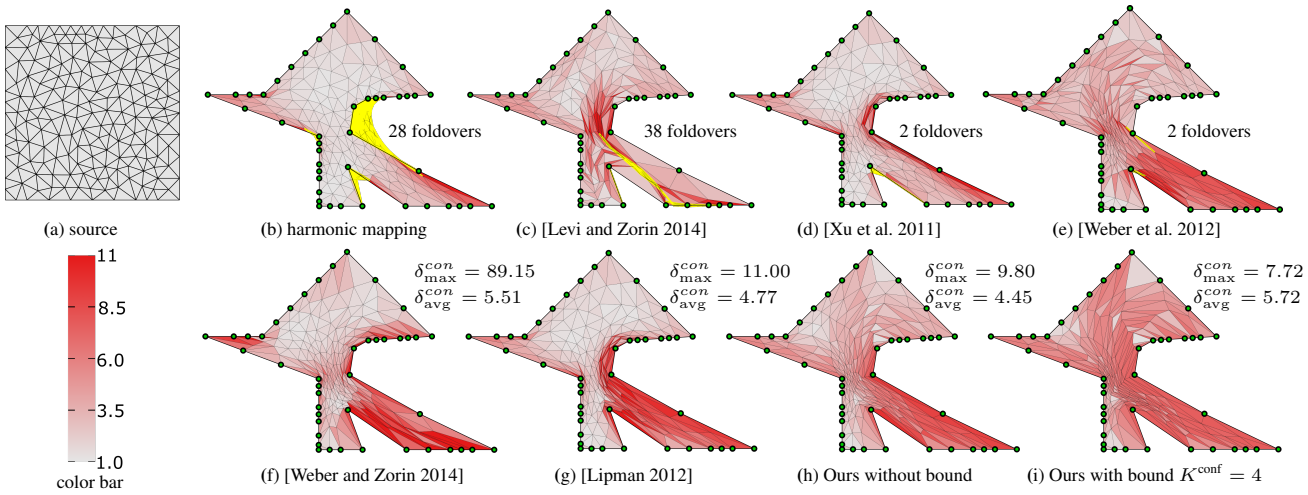


Figure 1: *2D conformal fixed-boundary mapping. A square-shaped triangle mesh (a) is mapped onto a non-convex polygonal domain. A harmonic mapping (b) contains 28 foldovers (colored in yellow). With this poor mapping as initialization, our method can easily achieve inversion-free results with low distortion ($K^{conf} = \infty$) (h) and bounded distortion with bound $K^{conf} = 4$ (i). $E_m = 0$ in this example. We compare our method with other state-of-the-art methods. The methods of [Levi and Zorin 2014] (c), [Xu et al. 2011] (d) and [Weber et al. 2012] (e) cannot prevent foldovers. The method of [Weber and Zorin 2014] needs to insert 9 additional vertices into the triangulation to guarantee inversion-free results (f). Bounded distortion methods like [Lipman 2012] can find a valid mapping (g), but the bound is hard to determine. It fails when $\delta^{con}_{bound} < 11$ using the authors' implementation with the defaulting setting. The color on triangles encodes the conformal distortion, with white being optimal.*

Abstract

We present a novel method, called *Simplex Assembly*, to compute inversion-free mappings with low or bounded distortion on simplicial meshes. Our method involves two steps: simplex disassembly and simplex assembly. Given a simplicial mesh and its initial piecewise affine mapping, we project the affine transformation associated with each simplex into the inversion-free and distortion-bounded space. The projection disassembles the input mesh into disjoint simplices. The disjoint simplices are then assembled to recover the original connectivity by minimizing the mapping distortion and the difference of the disjoint vertices with respect to the piecewise affine transformations, while the piecewise affine mapping is restricted inside the feasible space. Due to the use of affine transformations as variables, our method explicitly guarantees that no inverted simplex occurs, and that the mapping distortion is below the bound during the optimization. Compared with existing methods, our method is robust to an initialization with many inverted elements and positional constraints. We demonstrate the efficiency and robustness of our method through a variety of geometric processing tasks.

Keywords: inversion-free, bounded distortion, simplex assembly

Concepts: •**Computing methodologies** → **Mesh models; Mesh geometry models; Volumetric models;**

1 Introduction

Computing inversion-free mappings with low or bounded distortion on simplicial meshes is essential to many computer graphics and geometric processing tasks, such as mesh parameterization and mesh deformation. The term *inversion-free* means that the determinant of the Jacobian of the mapping f is always positive: $\det \mathbf{J}(f) > 0$. Its geometric interpretation is simple: the mapping does not introduce any *inverted simplices* (that are also called *foldovers*, *flips*). The low distortion usually means that the simplex after the mapping reserves similarity or isometry to its original shape as much as possible.

The computation of inversion-free mappings is usually formulated as an optimization problem where the objective characterizes the distortion of the mapping, and the inversion-free requirement is posed as soft or hard constraints. Numerous techniques have been proposed to solve this problem. Methods like [Hormann and Greiner 2000; Degener et al. 2003; Schüller et al. 2013; Fu et al. 2015] start with an inversion-free initial mapping and minimize an objective function that measures the mapping distortion and goes to infinity if $\det \mathbf{J}(f)$ approaches zero. These methods usually perform well both in efficiency and quality if there are only a few positional constraints, like handles used in mesh deformation. However, they converge slowly or even fail when the initialization is far away from the optimal, or there are too many positional constraints (Figure 15(b) shows a failure case). To resolve these issues, Jin *et al.* [2014] remesh the input adaptively to expand the solution space. However, their method

SA '16 Technical Papers, December 05-08, 2016, Macao
ISBN: 978-1-4503-4514-9/16/12
DOI: http://dx.doi.org/10.1145/2980179.2980231

ACM Reference Format
Fu, X., Liu, Y. 2016. Computing Inversion-Free Mappings by Simplex Assembly. ACM Trans. Graph. 35, 6, Article 216 (November 2016), 12 pages. DOI = 10.1145/2980179.2980231
http://doi.acm.org/10.1145/2980179.2980231.

may fail when the positions of handles are very different from the initial positions. Weber *et al*. [2014] insert new vertices into the triangulation to find a solution for 2D fixed-boundary mappings. Both Jin *et al*.'s and Weber *et al*.'s methods modify the input mesh structure and are not trivial to extend to tetrahedral meshes. Lipman and his colleagues propose novel ways to bound the mapping distortion explicitly. Their methods [2012; 2013; 2014] do not require an inversion-free initialization and work for high-dimensional mappings. However, the computational efficiency is relatively low due to the heavy use of quadratic programming and/or semidefinite programming. Kovalsky *et al.* [2015] speed up the bounded distortion mapping method by linearizing the constraint space and utilizing matrix pre-factorization for fast computation. However, how to choose a low distortion bound while ensuring there is a feasible solution is still being studied. Given a tight bound, their method may fail to find a feasible mapping (Figure 14 shows a failure case).

We propose a novel method called *Simplex Assembly* to compute inversion-free mappings with low or bounded distortion. Our essential idea includes two steps: (1) we first disassemble the simplicial mesh into disjoint simplices that are inversion-free and stay inside the feasible distortion space; (2) we treat piecewise affine transformations as variables and assemble the disjoint simplices by solving an unconstrained optimization problem whose objective penalizes disjoint simplices and restricts the affine mappings inside the feasible distortion space by a simple and novel barrier function that avoid inversion and control distortion. Our method is capable of achieving bounded distortion mappings and locally injective mappings, and is robust to various initial mappings with many inverted simplices and positional constraints. We demonstrate the efficiency and robustness of our method in various applications including mesh parameterization, 2D/3D fixed-boundary mappings, and mesh deformation.

2 Related Work

Mesh-based mappings Numerous techniques for mesh-based mappings have been developed (cf. the surveys in [Floater and Hormann 2005; Botsch et al. 2010]). For computing the mapping efficiently, many methods are linear-based techniques, which cannot guarantee inversion-free results. For instance, the LSCM method [Lévy et al. 2002] for mesh parameterization may generate foldovers if the geometry of the mesh is complex and the cut of the mesh is too short. The As-Rigid-As-Possible (ARAP) methods [Liu et al. 2008; Chao et al. 2010] also do not prevent inversion. The strict minimizer [Levi and Zorin 2014] provides a minimal L_∞ solution to control the distortion but has no guarantee on inversion-free results. Our method is robust to initializations with many inverted elements, and we can utilize existing methods to obtain an initialization and generate a result close to it but with low distortion and no foldovers.

Inversion-free mappings Many inversion-free algorithms require an inversion-free initialization so that they can keep this property during the optimization. For mesh parameterization, the initial map is usually obtained from the Tutte embedding [Tutte 1963; Floater 2003]; for mesh deformation, the meshes in the rest-pose serve as initializations. The MIPS energy [Hormann and Greiner 2000] and its variants [Degener et al. 2003; Fu et al. 2015; Smith and Schaefer 2015] or other energies with barrier terms [Schüller et al. 2013; Jin et al. 2014; Rabinovich et al. 2016] are used to prevent degenerate simplices and penalize high distortion. By observing that the tessellations of meshes affect the mapping quality and the convergence speed, Jin *et al*. [2014] propose an adaptive remeshing scheme to expand the solution space for 2D deformation, and Weber *et al*. [2014] allow insertion of vertices into both the source and target tessellations to find a valid 2D fixed-boundary mapping. To further improve the mapping computation when positional constraints are imposed, Schuller *et al*. [2013] propose the *substepping* strategy that replaces the target handle positions by intermediate positions. The substepping strategy creates moving paths of handles to assist in the optimization. However, when there are too many positional constraints such as fixed boundary mappings, it rarely works because of the conflict of handle paths as observed by [Jin et al. 2014; Fu et al. 2015]. Recently, Liu *et al*. [2016] used a quadratic ex-rotate elastic energy with some convex terms to penalize the presence of inverted elements. Their method works well for mesh skinning but it requires an approximate ex-rotated field as input.

Bounded distortion mappings The authors of [Lipman 2012; Aigerman and Lipman 2013; Kovalsky et al. 2014; Poranne and Lipman 2014; Chen and Weber 2015] seek a mapping whose conformal or isometric distortion is under a given bound. They approximate the constraint space by a maximal convex subspace and solve it by constrained nonlinear optimizations. The results are guaranteed to be inversion-free with bounded distortion if their methods converge. However, the speed and the scalability of their methods limit their practical use. Recently Kovalsky *et al*. [2015] present a first-order approximation approach to solving these issues. Their method is very promising, but an inappropriate bound can result in an empty solution space. Our method is also capable of achieving bounded distortion mappings and can be used to seek the optimal bound.

Transformation representations Manipulating mesh vertex positions directly is a common approach in mapping computation. However, it is known that a more appropriate representation can significantly improve the efficiency and robustness of the algorithm for dedicated tasks. Angle based mesh representations are effective for 2D conformal parameterization [Sheffer and de Sturler 2001; Sheffer et al. 2005] and their 3D version – dihedral angle based representations – are proved to be very useful in tetrahedral mesh mappings [Paillé et al. 2015]. Metric scaling, i.e., scalings of mesh edge lengths, is another good representation for conformal embedding [Springborn et al. 2008; Ben-Chen et al. 2008]. Crane *et al*. [2011; 2013] manipulate the curvature space to control spin transformation and Willmore flow. In this paper we find that piecewise affine transformations on simplices are nice representations to control distortion and avoid inversion.

3 Problem definition

We study inversion-free mappings on simplicial meshes. The source of the mapping is a d-dimensional simplicial mesh $\mathbf{M}$ containing N_v vertices $\{\mathbf{v}_i, i = 1, \ldots, N_v\}$, N_e edges $\{\mathbf{e}_i, i = 1, \ldots, N_e\}$ and N simplices $\{\mathbf{s}_i, i = 1, \ldots, N\}$. For fixed-boundary mappings or mesh deformation, some vertices are handles that provide positional constraints. We denote these handles and their target positions as $\{\mathbf{v}_h, h = 1, \ldots, N_h\}$ and $\{\mathbf{v}_h^\star, h = 1, \ldots, N_h\}$, respectively. We use symbols $\widehat{\mathbf{M}}, \widehat{\mathbf{v}}, \widehat{\mathbf{e}}, \widehat{\mathbf{s}}$ to denote the image of the mesh, vertex, edge and simplex under the mapping. The affine transformation $\mathcal{A}_i$ defined on simplex $\mathbf{s}_i$ is denoted by $\{\mathbf{A}_i, \mathbf{T}_i\}$, where $\mathbf{A}_i$ is the linear transformation and $\mathbf{T}_i$ is the translation vector. $\mathcal{A}_i$ maps any vector $\mathbf{p} \in \mathbb{R}^d$ to $\mathbf{A}_i\mathbf{p} + \mathbf{T}_i$. A mapping on $\mathbf{M}$ is inversion-free if and only if $\det \mathbf{A}_i > 0,\ i = 1, \ldots, N$.

Mapping distortions The quality of a mapping is usually measured by the mapping distortion. The types and definitions of distortions vary across applications. Conformal distortion, areal/volumetric distortion and isometric distortion are commonly used in mesh parameterization and mesh deformation. We use the MIPS energy and its extensions [Hormann and Greiner 2000; Fu et al. 2015] to define them. Denote $\sigma_{i,1}, \ldots, \sigma_{i,d}$ as the singular values of $\mathbf{A}_i$, where $\det \mathbf{A}_i > 0$. The **conformal distortion** on $\mathcal{A}_i$

is defined as:

$$d_i^c := \frac{1}{2^d}\prod_{k<l}\left(\sigma_{i,k}\,\sigma_{i,l}^{-1}+\sigma_{i,k}^{-1}\,\sigma_{i,l}\right) = \begin{cases}\frac{1}{2}\|\mathbf{A}_i\|_F\|\mathbf{A}_i^{-1}\|_F = \frac{\operatorname{trace}\mathbf{A}_i^T\mathbf{A}_i}{2\det\mathbf{A}_i}, & d=2\\ \frac{1}{8}(\|\mathbf{A}_i\|_F^2\|\mathbf{A}_i^{-1}\|_F^2-1). & d=3.\end{cases}$$

Here $\|\cdot\|_F$ denotes the Frobenius matrix norm. The **volumetric distortion** on $\mathcal{A}_i$ is defined as:

$$d_i^{vol} := \frac{1}{2}\left(\det\mathbf{A}_i + (\det\mathbf{A}_i)^{-1}\right).$$

Since a mapping is isometric if and only if it is both conformal and equiareal, we combine the above distortions linearly to define the **isometric distortion** on $\mathcal{A}_i$:

$$d_i^{iso} := \gamma\, d_i^c + (1-\gamma)\, d_i^{vol}.$$

We choose $\gamma = 0.5$ in our paper.

Other types of distortions include the Dirichlet energy: $d_i^{Dirichlet} := \sum_{k=1}^d \sigma_{i,k}^2$, the Green-Lagrange energy: $d_i^{GL} := \sum_{k=1}^d(\sigma_{i,k}^2-1)^2$, the As-Similar-As-Possible energy: $d_i^{ASAP} := \sum_{j\neq k}(\sigma_{i,j}-\sigma_{i,k})^2$, the As-Rigid-As-Possible energy: $d_i^{ARAP} := \sum_{k=1}^d(\sigma_{i,k}-1)^2$, and the difference from a given mapping $\mathcal{A}_i^0$: $d_i^{diff} := \|\mathbf{A}_i - \mathbf{A}_i^0\|_F^2$, or other user-desired energies. The summation of one type of distortion over all the piecewise affine mappings forms the mapping energy E_m:

$$E_m := \sum_{i=1}^N w_i\, d_i^{\mathtt{type}}.$$

Here $d^{\mathtt{type}}$ corresponds to one type of distortion, w_i is a weighting factor usually set as the volume of $\mathbf{s}_i$.

Bounded distortion Extremely distorted simplices are not desired in many applications, such as mesh parameterization. Restricting the mapping distortion inside a given interval can provide a quality-guaranteed mapping. For conformal distortion and volumetric distortion, we impose the constraints: $1 \le d_i^c \le K^{\text{conf}}$ and $1 \le d_i^{vol} \le K^{\text{vol}}$, respectively. K^{conf} and K^{vol} are the upper bounds set by the user. When $\det\mathbf{A}_i > 0$, the left sides of these inequalities are always satisfied. The isometric distortion d_i^{iso} is bounded when both d_i^c and d_i^{vol} are bounded. Other types of distortions can be constrained too, such as d_i^{ARAP} in [Levi and Zorin 2014], but our main interest is in bounding the conformal and isometric distortion.

Constrained problem w.r.t. vertex positions Finding an inversion-free and distortion-bounded mapping can be solved as a nonlinear constrained optimization problem:

$$\begin{aligned}&\underset{\hat{\mathbf{v}}_1,\ldots,\hat{\mathbf{v}}_{N_v}}{\textbf{minimize}} && E_m\\ &\text{subject to} && \det\mathbf{A}_i > 0, \qquad i=1,\ldots,N,\\ &&& d_i^{\mathtt{type}} \le K^{\mathtt{type}}, \quad i=1,\ldots,N.\end{aligned} \tag{1}$$

Here $K^{\mathtt{type}}$ denotes the upper bound of the distortion $d^{\mathtt{type}}$, where the distortion type of $d^{\mathtt{type}}$ can be conformal or isometric. The interior-point method or the augmented Lagrangian method can be employed to solve the above problem. However, these methods are usually slow as observed by the authors of [Schüller et al. 2013]. Schuller *et al.* show that their barrier method can be 1000 times faster than the state-of-the-art interior-point implementation of KNITRO and Ipopt. However, the initialization of [Schüller et al. 2013] needs to be inversion-free. For mesh parameterization, the initialization from a convex combinatorial mapping may slow down the convergence when it is far away from the optimal. In the next section, we show that by treating affine transformations as variables, an inversion-free initialization is not required.

4 Computing Inversion-Free Mappings

Instead of optimizing the vertex positions, we treat piecewise affine transformations as unknowns and formulate the mapping computation as an unconstrained optimization problem (Section 4.1). By projecting illegal affine transformations of the initial mapping into the feasible space (Section 4.2), we can keep affine transformations inside the feasible space during the optimization. We present the details of our minimization method in Sections 4.3 and 4.4. We show that finding an inversion-free mapping is equivalent to computing integrable frame fields (Section 4.5).

4.1 Objective formulation

4.1.1 Assembly constraints

For a mapping $\Phi = \{\mathcal{A}_1,\ldots,\mathcal{A}_N\}$ on a simplicial mesh $\mathbf{M}$, its piecewise affine transformations should satisfy the following constraints.

Edge-assembly constraints. For any two neighboring simplices $\mathbf{s}_i$ and $\mathbf{s}_j$ who share an edge $\mathbf{e}_k := \overline{\mathbf{v}_a\mathbf{v}_b}$, the mapping images of any end points of $\mathbf{e}_k$ by $\{\mathbf{A}_i,\mathbf{T}_i\}$ and $\{\mathbf{A}_j,\mathbf{T}_j\}$ should be the same. We impose the following constraints:

$$\begin{aligned}(\mathbf{A}_i\mathbf{v}_a+\mathbf{T}_i)-(\mathbf{A}_j\mathbf{v}_a+\mathbf{T}_j)=0;\\ (\mathbf{A}_i\mathbf{v}_b+\mathbf{T}_i)-(\mathbf{A}_j\mathbf{v}_b+\mathbf{T}_j)=0.\end{aligned} \tag{2}$$

Handle-assembly constraints. For any simplex $\mathbf{s}_i$ which contains handle $\mathbf{v}_h$, the mapping $\{\mathbf{A}_i,\mathbf{T}_i\}$ should map $\mathbf{v}_h$ to $\mathbf{v}_h^\star$. We have the following constraint:

$$(\mathbf{A}_i\mathbf{v}_h+\mathbf{T}_i)-\mathbf{v}_h^\star = 0. \tag{3}$$

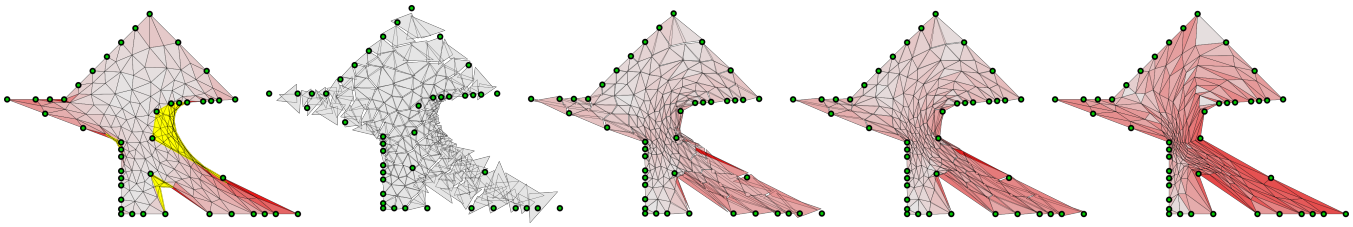
4.1.2 Objective function

By treating the affine transformations as variables, we formulate the following constrained nonlinear minimization problem to find a low or bounded distortion mapping:

$$\begin{aligned}&\underset{\substack{\mathbf{A}_1,\ldots,\mathbf{A}_N\\ \mathbf{T}_1,\ldots,\mathbf{T}_N}}{\textbf{minimize}} && \mu\, E_m + E_{\mathrm{C}}.\\ &\text{subject to:} && (2),\ (3);\end{aligned} \tag{4}$$

Here $\mu > 0$ is the coefficient of E_m and its default value is 1000 in our work. E_{C} is a novel barrier function which is designed to prevent inverted and bound-violated transformations and is used to replace the inequalities of Eq. (1). Its design is as follows.

Design of E_{C} Note that the expressions d_i^c, d_i^{vol} and d_i^{iso} naturally penalize degenerate simplices because they go to infinity if $\det\mathbf{A}_i$ approaches zero. We use these expressions as components to design E_{C} to bound distortions and prevent degenerate transformations. We also follow [Fu et al. 2015]'s approach that combines the exponent function to penalize the maximal distortion as much as possible. E_{C} is defined as follows:



(a) (b) (c) (d) (e)

Figure 2: *Mesh disassembly and optimization process. The input mesh and the initial map are the same as in Figure 1. (a) the initial map; (b) the disassembled mesh from (a); (c,d): the intermediate optimization results after 4 and 6 iterations; (e) the final result after 34 iterations.*

1. If the type of distortion $d^{\texttt{type}}$ and its bound $K^{\texttt{type}} > 1$ are given,

$$E_{\mathrm{C}} := \begin{cases} +\infty, & \exists i,\ \det A_i \le 0 \text{ or } K^{\texttt{type}} < d_i^{\texttt{type}}; \\ \sum_{i=1}^{N} \frac{e^{s\times d_i^{\texttt{type}}}}{K^{\texttt{type}} - d_i^{\texttt{type}}} & \text{otherwise.} \end{cases}$$

2. If the type of distortion $d^{\texttt{type}}$ is given and $K^{\texttt{type}} = +\infty$,

$$E_{\mathrm{C}} := \begin{cases} +\infty, & \exists i,\ \det A_i \le 0; \\ \sum_{i=1}^{N} e^{s\times d_i^{\texttt{type}}} & \text{otherwise.} \end{cases}$$

Here the type of distortion may be conformal, volumetric or isometric distortion. s controls the level of penalty on the maximal distortion and we choose $s = 1$ in all our experiments.

If the user prefers a lower E_m, one can choose a large μ or tune the barrier to let E_{C} go to zero when the distortion is much lower than its bound, for instance, by using the inverse of a cubic function proposed in [Schüller et al. 2013].

The minimization of Eq. (4) can be converted to an unconstrained problem by the quadratic penalty approach [Nocedal and Wright 2006]:

$$\underset{\substack{\mathbf{A}_1,\ldots,\mathbf{A}_N \\ \mathbf{T}_1,\ldots,\mathbf{T}_N}}{\textbf{minimize}} \quad \mu\, E_m + E_{\mathrm{C}} + \lambda\, E_{\text{assembly}}. \tag{5}$$

Here E_{assembly} is the summation of squares of all the left sides of Equations (2) and (3). The above penalty approach requires that $\det(\mathbf{A}_i) > 0$, $d_i^{\texttt{type}} < K^{\texttt{type}}$, $\forall i$ is satisfied at the beginning, so that the barrier function E_{C} can keep them inside the feasible space. We discuss the initialization in Section 4.2. Coefficient $\lambda > 0$ is adjusted dynamically during the minimization to enforce all the simplices to be assembled together at the final stage. The adjustment strategy of λ is presented in Section 4.4.

Computing $\widehat{\mathbf{M}}$ When an optimal solution of Eq. (4) is found, we have $E_{\text{assembly}} \approx 0$. The vertex positions of $\widehat{\mathbf{M}}$ can be recovered by solving the following least squares problem:

$$\underset{\widehat{\mathbf{v}}_1,\ldots,\widehat{\mathbf{v}}_n}{\textbf{minimize}} \sum_{k=1}^{N_e} \left(\|\mathbf{A}_i(\mathbf{v}_a - \mathbf{v}_b) - (\widehat{\mathbf{v}}_a - \widehat{\mathbf{v}}_b)\|_2^2 + \|\mathbf{A}_j(\mathbf{v}_a - \mathbf{v}_b) - (\widehat{\mathbf{v}}_a - \widehat{\mathbf{v}}_b)\|_2^2 \right). \tag{6}$$

Here edge $\overline{\mathbf{v}_a\mathbf{v}_b}$ is adjacent to simplices $\mathbf{s}_i$ and $\mathbf{s}_j$, and the known handle positions are fixed.

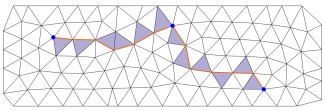
4.2 Initialization and mesh disassembly

An initial mapping is required in the minimization of Eq. (5) due to the nature of nonlinear optimization. An initial mapping can be provided by the user or obtained by minimizing E_m directly. We assume that the initial mapping $\mathcal{M}^0$ is available and the corresponding mesh is denoted as $\mathbf{M}^0 := \mathcal{M}^0(\mathbf{M})$. The initial $\mathbf{A}_i^0$s and $\mathbf{T}_i^0$s can be computed from the corresponding simplices $\triangle\mathbf{v}_{i,0}\cdots\mathbf{v}_{i,d}$ and $\triangle\mathbf{v}_{i,0}^0\cdots\mathbf{v}_{i,d}^0$ by solving $\mathbf{A}_i^0\mathbf{v}_{i,k} + \mathbf{T}_i^0 = \mathbf{v}_{i,k}^0, k = 0,\ldots,d$. To minimize the objective (5), the initial $\mathbf{A}_i^0$s should be inside the feasible space Ω defined by inversion-free and bounded distortion constraints so that E_{C} can serve as barriers. In our paper, we assign $\mathbf{A}_i^0$ to its closest similar matrix for conformal mappings and the closest rotation matrix for isometric mappings [Liu et al. 2008]. An alternative way is to employ the bounded distortion projection [Kovalsky et al. 2015] to project $\mathbf{A}_i^0$ into the feasible space. By applying the modified transformation to the source $\mathbf{M}$, $\mathbf{M}$ is actually disassembled into disjoint simplices. We call this procedure *mesh disassembly*. Figure 2(b) shows a disassembled result.

4.3 Variable reduction and E_{assembly} modification

The number of variables in Eq. (5) can be reduced. Geometrically, if the disjoint simplices can be assembled, we only need to fix one simplex's position and the translations of the other simplices are automatically determined. So for achieving the assemblable simplices, we do not need to know all the $\mathbf{T}_i$s. We propose the following method to remove their redundancy.

Denote $\mathbf{G}$ as the unweighted and undirected vertex graph of the mesh $\mathbf{M}$. We compute a minimal tree $\mathcal{T}$ of $\mathbf{G}$ that passes through all the handles $\{\mathbf{v}_h\}_{h=1}^{N_h}$, i.e., a sub-graph that contains all the handles with the least number of edges. For each edge $\mathbf{e} \in \mathcal{T}$ we tag one of its adjacent simplices. We denote such a tagged simplex set as Ω_s. The right inset illustrates $\mathcal{T}$ (orange edges) and Ω_s (colored triangles) on a simple 2D mesh with three handles (blue vertices). We redefine the assembly constraints based on this structure and only introduce translation variables $\mathbf{T}$ defined on the simplices in Ω_s.



Edge assembly constraints. For any two neighboring simplices $\mathbf{s}_i$ and $\mathbf{s}_j$ who share edge $\mathbf{e}_k = \overline{\mathbf{v}_a\mathbf{v}_b}$, we impose the constraint:

$$\mathbf{A}_i(\mathbf{v}_a - \mathbf{v}_b) - \mathbf{A}_j(\mathbf{v}_a - \mathbf{v}_b) = 0. \tag{7}$$

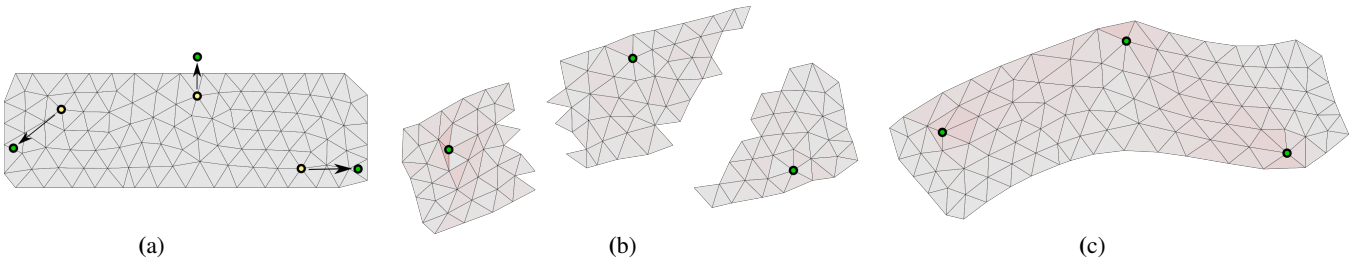
When the above constraint satisfies, the disjoint edges on $\mathbf{s}_i$ and $\mathbf{s}_j$ are parallel and have the same length. We also normalize $\mathbf{v}_a - \mathbf{v}_b$ to make the equation insensitive to the mesh scale.

Handle assembly constraints. For any handle $\mathbf{v}_h$ and each of its neighboring simplices $\mathbf{t}_p \in \Omega_s$, we impose:

$$(\mathbf{A}_p\mathbf{v}_h + \mathbf{T}_p) - \mathbf{v}_h^\star = 0. \tag{8}$$

Vertex assembly constraints. For any vertex $\mathbf{v} \in \mathcal{T}$ and each pair of adjacent edges defined by $\mathbf{e}_p \in \mathcal{T}, \mathbf{e}_q \in \mathcal{T}$, we impose the constraint:

$$(\mathbf{A}_p\mathbf{v} + \mathbf{T}_p) - (\mathbf{A}_q\mathbf{v} + \mathbf{T}_q) = 0. \tag{9}$$



(a) (b) (c)

Figure 3: *Mesh deformation. (a) An input mesh with three handles (yellow circles). The target positions of handles are colored in green. (b) Deformation result that only satisfies edge assembly constraints (Eq.* (7)*) and handle assembly constraints (Eq.* (8)*). (c) Deformation result that satisfies all the constraints (Eq.* (7)*,* (8) *and* (9)*).*

Here affine transformations $\{\mathbf{A}_p, \mathbf{T}_p\}$ and $\{\mathbf{A}_q, \mathbf{T}_q\}$ are associated with the tagged simplices of $\mathbf{e}_p$ and $\mathbf{e}_q$.

We denote $\widetilde{E}_{\text{assembly}}$ as the summation of squares of all the left sides of the above constraint equations (7), (8) and (9). In the Appendix we prove that the dissembled mesh can be glued together if $\widetilde{E}_{\text{assembly}} = 0$. Note that the vertex assembly constraints are essential to mesh assembly. Figure 3 shows that a mapping that does not satisfy Eq. (9) can yield a broken mesh (Figure 3-b).

By replacing E_{assembly} with $\widetilde{E}_{\text{assembly}}$ in Eq. (5), we can reduce the number of variables from $(d^2+d)N$ (d^2N from $\mathbf{A}_i$s, dN from $\mathbf{T}_i$s) to $d^2N + d\,|\Omega_s|$, where $|\Omega_s|$ counts the number of tagged simplices in $\mathcal{T}$ and is much smaller than N. Note that if there is no handle, then $\Omega_s = \emptyset$.

Computation of $\mathcal{T}$ Since we want to reduce translation variables as much as possible, the number of edges in $\mathcal{T}$ should be as small as possible. So finding $\mathcal{T}$ is indeed a minimum Steiner tree problem which is NP-complete [Ivanov and Tuzhilin 1994]. We approximate it by the following simple way.

1. Set $\mathcal{T}$ to be empty. Divide the handle set $\mathbf{H} = \{\mathbf{v}_h\}$ into disjoint groups $g_k, k = 1, \ldots, n$, where the handles in each group are connected in the vertex graph $\mathbf{G}$. Each g_i is a subgraph of $\mathbf{G}$. Compute a minimal spanning tree T_k of g_k and add T_k into $\mathcal{T}$, $\forall k$.

2. Regard g_k as a node of $\mathbf{G}$, i.e., nodes in g_k are merged virtually while the connectivity of the unaffected part of $\mathbf{G}$ is not changed. Compute the shortest paths between any pair of merged nodes $\tilde{g}_k$. Construct a complete graph S of $\tilde{g}_k$ whose edge length between $\tilde{g}_i$ and $\tilde{g}_j$ is the length of their shortest path.

3. Construct a minimal spanning tree S_t of S. Each edge of S_t corresponds to a shortest path in $\mathbf{G}$ and we add these corresponding paths into $\mathcal{T}$.

4.4 Numerical optimization

We stack all $\mathbf{A}_i$ and $\mathbf{T}_p \in \Omega_s$ in column format to form the variable vector $\mathbf{X}$ and employ the Newton method to minimize the following modified objective function:

$$E = \mu\, E_m + E_{\text{C}} + \lambda\, \widetilde{E}_{\text{assembly}}. \tag{10}$$

Our Newton-based updating scheme is:

$$\begin{aligned} \mathbf{H}_k &= \mu\, \nabla^2 E_m(\mathbf{X}_k) + \nabla^2 E_{\text{C}}(\mathbf{X}_k) + \lambda_k\, \nabla^2 \widetilde{E}_{\text{assembly}}(\mathbf{X}_k); \\ \delta\mathbf{X}_{k+1} &= -\mathbf{H}_k^{-1} \cdot \nabla E(\mathbf{X}_k); \\ \mathbf{X}_{k+1} &= \mathbf{X}_k + \alpha_k \cdot \delta\mathbf{X}_{k+1}. \end{aligned}$$

α_k is the step size determined by a backtracking line search to decrease the objective. At the k-th step, α_k is initialized with α_{k-1} and it is multiplied by two if the initial α_k decreases the energy. We set $\alpha_0 = 1$. λ is adaptively adjusted to enforce $\widetilde{E}_{\text{assembly}}$ to approach zero by the following scheme.

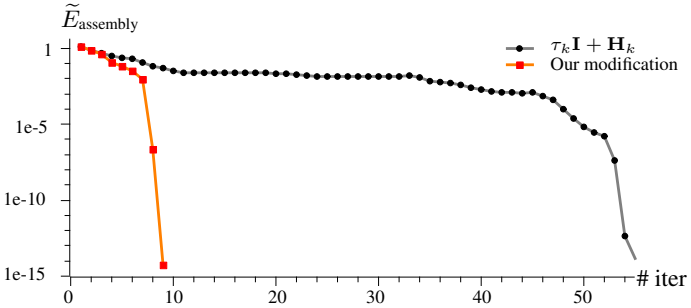


Figure 4: *#Iteration verus $\widetilde{E}_{assembly}$. The black curve is for the approach of $\tau_k\mathbf{I} + \mathbf{H}_k$. The orange curve represents our Hessian modification approach which converges faster.*

$$\lambda_{k+1} = \min\left(\lambda_{\min} \cdot \max\left(\frac{\mu\, E_m(\mathbf{X}_k) + E_{\text{C}}(\mathbf{X}_k)}{\widetilde{E}_{\text{assembly}}(\mathbf{X}_k)}, 1\right), \lambda_{\max}\right).$$

$\lambda_{\min}$ and $\lambda_{\max}$ are set as $\lambda_{\min} = 10^3$ and $\lambda_{\max} = 10^{16}$ in our experiments. Our λ-updating scheme makes $\widetilde{E}_{\text{assembly}}$ dominate the objective at the next iteration in general. But when E_{C} gets extremely large due to nearly degenerate simplices, E_{C} can take the dominant role in E since λ_k has an upper bound, and the optimizer will penalize E_{C} more. When E_{C} gets smaller, $\widetilde{E}_{\text{assembly}}$ will take the dominant role back because of the adjusted λ_k. This idea is similar to [Schüller et al. 2013] but we do not let λ_k always increase.

Hessian modification On each Newton-based iteration, any indefinite Hessian should be modified to be positive-definite. We modify our Hessian as follows.

1. Notice that $\nabla^2 E_{\text{C}}(\mathbf{X}_k)$ is a block diagonal matrix, where each block $\mathbf{B}_i$ is a $d^2 \times d^2$ matrix. We add $(-\kappa_i + 10^{-8})\mathbf{I}_{d^2\times d^2}$ to $\mathbf{B}_i$ if the smallest eigenvalue κ_i of $\mathbf{B}_i$ is non-positive.

2. Add $10^{-6}\lambda_k$ to the diagonal entries of $\mathbf{H}_k$ that correspond to the translation variables. This operation is equal to adding a regularization term $10^{-6}\lambda_k \|\mathbf{T}\|^2$ to E. We add this regularization to handle the possible redundant translation variables due to our approximation of the minimum Steiner tree.

3. If the modified Hessian by steps 1 and 2 is still not positive-definite (detected by Cholesky decomposition), we add τ to the diagonal entries of $\mathbf{H}_k$ that correspond to variables of $\mathbf{A}_i$s. τ is updated iteratively by $\tau_{l+1} = \min(10^{16}, 10\tau_l), \tau_0 = 10^{-5}$ until the modified Hessian is positive-definite.

In practice, we found that our dedicated Hessian modification significantly reduces the number of iterations compared to adding $\tau_k\mathbf{I}$ to $\mathbf{H}_k$ globally. Figure 4 shows a comparison on the example of Figure 1(h).

Termination Our algorithm terminates when $\widetilde{E}_{\text{assembly}}$ is less than 10^{-12} and the relative change of E is below 10^{-8}, or $k > 200$. For most examples in the paper, our algorithm takes at most 50 iterations to converge.

Speedup The memory and time cost of the Newton solver is expensive when the number of simplices is very large. We propose a submesh-based optimization strategy to speed up the computation. The basic idea is simple: minimize the objective with respect to a subset of variables at one time.

Assume an initial mapping Φ^0 is available; we improve the mapping as follows.

(1). Set $l = 0$.
(2). From the mapping Φ^l, map the input mesh $\mathbf{M}$ to the mesh $\mathbf{M}^l$ by minimizing Eq. (6).
(3). For each simplex $\mathbf{s}$ in $\mathbf{M}$, we create a submesh $\mathbf{S}$ by grouping simplices that are inside the m-ring of $\mathbf{s}$. In this way, we obtain N submeshes.
(4). For each submesh $\mathbf{S}$, the original handles of $\mathbf{M}$ are still handles. We treat each boundary vertex of $\mathbf{S}(\mathbf{s})$ as an additional fixed handle if the vertex is not the boundary vertex of $\mathbf{M}$. We then formulate the mapping computation problem of Eq. (10) on $\mathbf{S}$ and minimize it by our Newton method. When the minimization terminates, if $\widetilde{E}_{\text{assembly}}$ of $\mathbf{S}$ cannot reach zero ($\widetilde{E}_{\text{assembly}} > 10^{-12}$), we recursively enlarge $\mathbf{S}$ by adding 2 more rings and run the minimization again until $\widetilde{E}_{\text{assembly}}$ reaches zero or the ring size is larger than $m_{\max}$. We accept the optimization results even if $\widetilde{E}_{\text{assembly}} > 10^{-12}$ when $m = m_{\max}$.
(5). If all the submeshes satisfy $\widetilde{E}_{\text{assembly}} \leq 10^{-12}$ or $l \geq l_{\max}$, stop the algorithm; otherwise let $l := l + 1$ and treat the current affine transformations as ϕ^l and goto Step 2.

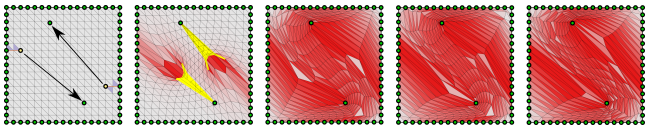
We set $m = 4$, $m_{\max} = 20$ and $l_{\max} = 2$ in our implementation. In step 4, we optimize the submeshes in parallel by using a graph coloring approach similar to the speedup strategy in [Fu et al. 2015]. To further speed up the optimization, we also skip the minimization of any submesh in step 4 if the distortions of its simplices are below the bound and $\widetilde{E}_{\text{assembly}} \leq 10^{-12}$.

4.5 Relation to integrable frame fields

The row vectors of the linear transformation matrix $\mathbf{A}$ can be regarded as a discrete d-dimensional frame (d vectors). On a simplicial mesh $\mathbf{M}$ associated with piecewise affine transformations, by collecting all the frames from affine transformations, we can form a discrete frame field Φ. If the edge assembly constraints (7) are satisfied, Φ is *curl free* across edges of $\mathbf{M}$. Φ is called an *integrable frame field* [Diamanti et al. 2015]. We conclude that an inversion-free mapping on a simplicial mesh naturally yields an integrable frame field. Diamanti *et al.* have shown the advantage of integrable frame fields in mesh parameterization: *the gradients of the parameterization coordinate functions exactly align with the designed fields*. For an input frame field with fixed singularities, our method is capable of refining it to an integrable frame field without altering the singularities. In Section 5.2, we show the application in global seamless mesh parameterization.

5 Experiments and Applications

Our method is capable of finding inversion-free mappings robustly and efficiently. In this section we first discuss the strengths of our method from different aspects in Section 5.1, then show its power in different tasks: mesh parameterization, fixed boundary mapping and mesh deformation. We also compare our method with the state-of-the-art methods. Our experiments were conducted on a desktop PC with a 3.4 GHz Intel Core i7 and 16 GB of RAM. Our speedup strategy is disabled by default and all the parameters in our algorithm use the suggested values always. The Cholesky decomposition used in the Newton solver is from the Eigen library [Guennebaud et al. 2010]. Our C++ implementation for triangle mesh mappings is released at http://staff.ustc.edu.cn/~fuxm/projects/SimplexAssembly/.



(a) (b) (c) (d) (e)

Figure 5: *(a) the source mesh and handle trajectories. (b) the initial map by solving a Laplacian smoothing problem; (c) r-SA result; (d) f-SA result ; (e) v-SA result. Here we set $K^{conf} = +\infty$, $E_m = 0$.*

Quality metrics To compare the quality of mappings from different methods, we unify the metric by following the approach of [Fu et al. 2015]. For each simplex $\mathbf{s}_j$, its isometric distortion metric is defined as $\delta^{iso}_{\mathbf{s}_j} := \max\{\sigma_{j,1}, 1/\sigma_{j,1}, \ldots, \sigma_{j,d}, 1/\sigma_{j,d}\}$ where $\sigma_{j,k}, k = 1, \ldots, d$ are the singular values of A_j. $\delta^{con}_{\mathbf{s}_j} := \max\{\sigma_{j,1}, \ldots, \sigma_{j,d}\}/\min\{\sigma_{j,1}, \ldots, \sigma_{j,d}\}$ is the conformal distortion metric. We use $\delta^{vol}_{\mathbf{s}_j} := \max\{\det(\mathbf{A}_j), \det(\mathbf{A}_j)^{-1}\}$ to measure the volumetric distortion metric. We denote $\delta^{iso}_{\max}, \delta^{iso}_{\text{avg}}$, $\delta^{con}_{\max}, \delta^{con}_{\text{avg}}$, and $\delta^{vol}_{\max}, \delta^{vol}_{\text{avg}}$ as the maximum and average of the above distortion metrics over all the simplices.

Our distortion definitions d^c, d^{vol} and d^{iso} are tightly connected to the above unified metrics. From the inequality $2\, d^c_{\max} \geq (\delta^{con}_{\max} + 1/\delta^{con}_{\max})$, we can see that δ^{con} is bounded if d^c is bounded. A similar conclusion exists for the volumetric distortion. The bounded δ^{con} and δ^{vol} also bound δ^{iso}.

In the works of [Kovalsky et al. 2015] and [Lipman 2012], the conformal bound δ^{con}_{bound} is defined such that $\delta^{con}_{\mathbf{s}_j} < \delta^{con}_{bound}, \forall j$. We can relate our bound K^{conf} to it via the formula $K^{\text{conf}} = \frac{1}{2}\left(\delta^{con}_{bound} + (\delta^{con}_{bound})^{-1}\right)$ for 2D mapping.

5.1 Strengths of simplex assembly

Variants of simplex assembly The concept of *simplex assembly* (SA) leads to two objectives: (5) and (10). The benefit of (10) is obvious: the number of variables is reduced significantly. The objective (5) can still be minimized with the help of Hessian modification. We name them *reduced-variable* SA (r-SA) and *full-variable* SA (f-SA).

Notice that the affine transformations can be linearly determined by the vertex positions of simplices:

$$\mathbf{A}_i = (\widehat{\mathbf{v}}_{i,0} - \widehat{\mathbf{v}}_{i,1}| \cdots |\widehat{\mathbf{v}}_{i,0} - \widehat{\mathbf{v}}_{i,d})\,(\mathbf{v}_{i,0} - \mathbf{v}_{i,1}| \cdots |\mathbf{v}_{i,0} - \mathbf{v}_{i,d})^{-1}$$
$$\mathbf{T}_i = \widehat{\mathbf{v}}_{i,0} - \mathbf{A}_i \mathbf{v}_{i,0}.$$

Here $\triangle \mathbf{v}_{i,0} \cdots \mathbf{v}_{i,d}$ is the simplex on the source mesh and $\triangle \widehat{\mathbf{v}}_{i,0} \cdots \widehat{\mathbf{v}}_{i,d}$ is its image. By disassembling all the simplices and treating their vertex positions as variables, we can represent (5) or (10) with respect to vertex positions. The number of variables is $(d^2 + d)N$ and is hard to reduce by following our reduction approach. We name this formulation *vertex-variable* SA (v-SA).

We compare these methods on a conformal 2D mesh deformation problem. The conformal bound is $+\infty$ and $E_m = 0$. The initial map is computed by solving a simple Laplacian energy, and its affine transformations are projected as the closest similar matrix. Figure 5 lists the results of these three methods. r-SA only introduces four translation vectors (associated triangles are colored in Figure 5(a)).

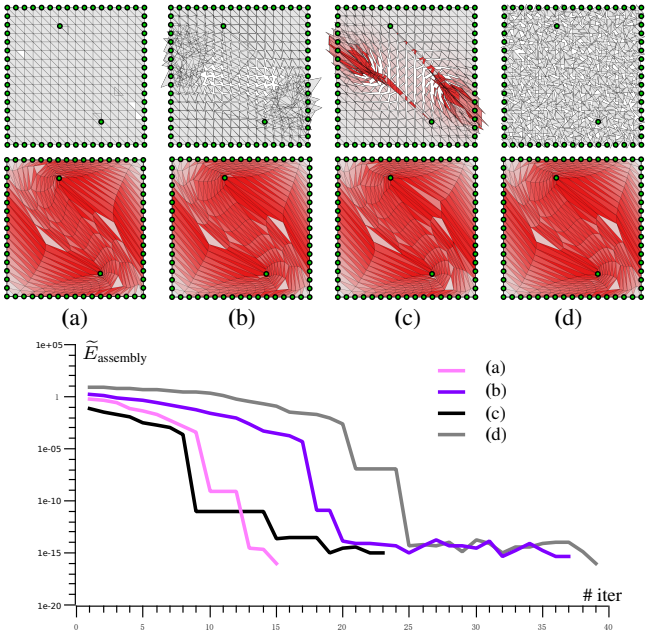


Figure 6: *Different initializations (1^{st} row) and their results (2^{nd} row). (a) Identity transformations; (b) closest similar transformations; (c) Bounded distortion space; (d) Project the random affine transformations to their closest rotation transformations. The bottom figure plots $\widetilde{E}_{assembly}$ with respect to the iteration number.*

r-SA took only 37 iterations (0.11 seconds) to converge while f-SA took 66 iterations (0.29 seconds), and v-SA took 219 iterations (0.71 seconds). The maximal and average conformal distortions of r-SA, f-SA and v-SA are (12.84, 7.72), (13.59, 7.90) and (13.44, 8.12), respectively. In our paper, r-SA is the default method.

Various initializations Our method is not very sensitive to different initial transformations. We use the previous example to demonstrate this. The initializations are (a) $\mathbf{A}_i = \mathbf{I}, \forall i$; (b) the closest similar transformation [Liu et al. 2008]; (c) the bounded distortion projection with bound $\delta_{bound}^{con} = 10$ [Kovalsky et al. 2015]; (d) random affine transformations (matrix entries are randomly selected from interval $[0, 1]$) and project them onto the closest rotation transformations. The upper row of Figure 6 shows the four initial states. The iteration number and the computational time are (16, 0.06 seconds), (37, 0.11 seconds), (23, 0.08 seconds) and (40, 0.13 seconds), respectively. δ_{max}^{con} and δ_{avg}^{con} from (a) to (d) are (13.27, 7.86), (12.84, 7.72), (14.72, 7.78) and (12.87, 7.72), respectively. We found that the convergence speed is not seriously affected by the initial mappings.

Parameter sensitivity The formulation of our E_C is similar to the AMIPS energy function in [Fu et al. 2015]. Fu *et al.* show that their method is sensitive to parameter s of the exponential function when the initial mapping has an extremely high distortion. Our method does not suffer from this issue because our initial mapping has a low distortion. Figure 7(a) shows a failure case for their method (isometric parameterization of a hand model). We start with the same Tutte mapping and set $E_m = 0$. Our method converges after 22 iterations using the same s value in Figure 7(c), 7 iterations for Figure 7(b) and 38 iterations for Figure 7(d).

Optimal bound In practice, setting a low bound of the distortion usually results in high-quality mappings as shown in [Lipman 2012; Kovalsky et al. 2014]. Kovalsky *et al.* propose an optimization strategy that uses the current maximal distortion as the bound for the next round of minimization. This strategy has been integrated into the implementation of [Lipman 2012]. We utilize their strategy

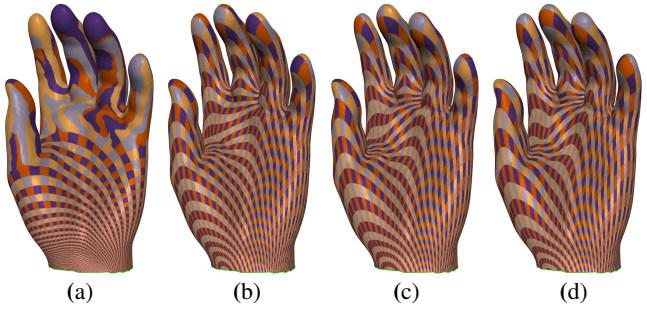


Figure 7: *Failure case for AMIPS using default parameters in [Fu et al. 2015] for isometric parameterization. (a) is a failure example of AMIPS using $s = 4$. (b), (c) and (d) are our results using different s values 1, 4, 8. Their maximal and average isometric distortion are (10.58, 2.86), (5.19, 2.86) and (4.62, 2.96), respectively. We set $E_m = 0$.*

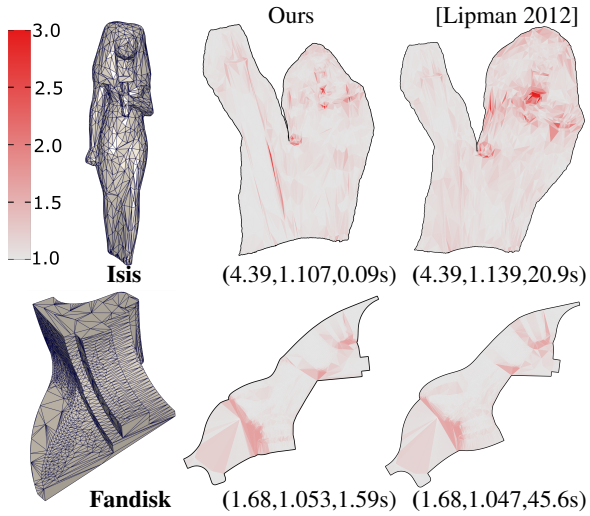
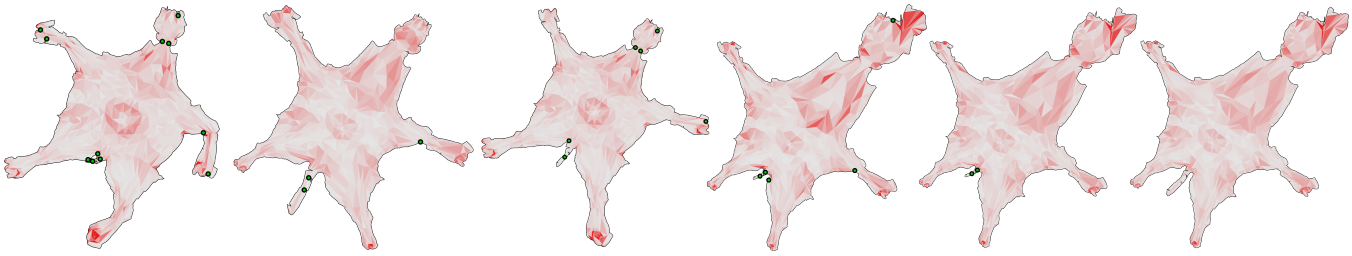


Figure 8: *Optimal conformal bound. We test on two models Isis and Fandisk. The text shows the maximal conformal distortion, average conformal distortion and running time. We use the authors' implementation of [Lipman 2012] with the default setting to find the optimal bound.*

to find the optimal bound in our implementation. We compute a mapping by setting an initial bound of $K = +\infty$, then pick the maximal distortion of the mapping as K and repeat the optimization until no further improvement can be found. We compared with Lipman's method on the Isis and Fandisk models and found that our method obtains conformal bounds similar to theirs as shown in Figure 8 but our method is at least one-order of magnitude faster.

Locally injective mapping Locally injective mapping is an important property for many applications. For a triangle mesh mapping, local injectivity requires that (1) the mapping is inversion-free; (2) for any interior vertex $\mathbf{v}$ in the image of the mapping, $\theta_{\mathbf{v}}$ (the sum of triangle angles around $\mathbf{v}$) is 2π; (3) for any boundary vertex $\mathbf{v}$ in the image of the mapping, $\theta_{\mathbf{v}}$ is less than 2π. Note that condition (1) and (3) imply that the map on the boundary is locally injective, and (2) is satisfied [Lipman 2014]. Thus for achieving a locally injective mapping, we only need to consider the constraints of (1) and (3). We simply add a new barrier function $E_\theta = \sum_{\mathbf{v}\in\partial\mathbf{M}} \frac{1}{2\pi-\theta_{\mathbf{v}}}$ to our objective function E. In the mesh disassembly step, we can easily find new affine transformations that can result in $\theta_{\mathbf{v}} < 2\pi, \forall \mathbf{v} \in \partial\mathbf{M}$ by stretching the triangles to make the boundary angle smaller while they do not trigger any triangle inversion. Figure 9 shows a comparison with different methods. With the barrier function E_θ, we can achieve locally injective parameterization (Figure 9(f)).



(a) [Lipman 2012] (b) AMIPS (c) [Schüller et al. 2013] (d) FBDM (e) Ours without E_θ (f) Ours with E_θ

Figure 9: *Conformal parameterizations of Camel model of Figure 10. From (a) to (f), there are 11, 3, 6, 5, 2 and 0 boundary vertices (green circles) whose the sum of triangle angles around it is greater than 2π. The mapping is not locally injective at these vertices. We set $K^{conf} = +\infty$, $E_m = \sum_i \text{area}(\mathbf{s}_i)\|\mathbf{A}_i - \mathbf{A}_i^0\|_F^2$. The speedup strategy is employed here and the initial mapping is from Linear ABF [Zayer et al. 2007] for (e) and (f). Triangles are color-coded using the same setting as in Figure 8.*

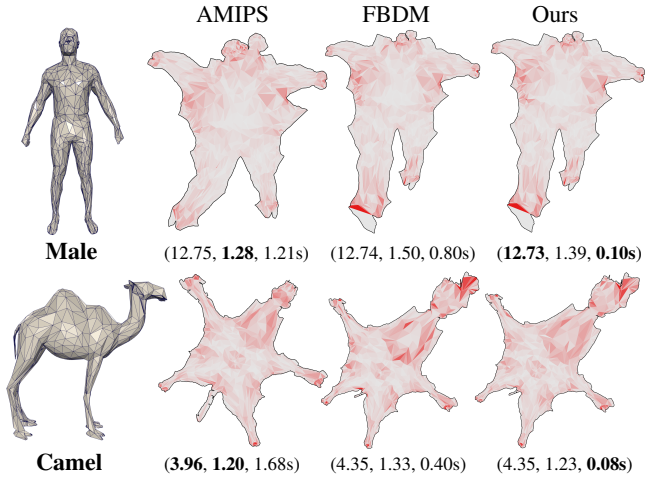


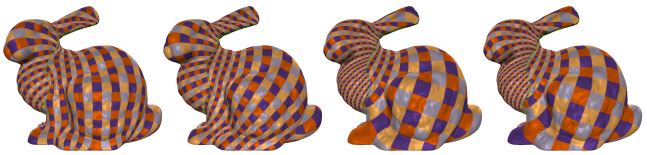
Figure 10: *Conformal parameterization for two models. The initial mapping is from Linear ABF [Zayer et al. 2007]. The text shows the maximal conformal distortion, average conformal distortion and running time.*

For tetrahedral meshes, there are similar conditions on the angles except for the inversion-free requirement: (1) for each boundary edge $\mathbf{e}$, $\theta_\mathbf{e}$ (the sum of dihedral angles around $\mathbf{e}$) is less than 2π; (2) for each boundary vertex $\mathbf{v}$, $\theta_\mathbf{v}$ (the sum of solid angles around $\mathbf{v}$) is less than 4π; and (3) the sum of solid angles around interior vertex is 4π. The conditions on boundary edges and vertices also imply the condition on interior vertices. Similar barrier functions $\sum_{\mathbf{e}\in\partial\mathbf{M}} \frac{1}{2\pi-\theta_\mathbf{e}}$ and $\sum_{\mathbf{v}\in\partial\mathbf{M}} \frac{1}{4\pi-\theta_\mathbf{v}}$ can be added to E.

Difference from the local/global strategy Our method is somewhat similar to the local/global strategy used in ARAP/ASAP [Liu et al. 2008] and Poisson editing [Yu et al. 2004] at first glance, where Liu *et al.* repeatedly disassemble the simplices (local step) and reconstructs vertex positions (global step), Yu *et al.* apply one round of the local and global step. However, in the global step, their least-squares assembling approach cannot prevent simplex inversion. In contrast, we only disassemble the simplices once, and our optimization (assembling) helps find an inversion-free mapping.

5.2 Mesh parameterization

Planar parameterization Figure 10 shows an example of conformal parameterization on two models. We compare our method with FBDM [Kovalsky et al. 2015] and AMIPS [Fu et al. 2015]. In order to make the maximal distortion close to the set bound for FBDM, we revise their stopping condition to be $\delta_{max}^{con} < \delta_{bound}^{con} + 10^{-3}$ or the iteration number exceeding 5000 for all the following comparisons. Our speedup strategy is employed here and $K^{\text{conf}} = +\infty$. We set



(a) (b) (c) (d)

Figure 11: *Conformal parameterization for a Bunny model (1251046 vertices). (a) and (b) are the results of [Fu et al. 2015] initialized with a circular convex mapping and the result of Linear ABF [Zayer et al. 2007].(c) FBDM. (d) Our result.*

the $E_m = \sum_i \text{area}(\mathbf{s}_i)\|\mathbf{A}_i - \mathbf{A}_i^0\|_F^2$ which is also used by FBDM. We set the conformal bound of FBDM by the maximal distortion of our result. The experiment shows that our method is the fastest and can achieve similar distortion to AMIPS.

Figure 11 shows the conformal parameterization on a Bunny model (1.25 million vertices and 2.5 million faces). We use the result of LSCM [Lévy et al. 2002] which contains foldovers as initialization and set E_m = LSCM in our solver with $K^{conf} = 1.45$ (same to $\delta_{bound}^{con} = 2.5$). Our method with the speedup strategy took 10.2 seconds while the AMIPS method [Fu et al. 2015] took 2 hours with the Tutte embedding initialization and 2 minutes with the Linear ABF [Zayer et al. 2007] initialization. FBDM took 82.3 seconds to converge using the same bound and the LSCM initialization. The maximal and average conformal distortion of the result of AMIPS (Figure 11(a)&(b)), FBDM (Figure 11(c)) and our method (Figure 11(d)) are (2.29, 1.006), (2.25, 1.003), (2.50, 1.004) and (2.42, 1.004) respectively. Our method obtains a comparable result in the least time.

Global seamless parameterizations For frame field driven mesh parameterization, the non-integrability of the frame field is usually the source of foldovers of the parameterization. For a given input frame field ϕ on a mesh, we seek a new frame field ϕ' that is close to ϕ and is integrable with the same singularities. The computation is simple: (1) build a local coordinate system on each face and convert each frame ϕ_i' as a 2D linear transformation matrix $\mathbf{A}_i$. (2) we impose the constraints on each edge $\mathbf{e}$ of the mesh:

$$\mathbf{A}_l\,\mathbf{e}_l = \mathbf{R}_e\,\mathbf{A}_r\,\mathbf{e}_r. \tag{11}$$

$\mathbf{A}_l$ and $\mathbf{A}_r$ are defined on the neighboring facets of $\mathbf{e}$, and $\mathbf{e}_l$ and $\mathbf{e}_r$ are 2D normalized edge vectors of $\mathbf{e}$ under their corresponding local coordinate systems. $\mathbf{R}_e$ is the rotation matrix defined on $\mathbf{e}$ and it is fixed since we do not change the singularities of the frame field. $\|\mathbf{A}_l\,\mathbf{e}_l - \mathbf{R}_e\,\mathbf{A}_r\,\mathbf{e}_r\|^2$ measures the squared edge-curl defined on edge $\mathbf{e}$. The summation of the squares of (11) forms $\widetilde{E}_{\text{assembly}}$ and we use our simplex assembly method to find ϕ'. We set $E_m = \|\phi' - \phi\|^2$ to minimize the derivation from the original field. We also set $K^{\text{conf}} = +\infty$, $K^{\text{vol}} = +\infty$ and use our speedup strategy.

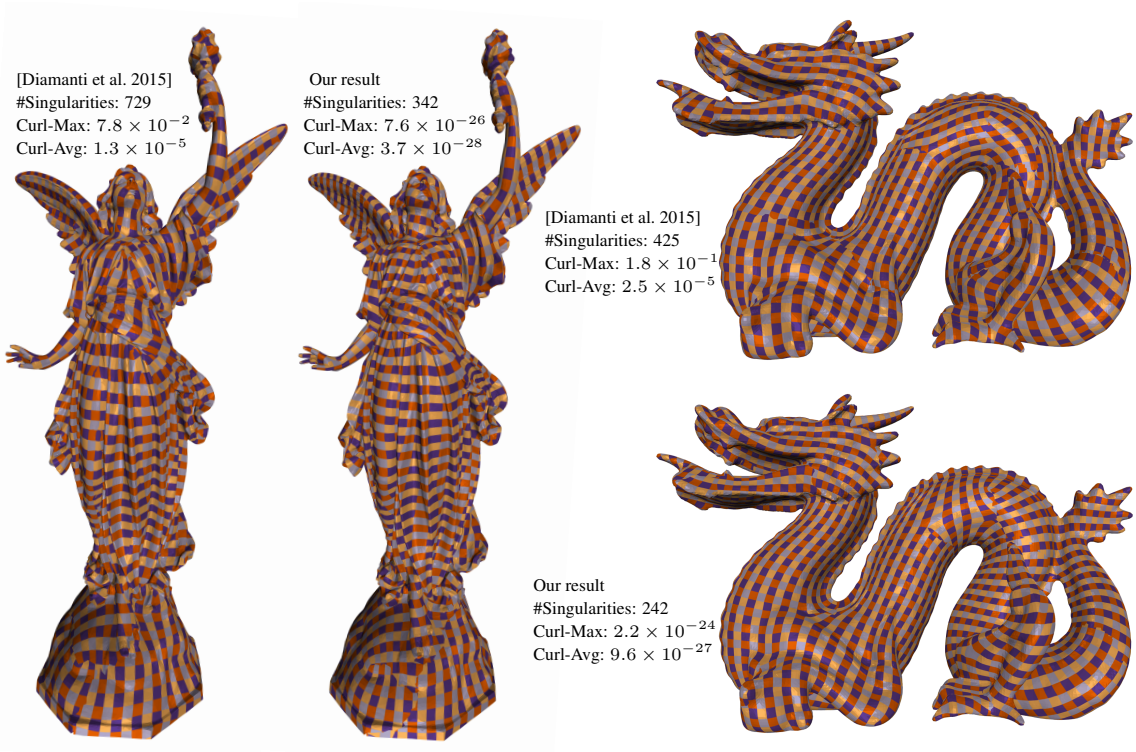
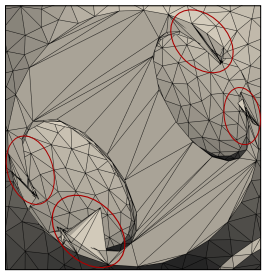


Figure 12: *Global seamless parameterization. Compared with [Diamanti et al. 2015], our method maintains the original singularities and has much lower edge-curl values.*

The parameterization result can be obtained by solving the Poisson equation which finds two piecewise linear scalar functions defined on mesh vertices whose gradients align to the frame field [Diamanti et al. 2015].

We test our algorithm on a benchmark dataset [Myles et al. 2014; Diamanti et al. 2015] (114 models) where there are 80 models whose parameterizations solved by the initial frame field have foldovers. Our method successfully finds the integrable frame fields for 73 models whose maximal edge-curl value is less than 10^{-18}. For the remaining 7 models, the maximal and average edge-curl values are about 10^{-7} and 10^{-11}. We can generate foldover-free parameterizations for these seven models by solving the Poisson equation. However, the results are not exactly seamless although the seam is invisible. If we add the hard constraints (11) into the Poisson equation, the parameterization results of these seven models have a few foldovers. We observe that these seven models contain a few long and slim spikes (see the tangled triangles in the inset) where the input frame field is non-smooth. By smoothing the positions of vertices in these local regions by several steps of Laplacian smoothing and projecting the frame field onto the modified faces, our method can successfully find integrable frame fields. The full statistics of our test is provided in the supplemental material. The average computation time of each model is about 15 seconds.



Comparisons Myles *et al.* [2014] reported that the method of [Lipman 2012] failed on 20 models when the conformal bound is $\delta_{bound}^{con} = 199$. Diamanti *et al.* [2015] introduce the PolyVector technique to compute integrable frame fields. They stop the optimization when the average residual of the Poisson step is lower than 10^{-3}.

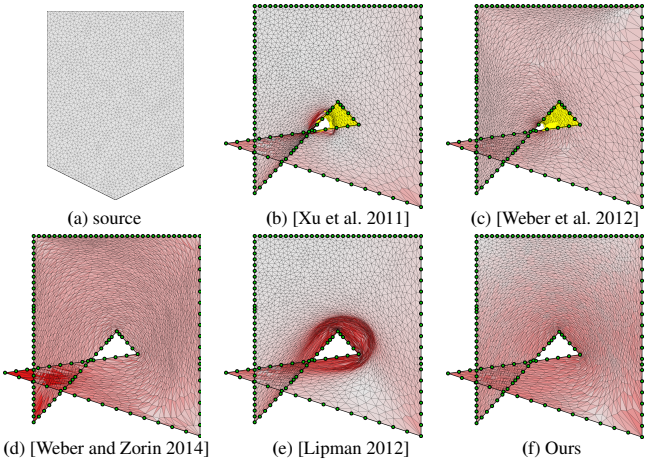


Figure 13: *Comparisons for 2D conformal fixed boundary mapping. (a) input. (b) the result of [Xu et al. 2011] with 150 foldovers. (c) [Weber et al. 2012]'s result with 162 foldovers. (d) extremal map with 4 new vertices of [Weber and Zorin 2014]. (e) [Lipman 2012] with bound = 12. (f) our result from the initialization of [Xu et al. 2011]. Triangles are color-coded by the mapping distortion. The results of (b,c,d) are from [Weber and Zorin 2014].*

Therefore, their frame fields are not exactly integrable because of the relatively high value of edge curl. Their parameterization results are visually seamless, and the number of singularities is much higher than the original one. Figure 12 shows a comparison between our method and Diamanti *et al.*'s on two models. Our method maintains the low singularities of the input frame field. Thus, it is more suitable for parameterizing meshes without interfering with the users' design of frame fields. However, similar to their work, our method has no guarantee of generating a foldover-free quadrangulation due to the integer rounding.

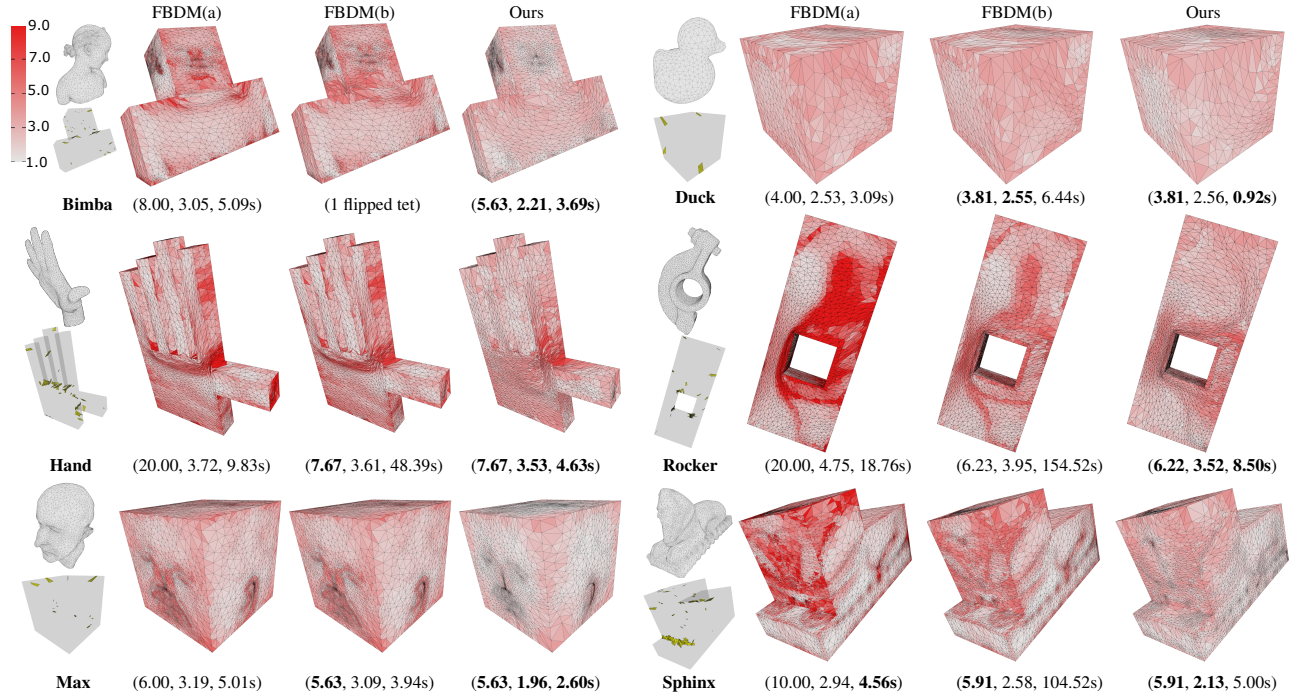


Figure 14: *Conformal PolyCube mapping of six models using our algorithm and FBDM with different settings. The initial mappings of FBDM(a) and FBDM(b) are the same as in [Aigerman and Lipman 2013]. The bound of FBDM(a) is the same as in [Kovalsky et al. 2015]. FBDM(b) uses our maximal conformal distortion as the bound. The bottom-left insets show the initial mappings with inverted tetrahedrons (colored in yellow). The boundary tetrahedrons are color-coded using the magnitude of δ^{con}. The three numbers below each figure report the maximal conformal distortion, average conformal distortion and running time. The number of vertices and tetrahedrons of the models are: Bimba (10790,45422), Duck (2464,12601), Hand (8366,40627), Rocker (12428,60301), Max (9867,40076), Sphinx (10528,43371).*

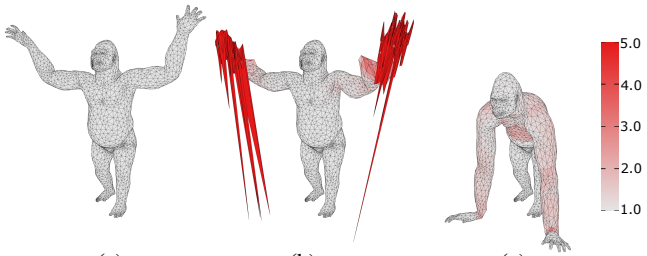


Figure 15: *Mesh deformation. (a) is the source mesh. (b) is a failure result of AMIPS. Our result is shown in (c).*

5.3 Fixed boundary mapping

2D fixed boundary mapping Figure 1 shows a 2D conformal fixed boundary mapping example computed using different methods. Only our method and the bounded distortion mapping method [Lipman 2012] can find valid results without adding vertices. We also show the intermediate results of our method in Figure 2. For illustration purposes, we use f-SA in Figure 2 so that all the translation vectors can be determined at each iteration.

Figure 13 shows another example that is used in [Weber and Zorin 2014]. The methods of [Weber et al. 2012; Xu et al. 2011] fail to generate inversion-free results. The method of [Weber and Zorin 2014] inserts four vertices into the triangulation to avoid inversion. In our method, we set $K^{\text{conf}} = \infty$, $E_m = 0$ and the speedup strategy is not employed. Initialized with the result of [Xu et al. 2011], our method produces a low-distortion inversion-free mapping in 1.85 seconds. The maximal and average conformal distortion of our result, [Weber and Zorin 2014]'s and [Lipman 2012]'s are (8.59, 3.48), (31.35, 4.22) and (12.00, 4.29). The bounded distortion mapping method [Lipman 2012] (using the authors' implementation) fails when δ^{con}_{bound} is below 12.

PolyCube mapping Because PolyCube mapping needs as low as possible distortion for many applications, we apply our method to seek the optimal conformal bound. Our speedup strategy is employed and $E_m = 0$. We use the same initial mappings of [Aigerman and Lipman 2013] to construct PolyCube parameterizations. We also allow boundary vertices to move on its corresponding plane or edge. This movement is treated as linear constraints with respect to affine transformations, and their squares are added to $\widetilde{E}_{\text{assembly}}$. Figure 14 shows our results on six models. We also compare our method with FBDM. FBDM may not converge on a low distortion bound and it fails for the Bimba model. In general FBDM spends more time to converge except for the Sphinx model, and its average conformal distortion is larger than ours.

5.4 Mesh deformation

Figure 15 shows an example of isometric tetrahedral mesh deformation. The palms of the gorilla are pushed to the floor where all the vertices on hands are handles. AMIPS, the method of [Fu et al. 2015], fails to generate an inversion-free deformation because their substepping strategy does not work due to the conflict of handle trajectories (Figure 15(b)). Our method starts with the ARAP result that contains inverted tetrahedrons. We set $E_m = \sum_i \|\mathbf{A}_i - \mathbf{A}_i^0\|_F^2$ to minimize the change to the initial ARAP mapping. We also seek the optimal bound both on conformal and volumetric distortion simultaneously on this example. We achieve an inversion-free mapping in 1.05 seconds with low maximal conformal and volumetric distortion ($\delta^{con}_{max} = 3.55$ and $\delta^{vol}_{max} = 4.34$). $E_m = 250.05$ and its average is 0.012 on each tetrahedron in the final result. The speedup strategy is used in this example. We also compared our method with

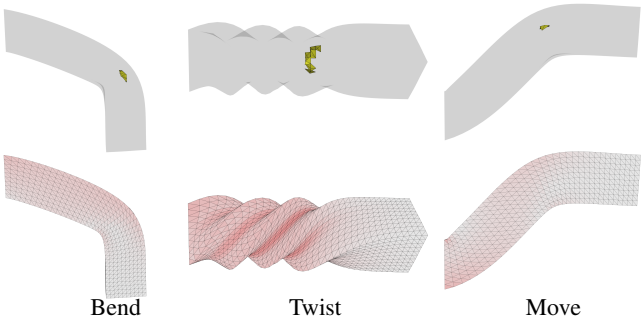


Figure 16: *Bend, twist and translate a bar. The initial mapping results of the first row are from [Aigerman and Lipman 2013] and contain some inverted tetrahedrons. The second row shows our low distortion and inversion-free results. The color coding is the same as Figure 15.*

the method of [Kovalsky et al. 2014] and FBDM with $\delta_{bound}^{con} = 5$ on this example. All the methods achieve comparable results and ours is slightly better. The method of [Kovalsky et al. 2014] took 264.96 seconds and FBDM is the fastest and took 0.59 seconds.

Figure 16 shows three isometric deformation examples on a bar model. $E_m = \sum_i \|\mathbf{A}_i - \mathbf{A}_i^0\|_F^2$ is also incorporated into our energy and we seek the optimal bound both on conformal and volumetric distortion. The maximal and average isometric distortions are (5.42, 1.30), (5.22, 1.45), (4.55, 1.21) respectively, and E_m are 257.64, 256.43, 67.21. The speed of our method (using the speedup strategy) is slightly better than FBDM [Kovalsky et al. 2015] with $\delta_{bound}^{con} = 5$, (0.68 seconds, 0.58 seconds, 0.60 seconds) versus (1.01 seconds, 0.71 seconds, 0.85 seconds).

6 Conclusion

Our novel simplex assembly method enables a robust and efficient way to compute inversion-free mappings with or without distortion bound. We demonstrate the advantages of our method in various applications and challenging examples. The existing techniques without inversion-free guarantees can benefit from our approach. The intuitiveness of our algorithm – *disassembly* + *assembly* – makes the implementation easy. Our formulation also naturally yields integrable frame fields.

Convergence Since our objective is nonlinear and non-convex, we have no theoretical guarantee that the result of our method always satisfies all the constraints. As shown earlier, our method fails on 7 models in the global seamless parameterization test. Another potential limitation comes from the minimization of Eq. (6). The least squares approach may introduce flips due to numerical error since $\widetilde{E}_{\text{assembly}} < 10^{-12}$ is not exactly zero; however, we have not observed this phenomenon.

Efficiency Compared with the approach of [Kovalsky et al. 2015], our method has a limitation: the Hessian in our computation cannot be pre-factorized for speedup. However, thanks to our speedup strategy, the speed of our method is comparable to or better than [Kovalsky et al. 2015]. Furthermore, E_m in our method is not required to be a quadratic energy while it is demanded by [Kovalsky et al. 2015].

Acknowledgements

Mesh models are courtesy of the Aim@Shape Repository, the Stanford 3D Scanning Repository and the INRIA Alice group. We thank the authors of [Kovalsky et al. 2015] for providing their implementation, and the anonymous reviewers for their constructive feedback.

References

AIGERMAN, N., AND LIPMAN, Y. 2013. Injective and bounded distortion mappings in 3D. *ACM Trans. Graph. (SIGGRAPH) 32*, 4, 106:1–106:14.

BEN-CHEN, M., GOTSMAN, C., AND BUNIN, G. 2008. Conformal flattening by curvature prescription and metric scaling. *Comput. Graph. Forum (EG) 27*, 2, 449–458.

BOTSCH, M., KOBBELT, L., PAULY, M., ALLIEZ, P., AND LÉVY, B. 2010. *Polygon Mesh Processing*. A K Peters/CRC Press.

CHAO, I., PINKALL, U., SANAN, P., AND SCHRÖDER, P. 2010. A simple geometric model for elastic deformations. *ACM Trans. Graph. (SIGGRAPH) 29*, 4, 38:1–38:6.

CHEN, R., AND WEBER, O. 2015. Bounded distortion harmonic mappings in the plane. *ACM Trans. Graph. (SIGGRAPH) 34*, 4, 73:1–73:12.

CRANE, K., PINKALL, U., AND SCHRÖDER, P. 2011. Spin transformations of discrete surfaces. *ACM Trans. Graph. (SIGGRAPH) 30*, 104:1–104:10.

CRANE, K., PINKALL, U., AND SCHRÖDER, P. 2013. Robust fairing via conformal curvature flow. *ACM Trans. Graph. (SIGGRAPH) 32*, 61:1–61:10.

DEGENER, P., MESETH, J., AND KLEIN, R. 2003. An adaptable surface parameterization method. In *Int. Meshing Roundtable*, 201–213.

DIAMANTI, O., VAXMAN, A., PANOZZO, D., AND SORKINE-HORNUNG, O. 2015. Integrable PolyVector fields. *ACM Trans. Graph. (SIGGRAPH) 34*, 4, 38:1–38:12.

FLOATER, M. S., AND HORMANN, K. 2005. Surface parameterization: a tutorial and survey. In *In Advances in Multiresolution for Geometric Modelling*, Springer, 157–186.

FLOATER, M. S. 2003. One-to-one piecewise linear mappings over triangulations. *Math. Comput. 72*, 685–696.

FU, X.-M., LIU, Y., AND GUO, B. 2015. Computing locally injective mappings by advanced MIPS. *ACM Trans. Graph. (SIGGRAPH) 34*, 4, 71:1–71:12.

GUENNEBAUD, G., JACOB, B., ET AL., 2010. Eigen v3. http://eigen.tuxfamily.org.

HORMANN, K., AND GREINER, G. 2000. MIPS: An efficient global parametrization method. In *Curve and Surface Design: Saint-Malo 1999*. Vanderbilt University Press, 153–162.

IVANOV, A. O., AND TUZHILIN, A. A. 1994. *Minimal Networks: The Steiner Problem and Its Generalizations*. CRC Press.

JIN, Y., HUANG, J., AND TONG, R. 2014. Remeshing-assisted optimization for locally injective mappings. *Comput. Graph. Forum (SGP) 33*, 5, 269–279.

KOVALSKY, S. Z., AIGERMAN, N., BASRI, R., AND LIPMAN, Y. 2014. Controlling singular values with semidefinite programming. *ACM Trans. Graph. (SIGGRAPH) 33*, 4, 68:1–68:13.

KOVALSKY, S. Z., AIGERMAN, N., BASRI, R., AND LIPMAN, Y. 2015. Large-scale bounded distortion mappings. *ACM Trans. Graph. (SIGGRAPH ASIA) 34*, 6, 191:1–191:10.

LEVI, Z., AND ZORIN, D. 2014. Strict minimizers for geometric optimization. *ACM Trans. Graph. (SIGGRAPH) 33*, 6, 185:1–185:14.

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

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

LIPMAN, Y. 2014. Bijective mappings of meshes with boundary and the degree in mesh processing. *SIAM J. Imaging Sciences 7*, 2, 1263–1283.

LIU, L., ZHANG, L., XU, Y., GOTSMAN, C., AND GORTLER, S. J. 2008. A local/global approach to mesh parameterization. *Comput. Graph. Forum (SGP) 27*, 5, 1495–1504.

LIU, T., GAO, M., ZHU, L., SIFAKIS, E., AND KAVAN, L. 2016. Fast and robust inversion-free shape manipulation. *Comput. Graph. Forum (EG) 35*, 2.

MYLES, A., PIETRONI, N., AND ZORIN, D. 2014. Robust field-aligned global parametrization. *ACM Trans. Graph. (SIGGRAPH) 33*, 4, 135:1–135:14.

NOCEDAL, J., AND WRIGHT, S. 2006. *Numerical Optimization*, 2nd ed. Springer.

PAILLÉ, G.-P., RAY, N., POULIN, P., SHEFFER, A., AND LÉVY, B. 2015. Dihedral angle-based maps of tetrahedral meshes. *ACM Trans. Graph. (SIGGRAPH) 34*, 4, 54:1–54:10.

PORANNE, R., AND LIPMAN, Y. 2014. Provably good planar mappings. *ACM Trans. Graph. (SIGGRAPH) 33*, 4, 76:1–76:11.

RABINOVICH, M., PORANNE, R., PANOZZO, D., AND SORKINE-HORNUNG, O. 2016. Scalable locally injective maps. Tech. rep., ETH.

SCHÜLLER, C., KAVAN, L., PANOZZO, D., AND SORKINE-HORNUNG, O. 2013. Locally injective mappings. *Comput. Graph. Forum (SGP) 32*, 5, 125–135.

SHEFFER, A., AND DE STURLER, E. 2001. Parameterization of faceted surfaces for meshing using angle-based flattening. *Eng. Comput. 17*, 3, 326–337.

SHEFFER, A., LÉVY, B., MOGILNITSKY, M., AND BOGOMYAKOV, A. 2005. ABF++: fast and robust angle based flattening. *ACM Trans. Graph. 24*, 2, 311–330.

SMITH, J., AND SCHAEFER, S. 2015. Bijective Parameterization with Free Boundaries. *ACM Trans. Graph. (SIGGRAPH) 34*, 4, 70:1–70:9.

SPRINGBORN, B., SCHRÖDER, P., AND PINKALL, U. 2008. Conformal equivalence of triangle meshes. *ACM Trans. Graph. (SIGGRAPH) 27*, 3, 77:1–77:11.

TUTTE, W. T. 1963. How to draw a graph. In *Proceedings of the London Mathematical Society*, vol. 13, 747–767.

WEBER, O., AND ZORIN, D. 2014. Locally injective parametrization with arbitrary fixed boundaries. *ACM Trans. Graph. (SIGGRAPH) 33*, 4, 75:1–75:12.

WEBER, O., MYLES, A., AND ZORIN, D. 2012. Computing extremal quasiconformal maps. *Comput. Graph. Forum 31*, 5, 1679–1689.

XU, Y., CHEN, R., GOTSMAN, C., AND LIU, L. 2011. Embedding a triangular graph within a given boundary. *Comput. Aided Geom. Des. 28*, 6, 349–356.

YU, Y., ZHOU, K., XU, D., SHI, X., BAO, H., GUO, B., AND SHUM, H.-Y. 2004. Mesh editing with Poisson-based gradient field manipulation. *ACM Trans. Graph. (SIGGRAPH) 23*, 3, 644–651.

ZAYER, R., LÉVY, B., AND SEIDEL, H.-P. 2007. Linear angle based parameterization. In *Symp. Geom. Proc.*, 135–141.

Appendix: variable reduction

We shall prove that the disassembled mesh can be assembled together when $\widetilde{E}_{\text{assembly}} = 0$. The following simple proof is also a constructive way to find unknown $\mathbf{T}_i$s.

For a mapping ϕ that satisfies $\widetilde{E}_{\text{assembly}} = 0$, we only know values of $\mathbf{T}_i$ in Ω_s and the remaining $\mathbf{T}_i$ are unknown. We can set arbitrary values to them first and denote $\phi(\mathbf{M})$ as $\mathbf{M}'$. We say two simplices in $\mathbf{M}'$ are *glued* if their preimages share at least one common vertex and the images of this vertex have the same coordinate in $\mathbf{M}'$. Since the vertex assembly constraints (9) are always satisfied due to $\widetilde{E}_{\text{assembly}} = 0$, all the images of the simplices of Ω_s are connected via the handle vertices and form a submesh $\mathbf{S}$.

Now we consider any simplex $\mathbf{s}_o \notin \mathbf{S}$ whose preimage shares a d-dimensional face with simplex $\mathbf{s}$ in $\mathbf{S}$. Due to the edge assembly constraints (7), we can glue $\mathbf{s}_o$ and $\mathbf{s}$ by freely translating $\mathbf{s}_o$, and we enlarge $\mathbf{S}$ by $\mathbf{S} \bigcup \mathbf{s}_o$. We can recursively process the remaining simplices out of $\mathbf{S}$ in this way until all the simplices are glued together. The composition of the initial translation and the later translation for gluing recovers the unknown $\mathbf{T}_i$s. ■