Computational Visual Media
https://doi.org/10.1007/s41095-021-0233-9 Vol. 7, No. 3, September 2021, 289–318

Review Article

Inversion-free geometric mapping construction: A survey

Xiao-Ming Fu[1] (✉), Jian-Ping Su[1], Zheng-Yu Zhao[1], Qing Fang[1], Chunyang Ye[1], and Ligang Liu[1]

Abstract A geometric mapping establishes a correspondence between two domains. Since no real object has zero or negative volume, such a mapping is required to be inversion-free. Computing inversion-free mappings is a fundamental task in numerous computer graphics and geometric processing applications, such as deformation, texture mapping, mesh generation, and others. This task is usually formulated as a non-convex, nonlinear, constrained optimization problem. Various methods have been developed to solve this optimization problem. As well as being inversion-free, different applications have various further requirements. We expand the discussion in two directions to (i) problems imposing specific constraints and (ii) combinatorial problems. This report provides a systematic overview of inversion-free mapping construction, a detailed discussion of the construction methods, including their strengths and weaknesses, and a description of open problems in this research field.

Keywords inversion-free mapping; Jacobian matrix; distortion; first-order methods; second-order methods

1 Introduction

In computer graphics and geometric processing, a geometric mapping $f : \Omega \subset \mathbb{R}^m \to \mathbb{R}^n$ transforms its domain Ω into another domain. The task of computing geometric mappings is expected and essential. For example, the difficulty of geometric processing tasks can be significantly reduced in the mapped domains. In texture mapping, the parameterization is used to map an existing 2D image onto the 3D model. In FEM simulation, the geometric mapping is optimized to improve the mesh quality, thereby improving the simulation accuracy.

There is no zero or even negative volume in any natural material. Intuitively, any physical deformation will not result in zero or even negative volume. Thus, the determinant of the Jacobian matrix of f at any $\boldsymbol{x} \in \Omega$, which indicates the ratio between the original volume and the transformed volume, must be positive. This constraint is called an *inversion-free* condition. In other literature, other names are used, such as *flip-free* constraint, *foldover-free* constraint, and *orientation-preserving* constraint. We are interested in surveying the body of work that focuses on the active creation of such mappings.

If not specified, the models we focus on are all the d-dimensional simplicial meshes on $\mathbb{R}^m$, which will be introduced in Section 2. The mapping we deal with then will be the piecewise linear mapping on the simplicial meshes:

$$f(\boldsymbol{x}) = \boldsymbol{J}_i \boldsymbol{x} + \boldsymbol{t}_i, \quad \forall \boldsymbol{x} \in \boldsymbol{s}_i$$

where $\boldsymbol{x}$ is a point on the simplex $\boldsymbol{s}_i$ of a mesh, and $\boldsymbol{J}_i$ is the discrete Jacobian matrix, which is constant on $\boldsymbol{s}_i$. $\boldsymbol{t}_i$ is a transformation vector.

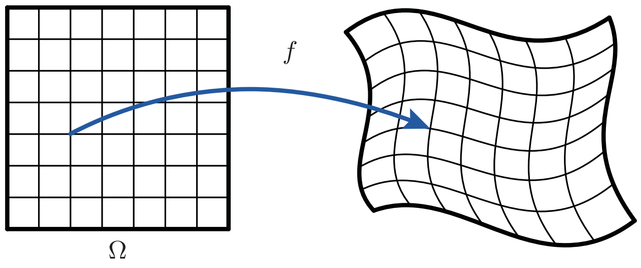


Fig. 1 Illustration for the geometric mapping f.

1 School of Mathematical Sciences, University of Science and Technology of China, Hefei, China. E-mail: X.-M. Fu, fuxm@ustc.edu.cn (✉); J.-P. Su, SJPing@mail.ustc.edu.cn; Z.-Y. Zhao, zyzhao18@mail.ustc.edu.cn; Q. Fang, fq1208@mail.ustc.edu.cn; C. Ye, yechyang@mail.ustc.edu.cn; L. Liu, lgliu@ustc.edu.cn.

Manuscript received: 2020-12-29; accepted: 2021-03-27

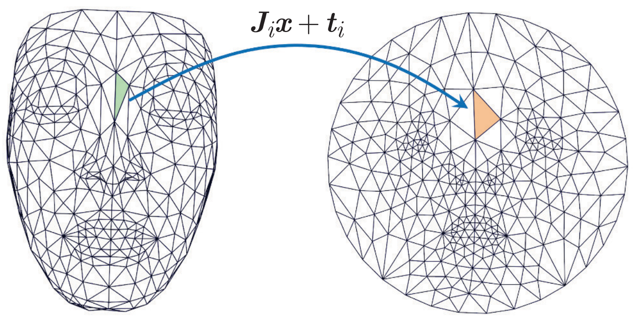


Fig. 2 Illustration for the piecewise linear mapping.

1.1 Common applications

An increasing number of computer graphics and geometry processing methods rely on inversion-free geometric mappings. Here, we first discuss three typical applications, including deformation, mesh parameterizations, and mesh quality improvement [1].

Deformation. Given the desired positions of the manipulation handles in shape deformation, we seek a new shape that satisfies the following properties:

- The shape distortion is small.
- The resulting model meets the positional constraints of the handles.
- The deformation from the input shape to the result is inversion-free.

Parameterizations. Parameterizations map 3D triangular surfaces onto the plane. By building a local coordinate frame on each triangle, the parameterization is a continuous piecewise affine map. The Jacobian matrix on each triangle is constant and is a 2×2 matrix. In addition to satisfying the inversion-free constraint, the parameterization is also required to contain small distortion. As a consequence, the texture image can be projected onto the 3D surface with small distortion. Figure 3 shows a comparison between parameterizations with and without inversion-free constraints. The inverted triangles cause significantly visual artifacts in texture mapping.

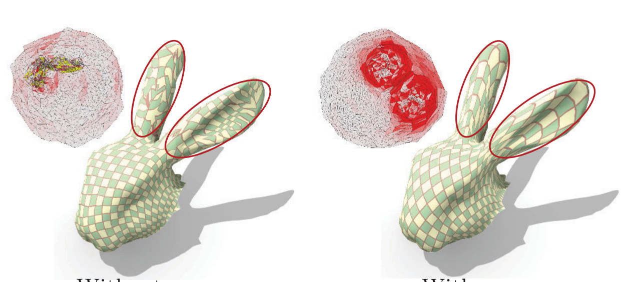


Fig. 3 Parameterizations with/without inversion-free constraints. The yellow triangles in the left image indicate the inverted triangles. The color on triangles encodes the distortion, with white being optimal.

Mesh quality improvement. To improve the accuracy in FEM simulation, the mesh elements are required to approach their ideal shapes. Since the element degeneration threatens the robustness and realism of FEM, the element should be with positive volume. Thus, the mesh quality improvement problem can also be treated as an inversion-free mapping construction problem.

1.2 General formulation

Based on the introduction of the three common problems, inversion-free geometric mappings can be constructed by a computational model that optimizes mapping objectives while satisfying inversion-free constraints.

$$\begin{aligned} \min_{\boldsymbol{u}} \quad & E(\boldsymbol{u}) \\ \text{s.t.} \quad & \det \boldsymbol{J}(\boldsymbol{x}) > 0, \forall \boldsymbol{x} \in \Omega \\ & A\boldsymbol{u} = \boldsymbol{b} \end{aligned} \tag{1}$$

where $\boldsymbol{J}(\boldsymbol{x})$ is the Jacobian matrix of f at $\boldsymbol{x}$, $\boldsymbol{u}$ indicates the vector representation of the optimization variables, $E(\boldsymbol{u})$ means the optimization objective, and $A\boldsymbol{u} = \boldsymbol{b}$ denotes the linear constraint.

For example, in shape deformation, $E(\boldsymbol{u})$ represents the shape distortion, and the linear constraint indicates the positional constraint of handles.

1.3 Overview

To provide a comprehensive overview of the recent contributions that have been made to this topic, we study problem (1) in the following aspects:

Variables. Different representations of $\boldsymbol{u}$ have various algorithm performances. The mesh vertex positions are commonly used variables in geometric mapping computation. However, a more appropriate representation can significantly improve the algorithm's efficiency and robustness for dedicated geometric process tasks. We introduce these representations in Section 2.

Objectives. Different applications have their own goals. For example, shape deformation and mesh parameterizations usually optimize the distortion from the reference mesh. In general, a lot of distortion metrics exist. Various optimization objective metrics are covered in Section 3.

Inversion-free constraints. The inversion-free constraint is nonlinear. The method of handling constraints determines the difficulty of the algorithm. More mathematical analyses of the inversion-free constraints and different approaches to realize the inversion-free goal are described in Section 4.

Methods. Many methods have been proposed to solve the problem (1) to construct inversion-free geometric mappings. If the initial geometric mappings are inversion-free, keeping the mappings always staying in the inversion-free space can theoretically guarantee the inversion-free constraint to be satisfied. Otherwise, various methods are presented to push the invalid mappings into the inversion-free space. We analyze their strengths and weaknesses in Section 5.

More constraints. The formulation (1) is not suitable for all applications. For the special applications, it should be added with some other constrains that are linear or non-linear. There are more specific constraints demanded by applications. In Section 6, we outline the bijective, bijective inter-surface, axis-aligned, and global seamless constraints. These constraints present more difficulties in constructing geometric mappings, thereby requiring the design of dedicated algorithms.

Combinatorial problems. Discrete variables or constraints may arise in geometric mapping construction problems, such as cone singularity detection, cut construction for parameterizations, and hex mesh structure simplification. These combinatorial problems are difficult to be resolved. We present these problems and their existing solutions in Section 7.

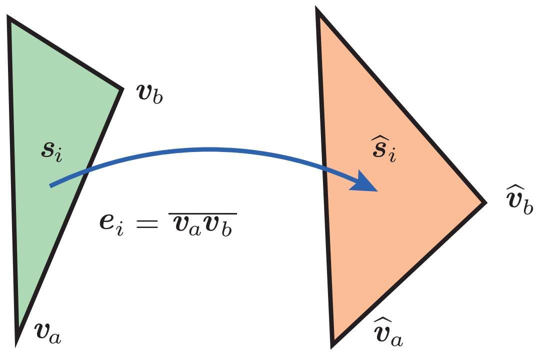
Open questions. By discussing open problems, shortcomings, and remaining questions, we conclude in Section 8.

2 Variables

Many representations of the variable $\boldsymbol{u}$ have been used for computing inversion-free mappings. Each representation has its strengths and weaknesses. Thus, given a geometric process task, an appropriate representation can significantly improve algorithm efficiency and robustness.

2.1 Mesh-based mappings

We first study mappings on simplicial meshes (2D triangular meshes or 3D tetrahedral meshes). The domain Ω of the mapping f is a d-dimensional



$\boldsymbol{v}_b$
$\boldsymbol{s}_i$
$\boldsymbol{e}_i = \overline{\boldsymbol{v}_a\boldsymbol{v}_b}$
$\boldsymbol{v}_a$
$\widehat{\boldsymbol{s}}_i$
$\widehat{\boldsymbol{v}}_b$
$\widehat{\boldsymbol{v}}_a$

Fig. 4 Illustration for the symbols.

simplicial mesh $\mathcal{M}$ containing N_{v} vertices $\{\boldsymbol{v}_i, i = 1, \cdots, N_{\text{v}}\}$, N_{e} edges $\{\boldsymbol{e}_i, i = 1, \cdots, N_{\text{e}}\}$, and N simplices $\{\boldsymbol{s}_i, i = 1, \cdots, N\}$. The mapping f is a continuous piecewise linear mapping. The Jacobian matrix of f on each simplex $\boldsymbol{s}_i$ is constant, denoted as $\boldsymbol{J}_i$. We denote the image of the mesh, vertex, edge, and simplex under the mapping f as $\widehat{\mathcal{M}}, \widehat{\boldsymbol{v}}, \widehat{\boldsymbol{e}}, \widehat{\boldsymbol{s}}$, respectively.

2.1.1 Vertex positions

Manipulating mesh vertex positions $\widehat{\boldsymbol{v}}$ is a common approach in mapping computation.

Jacobian matrices. We denote the simplex $\boldsymbol{s}_i$ with $(d+1)$ corresponding vertices as $\boldsymbol{s}_i = \triangle \boldsymbol{v}_{i,0}, \cdots, \boldsymbol{v}_{i,d}$ and its image as $\widehat{\boldsymbol{s}}_i = \triangle \widehat{\boldsymbol{v}}_{i,0}, \cdots, \widehat{\boldsymbol{v}}_{i,d}$. Then, we get a simple form of the Jacobian matrix $\boldsymbol{J}_i$:

$$\boldsymbol{J}_i = [\widehat{\boldsymbol{v}}_{i,0} - \widehat{\boldsymbol{v}}_{i,1}, \cdots, \widehat{\boldsymbol{v}}_{i,0} - \widehat{\boldsymbol{v}}_{i,d}][\boldsymbol{v}_{i,0} - \boldsymbol{v}_{i,1}, \cdots, \boldsymbol{v}_{i,0} - \boldsymbol{v}_{i,d}]^{-1}$$

The Jacobian matrix $\boldsymbol{J}_i$ is a linear function of the vertex positions $\widehat{\boldsymbol{v}}$.

Discussions. If the initial mapping is inversion-free, vertex positions are very appropriate. We can use the explicit checks in combination with line search to always satisfy the inversion-free constraint when performing mapping distortion reduction. However, the initializations with inverted simplices increase the difficulty of creating inversion-free results. As the inversion-free constraint concerning vertex positions is nonlinear and non-convex, eliminating the inverted simplices is difficult and non-trivial.

2.1.2 Jacobian matrices

Motivation. To effectively handle the inverted initializations, Jacobian matrices are used [2]. Since the Jacobian matrices become the variables, we can easily project inverted Jacobian matrices of the initial mapping into the inversion-free space. Then, the explicit checks combined with line search can keep

Jacobian matrices inside the inversion-free space during the mapping construction process.

Assembly constraints. However, the individual Jacobian matrices disassemble the input mesh into disjoint simplices. To assemble disjoint simplices, two assembly constraints should be satisfied [2]:

- *Edge assembly constraints.* For any two neighboring simplices $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$ who share an edge $\boldsymbol{e}_k$, the mapping images of any end points of $\boldsymbol{e}_k$ by the individual mappings should be the same.
- *Positional assembly constraints.* For any simplex $\boldsymbol{s}_i$ which contains positional constraints, the individual mapping should satisfy the positional constraints.

Recover $\widehat{\mathcal{M}}$. When the optimal Jacobian matrices, which satisfy the edge assembly constraints and positional assembly constraints, are achieved, the vertex positions of $\widehat{\mathcal{M}}$ can be recovered by solving the following least squares problem:

$$\min_{\widehat{\boldsymbol{v}}_1,\cdots,\widehat{\boldsymbol{v}}_n} \sum_{k=1}^{N_e} \left(\|\boldsymbol{J}_i(\boldsymbol{v}_a-\boldsymbol{v}_b)-(\widehat{\boldsymbol{v}}_a-\widehat{\boldsymbol{v}}_b)\|_2^2 + \|\boldsymbol{J}_j(\boldsymbol{v}_a-\boldsymbol{v}_b)-(\widehat{\boldsymbol{v}}_a-\widehat{\boldsymbol{v}}_b)\|_2^2\right)$$

Here the edge $\overline{\boldsymbol{v}_a\boldsymbol{v}_b}$ is adjacent to simplices $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$, and the positional constraints are fixed.

2.1.3 Angles

In addition to vertex positions, other geometric concepts can also be used to compute inversion-free mappings. Triangle angles are used to compute inversion-free parameterizations with small conformal distortion [3, 4]. For tetrahedral meshes, dihedral angles are used to compute inversion-free mappings [5].

Angle-based flattening. To reconstruct a valid parameterized mesh from angles, the following consistency conditions are required [3]:

- *Triangle consistency.* For each triangular face with angles $\widehat{\alpha}$, $\widehat{\beta}$, $\widehat{\gamma}$:
$$\widehat{\alpha}+\widehat{\beta}+\widehat{\gamma}=\pi$$
- *Vertex consistency.* For each internal vertex $\widehat{\boldsymbol{v}}$, with central angles $\widehat{\alpha}_1,\cdots,\widehat{\alpha}_n$:
$$\sum_{i=1}^{n}\widehat{\alpha}_i=2\pi$$
- *Wheel consistency.* For each internal vertex $\widehat{\boldsymbol{v}}$ with left angles $\widehat{\beta}_1,\cdots,\widehat{\beta}_n$ and right angles $\widehat{\gamma}_1,\cdots,\widehat{\gamma}_n$:
$$\prod_{i=1}^{n}\frac{\sin\widehat{\beta}_i}{\sin\widehat{\gamma}_i}=1$$

To satisfy the inversion-free constraint, the angles are required to be positive. There are two common methods to recover a parameterized mesh from the angles:

- The ABF technique [3] proposes an unfolding mechanism that constructs the parameterization coordinates for one vertex at a time in a front propagation procedure. This method suffers from error accumulation that breaks the parameterized mesh.
- The ABF++ technique [4] formulates the reconstruction problem as a global linear system and computes all the vertex coordinates simultaneously by the least-square method.

Dihedral angles of tetrahedral meshes. Dihedral angles are used to determine the shape of a tetrahedral mesh [5]. To obtain this mapping for tetrahedral meshes, three types of structural constraints and some inequalities are needed. Then the dihedral angle determination process is formulated as a constrained nonlinear optimization problem. Finally, a robust linear spectral reconstruction method, which distributes numerical errors uniformly across the mesh, is proposed to recover positions.

2.1.4 Metrics

Metric scaling, i.e., scalings of mesh edge lengths, is another good representation for conformal embedding [6–9].

Discrete conformality. According to Refs. [10, 11], two discrete metrics $\widehat{l}_i$ and l_i on $\mathcal{M}$ are conformally equivalent if the metrics are related by

$$\widehat{l}_i=\mathrm{e}^{u_a+u_b}l_i,\quad \boldsymbol{e}_i=\overline{\boldsymbol{v}_a\boldsymbol{v}_b}$$

where $u_i\in\mathbb{R}$ is the conformal factor assigned for $\boldsymbol{v}_i$. This metric is called *piecewise linear metric* (PL metric). Actually, l_i is the edge length of $\boldsymbol{e}_i$.

Conformal parameterizations via intrinsic flow. Since the parameterized mesh is planar, its curvature is zero everywhere. From Gauss's Theorema Egregium, Gaussian curvature is an intrinsic invariance of a surface determined by a metric. The conformal parameterizations can be achieved by intrinsic flows (e.g., Ricci flow, Calabi flow, Yamabe flow) that evolve the surface metric into a flat one. The final parametrization is obtained by embedding the surface of the flat metric to the 2D plane.

Recovering vertices. After the convergence of the intrinsic flows, we achieve a new length $\widehat{l}_i$ for each edge $\boldsymbol{e}_i = \overline{\boldsymbol{v}_a\boldsymbol{v}_b}$ to reconstruct a parameterized mesh. The first reconstruction method assembles the triangle one by one [12]; however, it accumulates the numerical error so that the resulting parameterized mesh breaks. To distribute the numerical errors evenly, a novel extrinsic shape optimization procedure is proposed in Ref. [8]. It optimizes the edge length to the target and minimizes the mean curvatures to zero by a local-global solver. Nevertheless, its solver cannot theoretically guarantee no inverted triangles. To this end, we propose a new optimization problem to recover vertices:

$$\min_{\widehat{\boldsymbol{v}}_1,\cdots,\widehat{\boldsymbol{v}}_n} \sum_{i=1}^{N_\mathrm{e}} d_i^\mathrm{l} + \sum_{j=1}^{N} d_j^\mathrm{b} \tag{2}$$

d_i^l measures the length difference:

$$d_i^\mathrm{l} = \frac{|\widehat{\boldsymbol{v}}_a - \widehat{\boldsymbol{v}}_b|_2^2}{\widehat{l}_i^2} + \frac{\widehat{l}_i^2}{|\widehat{\boldsymbol{v}}_a - \widehat{\boldsymbol{v}}_b|_2^2}$$

d_i^b is a triangle inequality-based barrier function [13] defined on the triangle $\widehat{\boldsymbol{s}}_i = \triangle\widehat{\boldsymbol{v}}_{i,0}, \widehat{\boldsymbol{v}}_{i,1}, \widehat{\boldsymbol{v}}_{i,2}$ to avoid inversion.

$$\begin{aligned} d_j^\mathrm{b} &= \frac{1}{|\widehat{\boldsymbol{v}}_{i,1} - \widehat{\boldsymbol{v}}_{i,2}| + |\widehat{\boldsymbol{v}}_{i,1} - \widehat{\boldsymbol{v}}_{i,3}| - |\widehat{\boldsymbol{v}}_{i,2} - \widehat{\boldsymbol{v}}_{i,3}|} \\ &= \frac{1}{|\widehat{\boldsymbol{v}}_{i,2} - \widehat{\boldsymbol{v}}_{i,1}| + |\widehat{\boldsymbol{v}}_{i,2} - \widehat{\boldsymbol{v}}_{i,3}| - |\widehat{\boldsymbol{v}}_{i,1} - \widehat{\boldsymbol{v}}_{i,3}|} \\ &= \frac{1}{|\widehat{\boldsymbol{v}}_{i,3} - \widehat{\boldsymbol{v}}_{i,1}| + |\widehat{\boldsymbol{v}}_{i,3} - \widehat{\boldsymbol{v}}_{i,2}| - |\widehat{\boldsymbol{v}}_{i,1} - \widehat{\boldsymbol{v}}_{i,2}|} \end{aligned}$$

Starting from an inversion-free parameterization, solving Eq. (2) can achieve a desired result.

2.2 Meshless mappings

In general, the mesh-based mapping is C^0 and lacks high-order smoothness. To this end, we then study meshless mappings to achieve high-order smoothness (Fig. 5).

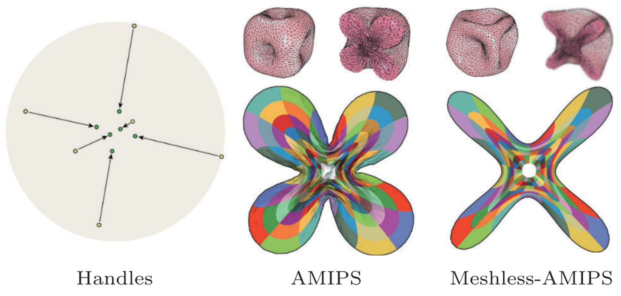


Fig. 5 3D meshless deformation of a spherical tet mesh using AMIPS. Deformed meshes and their cut-views are shown. The meshless deformation generates smooth results. Reproduced with permission from Ref. [14], © ACM 2015.

Mapping representation. A meshless mapping f is usually defined as

$$f(\boldsymbol{x}) = \sum_{j=1}^{m} \boldsymbol{c}_j B_j(\boldsymbol{x})$$

where $\mathcal{B} = \{B_j\}_{j=1}^m$ is a set of basis functions and the coefficients of basis functions $\boldsymbol{c} = [\boldsymbol{c}_1, \cdots, \boldsymbol{c}_m]$ are unknowns. For example, the uniform cubic tensor product B-splines and RBF basis functions can be used as the basis functions.

Jacobian matrices. Then, the Jacobian matrix of f at $\boldsymbol{x}$ has the form:

$$\boldsymbol{J}_{\boldsymbol{x}} = \sum_{j=1}^{m} \boldsymbol{c}_j \nabla_{\boldsymbol{x}} B_j(\boldsymbol{x})$$

Based on Ref. [15], a meshless mapping is considered to be inversion-free if the mapping is inversion-free on dense sampling points. Without loss of generality, we denote the sampling points as $\{\boldsymbol{p}_i \in \Omega, i = 1, ..., N\}$ and the $\boldsymbol{J}_{\boldsymbol{p}_i}$ as $\boldsymbol{J}_i$. $\boldsymbol{J}_i$ is also a linear function of the unknown coefficients $\boldsymbol{c}$.

Discussions. In shape deformation [15–19], the rest pose indicates an identity map that is inversion-free and contains the least distortion. Then, after the handles are moved to the desired positions, we optimize $\boldsymbol{c}$ to reduce shape distortion while satisfying the inversion-free constraints. The size of $\boldsymbol{c}$ is usually small enough to enable real-time interaction.

In isogeometric analysis [20], domain parameterizations are generated by mapping parametric domains (generally unit cubes) to computational domains. Usually, the basis functions of the parameterizations are formulated as spline functions. Inversion-free parameterizations are required to improve the subsequent accuracy and computational robustness for solving partial differential equations. However, the initial parameterizations often contain inverted regions, so the challenges are to eliminate them [21, 22].

Smooth mappings are used to seamlessly transform the floor plan of a large virtual scene into a small physical space for real walking in virtual reality [23–25]. The mapping is required to be inversion-free for avoiding visual artifacts and be with low isometric distortion for keeping the real sence of walking.

3 Objectives

The distortion of the input domain Ω under the mapping f is expected to be as small as possible and often treated as the optimization objective. To measure the distortion, Jacobian matrices are commonly used in the variable representations of vertex positions and Jacobian matrices. Other representations have their own methods to define the distortion objectives.

3.1 Jacobian matrix-based energies

Except for three commonly used distortion metrics, i.e., (1) conformal distortion, (2) area-preserving distortion, and (3) isometric distortion, other types of distortions still exist. These three types of distortion energies are formulated by the singular values of the Jacobian matrices.

Signed singular value decomposition. The singular value decomposition (SVD) of $\boldsymbol{J}_i$ is denoted as

$$\boldsymbol{J}_i = \boldsymbol{U}_i \boldsymbol{S}_i \boldsymbol{V}_i^{\mathrm{T}}$$

where $\boldsymbol{U}_i$ and $\boldsymbol{V}_i$ are the orthogonal matrices, and $\boldsymbol{S}_i = \mathrm{diag}(\sigma_{i,1}, ..., \sigma_{i,d})$ is a diagonal matrix with singular values on the diagonal. Without loss of generality, we assume $\sigma_{i,j} \geqslant \sigma_{i,k}, \forall 1 \leqslant j < k \leqslant d$. To define inversion-free constraints and mapping distortions, signed singular value decomposition (SSVD) [26] is introduced. If $\det \boldsymbol{J}_i < 0$, $\boldsymbol{U}_i$ and $\boldsymbol{V}_i$ are modified to be rotation matrices, and the smallest singular value $\sigma_{i,3}$ becomes negative; otherwise, SSVD is the same as SVD. Then, the squared Frobenius matrix norm $\|\boldsymbol{J}_i\|_{\mathrm{F}}^2$ of $\boldsymbol{J}_i$ is equal to $\sum_{j=1}^d \sigma_{i,j}^2$ and $\det \boldsymbol{J}_i = \prod_{j=1}^d \sigma_{i,j}$.

Conformal distortion metrics. Conformal distortion energies measure the deviation of the Jacobian matrices $\boldsymbol{J}_i$ from similar transformations. When $\sigma_{i,1} = \sigma_{i,d}$, the energy reaches the optimal.

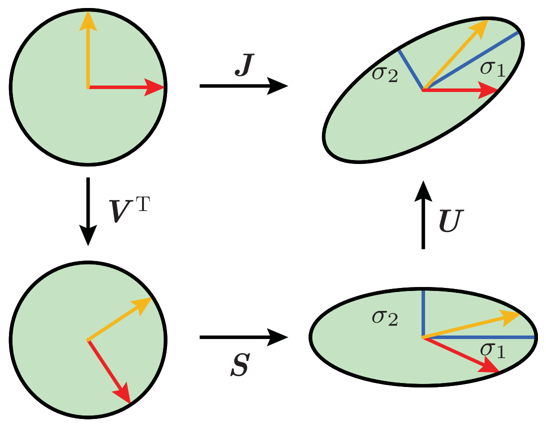


Fig. 6 Illustration for the SVD.

Commonly used energies are proposed in literature:

- *Conformal distortion* [27]: $\sigma_{i,1}/\sigma_{i,d}$
- *MIPS energy* [28]:
$$\frac{\sum_{j=1}^d \sigma_{i,j}^2}{d(\prod_{j=1}^d \sigma_{i,j})^{2/d}}$$
- *As-similar-as-possible energy* [29, 30]:
$$\sum_{j\neq k} (\sigma_{i,j} - \sigma_{i,k})^2$$

For 3D MIPS energy, Fu et al. [16] propose a different formulation:

$$\frac{1}{8}\left(\frac{\sigma_{i,1}}{\sigma_{i,2}} + \frac{\sigma_{i,2}}{\sigma_{i,1}}\right)\left(\frac{\sigma_{i,2}}{\sigma_{i,3}} + \frac{\sigma_{i,3}}{\sigma_{i,2}}\right)\left(\frac{\sigma_{i,3}}{\sigma_{i,1}} + \frac{\sigma_{i,1}}{\sigma_{i,3}}\right)$$

Area distortion metrics. Preserving area in mapping construction is important. As we know, the determinant of the Jacobian matrix indicates the ratio between the original volume and the mapped volume. An area-preserving mapping requires the determinant to be 1. There are three common approaches to compute the difference from 1:

- *Area distortion*: $\max\{\prod_{j=1}^d \sigma_{i,j}, \frac{1}{\prod_{j=1}^d \sigma_{i,j}}\}$
- *Ratio form* [16]: $\prod_{j=1}^d \sigma_{i,j} + \frac{1}{\prod_{j=1}^d \sigma_{i,j}}$
- *Difference form*: $(\prod_{j=1}^d \sigma_{i,j} - 1)^2$

The ratio form penalizes degenerate simplices since it goes to infinity when $\det \boldsymbol{J}_i$ approaches to zero. Thus, the ratio form is more popular than the difference form.

Isometric distortion metrics. A mapping is isometric if and only if it is both conformal and area-preserving. Thus, when $\sigma_{i,1} = \sigma_{i,d} = 1$, the isometric energy reaches the optimal.

- *Isometric distortion*: $\max\{\sigma_{i,1}, \frac{1}{\sigma_{i,d}}\}$
- *Symmetric dirichlet energy* [31, 32]:
$$\sum_{j=1}^d (\sigma_{i,j}^2 + \sigma_{i,j}^{-2})$$
- *AMIPS energy* [16]:
$$\frac{1}{d}\left(\frac{\sum_{j=1}^d \sigma_{i,j}^2}{(\prod_{j=1}^d \sigma_{i,j})^{2/d}}\right) + \frac{1}{2}\left(\prod_{j=1}^d \sigma_{i,j} + \frac{1}{\prod_{j=1}^d \sigma_{i,j}}\right)$$
- *As-rigid-as-possible energy* [30]:
$$\sum_{j=1}^d (\sigma_{i,j} - 1)^2$$

The AMIPS energy linearly combines the MIPS energy and the ratio form of the area distortion metric.

Other energy metrics. There are many other energy metrics:

- *Dirichlet energy*: $\sum_{j=1}^{d} \sigma_{i,j}^2$
- *Green–Lagrange energy*: $\sum_{j=1}^{d} (\sigma_{i,j}^2 - 1)^2$
- *Hencky strain energy*: $\| \log \boldsymbol{J}_i^{\mathrm{T}} \boldsymbol{J}_i \|_{\mathrm{F}}^2$
- *The difference from a given mapping* [26]: $\|\boldsymbol{J}_i - \boldsymbol{J}_i^{\mathrm{init}}\|_{\mathrm{F}}^2$, where $\boldsymbol{J}_i^{\mathrm{init}}$ is the Jacobian matrix of the initial mapping.

For a mesh or all sampling points, the energy should be added up over all elements.

3.2 Energies without Jacobian matrices

For the angle-based and metric-based representations, the distortion energy is usually defined as the difference from the ideal reference.

Angle-based flattening. As reported in Ref. [3], the energy function is simply:

$$\sum_{\boldsymbol{s}_i} \sum_{j=1}^{3} \frac{1}{\omega_{i,j}} (\widehat{\alpha}_{i,j} - \alpha_{i,j}^{\star})^2$$

where $\widehat{\alpha}_{i,j}$ are the unknown planar angles, and $\alpha_{i,j}^{\star}$ are the optimal angles. The weights $\omega_{i,j}$ are set to $(\alpha_{i,j}^{\star})^{-2}$ to reflect relative rather than absolute angular distortion. In general, $\alpha_{i,j}^{\star}$ is computed as follows [3, 4]:

$$\alpha_{i,j}^{\star} = \begin{cases} \alpha_{i,j}^0 \frac{2\pi}{\sum_{k=1}^{n} \alpha_{i,k}^0}, & \text{around an interior vertex} \\ \alpha_{i,j}^0, & \text{around a boundary vertex} \end{cases}$$

where $\alpha_{i,j}^0$ is the angle in the input mesh. As shown in Ref. [5], the objective function for dihedral angles in the tetrahedral mesh is similarly defined.

Metric-based flattening. The intrinsic flows for conformal parameterizations output a metric to agree with the input Gaussian curvature. For example, the Calabi energy is squared difference between current Gaussian curvature vector and target Gaussian curvature:

$$\sum_{i=1}^{N_{\mathrm{v}}} (K(\widehat{\boldsymbol{v}}_i) - K_i^{\mathrm{t}})^2$$

where $K(\widehat{\boldsymbol{v}}_i)$ is the Gaussian curvature at $\widehat{\boldsymbol{v}}_i$ and K_i^{t} is the target Gaussian curvature at $\widehat{\boldsymbol{v}}_i$. The Calabi energy can be minimized by the Calabi flow [12].

4 Inversion-free constraints

4.1 Relationship with volume

Here we study the mappings on simplicial meshes and remind that no zero or negative volume exists in the real world. For a transformed simplex $\widehat{\boldsymbol{s}}_i = \triangle \widehat{\boldsymbol{v}}_{i,0}, \cdots, \widehat{\boldsymbol{v}}_{i,d}$, its signed volume is computed as a determinant:

$$\frac{1}{d!} |\widehat{\boldsymbol{v}}_{i,1} - \widehat{\boldsymbol{v}}_{i,0}, \cdots, \widehat{\boldsymbol{v}}_{i,d} - \widehat{\boldsymbol{v}}_{i,0}|$$

Note that the signed volume may be negative. Thus, the inversion-free constraint requires that the sign of the signed volume before and after the transformation is unchanged. The signed volume is a polynomial with degree d and is non-convex.

4.2 Relationship with singular values

Here we study the inversion-free constraints using the Jacobian matrices.

Determinant and conformal distortion. If $\det \boldsymbol{J}_i > 0, i = 1, \cdots, N$, the map f is inversion-free. From the view of SSVD, the inversion-free constraint requires the smallest singular value of each Jacobian matrix to be positive. If the conformal distortion $\tau(\boldsymbol{J}_i) = \sigma_{i,1}/\sigma_{i,d} < 0, \exists i \in [1, N]$, the map f has negative $\sigma_{i,d}$. Thus, inversion-free property requires $\tau(\boldsymbol{J}_i) \geqslant 1, i = 1, \cdots, N$.

Bounded conformal distortion. Bounded conformal distortion mappings require:

$$1 \leqslant \tau(\boldsymbol{J}_i) \leqslant k_i, \quad i = 1, \cdots, N$$

Here, k_i denotes the upper bound of the conformal distortion $\tau(\boldsymbol{J}_i)$. Since $\sigma_{i,1}$ is always positive and $\sigma_{i,1} \geqslant |\sigma_{i,d}|$, the bounded conformal distortion constraint $1 \leqslant \tau(\boldsymbol{J}_i) \leqslant k_i$ is equivalent to require $\sigma_{i,1} \leqslant k_i \sigma_{i,d}$. If $\sigma_{i,1} \leqslant k_i \sigma_{i,d}$, it means that $\sigma_{i,d} > 0$ indicating the mapping is inversion-free. If $\det \boldsymbol{J}_i > 0$, it is trivial to choose k_i that makes the constraint $\tau(\boldsymbol{J}_i) \leqslant k_i$ hold, for example, $k_i = \tau(\boldsymbol{J}_i)$. Accordingly, inversion-free constraints can be converted to bounded conformal distortion constraints.

More analyses for 2D case. Similar to Ref. [33], we rewrite the 2×2 Jacobian matrix $\boldsymbol{J}_i$ as

$$\begin{bmatrix} a_i + c_i & d_i - b_i \\ d_i + b_i & a_i - c_i \end{bmatrix}$$

Then, we have analytical expressions for the two singular values:

$$\sigma_{i,1} = \sqrt{a_i^2 + b_i^2} + \sqrt{c_i^2 + d_i^2}$$

$$\sigma_{i,2} = \sqrt{a_i^2 + b_i^2} - \sqrt{c_i^2 + d_i^2}$$

Then inversion-free condition can be rewritten as

$$\sqrt{a_i^2 + b_i^2} > \sqrt{c_i^2 + d_i^2} \tag{3}$$

The bounded conformal distortion constraint is similarly reformulated as [33]:

$$\frac{k_i - 1}{k_i + 1} \sqrt{a_i^2 + b_i^2} > \sqrt{c_i^2 + d_i^2} \tag{4}$$

These two constraints are nonlinear and non-convex. For 3D case, the condition formulations are more complicated due to the complex forms of roots of a cubic equation.

5 Methods

In this section, we focus on the recent works closely related to generating inversion-free geometric mappings. Three main families of methods have been proposed to deal with the challenging inversion-free constraint: maintenance-based methods for inversion-free initializations, elimination of foldovers for inverted initializations, and expanding the feasible region by connectivity-updated methods.

5.1 Inversion-free initializations

If the initial geometric mappings are inversion-free, keeping the mappings always staying in the inversion-free space can theoretically guarantee the inversion-free constraint to be satisfied. A few methods use maintenance-based methods to optimize "barrier"-type energies, in which the objective function includes terms that grow asymptotically as an element becomes degenerate.

5.1.1 Pipeline

Starting from inversion-free initializations $\boldsymbol{x}_0$, maintenance-based methods minimize objective functions which avoid inverted elements. The optimization approach is very simple and is described in Algorithm 1.

Algorithm 1 Maintenance-based methods

```
Input: inversion-free initialization $\boldsymbol{x}_0$;
Initialize: Set iteration number $n = 0$;
while not converged do
    Compute descent direction $\boldsymbol{p}_n$;
    Find max step size $\alpha_{\max}$;
    Perform line search to find step size $\alpha$;
    $\boldsymbol{x}_{n+1} \leftarrow \boldsymbol{x}_n + \alpha\, \boldsymbol{p}_n$;
    $n \leftarrow n + 1$;
end while
```

It is an iterative algorithm producing a sequence of approximations $\boldsymbol{x}_n$ to the optimal point $\boldsymbol{x}^\star$. There are three intermediate steps in each iteration: computing descent direction ($\boldsymbol{p}_n$), finding max step size ($\alpha_{\max}$), and performing a line search to find step size (α) starting from $\alpha_{\max}$.

Generating initializations. Tutte's embedding [34] is guaranteed to create bijective mappings under the minimal assumptions that both domains are simply connected and the target planar domain is convex. Since Tutte's embedding guarantees inversion-free, it has achieved great success in the field of mesh parameterizations [32, 35–37]. Although several works extended it [38, 39] to other specific classes of mappings, its essential limitations remain: it can only map injectively to a prescribed convex boundary, without any interior constraints.

Furthermore, it is very challenging to compute inversion-free initializations in 3D. For example, a tetrahedral mesh can be bijectively mapped to a cube or a ball [40]; but it cannot be used for general boundary shapes. For tetrahedral mesh deformation, they use the meshes in the rest-pose as initializations and treat the handle positions as soft constraints.

Barrier functions. Barrier functions diverge to infinity when elements become degenerate, thus inhibiting inversion. Ref. [41] used the log of the determinant as a barrier term and Ref. [42] followed a similar path by solving a sequence of convex programs. Instead of using an auxiliary injectivity barrier, several methods directly optimize distortion metrics that explode near degeneracies, such as the MIPS energy, the AMIPS energy, and the symmetric Dirichlet energy in Section 3.

Descent directions. These non-linear energies are difficult to minimize. Existing optimization algorithms typically produce a sequence of approximations, $\boldsymbol{x}_n$, designed to converge to an optimal point $\boldsymbol{x}^\star$. To this end, most approaches use a local quadratic approximation of the objective function, or *proxy*:

$$\begin{aligned} E_n(\boldsymbol{x}) =& E(\boldsymbol{x}_n) + (\boldsymbol{x} - \boldsymbol{x}_n)^{\mathrm{T}} \nabla E(\boldsymbol{x}_n) \\ &+ \frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}_n)^{\mathrm{T}} \boldsymbol{H}_n (\boldsymbol{x} - \boldsymbol{x}_n) \end{aligned}$$

where $E(\boldsymbol{x})$ is the objective function and $\boldsymbol{H}_n$ is a symmetric matrix, named the *proxy matrix*. Thus, $E_n(\boldsymbol{x})$ is an osculating convex quadric approximation to E at $\boldsymbol{x}_n$ and its minimization determines the next approximation $\boldsymbol{x}_{n+1}$. From this point of view, the difference between the various methods lies in the choice of $E_n(\boldsymbol{x})$, or more precisely, the choice of its proxy matrix $\boldsymbol{H}$. Broadly, existing methods for the local energy approximation fall into three rough categories that vary in the construction of proxy matrix $\boldsymbol{H}$.

- *First-order methods (Section 5.1.2)*: the methods use only first derivatives and do not directly use second order derivatives of the energy;
- *Quasi-Newton methods (Section 5.1.3)*: the methods iteratively update $\boldsymbol{H}$ to approximate second derivatives using just differences in gradients;
- *Newton-type methods (Section 5.1.4)*: the methods exploit expensive second-order derivative information.

Line search. Consider a non-degenerate 2D triangle with vertices $\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3$ with corresponding search direction vectors $\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3$. The triangle becomes degenerate when its signed area becomes zero [32]:

$$\det\begin{pmatrix} (\boldsymbol{u}_2+\boldsymbol{v}_2 t)-(\boldsymbol{u}_1+\boldsymbol{v}_1 t) \\ (\boldsymbol{u}_3+\boldsymbol{v}_3 t)-(\boldsymbol{u}_1+\boldsymbol{v}_1 t) \end{pmatrix} = 0 \tag{5}$$

Fortunately, Eq. (5) is quadratic in t and the max step sizes are simply given by the roots of this quadratic. Given that we are only concerned with searches in the positive direction, the smallest positive root gives the max step size for this triangle. The max step size $t_{\max}$ for the line search is given by computing the minimum parameter over all triangles [32]. For a tetrahedron, it is also easy to generalize by replacing the signed area with the signed volume.

Termination conditions. The iteration continues until we are able to stop with a "good enough" solution, but the termination requires a precise computational definition. The common termination conditions are:

- The gradient is small $\|\nabla E\| < \epsilon$, for a specified tolerance $\epsilon > 0$;
- A fixed number of iterations [43];
- The absolute or relative error in energy $\|E_{n+1}–E_n\|$ and/or position $\|\boldsymbol{x}_{n+1} - \boldsymbol{x}_n\|$ are small [37, 44].

However, an appropriate value of ϵ for a given application is highly dependent on the other conditions, such as the mesh and the energy. To provide reassuring termination criteria in practice, the Blended Cured Quasi-Newton (BCQN) method develops a gradient-based stopping criterion [45]. The proposed termination condition remains consistent for optimization problems even as we vary scale, mesh resolution and energy type:

$$\|\nabla E\| < \epsilon\langle \boldsymbol{W}\rangle\|\boldsymbol{l}(T)\| \tag{6}$$

where $\langle \boldsymbol{W}\rangle$ is the 2-norm of a matrix related to $\boldsymbol{W}(\cdot)$ and $\boldsymbol{l}(T)$ ia a vector. The specific definitions can reference Section 6 in Ref. [45]. The gradient-based stopping criterion allows users to set a default convergence tolerance ϵ in the solver once and leave it unchanged, independent of scale, mesh, and energy.

5.1.2 *First-order methods*

Block coordinate descent solvers. The Block Coordinate Descent (BCD) method is a popular optimization tool suitable for solving large-scale problems. Considering the optimization problem:

$$\min_{\boldsymbol{x}} E(B_1, \cdots, B_l, \cdots, B_m)$$

where E is the objective energy and the variable $\boldsymbol{x}$ is partitioned into m blocks $\{B_i, i = 1, \cdots, m\}$. In each iteration, for every l, $l \in \{1, \cdots, m\}$, a subproblem is solved by treating the block B_l as the free variables of the optimization problem while keeping the rest variables fixed.

The BCD method is categorized into two types: exact BCD and inexact BCD. The MIPS energy is locally convex with respect to each vertex around its 1-ring region. The standard MIPS algorithm [46] employs the exact BCD where each vertex forms a block of variables and the Newton method is adopted to solve each subproblem exactly. However, solving the subproblem exactly is usually time-consuming, and seeking an approximate solution is a common way to accelerate the algorithm. The inexact BCD method is employed to optimize the AMIPS energy by applying only one step of gradient descent [16]. The experiment demonstrates that exact BCD is easily trapped in local minimum early while inexact BCD always yields lower energy. Note that the BCD method can be accelerated using parallel technology by partitioning the variables into blocks where any two variables in the same block are independent.

AQP. Given the current iteration $\boldsymbol{x}_n$, the Accelerated Quadratic Proxy (AQP) method computed the next iteration $\boldsymbol{x}_{n+1}$ by an intermediate guess $\boldsymbol{y}_{n+1}$ [47]. AQP used an affine combination of current iteration $\boldsymbol{x}_n$ and previous iteration $\boldsymbol{x}_{n-1}$ with a constant $\theta > 0$ to produce $\boldsymbol{y}_{n+1}$, namely:

$$\boldsymbol{y}_n = (1+\theta)\boldsymbol{x}_{n-1} - \theta\boldsymbol{x}_{n-2}$$

An optimal choice θ leads to an optimal convergence rate, which is proved by Lemma 2 in Ref. [47]. Then, AQP uses a quadratic polynomial proxy, whose Hessian is taken to be the Laplacian, to compute a descent direction $\boldsymbol{p}_n$.

AQP utilizes the common structure of optimization problems over meshes to improve iteration efficiency and incorporate acceleration in an almost universal way (i.e., insensitive to different energy types and mesh sizes). However, AQP does not have a principled way of determining how effective the Laplacian approximation for Hessian of arbitrary energy. Besides, the optimal choice θ requires the condition number of a matrix, which is challenging to obtain.

SLIM. The local/global method is used in Ref. [30] to minimize the ARAP energy:

$$\boldsymbol{D}_{\mathrm{ARAP}}(\boldsymbol{J}) = \|\boldsymbol{J} - \boldsymbol{R}\|_{\mathrm{F}}^2$$

where $\boldsymbol{R}$ is the closest rotation to $\boldsymbol{J}$ in the Frobenius norm and $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm. The local/global algorithm iteratively alternates between a local step and a global step. In the local step, each element is individually perfectly mapped (without any distortion), and in the global step, a linear system is solved to stitch all elements back together.

The Scalable Locally Injective Mappings (SLIM) method extends the local–global strategy to a wide range of distortion energies [43]. It uses the local/global paradigm and enriches it with a reweighting scheme to efficiently minimize nonlinear and flip-preventing energies. The proxy functions is

$$\mathcal{P}_{\boldsymbol{W}}(\boldsymbol{J}) = \|\boldsymbol{W}(\boldsymbol{J} - \boldsymbol{R})\|_{\mathrm{F}}^2$$

where $\boldsymbol{W}$ is the weighted matrix.

SLIM is a scalable approach for optimizing flip-preventing energies in the general context of simplicial mappings. The central theoretical limitation and advantage of SLIM are both inherited from the local/global method. The algorithm is high-speed while approaching a local minimum, but it requires many iterations to converge to a numerical minimum. Besides, the proxy energy definition only works for the rotation invariant distortion energies.

AKVF. The Approximate Killing Vector Fields (AKVF) method formulates a new preconditioner specifically designed for parameterization problems, using the language of vector field design [36]. The Killing operator $\boldsymbol{K}(\boldsymbol{x})$ measures the deviation of a vector field on $\boldsymbol{x}$ from being a rigid motion, and AKVF applies the Moore–Penrose pseudoinverse $\boldsymbol{K}(\boldsymbol{x})^+$ of Killing vector field operator $\boldsymbol{K}(\boldsymbol{x})$ as the proxy matrix. Then the descent direction $-\nabla_{\boldsymbol{x}} E(\boldsymbol{x})$ is transformed into an approximately rigid motion $-\boldsymbol{K}(\boldsymbol{x})^+ \nabla_{\boldsymbol{x}} E(\boldsymbol{x})$ by the proxy matrix $\boldsymbol{K}(\boldsymbol{x})^+$ when possible.

For planar case and volumetric case, $\boldsymbol{K}(\boldsymbol{x})$ can be computed as Eq. (6) and Eq. (10) in Ref. [36], respectively.

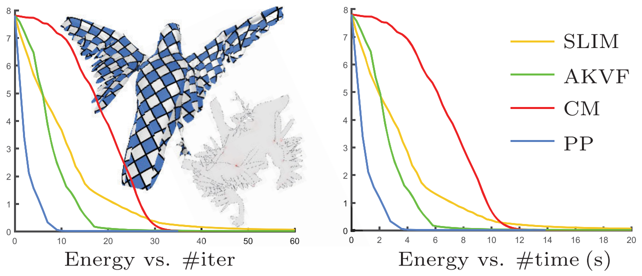


Fig. 7 Comparison for four competing methods, including CM [37], PP [35], AKVF [36], and SLIM [43]. Reproduced with permission from Ref. [35], © ACM 2018.

SLIM and AKAP converge faster than AQP. However, they require re-assembly and factorization of their proxies for each iteration. Besides, they do not match the convergence quality of the second-order methods, such as CM and PN.

5.1.3 Quasi-Newton method

L-BFGS. L-BFGS directly approximates the inverse of the Hessian, requiring only the position and gradient information of a few previous iterations. While L-BFGS iterations are fast, they typically require many iterations to converge. L-BFGS convergence can be improved with the choice of a preconditioner, such as the diagonal of the Hessian [48], application-specific structure [49], or even the Laplacian [50]. However, so far, for distortion optimization problems, L-BFGS has consistently and surprisingly failed to perform competitively irrespective of the choice of preconditioner [43, 47]. Moreover, Ref. [48] points out that the secant approximation can implicitly create a dense proxy, unlike the sparse true Hessian, directly and incorrectly coupling distant vertices.

BCQN. For the aforementioned issue of a dense proxy incorrectly coupling distant vertices in L-BFGS, the Laplacian provides the correct structure for the proxy essentially. It only directly couples neighboring elements in the mesh and is well-behaved initially when far from the solution. However, the Laplacian is constant, and thus it ignores valuable local curvature information, thereby leading to prohibitively slow convergence.

Fortunately, the L-BFGS offers superlinear convergence near solutions, and Ref. [45] develops a new quasi-Newton method, which adaptively blends gradient information with the matrix Laplacian at each iteration. Then, it can regain improved and robust convergence with efficient per-iterated storage and computation across scales while avoiding the current pitfalls of L-BFGS methods.

5.1.4 Second-order methods

Overview. Second-order methods generally can achieve the most rapid convergence but require the costly assembly, factorization, and backsolve of new linear systems per step. At each iteration, second-order methods use the energy Hessian, $\nabla^2 E$, to form a proxy matrix $\boldsymbol{H}$. This works well for convex energies, but it requires modification for non-convex energies [51] to ensure that $\boldsymbol{H}$ is at least positive semi-definite (PSD).

A general solution is to add small multiples of the identity and project the Hessian to the PSD cone, but this generally damps convergence too much [37, 51]. The global Hessian matrix of the objective function is constructed from the element Hessian matrix, which is based on locally individual elements (triangle or tetrahedra). As long as the Hessian matrices of all elements are PSD, then the global Hessian matrix is PSD. Thus, most second-order methods locally modify the element Hessian matrices, whose dimensions are far lower than the global Hessian matrix.

Locally modifying Hessian matrices. Projected Newton (PN) does eigendecomposition on per-element Hessian and clamps all negative eigenvalues to zero, to project per-element Hessian to the PSD [52]. PN is an effective and general purpose for 2D and 3D problems. However, PN introduces a significant computational overhead on eigendecomposition and becomes computationally prohibitive.

The Composite Majorization (CM) method provides an analytic formula to modify element Hessian [37]. Composite majorization, a tight convex majorizer, was recently proposed as an analytic PSD approximation of the Hessian. CM method is concerned with objective functions that can be represented as the composition of simpler functions for which convex–concave decompositions are known.

$$f(\boldsymbol{x}) = h(\boldsymbol{g}(\boldsymbol{x})) = h(g_1(\boldsymbol{x}), \cdots, g_k(\boldsymbol{x}))$$

where $h : \mathbb{R}^k \rightarrow \mathbb{R}$ and $g_j(\boldsymbol{x}) : \mathbb{R}^n \rightarrow \mathbb{R}$ are C^2 functions with convex–concave decompositions. That is, they each decompose as

$$h = h^+ + h^-, g_j = g_j^+ + g_j^-$$

with h^+ and g_j^+ convex and, respectively, h^- and g_j^- concave.

CM's strategy for picking a convex osculating quadric at x_n is based on: (i) exploiting the composite structure for constructing a convex majorizer to f centered at x_n, and (ii) computing its Hessian at x_n. The majorizer provides a tight convex upper bound to f and therefore provides a well justified choice of a PSD proxy matrix $\boldsymbol{H}$ at x_n:

$$\begin{aligned}\boldsymbol{H} =& \frac{\partial[\boldsymbol{g}]^{\mathrm{T}}}{\partial x}\nabla^2 h^+ \frac{\partial[\boldsymbol{g}]}{\partial x} \\ &+ \sum_j \left(\frac{\partial h}{\partial u_j}\right)_+ \nabla^2 g_j^+ + \sum_j \left(\frac{\partial h}{\partial u_j}\right)_- \nabla^2 g_j^-\end{aligned}$$

where $(\cdot)_+$ keeps only positive numbers (linear rectifier), $(\cdot)_-$ only negative numbers. CM is efficient and is even better relative performance improvement over PN. However, it is limited to two-dimensional problems.

The KP-Newton method [53] has applied the complex view to the piecewise linear mapping. It shows that simple analytic expressions of the Hessian are obtained, which allows simple and close to optimal analytic PSD projection. Based on the complex view of the linear mapping, KP-Newton speed-ups the numerical projection for PN by reducing the matrix size (reducing the full 6×6 projection to the 4×4 case).

CM needs to construct a convex–concave decomposition of the objective function. The choice of this decomposition is not unique and is likely to result in different PSD matrices and consequently affects the convergence behavior. In contrast, KP-Newton only requires the partial derivatives of a simple-to-obtain energy formulation. Additionally, KP-Newton has the property that the element Hessians are not modified if they are already PSD, which is not necessarily the case for CM. However, KP-Newton is also limited to two-dimensional problems.

Analytic Eigensystem (AE) provides compact expressions to optimize problems both in 2D and 3D, and does not introduce spurious degeneracies [54]. At its core, AE utilizes the invariants of the stretch tensor $\boldsymbol{S}$ that arises from the polar decomposition of the deformation gradient $\boldsymbol{J} = \boldsymbol{R}\boldsymbol{S}$:

$$I_1 = \mathrm{tr}(\boldsymbol{S}) = \sum_j \sigma_j$$

$$I_2 = \|\boldsymbol{S}\|^2 = \sum_j \sigma_j^2$$

$$I_3 = \det(\boldsymbol{S}) = \prod_j \sigma_j$$

The majority of distortion energies used in geometry optimization are isotropic and can be expressed in terms of invariants, such as, the ARAP energy $D_{\text{ARAP}}(\boldsymbol{J}) = \sum_{j=1}^{d}(\sigma_j - 1)^2 = I_2 - 2I_1 + d$. AE provides closed-form expressions for the eigensystems for all these invariants, and uses them to systematically derive the eigensystems of any isotropic energy. Then these systems can be used to project energy Hessian to PSD analytically.

Different from the aforementioned methods, PP [35] observes that when the distortion between each parameterized triangle and its corresponding reference triangle is below a threshold K, only a few iterations are needed to reach a result that is comparable with the convergent one. Based on this key observation, PP iteratively updates the optimization objective by constructing the new reference triangles, which makes distortion between the new reference and the current parameterizations bounded. Combined with a hybrid solver, PP outperforms the competitors.

5.2 Inverted initializations

5.2.1 Quasi-conformal mappings

A quasi-conformal mappings (QC mapping) is an extension of conformal mapping. For conformal mappings, there is no angular distortion. For QC mappings, the angular distortion is bounded and is introduced by the Beltrami coefficient or Beltrami differentials.

QC mappings of plane domains. For conformal mapping f from the complex plane $\mathbb{C}$ to $\mathbb{C}$, the Cauchy–Riemann equation $\frac{\partial f}{\partial \bar{z}} = 0$ is satisfied. Correspondingly, f is called a QC mapping, if f satisfies the following Beltrami equation:

$$\frac{\partial f}{\partial \bar{z}} = \mu(z)\frac{\partial f}{\partial z}$$

Here, μ is called the Beltrami coefficient of f. In this case, the Jacobbi of f : $J(f) = |\frac{\partial f}{\partial z}|^2(1 - |\mu|^2)$. Thus the Beltrami coefficient $||\mu||_\infty < 1$ must hold for f to be orientation-preserving.

QC mapping between Riemann surfaces. QC mapping can also be defined on Riemann surfaces. For two surfaces S_1, S_2 embedded in $\mathbb{R}^3$, let $\phi_1 : U \subset S_1 \to \mathbb{C}$, $\phi_2 : V \subset S_2 \to \mathbb{C}$ be two local conformal parameterization, $\phi_1(U)$ or $\phi_2(V)$ forms the isothermal coordinate chart of S_1 or S_2. Then $f : S_1 \to S_2$ is quasi-conformal if for any ϕ_1, ϕ_2,

$$f_{12} = \phi_2 \circ f \circ \phi_1^{-1} : \phi_1(U) \to \phi_2(V)$$

is quasi-conformal. Instead of Beltrami coefficient, the Beltrami differential $\mu\frac{\mathrm{d}\bar{z}}{\mathrm{d}z}$ is used, which is kept unchanged under different coordinate charts. According to Teichmüller theory, there is a one-to-one correspondence between the set of Beltrami differentials and the set of QC surface mappings under normalization conditions.

Applications. QC mapping has been widely used in computer graphics, such as parameterization, deformation, and shape registration [55–58]. The research focuses on two main areas:

- How to calculate QC mapping under boundary and landmark constraints?
- How to find a QC map that satisfies the described Beltrami coefficients?

Boundary and landmark constraints. The most popular method is to optimize the angular distortion energy instead, when there is no restriction on Beltrami coefficients or Beltrami differentials. If the QC mapping is specified to Teichmüller mapping with uniform conformality distortion over the whole domain, Ref. [58] locally projects Beltrami coefficient μ into the one with constant norm after computing a global harmonic mapping and iterated until convergence. An alternating-descent algorithm is proposed in Ref. [57] to minimize the difference error of the Beltrami equation efficiently, although there is no theoretical guarantee to reach the global minimum.

Described Beltrami coefficients. There are different algorithms to compute QC maps on planar domains [59–61]. For arbitrary Riemann surfaces, Refs. [62] and [63] establish a discrete Beltrami flow to evolve an identity map to the desired QC mapping. An auxiliary metric is proposed in Ref. [64], and the original QC mapping becomes conformal under the auxiliary metric. Then, the desired QC mapping can be obtained by using the conformal mapping method.

5.2.2 Bounded distortion mappings

Bounded distortion mapping methods [15, 18, 19, 26, 33, 65] tried to bound the distortion of the mappings. Different from the aforementioned quasi-conformal mappings, these methods study the mappings from the discrete view. Namely, the bounded distortion constraint is enforced on each Jacobian matrix. Then, a constrained optimization problem with non-convex constraints is achieved. To this end, some elegant methods are proposed. Based on the strategies processing the bounded distortion constraints, these methods can be classified into two categories: (1) extracting convex subspace and (2) linearizing the constraints.

Maximal convex subspace. For triangular meshes, the bounded distortion constraint (4) and inversion-free constraint (3) are nonlinear and non-convex. By introducing a new variable $r_i \in \mathbb{R}$, they can be simplified as [33]:

$$\sqrt{c_i^2 + d_i^2} \leqslant r_i \frac{k_i - 1}{k_i + 1}$$
$$\sqrt{a_i^2 + b_i^2} \geqslant r_i$$
$$r_i > 0$$

Then, the maximal convex subspace can be achieved as follows (see more details in Ref. [33])

$$\sqrt{c_i^2 + d_i^2} \leqslant r_i \frac{k_i - 1}{k_i + 1}$$
$$a_i \cos\theta_i + b_i \sin\theta_i \geqslant r_i, \theta_i \in [0, 2\pi)$$
$$r_i > 0$$

θ_i is a parameter and is adaptively adjusted during the optimization. Then, a convex problem is built and can be solved effectively. The linear matrix inequality [65] is used to extend this idea to the tetrahedral meshes.

Quadratic programming. For the SSVD $\boldsymbol{J}_i = \boldsymbol{U}_i \boldsymbol{S}_i \boldsymbol{V}_i^{\mathrm{T}}$, if $\boldsymbol{U}_i$ and $\boldsymbol{V}_i$ are known, then the singular values are linear functions with respect to the vertex positions. Based on this fact, Aigerman and Lipman [26] use $\boldsymbol{U}_i$ and $\boldsymbol{V}_i$ in the last iteration as the estimator in the current iteration. Then, the non-convex constrained problem becomes a quadratic programming problem that can be effectively solved. By iteratively performing these two steps, the method [26] usually converges within a small number of iterations. This method works for both 2D and 3D.

Discussions. Bounded distortion mapping methods ensure no inversion if the resulting mappings fall into the bounded distortion space; however, setting an appropriate distortion bound remains an open problem.

5.2.3 Projection-based methods

Motivation. Our goal is to project the inverted initializations into the inversion-free mapping space. As mentioned before, inversion-free constraints can be converted to bounded conformal distortion constraints. In practice, we can try to minimize the distance from the mapping to the bounded conformal distortion mapping space. Then, the optimization problem can be formulated as

$$\begin{aligned} \min_{\mu} \quad & \sum_{i=1}^{N} \|\boldsymbol{J}_i - \boldsymbol{H}_i\|_{\mathrm{F}}^2 \\ \text{s.t.} \quad & \boldsymbol{H}_i \in \mathbf{H}_i, \quad i = 1, \cdots, N \\ & A\boldsymbol{u} = \boldsymbol{b} \end{aligned} \tag{7}$$

Here, $\mathbf{H}_i = \{\boldsymbol{H}_i | 1 \leqslant \tau(\boldsymbol{H}_i) \leqslant k_i\}$ denotes the bounded conformal distortion space with bound k_i. k_i is a variable in the optimization. Next, we first fix it and then discuss the updating cases.

Algorithms. Given distortion bounds k_i, it is difficult to solve the problem (7) due to the nonlinear bounded distortion constraints $\boldsymbol{H}_i \in \mathbf{H}_i$. Thus the projection-based methods decouple the bounded distortion constraints from the problem (7) and devise an alternating pipeline. Generally, the methods can be classified based on different projection approaches: (1) tangential projection and (2) closest point projection.

Closest point projection. The most common projection method is the closest point projection [66]. Given fixed $\boldsymbol{J}_i$, we want to solve $\boldsymbol{H}_i$. $\boldsymbol{H}_i$ is separated, and we compute it one by one through solving the following problem (local step):

$$\begin{aligned} \min_{H_i} \quad & \|\boldsymbol{J}_i - \boldsymbol{H}_i\|_{\mathrm{F}}^2 \\ \text{s.t.} \quad & \boldsymbol{H}_i \in \mathbf{H}_i \end{aligned}$$

This problem has a closed-form solution [67]. Then, given fixed $\boldsymbol{H}_i$, we solve $\boldsymbol{\mu}$ as follows (global step):

$$\begin{aligned} \min_{\mu} \quad & E_d = \sum_{i=1}^{N} \|\boldsymbol{J}_i - \boldsymbol{H}_i\|_{\mathrm{F}}^2 \\ \text{s.t.} \quad & A\boldsymbol{\mu} = \boldsymbol{b} \end{aligned}$$

This problem can also be easily solved. Although the local–global method monotonically decreases the objective function, it converges slowly [68]. Then, the Anderson acceleration method [68] is used for acceleration.

Tangential projection. In the global step, the tangential projection method [67] restricts the Jacobian matrix to belong to a single hyperplane locally supporting $\boldsymbol{H}_i$ at the closest point projection. However this method may oscillate due to an inappropriate k_i; as a result, the distance from the mapping to the bounded distortion space may not consistently decrease. Thus, it works poorly in practice.

Updating bounds. Su et al. [66] devise a simple method to update the distortion bound. It gradually increases the distortion bound after one pass of the accelerated local–global solver converges. However, it has no theoretical guarantee of success for any model.

5.2.4 Area-based methods

As observed by Ref. [69], the Total Unsigned Area (TUA) is an upper bound to the sum of signed areas, which is constant for a fixed boundary, and equal if and only if the triangulation is injective. When the triangular mesh is inversion-free, it minimizes the sum of the unsigned triangle areas among all the triangulations of the given boundary. However, as Ref. [70] points out, directly minimizing TUA suffers from three deficiencies: (1) the triangulation containing degenerate elements is a global minimum of TUA but a non-injective embedding; (2) derivative discontinuity: TUA is not C^1 continuous when a vertex moves across the supporting line of its opposite edge; (3) vanishing gradient: TUA has zero gradients with respect to any vertex surrounded by a ring of consistently oriented elements. Based on those observations, Ref. [70] proposes a novel energy form, called Total Lifted Content (TLC), that lifts the simplices of the mesh into a higher dimension and then measures their contents:

$$\mathrm{TLC}_{\tilde{\boldsymbol{s}},\alpha}(\boldsymbol{s}) = \frac{1}{d!}\sqrt{\det(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X} + \alpha\tilde{\boldsymbol{X}}^{\mathrm{T}}\tilde{\boldsymbol{X}})}$$

where $\boldsymbol{s}$ is a d-dimensional simplex, $\tilde{\boldsymbol{s}}$ is the auxiliary simplex, and $\boldsymbol{X}$ ($\tilde{\boldsymbol{X}}$) is a $d \times d$ matrix whose column vectors are edge vectors of the simplex $\boldsymbol{s}$ ($\tilde{\boldsymbol{s}}$). TLC reduces to TUA when parameter $\alpha = 0$. TLC is smooth over the entire space and has only injective global minima for sufficiently small values of α. This simple energy can be efficiently minimized using quasi-Newton or projected-Newton solver.

5.2.5 Penalty-based methods

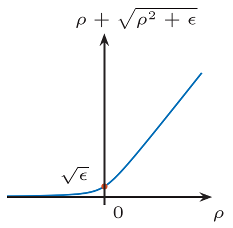
A simple idea for inversion elimination is to devise a penalty function having two main properties:

- it is very large to penalize the inverted Jacobian matrices;
- it is very small to accept inversion-free Jacobian matrices.

After designing a suitable penalty function, optimization solvers are then the challenges. This idea has been applied to many untangling problems [71, 72].

Here we discuss a popular penalty function as follows [72]:

$$E_{\text{penalty}} = \sum_{i=1}^{N} \frac{\|\boldsymbol{J}_i\|_{\mathrm{F}}^{d}}{\det \boldsymbol{J}_i + \sqrt{(\det \boldsymbol{J}_i)^2 + \epsilon}}$$



where ϵ is a small positive number that makes E_{penalty} very large when inversion exists (see the right inset). This penalty function works for both 2D and 3D domains. The key to this penalty function is the setting of ϵ. In Ref. [72], a detailed setting method is provided.

Then, two common methods for optimizing E_{penalty} are proposed.

- *Block coordinate descent method* [72] updates one block of variables each time.
- *Monotone preconditioned conjugate gradient method* [73] monotonically and efficiently reduces the objective function.

The block coordinate descent method is a local method. When the number of inverted elements is very large, it may be struggled and trapped by the local minimum. In the monotone preconditioned conjugate gradient method, the linear systems for computing descent directions have a fixed left-hand side; thus, it is pre-factorized once during the preprocessing, thereby making the solver efficient. This solver can eliminate most inverted elements, but its result often contains a small set of inverted elements. Thus, a practical solver can be devised as a hybrid one that first uses the monotone preconditioned conjugate gradient method and then uses the block coordinate descent method.

5.2.6 Representation-based methods

Simplex assembly [2]. Using the Jacobian matrices as the variables, the problem (1) is converted to a non-constrained optimization problem:

$$\min_{\boldsymbol{J}_1,\cdots,\boldsymbol{J}_N} \quad \mu\, E_{\mathrm{m}} + E_{\mathrm{c}} + \lambda\, E_{\text{assembly}}$$

where E_{m} is the mapping energy, E_{c} is a barrier function to keep each Jacobian matrix inside the feasible space, and E_{assembly} is the summation of squares of all the left sides of the two assembly constraints. Given an inverted initialization, it projects the Jacobian matrices associated with each simplex into the inversion-free and distortion-bounded space. The projected Newton's method is used to solve the optimization problem. λ is adaptively adjusted to enforce E_{assembly} to approach zero.

Angle-based methods. For ABF-based conformal parameterizations, three solvers are proposed to solve the constrained problem:

- *ABF* [3] uses Newton's method to solve an augmented objective function that formulates the constrained minimization problem using Lagrange multipliers.
- *ABF++* [4] uses the sequential linearly constrained programming.
- *Linear ABF* [74] linearizes the non-linear constraint and solves a linear system to obtain the resulting angles.

In practice, inverted triangles still arise, as demonstrated in Ref. [33].

Metric-based methods. Surface parameterization can be formulated as designing a Riemannian metric of the surface, such that all the interior points are with zero Gaussian curvatures, namely a flat metric. Discrete intrinsic flows were studied in recent decades. These methods evolve the curvature of the triangular meshes [75], or the piecewise linear metric of triangular meshes independent of embedding or immersion, such as discrete Ricci flow [76], Yamabe flow [11], and Calabi flow [77]. Based on these methods, powerful tools for conformal parameterization has been developed [78, 79].

5.3 Connectivity-updated methods

For highly non-linear optimization problems, the fixed connectivity may impose a strong restriction on the solution. As a consequence, the feasible region may be too small to contain an ideal solution. This leads to slow convergence, poor solution, or even that no solution can be found because of the nearly degenerated triangles generated during the iterations [80]. Thus, some methods are proposed to integrate connectivity-update into vertex optimization to solve this issue. Here, we introduce two connectivity-update techniques: adaptive refinement and hierarchical meshes.

Adaptive refinement. Ref. [80] proposes a connectivity-updated optimization method for locally injective mappings of 2D triangular meshes with position constraints. Their algorithm iteratively solves the vertex position and updates the connectivity according to the criteria based on residual, gradient and condition number of the energy. The connectivity-updated operators include edge-flip and edge-split. Ref. [81] applies adaptive refinement to the 3D deformation problem.

Hierarchical meshes. Ref. [82] focuses on computing high-quality spherical parameterizations with bijection and low isometric distortion. The method first simplifies the mesh until the model becomes a tetrahedron. After mapping the tetrahedron onto the sphere, the method alternately inserts vertices and do global distortion optimization to distribute the vertices uniformly on the sphere. Inspired by the similar idea, the progressive embedding is proposed in Ref. [83] with similar theoretical guarantees to Tutte's embedding, but it is more resilient to the rounding error of floating point arithmetic. Ref. [83] collapses edges on an invalid embedding to a valid, simplified mesh, and then inserts points back while maintaining validity.

6 More constraints

Inversion-free constraint is not the only one constraint in many applications. This section introduces the applications with other four constraints: (1) bijective mappings, (2) bijective inter-surface mappings, (3) axis-aligned structure construction, and (4) global seamless parameterizations.

6.1 Bijective constraints

In addition to being inversion-free, applications may ask for intersection-free boundaries [13, 32, 84, 85]. An inversion-free and intersection-free mapping is bijective. For example, bijective parameterizations can establish a one-to-one correspondence between the input surface and the parameterized mesh (Fig. 8). In fact, except for the negative or zero volume, physical objects also do not contain global overlaps. Thus the physical deformation/simulation should avoid intersecting boundaries and only contain

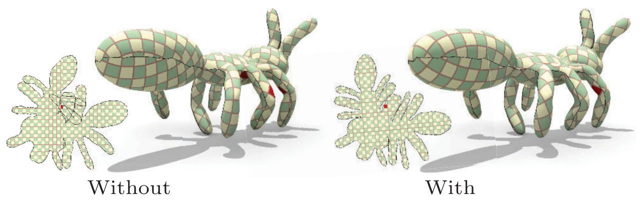


Fig. 8 Parameterizations with/without bijective constraints. Without bijective constraints, a 2D point may be mapped to more than one point on the surface.

positive volume. Here, we focus on the mesh-based mappings.

Constraint overview. This intersection-free constraint is more complicated than the inversion-free constraint. Preventing overlaps for the boundary leads to non-linear collision constraints. Besides, boundary collisions may occur everywhere on the boundary. Thus for a simplicial mesh, the number of potential collisions is quadratic in the number of boundary elements, thereby significantly increasing the computational cost.

There are two common strategies to handle the bijective constraints: (1) barrier functions and (2) scaffold meshes. Both of these approaches start from an intersection-free shape and avoid any overlap during the optimization process. For example, Tutte's embedding method [34] generates a bijective initial parameterization, and the rest shape in deformation is usually free of overlaps.

6.1.1 Barrier functions

Using barrier functions to avoid overlaps is a commonly used technique. Barrier functions for intersection-free constraints should satisfy a property: when the overlap is about to occur, the function goes to infinity. Thus, we need to answer the following questions: (1) how to use mathematical language to describe the occurrence of collisions and (2) what the concrete barrier function is?

Distance-based approach. When a boundary vertex approaches a boundary element (edge in 2D and triangle in 3D), the collision is about to occur. In 3D, when two boundary edges are close to each other, they will collide. For the first question, the distance from a boundary vertex to a boundary element or the distance between two boundary edges is used, denoted as d_{inter} [32, 85].

Triangle inequality approach. For 2D triangular mesh, Su et al. [13] propose a triangle inequality approach. A boundary vertex $\widehat{\boldsymbol{v}}$ and two end points of a boundary edge $\widehat{\boldsymbol{e}} = \overline{\widehat{\boldsymbol{v}}_a\widehat{\boldsymbol{v}}_b}$ form a triangle. Based on the triangle inequality, we have

$$\|\widehat{\boldsymbol{v}} - \widehat{\boldsymbol{v}}_a\|_2 + \|\widehat{\boldsymbol{v}} - \widehat{\boldsymbol{v}}_b\|_2 \geqslant \|\widehat{\boldsymbol{v}}_a - \widehat{\boldsymbol{v}}_b\|_2$$

The equality holds when $\widehat{\boldsymbol{v}}$ is on $\widehat{\boldsymbol{e}}$. Thus, $d_{\text{inter}} := (\|\widehat{\boldsymbol{v}} - \widehat{\boldsymbol{v}}_a\|_2 + \|\widehat{\boldsymbol{v}} - \widehat{\boldsymbol{v}}_b\|_2 - \|\widehat{\boldsymbol{v}}_a - \widehat{\boldsymbol{v}}_b\|_2)$ is used to answer the first question.

Concrete barrier functions. Given a distance threshold ϵ_{inter}, the barrier function is zero when $d_{\text{inter}} \geqslant \epsilon_{\text{inter}}$. Two barrier functions are commonly used when $d_{\text{inter}} < \epsilon_{\text{inter}}$:

- *Reciprocal-based barrier* [32]: $(\epsilon_{\text{inter}}/d_{\text{inter}} - 1)^2$
- *Log-based barrier* [85]: $-\ln(d_{\text{inter}}/\epsilon_{\text{inter}})(\epsilon_{\text{inter}} - d_{\text{inter}})^2$

They go to infinity when d_{inter} approaches zero.

Computational cost. The barrier functions are at least C^2 when $d_{\text{inter}} < \epsilon_{\text{inter}}$. Thus quasi-Newton solvers [32] and second-order solvers [13, 85] can be used. However, the number of potential collisions is quadratic in the number of boundary elements, and thus the density of the Hessian matrix in second-order solvers significantly increases, thereby causing much more time for optimization. For 2D triangular mesh, a coarse shell mesh is used [13] to reduce the computational cost; however, it extends this idea to 3D case.

6.1.2 Scaffold-based methods

Another idea to avoid overlaps is the use of a scaffold mesh. The scaffold mesh is introduced to convert the globally overlap-free constraint to a locally flip-free condition [84, 86–88].

Updating connectivity. During the optimization, the boundary of the scaffold mesh is fixed. To efficiently reduce distortion and prevent possible locking situations, the scaffold mesh must be frequently updated and optimized during the optimization [84]. This updating connectivity leads to a changed size and an updated nonzero structure of the sparse Hessian matrices for computing descent directions. Then, solving linear systems become more time-consuming, as observed by Ref. [13]. In addition, efficiently performing connectivity updates for tetrahedral meshes is difficult.

Very-large-scale bijective parameterizations. As high-precision 3D scanners become more and more widespread, it is easy to obtain very-large-scale meshes containing at least millions of vertices. However, due to the memory limitation of the used computer, the commonly developed methods for creating inversion-free mappings may fail for these models. Ye et al. [89] use the scaffold-based method to compute bijective parameterizations for very-large-scale models. Instead of computing descent directions using the mesh vertices as variables, they estimate descent directions for each vertex by optimizing a proxy energy defined in spline spaces. Since the spline functions contain a small set of control points, it significantly decreases memory requirement.

6.2 Bijective inter-surface mappings

Computing inter-surface mappings is a hot research topic [90–92]. Here, we focus on bijective inter-surface mappings that provide one-to-one correspondences between two shapes. Besides, inter-surface mappings can be used to generate compatible meshes that possess the same connectivity structures [73, 93, 94].

Common domain-based methods. Many approaches compute bijective inter-surface mappings via common domains, such as spheres [82, 95], coarse triangular meshes [31, 93, 96], and planar domains [97–99]. The algorithm workflow usually contains three steps: (1) constructing a common domain, (2) bijectively mapping the input models onto the common domain, and (3) determining the inter-surface mapping by composing one mapping with the inverse of the other.

Domain construction. Spheres are standard domains and only suitable for genus-zero shapes. In general, mapping the input shapes onto spheres (i.e., spherical parameterizations) contains very large distortion, and thus the resulting inter-surface mappings may be distorted severely [82, 100, 101]. In addition, robustly computing bijective spherical parameterizations without numerical issues still deserves more research.

Constructing coarse triangular meshes is non-trivial for arbitrary inputs. For example, progressive meshes [102] are used to define the base domain [31]. In Refs. [93, 96], common domains are built by consistently connecting feature points with equivalent paths over the two meshes.

The common planar domain is automatically obtained by computing bijective parameterizations with common boundary constraints [97–99]. The parameterizations require the two input meshes to be cut to disk topology. Thus, consistent cuts on two meshes are needed. However, it is difficult to construct them that will always lead to low distortion inter-surface mappings.

Distortions optimization. Since the inter-surface mapping is computed by composing one mapping with the inverse of the other, it is difficult to reduce the distortion. When two mappings are with low distortion, the final inter-surface mapping has a high probability of being low distortion; however, this is not absolute. Then, an end-to-end method is proposed to reduce the distortion of the final inter-surface mapping [31, 103, 104]. They represent the inter-surface mapping via a mutual tessellation and optimize the symmetric Dirichlet energy.

6.3 Axis-aligned constraints

If the boundary of a closed shape is axis-aligned, it is an *axis-aligned structure*. Axis-aligned structures (PolyCubes in 3D and PolySquares in 2D) provide compact representations for closed complex shapes. They have been proved to be very useful to many computer graphics applications, such as texture mapping [105–107], hex/quad meshing [108–116] (Fig. 9), GPU-based subdivision [117], and atlas refinement [118, 119].

Constraints. Generally, closed complex shapes are not axis-aligned. Thus, the goal of the axis-aligned structure construction method is to automatically and efficiently compute an axis-aligned structure. In our view, a high-quality construction algorithm usually satisfies the following properties:

- *Inversion-free constraint*: the axis-aligned structure contains no degenerate or inverted elements;
- *Distortion constraint*: the mapping distortion is as low as possible;
- *Corner constraint*: the number of corners of the axis-aligned structure is small.

Since the rest shape serves as the initialization, the initial mapping is an identity map. Thus, we can keep the axis-aligned map inversion-free by performing explicit checks combined with line search. Then, the left challenge is to strictly satisfy the axis-aligned constraint while reducing as many corners as possible.

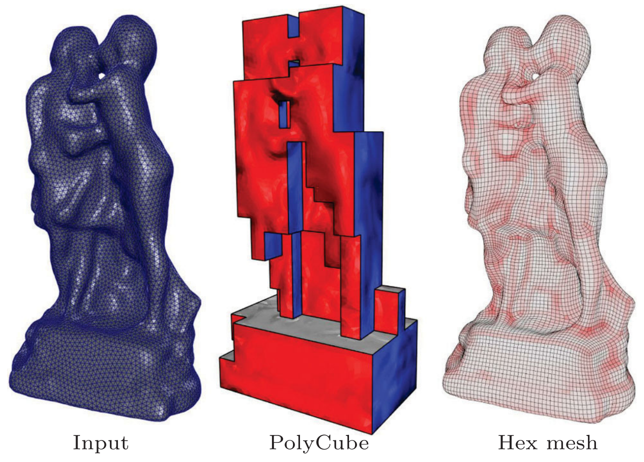


Fig. 9 PolyCubes for all-hex remeshing. It contains three steps: (1) constructing a PolyCube, (2) performing hex mesh generation of the PolyCube domain, and (3) mapping the hex mesh back to the input model.

Deformation-based methods. The deformation-based methods [108, 109, 111] contain three main steps:

- *Pre-axis-aligned deformation*: it deforms the input closed mesh to a pre-axis-aligned shape, whose face normals are almost aligned with the coordinate axes;
- *Boundary segmentation*: it determines whether the boundary surface is sufficient to form a valid axis-aligned structure [120];
- *Boundary flattening*: it maps the input to be strictly axis-aligned.

Many axis-aligned energy terms are proposed and optimized to drive the input shape to be pre-axis-aligned. There are three common strategies:

- *Rotation-driven strategy* [109]: it computes deformation gradients as the minimal rotation necessary to align each surface vertex normal with one of $\pm X, \pm Y, \pm Z$, and then uses the computed deformation gradients to deform the shape.
- *L_1-based energy* [111]: if normals are along axes, their L_1 norms reach the optimal.
- *Normal-smooth energy* [108]: it first computes target normals by Gaussian smoothing and closest axis projection, and then measures the difference between the current normals and the target normals as the objective energy.

In practice, high-quality results are usually achieved. However, these deformation-based methods have no theoretical guarantee that the valid axis-aligned topology can always be achieved under the inversion-free constraints.

Segmentation-based method. This method [115] first segments the input shape with valid axis-aligned topology and then deforms the input to be strictly axis-aligned. For the first step, a graph-cut based approach is proposed to control the corner counts. However, it contains two main limitations: (1) their algorithm could be time-consuming due to its local and greedy search and (2) their method cannot always achieve valid axis-aligned topology, as demonstrated in Ref. [121].

Construction-based methods. Given a closed mesh and a pre-axis-aligned shape, the construction-based methods first construct a valid axis-aligned structure and then compute a bijective correspondence between the constructed structure and the input mesh [110, 114, 121]. The pre-axis-aligned shape can be generated by the aforementioned deformation methods [108, 109, 111].

The goal of axis-aligned structure construction is to reduce the number of corners and generate low distortion mappings. However, axis-aligned structures are not determined during the construction process, and the distortion of the final mapping cannot be computed. Therefore, the distortion metric should be replaced with the approximation error between the pre-axis-aligned shape and the constructed axis-aligned structure. Morphological operations [110] and an erasing-and-filling solver [121] are proposed for construction. To build a bijective correspondence between the axis-aligned structure and the input mesh, Yang et al. [121] use a quad mesh optimization algorithm.

These construction-based methods can theoretically guarantee a valid axis-aligned structure. They have two main limitations: (1) they are unable to handle the pre-axis-aligned shapes containing global overlaps and (2) they do not adequately align the sharp features of the models.

Sharp features. Aligning the sharp features of the input models to the edges of the axis-aligned structures is non-trivial. To align most of sharp features, Guo et al. [116] use a feature-aware energy into the aforementioned deformation processes. However, strictly preserving sharp features remains a challenge.

6.4 Global seamless parameterizations

Problems. The global seamless parametrization is widely used in some specific applications, such as conforming quadrangulation and seamless texturing. For the seamless mapping $f : \mathcal{M} \to \Omega$, two kinds of constraints should be satisfied. The one is the inversion-free constraint in Eq. (1), the other one is the seamless constraints of the parametrization [122]:

$$\begin{pmatrix} u \\ v \end{pmatrix}_i = \boldsymbol{R}_{ij} \begin{pmatrix} u \\ v \end{pmatrix}_j + \boldsymbol{t}_{ij} \tag{8}$$

where $\boldsymbol{t}_{ij} \in \mathbb{R}^2$, and $(u,v)_i, (u,v)_j$ are the parameterization positions of any point on the the edge $\boldsymbol{e}_{ij}$ adjacent to simplices $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$ respectively, and

$$\boldsymbol{R}_{ij} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}^{k_{ij}}$$

is a rotation matrix with a seam rotation angle $w_{ij} = k_{ij}\pi/2, k_{ij} \in \mathbb{Z}$. It can also be written as follow [2, 123]:

$$\boldsymbol{J}_i \boldsymbol{e}_{ij} = \boldsymbol{R}_{ij} \boldsymbol{J}_j \boldsymbol{e}_{ij} \tag{9}$$

where $\boldsymbol{J}_i, \boldsymbol{J}_j$ are the Jacobian matrix mentioned in Section 2, $\boldsymbol{e}_{ij}$ is the edge adjacent to simplices $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$. Figure 10 shows an example with and without the seamless constraint (8).

There are mainly three kinds of methods, which are metric-based, field-based, and harmonic-based, to generate global seamless parameterizations.

Metric-based methods. The Jacobian matrix $\boldsymbol{J}_i$ can also be regarded as the metric of surfaces, so the direct way to get the seamless mappings is to construct an optimization problem with the constraint (9) [124].

However, there are many other methods with different representations of the metric. Based on the notion of the PL metric in Section 2.1.4, a precise notion of discrete conformal equivalence is presented in Ref. [6]. The parametrization is generated by finding a flat mesh that is discretely conformally equivalent to a given mesh. The problem is convex, and the seamless condition is transformed into the angle defect condition on the vertices. Different from Ref. [6], another conformal method, called BFF (boundary first flattening), is presented in Ref. [125]. The method is based on the Cauchy–Riemann equation, and the final parametrization is obtained by a linear system, so it is computed in real time. The seamless condition is also transformed into the cone condition.

Recently, some methods firstly cut the surface to topological disk(s), and then modify the cone metric so that the parametrization with the modified metric is seamless. The fact is demonstrated in Ref. [126] that, for (almost) any choice of cones, a corresponding global parametrization can be constructed without introducing additional cones. Based on this fact, their algorithm firstly cuts the surface to topological disk(s), then computes a cone metric on with prescribed boundary curvature, and the boundary is rectilinear. With map padding, the metric is modified into a seamless one. However, the nonlinear optimization convergence is not sure to the prescribed singularities, and there are some numerical issues such as precision limit, which affect the discrete conformal map computation. The method in Ref. [127] is a general combinatorial method, which eliminates the potential numerical issues in Ref. [126]. Similar to Ref. [126], the surface is cut firstly. Then the metapolygon will be constructed and modified to satisfy the seamless condition of each piece. The construction of the cone metric is explicit combinatorial, and numerical optimization is taken into account only for non-crucial decisions. Finally, the parametrization over the cut surface can then be obtained by existing techniques. Their method is reliable to generate the validity, seamlessness, and local injectivity parametrization with the expense of more time cost on the process of padding.

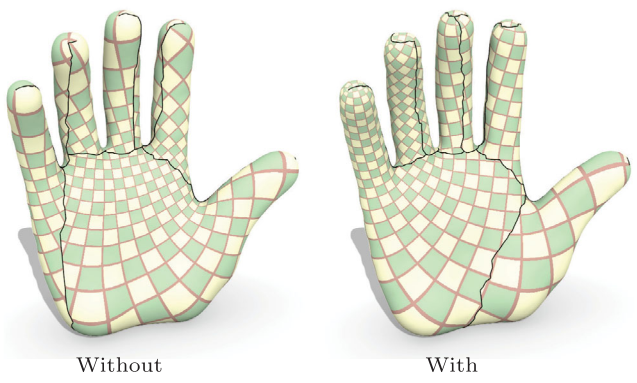


Fig. 10 Global seamless parameterizations.

Most of the metric-based methods are based on conformal mapping so that these methods may be with large area distortion. Recent popular distortion metrics are considered in Ref. [128] to achieve low metric distortion directly.

Field-based methods. The field-based methods are often computing the guiding field firstly. Then the parametrizations are from the field. The first field-based approach in Ref. [129] is also based on the conformal map. Their method computes seamless parametrizations of nonzero genus surfaces with boundaries. Since all conformal gradient fields (holomorphic 1-forms) form a linear space, the gradient field of the mapping can be got by solving a linear system with some constraints on the field. However, the final parametrization is not guaranteed to be injective, and the conformal mapping will also bring a large area distortion.

The methods based on two-direction field are more common than one-direction field. Based on cross-fields [130] and conformal map ideas [131], Ref. [132] proposes a feature-aligned method to reduce the metric distortion of parametrization. The seamless cone metric describes the seamless condition. In Ref. [122], a quad patch partition of the mesh is constructed by tracing the cross-field, and then the partition is modified to satisfy the global parametrization constraints, including seamless. With the partition, the problem to find a final parametrization is reduced to linear programs, i.e., an optimization problem with convex constraints, so the existence of a solution is always guaranteed. This method enforces a local bijective and feature-aligned, but singularities should be added during the modification of partition. These methods allow the feature-alignment, but the generation of the fields is also a complex problem.

In Ref. [133], the author presents a novel method to perform the quantization that satisfies the seamless condition. The quantization is performed efficiently by formulating the problem in alternative degrees of freedom. Ref. [131] describes a method to produce seamless parametrizations with low distortion. They prove that the parametrization f with a cone metric g is seamless if and only if the metric is also seamless, so the seamless condition is transformed into the seamless condition of cone metric g. Then, by evolving the surface's metric and finding a new metric g with zero Gaussian curvature almost everywhere, the method produces low-distortion, locally injective parametrization for surfaces of arbitrary topology, but the intrinsic method does not allow for feature alignment.

Harmonic-based methods. Given desired cone points and rational holonomy angles, Ref. [134] proposes a method, which called HGP (harmonic global parametrization), to compute seamless parametrization of surfaces with arbitrary topology. It is stated that if the cone and boundary triangles are positively oriented and achieve the correct cone and turning angles, the final parametrization is locally injective. By this result, the parametrization can be generated by solving the linear system, and the seamless condition is converted into the linear complex equations. In Ref. [135], an algorithm based on Ref. [134] is presented for low-distortion locally injective harmonic mappings. They construct a linear subspace from the solutions of the HGP system [134]. Then, the mapping is obtained by a nonlinear non-convex optimization from the reduced subspace. Their method achieves significant acceleration over HGP. The above two methods are fast and robust, but the local injectivity through convexification [134] will exclude the valid solutions. Also, Ref. [135] can only deal with the surfaces with genus zero.

7 Combinatorial problems

7.1 Cone singularity detection

Conformal parameterizations are easily computed. The main advantage of conformal parameterizations is free of angle distortion and inversions. However, conformal parameterizations suffer from severe area distortion (Fig. 11). Cone singularities [136] provide a way to mitigate area distortion.

Problem overview. In fact, the area distortion can always be reduced by adding more cones; however, too many cones usually result in a long cut for final parametrization. Thus, the goal of the cone singularity detection algorithm is to achieve a desired tradeoff between cone number, cone position, and the area distortion. The number and placement of cones are discrete, and thus this problem is *combinatorial*. Therefore, computing the best configuration (number, placement, and size) of cones is notoriously difficult. Many methods have been proposed to solve this challenging problem [6, 7, 131, 137].

Greedy methods. Cones are detected via a simple greedy algorithm [6]. In each iteration, it iteratively computes a conformal parameterization and places a new cone at the point with the greatest area distortion. The subsequent iterations treat cone points as punctures in the domain and can automatically determine cone angles by the conformal parameterization process. In Ref. [7],

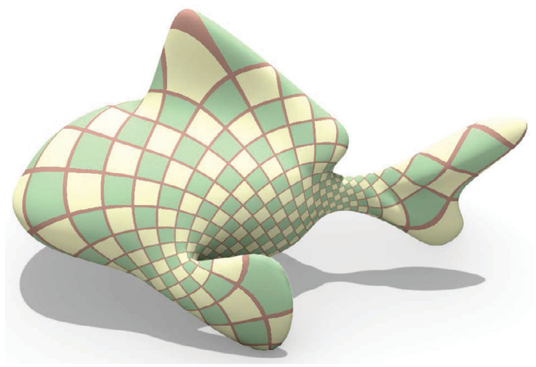


Fig. 11 Conformal parameterizations with large area distortion.

a parameterization algorithm is devised and cone locations are determined by the same greedy strategy. Different from Ref. [6], it develops a diffusion process involving Gaussian curvature to compute the cone angles.

Incremental methods. Cones are determined by incrementally flattening the surface [131]. Starting with the original metric, a fraction of the surface is incrementally constrained to have zero Gaussian curvature. Then, only a small set of vertices, i.e., cones, have non-zero curvature. However, there is no direct or explicit relationship between curvatures and cone configurations, as shown in Ref. [137].

Optimization methods. Below a fixed total cone angle bound, the method in Ref. [137] computes the cone configuration with the least total area distortion. However, the bound is not explicitly given in the optimization, whereas it is implicitly controlled by a weight for balancing its two energy terms. There is no intuitive nor direct mapping between the controlling parameter and the total cone angle bound. Judging from the results in Ref. [137], some important cones are not captured with default parameters, leading to high area distortion.

7.2 Cut construction for parameterizations

Parameterized 2D meshes are commonly used to store surface signals, such as colors, normals, and displacements. Before being parameterized to the plane, a closed mesh needs to be cut to a disk topology. The feasibility and practicality of parameterizations are affected by two major factors: (1) distortion and (2) cut length. *Short cuts* and *low isometric distortion* are both required for high-quality inversion-free parameterizations. Usually, these two requirements are contradictory (Fig. 12). Besides, cut generation can be used for more applications, such as peeling art design [138].

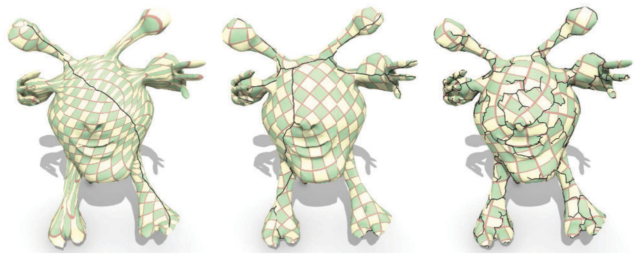


Fig. 12 Cut construction for parameterizations. In general, the longer the cut seam, the smaller the distortion.

Combinatorial problem. Solving this problem is very challenging. First, since a cut is discretely represented as mesh edges, it is a *combinatorial problem*. It is highly complex to reduce the length using combinatorial optimization techniques. Second, cut construction and parameterization generation are coupled. Parameterizations are usually computed after cuts are determined, and the distortion heavily depends on the cut location.

Method classification. Three types of methods are mainly proposed:

- *Segmentation-based methods* partition an input mesh into multiple charts [29, 139–141].
- *Optimization-based methods* simultaneously optimize the parameterization distortion and the cut length [142, 143].
- *Point-to-cut methods* first detect feature points where the distortion is usually concentrated and then connect these feature points to construct cuts [144–148].

Since the segmentation-based methods do not explicitly minimize cut lengths, we discuss other two methods in details. In addition, some greedy methods are developed. Gu et al. [149] alternately parameterize the mesh and connect the maximum distortion vertex to the existing cut via the shortest path. As observed by Ref. [146], this algorithm often terminates early, resulting in large isometric distortion. Triangles are parameterized one-by-one in Ref. [150] without violating the user-provided distortion bound. In general, the one-by-one way is too local to produce a shorter cut than the cut required to achieve a given bound, as observed by Refs. [90, 143].

7.2.1 Optimization-based methods

AutoCuts [143]. The energy function of AutoCuts is the weighted sum of the cut-penalty energy and the symmetric-Dirichlet distortion energy. During optimization, the parameterized mesh is treated as a fixed topology triangle soup, and the cut-penalty energy is optimized to pull separate triangles together. A balancing weight between the cut-penalty energy and the symmetric-Dirichlet distortion energy is required. However, it is non-trivial to determine the weight so that the desired tradeoff between cut length and parameterization distortion is obtained.

OptCuts [142]. OptCuts directly optimizes the cut length under bounded distortion constraint.

Since the optimization problem is combinatorial, they propose local topological operations, including boundary vertex split, interior vertex split, and corner merge. The local operations lead to early entrapment by local minimum, thereby resulting in long cuts, as shown in Ref. [147]. Besides, local operations also cause a high computational cost.

Discussions. Simultaneous optimization of the parameterization distortion and the cut length [142, 143] is a combinatorial problem. Since the nonlinear and non-convex optimization problem is very complicated, these methods are time-consuming and usually generate long cuts. Besides, they heavily rely on the initializations.

7.2.2 Point-to-cut methods

Detecting points. Since parameterizations are not determined during the feature point detection process, proxy metrics, such as the Gaussian curvature [144, 145] and distortion from spherical parameterizations [146], are used as predictors of anticipated parameterization distortion.

High curvature vertices have a high probability of producing high isometric distortion. However as observed by Ref. [137], the relationship between curvatures and distortions is not direct or clear.

A hierarchical clustering method uses distortion metrics from spherical parameterizations [146]. Since the spherical parameterization method [82] used in Ref. [146] may fail to generate bijective spherical parameterizations, distortion metrics from planar parameterizations are used [148]. However, the voting strategy requires ten times of planar parameterization. Similar to Ref. [148], Zhu et al. [147] also use planar parameterizations to generate proxy metrics. To detect necessary feature points to achieve low isometric distortion and prevent too many feature points, a greedy filtering process is proposed [147].

Conformal cone singularities [6, 7, 131, 137] can also be treated as feature points.

Connecting points. Given a graph and a set of terminal vertices in the graph, the Steiner tree problem seeks to find the minimum cost tree connecting all the terminal vertices. This is an NP-hard problem [151].

Algorithms for computing an exact solution to the Steiner tree problem have been proposed [152–154]. However, they cannot generate the exact solution in a reasonable amount of time for large-scale graphs or when there are many terminal vertices. In this problem, if the number of feature points is small and the size of the input mesh is moderate, the exact solution for the Steiner tree problem can be achieved within an acceptable time.

On this account, some approximation methods have been proposed [155–158]. Two commonly used approaches are based on the minimal spanning tree (MST) [159] and the shortest paths heuristic (SPH) [160]. The algorithm in Ref. [159] is used by Refs. [144–146]. A greedy algorithm [147] is proposed to compute an approximate solution driven by auxiliary points. As shown in Ref. [147], the greedy algorithm outperforms MST and SPH, and approximate the optimal solution better in the sense of relative error.

Discussions. In practice, the relationship between proxy metrics and parameterization distortion is not clear and direct, and the configuration (number and locations) of generated feature points is not always appropriate. For example, large distortions can occur if feature points are missing, whereas too many points produce long cuts.

7.3 Hex mesh simplification

Remehsing. Given a 3D mesh, the remeshing process computes another mesh so that its elements satisfy some quality requirements and approximate the input acceptably [161]. The mesh topology and vertex positions are the variables. Since the topology is discrete, the remeshing can be regarded as a *combinatorial* problem. In general, the inversion-free constraint is not explicitly enforced during the remeshing process. However, to improve the robustness and reality of FEM, the generated elements should not be inverted. For triangular and tetrahedral meshes, the Delaunay triangulation theoretically guarantees no inverted elements (triangles or tetrahedrons). For quad and hex meshes, it is challenging to achieve an inversion-free result.

Hex mesh simplification. Here we focus on the hex mesh simplification. A high-quality hex mesh should satisfy the following properties:

- *Local regularity*: each hex element approaches a cuboid and is free of negative scaled Jacobian.
- *Singularity complexity*: the singularity graph is simple and the number of patches in the hex layout is small.

The input of hex mesh simplification is an inversion-free hex mesh that contains no negative scaled Jacobian. The goal is to reduce the number of patches in the hex layout while avoiding any inverted hex and maintaining the input surface shape. Obviously, this is a *combinatorial* problem with inversion-free constraints.

Two robust collapse operations. Gao et al. [162] propose a robust structure simplification algorithm. The main idea is to greedily perform simplification operations, inducing sheet collapse and chord collapse, to reduce the complexity of the base complex of the input mesh. To keep the inversion-free property, they formulate the simplification operation as a deformation process that uses explicit checks in combination with line search to avoid inversions. In addition, the topological validity and geometrical fidelity are also guaranteed by explicit checks. In practice, these explicit checks limit the simplification operation space, thereby leaving room in reducing the singularity complexity.

8 Conclusions

We have presented the state-of-the-art in inversion-free geometric mapping construction. In this section, we discuss possible generalizations of existing methods, and interesting unsolved problems.

Theoretical guarantee. If the initial mapping is not inversion-free, no method has a theoretical guarantee that the result is always inversion-free. This is the most fundamental problem in studying and computing inversion-free mappings. More theoretical studies should be provided to achieve the inversion-free goal.

Bijective mappings in 3D. Bijective mappings in 3D are essential for many geometric processing tasks. In the future, it is worthwhile to study how to reduce computational costs in computing 3D bijective mappings. However, the cases of boundary collision in 3D are more complicated than 2D cases.

Time sequence data. Most geometric data in the aforementioned applications are single and static. One interesting future work is to explore optimization algorithms on the time sequence data, which is widely used in the reconstruction of the dynamic scene. Combined with the semantic information, the collaborative optimization for time sequence models is a possible research direction.

Generalization. Many methods or thoughts mentioned above can be generalized into a unified framework. For example, the parameter α used in TLC (Total Lifted Content) is fixed; but it can be modified to be a changing parameter $\alpha \to 0$, similar to the idea of homotopy optimization. Moreover, similar to the local–global method, these methods can be generalized into a framework that can be used in more applications.

Mesh cutting. The distortion in the mesh cutting algorithm [142] bounded; however, it usually generates long cuts to achieve this goal. The greedy method [147] often produces short cuts, but the distortion is not explicitly bounded. It is interesting to study the cut generation problem to achieve as short cut as possible while bounding the distortion.

Feature-preserving PolyCube construction. Although most features are aligned in Ref. [116], there are still some features that are not aligned. As shown in Ref. [121], a PolyCube corner, whose valence is equal to five, is always non-manifold. Thus, to match a feature point where five feature lines converge, a PolyCube corner with the valence of six is required. Preserving sharp features in the PolyCube construction is an intriguing direction for future research.

Quasi-conformal mappings in 3D. Conformal and quasi-conformal mappings are powerful tools for parameterizations or flattening of Riemann surfaces. Meanwhile, there is very little work to study 3D cases. According to Liouville's theorem, the conformal mappings in $\mathbb{R}^n (n \geqslant 3)$ are only Mübius transformations which is not flexible at all. Quasi-conformal mappings are sufficiently flexiable and still close to conformal in a suitable sense. To study 3D quasi-conformal mappings, Ref. [163] decouples the scaling and rotation in conformal deformation to generate a close-to-conformal mapping. However, generally measuring and optimizing the conformal quality of 3D quasi-conformal mappings are still open problem and need more research.

Poor triangulations for intrinsic flows. The computation process for intrinsic flows is affected by the triangulations. Poor triangulation will severely slow the convergence or even result in non-convergence of the discrete intrinsic flow. Even if an edge flip strategy is applied to improve the quality of triangulation, the process may terminate when extreme poor triangular meshes are used as inputs.

Hex mesh improvement. Improving the quality of a hex mesh requires optimizing the structure and vertex positions at the same time. There are several problems worth studying. First, if the input mesh contains inverted hex elements, how to effectively and efficiently eliminate them? Second, how to robustly compute a coarser structure while satisfying the geometric fidelity constraint and the topological constraint? Third, can we use the structure optimization technique to help us to eliminate inversion?

Acknowledgements

We would like to thank the anonymous reviewers for their constructive suggestions and comments. This work was supported by the National Natural Science Foundation of China (Nos. 61802359 and 61672482) and the USTC Research Funds of the Double First-Class Initiative (No. YD0010002003).

References

[1] Botsch, M.; Kobbelt, L.; Pauly, M.; Alliez, P.; Levy, B. *Polygon Mesh Processing*. New York: A K Peters/CRC Press, 2010.

[2] Fu, X. M.; Liu, Y. Computing inversion-free mappings by simplex assembly. *ACM Transactions on Graphics* Vol. 35, No. 6, Article No. 216, 2016.

[3] Sheffer, A.; de Sturler, E. Parameterization of faceted surfaces for meshing using angle-based flattening. *Engineering With Computers* Vol. 17, No. 3, 326–337, 2001.

[4] Sheffer, A.; Lévy, B.; Mogilnitsky, M.; Bogomyakov, A. ABF++: Fast and robust angle based flattening. *ACM Transactions on Graphics* Vol. 24, No. 2, 311–330, 2005.

[5] Paillé, G. P.; Ray, N.; Poulin, P.; Sheffer, A.; Lévy, B. Dihedral angle-based maps of tetrahedral meshes. *ACM Transactions on Graphics* Vol. 34, No. 4, Article No. 54, 2015.

[6] Springborn, B.; Schröder, P.; Pinkall, U. Conformal equivalence of triangle meshes. *ACM Transactions on Graphics* Vol. 27, No. 3, 1–11, 2008.

[7] Ben-Chen, M.; Gotsman, C.; Bunin, G. Conformal flattening by curvature prescription and metric scaling. *Computer Graphics Forum* Vol. 27, No. 2, 449–458, 2008.

[8] Fang, Q.; Zhao, Z. Y.; Liu, Z. Y.; Liu, L. G.; Fu, X. M. Metric first reconstruction for interactive curvature-aware modeling. *Computer-Aided Design* Vol. 126, 102863, 2020.

[9] Chien, E.; Levi, Z.; Weber, O. Bounded distortion parametrization in the space of metrics. *ACM Transactions on Graphics* Vol. 35, No. 6, Article No. 215, 2016.

[10] Roček, M.; Williams, R. M. The quantization of Regge calculus. *Zeitschrift Für Physik C Particles and Fields* Vol. 21, No. 4, 371–381, 1984.

[11] Luo, F. Combinatorial yamabe flow on surfaces. *Communications in Contemporary Mathematics* Vol. 6, No. 5, 765–780, 2004.

[12] Su, K.; Li, C.; Zhou, Y.; Xu, X.; Gu, X. Discrete calabi flow: A unified conformal parameterization method. *Computer Graphics Forum* Vol. 38, No. 7, 707–720, 2019.

[13] Su, J.-P.; Ye, C.; Liu, L.; Fu, X.-M. Efficient bijective parameterizations. *ACM Transactions on Graphics* Vol. 39, No. 4, Article No. 111, 2020.

[14] Fu, X. M.; Liu, Y.; Guo, B. N. Computing locally injective mappings by advanced MIPS. *ACM Transactions on Graphics* Vol. 34, No. 4, Article No. 71, 2015.

[15] Poranne, R.; Lipman, Y. Provably good planar mappings. *ACM Transactions on Graphics* Vol. 33, No. 4, Article No. 76, 2014.

[16] Fu, X. M.; Liu, Y.; Guo, B. N. Computing locally injective mappings by advanced MIPS. *ACM Transactions on Graphics* Vol. 34, No. 4, Article No. 71, 2015.

[17] Chen, R. J.; Weber, O. GPU-accelerated locally injective shape deformation. *ACM Transactions on Graphics* Vol. 36, No. 6, Article No. 214, 2017.

[18] Chen, R. J.; Weber, O. Bounded distortion harmonic mappings in the plane. *ACM Transactions on Graphics* Vol. 34, No. 4, Article No. 73, 2015.

[19] Levi, Z.; Weber, O. On the convexity and feasibility of the bounded distortion harmonic mapping problem. *ACM Transactions on Graphics* Vol. 35, No. 4, Article No. 106, 2016.

[20] Hughes, T. J. R.; Cottrell, J. A.; Bazilevs, Y. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Computer Methods in Applied Mechanics and Engineering* Vol. 194, Nos. 39–41, 4135–4195, 2005.

[21] Liu, H.; Yang, Y.; Liu, Y.; Fu, X. M. Simultaneous interior and boundary optimization of volumetric domain parameterizations for IGA. *Computer Aided Geometric Design* Vol. 79, 101853, 2020.

[22] Nian, X. S.; Chen, F. L. Planar domain parameterization for isogeometric analysis based on Teichmüller mapping. *Computer Methods in Applied Mechanics and Engineering* Vol. 311, 41–55, 2016.

[23] Dong, Z. C.; Fu, X. M.; Yang, Z. S.; Liu, L. G. Redirected smooth mappings for multiuser real walking in virtual reality. *ACM Transactions on Graphics* Vol. 38, No. 5, Article No. 149, 2019.

[24] Dong, Z. C.; Fu, X. M.; Zhang, C.; Wu, K.; Liu, L. G. Smooth assembled mappings for large-scale real walking. *ACM Transactions on Graphics* Vol. 36, No. 6, Article No. 211, 2017.

[25] Sun, Q.; Wei, L.-Y.; Kaufman, A. Mapping virtual and physical reality. *ACM Transactions on Graphics* Vol. 35, No. 4, Article No. 64, 2016.

[26] Aigerman, N.; Lipman, Y. Injective and bounded distortion mappings in 3D. *ACM Transactions on Graphics* Vol. 32, No. 4, Article No. 106, 2013.

[27] Degener, P.; Meseth, J.; Klein, R. An adaptable surface parameterization method. In: Proceedings of the 12th International Meshing Roundtable, 201–213, 2003.

[28] Hormann, K.; Greiner, G. MIPS: An efficient global parametrization method. In: *Curve and Surface Design: Saint-Malo 1999*. Laurent, P.-J.; Sablonniere, P.; Schumaker, L. L. Eds. Vanderbilt University Press, 153–162, 2000.

[29] Lévy, B.; Petitjean, S.; Ray, N.; Maillot, J. Least squares conformal maps for automatic texture atlas generation. *ACM Transactions on Graphics* Vol. 21, No. 3, 362–371, 2002.

[30] Liu, L. G.; Zhang, L.; Xu, Y.; Gotsman, C.; Gortler, S. J. A local/global approach to mesh parameterization. *Computer Graphics Forum* Vol. 27, No. 5, 1495–1504, 2008.

[31] Schreiner, J.; Asirvatham, A.; Praun, E.; Hoppe, H. Inter-surface mapping. *ACM Transactions on Graphics* Vol. 23, No. 3, 870–877, 2004.

[32] Smith, J.; Schaefer, S. Bijective parameterization with free boundaries. *ACM Transactions on Graphics* Vol. 34, No. 4, Article No. 70, 2015.

[33] Lipman, Y. Bounded distortion mapping spaces for triangular meshes. *ACM Transactions on Graphics* Vol. 31, No. 4, Article No. 108, 2012.

[34] Tutte, W. T. How to draw a graph. *Proceedings of the London Mathematical Society* Vol. s3-13, No. 1, 743–767, 1963.

[35] Liu, L. G.; Ye, C. Y.; Ni, R. Q.; Fu, X. M. Progressive parameterizations. *ACM Transactions on Graphics* Vol. 37, No. 4, Article No. 41, 2018.

[36] Claici, S.; Bessmeltsev, M.; Schaefer, S.; Solomon, J. Isometry-aware preconditioning for mesh parameterization. *Computer Graphics Forum* Vol. 36, No. 5, 37–47, 2017.

[37] Shtengel, A.; Poranne, R.; Sorkine-Hornung, O.; Kovalsky, S. Z.; Lipman, Y. Geometric optimization via composite majorization. *ACM Transactions on Graphics* Vol. 36, No. 4, Article No. 38, 2017.

[38] Aigerman, N.; Lipman, Y. Orbifold Tutte embeddings. *ACM Transactions on Graphics* Vol. 34, No. 6, Article No. 190, 2015.

[39] Floater, M. S. One-to-one piecewise linear mappings over triangulations. *Mathematics of Computation* Vol. 72, No. 242, 685–697, 2003.

[40] Campen, M.; Silva, C. T.; Zorin, D. Bijective maps from simplicial foliations. *ACM Transactions on Graphics* Vol. 35, No. 4, Article No. 74, 2016.

[41] Schüller, C.; Kavan, L.; Panozzo, D.; Sorkine-Hornung, O. Locally injective mappings. *Computer Graphics Forum* Vol. 32, No. 5, 125–135, 2013.

[42] Liu, T. T.; Gao, M.; Zhu, L. F.; Sifakis, E.; Kavan, L. Fast and robust inversion-free shape manipulation. *Computer Graphics Forum* Vol. 35, No. 2, 1–11, 2016.

[43] Rabinovich, M.; Poranne, R.; Panozzo, D.; Sorkine-Hornung, O. Scalable locally injective mappings. *ACM Transactions on Graphics* Vol. 36, No. 4, Article No. 16, 2017.

[44] Kovalsky, S. Z.; Galun, M.; Lipman, Y. Accelerated quadratic proxy for geometric optimization. *ACM Transactions on Graphics* Vol. 35, No. 4, Article No. 134, 2016.

[45] Zhu, Y. F.; Bridson, R.; Kaufman, D. M. Blended cured quasi-Newton for distortion optimization. *ACM Transactions on Graphics* Vol. 37, No. 4, Article No. 40, 2018.

[46] Hormann, K. Theory and applications of parameterizing triangulations. Ph.D. Thesis. Department of Computer Science, University of Erlangen, 2001.

[47] Kovalsky, S. Z.; Galun, M.; Lipman, Y. Accelerated quadratic proxy for geometric optimization. *ACM Transactions on Graphics* Vol. 35, No. 4, Article No. 134, 2016.

[48] Nocedal, J.; Wright, S. J. *Numerical Optimization*, 2nd edn. New York: Springer, 2006.

[49] Jiang, L. J.; Byrd, R. H.; Eskow, E.; Schnabel, R. B. A preconditioned L-BFGS algorithm with application to molecular energy minimization. Technical Report. CU-CS-982-04. Department of Computer Science, University of Colorado, 2004.

[50] Liu, T. T.; Bouaziz, S.; Kavan, L. Quasi-Newton methods for real-time simulation of hyperelastic materials. *ACM Transactions on Graphics* Vol. 36, No. 3, Article No. 23, 2017.

[51] Nocedal, J.; Wright, S. J. *Numerical Optimization.* New York: Springer, 1999.

[52] 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, 181–190, 2005.

[53] Golla, B.; Seidel, H. P.; Chen, R. J. Piecewise linear mapping optimization based on the complex view. *Computer Graphics Forum* Vol. 37, No. 7, 233–243, 2018.

[54] Smith, B.; De Goes, F.; Kim, T. Analytic eigensystems for isotropic distortion energies. *ACM Transactions on Graphics* Vol. 38, No. 1, Article No. 3, 2019.

[55] Ho, K. T.; Lui, L. M. QCMC: Quasi-conformal parameterizations for multiply-connected domains. *Advances in Computational Mathematics* Vol. 42, No. 2, 279–312, 2016.

[56] Zeng, W.; Gu, X. D. Registration for 3D surfaces with large deformations using quasi-conformal curvature flow. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2457–2464, 2011.

[57] Weber, O.; Myles, A.; Zorin, D. Computing extremal quasiconformal maps. *Computer Graphics Forum* Vol. 31, No. 5, 1679–1689, 2012.

[58] Ma, M.; Lei, N.; Chen, W.; Su, K. H.; Gu, X. F. Robust surface registration using optimal mass transport and Teichmüller mapping. *Graphical Models* Vol. 90, 13–23, 2017.

[59] Mastin, C. W.; Thompson, J. F. Discrete quasiconformal mappings. *Zeitschrift Für Angewandte Mathematik Und Physik ZAMP* Vol. 29, No. 1, 1–11, 1978.

[60] He, Z. X. Solving Beltrami equations by circle packing. *Transactions of the American Mathematical Society* Vol. 322, No. 2, 657–670, 1990.

[61] Daripa, P. A fast algorithm to solve the Beltrami equation with applications to quasiconformal mappings. *Journal of Computational Physics* Vol. 106, No. 2, 355–365, 1993.

[62] Wong, T. W.; Zhao, H. K. Computation of quasi-conformal surface maps using discrete beltrami flow. *SIAM Journal on Imaging Sciences* Vol. 7, No. 4, 2675–2699, 2014.

[63] Lui, L. M.; Wong, T. W.; Zeng, W.; Gu, X. F.; Thompson, P. M.; Chan, T. F.; Yau, S.-T. Optimization of surface registrations using beltrami holomorphic flow. *Journal of Scientific Computing* Vol. 50, No. 3, 557–585, 2012.

[64] Zeng, W.; Lui, L. M.; Luo, F.; Chan, T. F. C.; Yau, S. T.; Gu, D. X. Computing quasiconformal maps using an auxiliary metric and discrete curvature flow. *Numerische Mathematik* Vol. 121, No. 4, 671–703, 2012.

[65] Kovalsky, S. Z.; Aigerman, N.; Basri, R.; Lipman, Y. Controlling singular values with semidefinite programming. *ACM Transactions on Graphics* Vol. 33, No. 4, Article No. 68, 2014.

[66] Su, J. P.; Fu, X. M.; Liu, L. G. Practical foldover-free volumetric mapping construction. *Computer Graphics Forum* Vol. 38, No. 7, 287–297, 2019.

[67] Kovalsky, S. Z.; Aigerman, N.; Basri, R.; Lipman, Y. Large-scale bounded distortion mappings. *ACM Transactions on Graphics* Vol. 34, No. 6, Article No. 191, 2015.

[68] Peng, Y.; Deng, B. L.; Zhang, J. Y.; Geng, F. Y.; Qin, W. J.; Liu, L. G. Anderson acceleration for geometry optimization and physics simulation. *ACM Transactions on Graphics* Vol. 37, No. 4, Article No. 42, 2018.

[69] Xu, Y.; Chen, R. J.; Gotsman, C.; Liu, L. G. Embedding a triangular graph within a given boundary. *Computer Aided Geometric Design* Vol. 28, No. 6, 349–356, 2011.

[70] Du, X. Y.; Aigerman, N.; Zhou, Q. N.; Kovalsky, S. Z.; Yan, Y. J.; Kaufman, D. M.; Ju, T. Lifting simplices to find injectivity. *ACM Transactions on Graphics* Vol. 39, No. 4, Article No. 120, 2020.

[71] Toulorge, T.; Geuzaine, C.; Remacle, J. F.; Lambrechts, J. Robust untangling of curvilinear meshes. *Journal of Computational Physics* Vol. 254, 8–26, 2013.

[72] Escobar, J. M.; Rodríguez, E.; Montenegro, R.; Montero, G.; González-Yuste, J. M. Simultaneous untangling and smoothing of tetrahedral meshes. *Computer Methods in Applied Mechanics and Engineering* Vol. 192, No. 25, 2775–2787, 2003.

[73] Yang, Y.; Fu, X. M.; Chai, S. M.; Xiao, S. W.; Liu, L. G. Volume-enhanced compatible remeshing of 3D models. *IEEE Transactions on Visualization and Computer Graphics* Vol. 25, No. 10, 2999–3010, 2019.

[74] Zayer, R.; Lévy, B.; Seidel, H.-P. Linear angle based parameterization. In: Proceedings of the 5th Eurographics Symposium on Geometry Processing, 135–141, 2007.

[75] Crane, K.; Pinkall, U.; Schröder, P. Robust fairing via conformal curvature flow. *ACM Transactions on Graphics* Vol. 32, No. 4, Article No. 61, 2013.

[76] Jin, M.; Kim, J.; Luo, F.; Gu, X. F. Discrete surface ricci flow. *IEEE Transactions on Visualization and Computer Graphics* Vol. 14, No. 5, 1030–1043, 2008.

[77] Ge, H. B. Combinatorial Calabi flows on surfaces. *Transactions of the American Mathematical Society* Vol. 370, No. 2, 1377–1391, 2018.

[78] Wang, Y. L.; Shi, J.; Yin, X. T.; Gu, X. F.; Chan, T. F.; Yau, S. T.; Toga, A. W.; Thompson, P. M. Brain surface conformal parameterization with the ricci flow. *IEEE Transactions on Medical Imaging* Vol. 31, No. 2, 251–264, 2012.

[79] Zhao, H.; Li, X.; Ge, H. B.; Lei, N.; Zhang, M.; Wang, X. L.; Gu, X. F. Conformal mesh parameterization using discrete Calabi flow. *Computer Aided Geometric Design* Vol. 63, 96–108, 2018.

[80] Jin, Y.; Huang, J.; Tong, R. Remeshing-assisted optimization for locally injective mappings. *Computer Graphics Forum* Vol. 33, No. 5, 269–279, 2014.

[81] Zhang, W. J.; Ma, Y. W.; Zheng, J. M.; Allen, W. J. Tetrahedral mesh deformation with positional constraints. *Computer Aided Geometric Design* Vol. 81, 101909, 2020.

[82] Hu, X.; Fu, X. M.; Liu, L. G. Advanced hierarchical spherical parameterizations. *IEEE Transactions on Visualization and Computer Graphics* Vol. 24, No. 6, 1930–1941, 2018.

[83] Shen, H. X.; Jiang, Z. S.; Zorin, D.; Panozzo, D. Progressive embedding. *ACM Transactions on Graphics* Vol. 38, No. 4, Article No. 32, 2019.

[84] Jiang, Z. S.; Schaefer, S.; Panozzo, D. Simplicial complex augmentation framework for bijective maps. *ACM Transactions on Graphics* Vol. 36, No. 6, Article No. 186, 2017.

[85] Li, M. C.; Ferguson, Z.; Schneider, T.; Langlois, T.; Zorin, D.; Panozzo, D.; Jiang, C.; Kaufman, D. M. Incremental potential contact. *ACM Transactions on Graphics* Vol. 39, No. 4, Article No. 49, 2020.

[86] Zhang, E.; Mischaikow, K.; Turk, G. Feature-based surface parameterization and texture mapping. *ACM Transactions on Graphics* Vol. 24, No. 1, 1–27, 2005.

[87] Müller, M.; Chentanez, N.; Kim, T. Y.; Macklin, M. Air meshes for robust collision handling. *ACM Transactions on Graphics* Vol. 34, No. 4, Article No. 133, 2015.

[88] Misztal, M. K.; Bærentzen, J. A. Topology-adaptive interface tracking using the deformable simplicial complex. *ACM Transactions on Graphics* Vol. 31, No. 3, Article No. 24, 2012.

[89] Ye, C. Y.; Su, J. P.; Liu, L. G.; Fu, X. M. Memory-efficient bijective parameterizations of very-large-scale models. *Computer Graphics Forum* Vol. 39, No. 7, 1–12, 2020.

[90] Hormann, K.; Lévy, B.; Sheffer, A. Mesh parameterization: Theory and practice. In: Proceedings of the SIGGRAPH '07: ACM SIGGRAPH 2007 Courses, 1–es, 2007.

[91] Van Kaick, O.; Zhang, H.; Hamarneh, G.; Cohen-Or, D. A survey on shape correspondence. *Computer Graphics Forum* Vol. 30, No. 6, 1681–1707, 2011.

[92] Li, X.; Iyengar, S. S. On computing mapping of 3D objects. *ACM Computing Surveys* Vol. 47, No. 2, Article No. 34, 2015.

[93] Kraevoy, V.; Sheffer, A. Cross-parameterization and compatible remeshing of 3D models. *ACM Transactions on Graphics* Vol. 23, No. 3, 861–869, 2004.

[94] Yang, Y.; Zhang, W. X.; Liu, Y.; Liu, L. G.; Fu, X. M. Error-bounded compatible remeshing. *ACM Transactions on Graphics* Vol. 39, No. 4, Article No. 113, 2020.

[95] Alexa, M. Merging polyhedral shapes with scattered features. In: Proceedings of the International Conference on Shape Modeling and Applications, 202–210, 1999.

[96] Kwok, T. H.; Zhang, Y. B.; Wang, C. C. L. Efficient optimization of common base domains for cross parameterization. *IEEE Transactions on Visualization and Computer Graphics* Vol. 18, No. 10, 1678–1692, 2012.

[97] Aigerman, N.; Poranne, R.; Lipman, Y. Lifted bijections for low distortion surface mappings. *ACM Transactions on Graphics* Vol. 33, No. 4, Article No. 69, 2014.

[98] Aigerman, N.; Poranne, R.; Lipman, Y. Seamless surface mappings. *ACM Transactions on Graphics* Vol. 34, No. 4, Article No. 72, 2015.

[99] Aigerman, N.; Lipman, Y. Hyperbolic orbifold tutte embeddings. *ACM Transactions on Graphics* Vol. 35, No. 6, Article No. 190, 2016.

[100] Praun, E.; Hoppe, H. Spherical parametrization and remeshing. *ACM Transactions on Graphics* Vol. 22, No. 3, 340–349, 2003.

[101] Wang, C. X.; Hu, X.; Fu, X. M.; Liu, L. G. Bijective spherical parametrization with low distortion. *Computers & Graphics* Vol. 58, 161–171, 2016.

[102] Hoppe, H. Progressive meshes. In: Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques, 99–108, 1996.

[103] Schmidt, P.; Born, J.; Campen, M.; Kobbelt, L. Distortion-minimizing injective maps between surfaces. *ACM Transactions on Graphics* Vol. 38, No. 6, Article No. 156, 2019.

[104] Schmidt, P.; Campen, M.; Born, J.; Kobbelt, L. Inter-surface maps via constant-curvature metrics. *ACM Transactions on Graphics* Vol. 39, No. 4, Article No. 119, 2020.

[105] Tarini, M.; Hormann, K.; Cignoni, P.; Montani C. PolyCube-Maps. *ACM Transactions on Graphics* Vol. 23, No. 3, 853–860, 2004.

[106] Yao, C. Y.; Lee, T. Y. Adaptive geometry image. *IEEE Transactions on Visualization and Computer Graphics* Vol. 14, No. 4, 948–960, 2008.

[107] Chang, C.-C.; Lin, C.-Y. Texture tiling on 3D models using automatic PolyCube-maps and Wang tiles. *Journal of Information Science and Engineering* Vol. 26, No. 1, 291–305, 2010.

[108] Fu, X. M.; Bai, C. Y.; Liu, Y. Efficient volumetric PolyCube-map construction. *Computer Graphics Forum* Vol. 35, No. 7, 97–106, 2016.

[109] Gregson, J.; Sheffer, A.; Zhang, E. All-hex mesh generation via volumetric PolyCube deformation. *Computer Graphics Forum* Vol. 30, No. 5, 1407–1416, 2011.

[110] Yu, Y. Z.; Zhou, K.; Xu, D.; Shi, X. H.; Bao, H. J.; Guo, B. N.; Shum, H.-Y. Mesh editing with Poisson-based gradient field manipulation. *ACM Transactions on Graphics* Vol. 23, No. 3, 644–651, 2004.

[111] Huang, J.; Jiang, T. F.; Shi, Z. Y.; Tong, Y. Y.; Bao, H. J.; Desbrun, M. $\ell 1$-based construction of polycube maps from complex shapes. *ACM Transactions on Graphics* Vol. 33, No. 3, Article No. 25, 2014.

[112] Fang, X. Z.; Xu, W. W.; Bao, H. J.; Huang, J. All-hex meshing using closed-form induced polycube. *ACM Transactions on Graphics* Vol. 35, No. 4, Article No. 124, 2016.

[113] Liu, C. L.; Yu, W. Y.; Chen, Z. G.; Li, X. Distributed poly-square mapping for large-scale semi-structured quad mesh generation. *Computer-Aided Design* Vol. 90, 5–17, 2017.

[114] Xiao, S. W.; Kang, H. M.; Fu, X. M.; Chen, F. L. Computing IGA-suitable planar parameterizations by PolySquare-enhanced domain partition. *Computer Aided Geometric Design* Vol. 62, 29–43, 2018.

[115] Livesu, M.; Vining, N.; Sheffer, A.; Gregson, J.; Scateni, R. PolyCut: Monotone graph-cuts for PolyCube base-complex construction. *ACM Transactions on Graphics* Vol. 32, No. 6, Article No. 171, 2013.

[116] Guo, H. X.; Liu, X. H.; Yan, D. M.; Liu, Y. Cut-enhanced PolyCube-maps for feature-aware all-hex meshing. *ACM Transactions on Graphics* Vol. 39, No. 4, Article No. 106, 2020.

[117] Xia, J.; Garcia, I.; He, Y.; Xin, S. Q.; Patow, G. Editable polycube map for GPU-based subdivision surfaces. In: Proceedings of the Symposium on Interactive 3D Graphics and Games, 151–158, 2011.

[118] Liu, H. Y.; Fu, X. M.; Ye, C. Y.; Chai, S. M.; Liu, L. G. Atlas refinement with bounded packing efficiency. *ACM Transactions on Graphics* Vol. 38, No. 4, Article No. 33, 2019.

[119] Zhang, C.; Xu, M. F.; Chai, S. M.; Fu, X. M. Robust atlas generation via angle-based segmentation. *Computer Aided Geometric Design* Vol. 79, 101854, 2020.

[120] Eppstein, D.; Mumford, E. Steinitz theorems for orthogonal polyhedra. In: Proceedings of the 26th Annual Symposium on Computational Geometry, 429–438, 2010.

[121] Yang, Y.; Fu, X. M.; Liu, L. G. Computing surface PolyCube-maps by constrained voxelization. *Computer Graphics Forum* Vol. 38, No. 7, 299–309, 2019.

[122] Myles, A.; Pietroni, N.; Zorin, D. Robust field-aligned global parametrization. *ACM Transactions on Graphics* Vol. 33, No. 4, Article No. 135, 2014.

[123] Myles, A.; Zorin, D. Global parametrization by incremental flattening. *ACM Transactions on Graphics* Vol. 31, No. 4, Article No. 109, 2012.

[124] Fu, X. M.; Liu, Y. Computing inversion-free mappings by simplex assembly. *ACM Transactions on Graphics* Vol. 35, No. 6, Article No. 216, 2016.

[125] Sawhney, R.; Crane, K. Boundary first flattening. *ACM Transactions on Graphics* Vol. 37, No. 1, Article No. 5, 2018.

[126] Campen, M.; Shen, H. X.; Zhou, J. R.; Zorin, D. Seamless parametrization with arbitrary cones for arbitrary genus. *ACM Transactions on Graphics* Vol. 39, No. 1, Article No. 2, 2020.

[127] Zhou, J.; Tu, C.; Zorin, D.; Campen, M. Combinatorial construction of seamless parameter domains. *Computer Graphics Forum* Vol. 39, No. 2, 179–190, 2020.

[128] Levi, Z. Direct seamless parametrization. *ACM Transactions on Graphics* Vol. 40, No. 1, Article No. 6, 2021.

[129] Gu, X.; Yau, S.-T. Global conformal surface parameterization. In: Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing, 127–137, 2003.

[130] Bommes, D.; Zimmer, H.; Kobbelt, L. Mixed-integer quadrangulation. *ACM Transactions on Graphics* Vol. 28, No. 3, Article No. 77, 2009.

[131] Myles, A.; Zorin, D. Global parametrization by incremental flattening. *ACM Transactions on Graphics* Vol. 31, No. 4, Article No. 109, 2012.

[132] Myles, A.; Zorin, D. Controlled-distortion constrained global parametrization. *ACM Transactions on Graphics* Vol. 32, No. 4, Article No. 105, 2013.

[133] Campen, M.; Bommes, D.; Kobbelt, L. Quantized global parametrization. *ACM Transactions on Graphics* Vol. 34, No. 6, Article No. 77, 2015.

[134] Bright, A.; Chien, E.; Weber, O. Harmonic global parametrization with rational holonomy. *ACM Transactions on Graphics* Vol. 36, No. 4, Article No. 89, 2017.

[135] Hefetz, E. F.; Chien, E.; Weber, O. A subspace method for fast locally injective harmonic mapping. *Computer Graphics Forum* Vol. 38, No. 2, 105–119, 2019.

[136] Kharevych, L.; Springborn, B.; Schröder, P. Discrete conformal mappings via circle patterns. *ACM Transactions on Graphics* Vol. 25, No. 2, 412–438, 2006.

[137] Soliman, Y.; Slepčev, D.; Crane, K. Optimal cone singularities for conformal flattening. *ACM Transactions on Graphics* Vol. 37, No. 4, Article No. 105, 2018.

[138] Liu, H.; Zhang, X. T.; Fu, X. M.; Dong, Z. C.; Liu, L. G. Computational peeling art design. *ACM Transactions on Graphics* Vol. 38, No. 4, Article No. 64, 2019.

[139] Julius, D.; Kraevoy, V.; Sheffer, A. D-charts: Quasi-developable mesh segmentation. *Computer Graphics Forum* Vol. 24, No. 3, 581–590, 2005.

[140] Sander, P. V.; Gortler, S. J.; Snyder, J.; Hoppe, H. Signal-specialized parametrization. In: Proceedings of the 13th Eurographics Workshop on Rendering, 87–98, 2002.

[141] Zhou, K.; Synder, J.; Guo, B. N.; Shum, H. Y. Iso-charts: Stretch-driven mesh parameterization using spectral analysis. In: Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing, 45–54, 2004.

[142] Li, M.; Kaufman, D. M.; Kim, V. G.; Solomon, J.; Sheffer, A. OptCuts: Joint optimization of surface cuts and parameterization. *ACM Transactions on Graphics* Vol. 37, No. 6, Article No. 247, 2018.

[143] Poranne, R.; Tarini, M.; Huber, S.; Panozzo, D.; Sorkine-Hornung, O. Autocuts. *ACM Transactions on Graphics* Vol. 36, No. 6, Article No. 215, 2017.

[144] Sheffer, A.; Hart, J. C. Seamster: Inconspicuous low-distortion texture seam layout. In: Proceedings of the IEEE Visualization, 291–298, 2002.

[145] Sheffer, A. Spanning tree seams for reducing parameterization distortion of triangulated surfaces. In: Proceedings of the Shape Modeling International, 61–272, 2002.

[146] Chai, S. M.; Fu, X. M.; Hu, X.; Yang, Y.; Liu, L. G. Sphere-based cut construction for planar parameterizations. *Computers & Graphics* Vol. 74, 66–75, 2018.

[147] Zhu, T.; Ye, C. Y.; Chai, S. M.; Fu, X. M. Greedy cut construction for parameterizations. *Computer Graphics Forum* Vol. 39, No. 2, 191–202, 2020.

[148] Chai, S. M.; Fu, X. M.; Liu, L. G. Voting for distortion points in geometric processing. *IEEE Transactions on Visualization and Computer Graphics* Vol. 27, No. 4, 2469–2480, 2021.

[149] Gu, X. F.; Gortler, S. J.; Hoppe, H. Geometry images. *ACM Transactions on Graphics* Vol. 21, No. 3, 355–361, 2002.

[150] Sorkine, O.; Cohen-Or, D.; Goldenthal, R.; Lischinski, D. Bounded-distortion piecewise mesh parameterization. In: Proceedings of the IEEE Visualization, 355–362, 2002.

[151] Hwang, F. K.; Richards, D. S.; Winter, P. *The Steiner Tree Problem.* North Holland, 1992.

[152] Fomin, F. V.; Grandoni, F.; Kratsch, D. Faster Steiner tree computation in polynomial-space. In: *Algorithms - ESA 2008. Lecture Notes in Computer Science, Vol. 5193.* Halperin, D.; Mehlhorn, K. Eds. Springer Berlin Heidelberg, 430–441, 2008.

[153] Beasley, J. E. An SST-based algorithm for the Steiner problem in graphs. *Networks* Vol. 19, No. 1, 1–16, 1989.

[154] Hakimi, S. L. Steiner's problem in graphs and its implications. *Networks* Vol. 1, No. 2, 113–133, 1971.

[155] Berman, P.; Ramaiyer, V. Improved approximations for the steiner tree problem. *Journal of Algorithms* Vol. 17, No. 3, 381–408, 1994.

[156] Byrka, J.; Grandoni, F.; Rothvoss, T.; Sanità, L. Steiner tree approximation via iterative randomized rounding. *Journal of the ACM* Vol. 60, No. 1, 1–33, 2013.

[157] Robins, G.; Zelikovsky, A. Tighter bounds for graph steiner tree approximation. *SIAM Journal on Discrete Mathematics* Vol. 19, No. 1, 122–134, 2005.

[158] Pajor, T.; Uchoa, E.; Werneck, R. F. A robust and scalable algorithm for the Steiner problem in graphs. *Mathematical Programming Computation* Vol. 10, No. 1, 69–118, 2018.

[159] Kou, L.; Markowsky, G.; Berman, L. A fast algorithm for Steiner trees. *Acta Informatica* Vol. 15, No. 2, 141–145, 1981.

[160] Takahashi, H.; Matsuyama, A. An approximate solution for the Steiner problem in graphs. *Mathematica Japonica* Vol. 24, No. 6, 573–577, 1980.

[161] Alliez, P.; Ucelli, G.; Gotsman, C.; Attene, M. Recent advances in remeshing of surfaces. In: *Shape Analysis and Structuring. Mathematics and Visualization*. De Floriani, L.; Spagnuolo, M. Eds. Springer Berlin Heidelberg, 53–82, 2008.

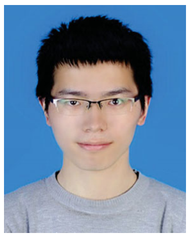
[162] Gao, X. F.; Panozzo, D.; Wang, W. P.; Deng, Z. G.; Chen, G. N. Robust structure simplification for hex re-meshing. *ACM Transactions on Graphics* Vol. 36, No. 6, Article No. 185, 2017.

[163] Chern, A.; Pinkall, U.; Schröder, P. Close-to-conformal deformations of volumes. *ACM Transactions on Graphics* Vol. 34, No. 4, Article No. 56, 2015.

Xiao-Ming Fu received his B.Sc. degree in 2011 and his Ph.D. degree in 2016 from the University of Science and Technology of China (USTC), where he is currently an assistant researcher in the School of Mathematical Sciences. His research interests include geometric processing and computer-aided geometric design. His research work is described at `http://staff.ustc.edu.cn/~fuxm`.

Jian-Ping Su received her B.Sc. degree in 2017 from Northeastern University. She is currently a Ph.D. candidate in the School of Mathematical Sciences, USTC. Her research interests include geometric processing and computer graphics.

Zheng-Yu Zhao received his B.Sc. degree in 2018 from Hunan University. He is currently a Ph.D. candidate in the School of Mathematical Sciences, USTC. His research interest is digital geometric processing.



Qing Fang received his bachelor degree in mathematics and applied mathematics from USTC in 2015. Now he is a Ph.D. candidate at USTC, supervised by Prof. Ligang Liu and Dr. Xiao-Ming Fu.

Chunyang Ye received his bachelor degree in engineering from USTC in 2012. Currently, he is a Ph.D. candidate in the School of Mathematical Sciences, USTC. His research interests include geometric processing and modeling.

Ligang Liu is a professor in the School of Mathematical Sciences, USTC. He received his B.Sc. (1996) and Ph.D. (2001) degrees from Zhejiang University, China. Between 2001 and 2004, he worked at Microsoft Research Asia. Then, he worked at Zhejiang University during 2004 and 2012. He paid an academic visit to Harvard University during 2009 and 2011. His research interests include digital geometric processing, computer graphics, and image processing. He serves as an associate editors for *IEEE Transactions on Visualization and Computer Graphics*, *IEEE Computer Graphics and Applications*, *Computer Graphics Forum*, *Computer Aided Geometric Design*, and *The Visual Computer*. He served as the conference co-chair for GMP 2017 and program co-chair for GMP 2018, CAD/Graphics 2017, CVM 2016, SGP 2015, and SPM 2014. His research work can be found at `http://staff.ustc.edu.cn/~lgliu`.