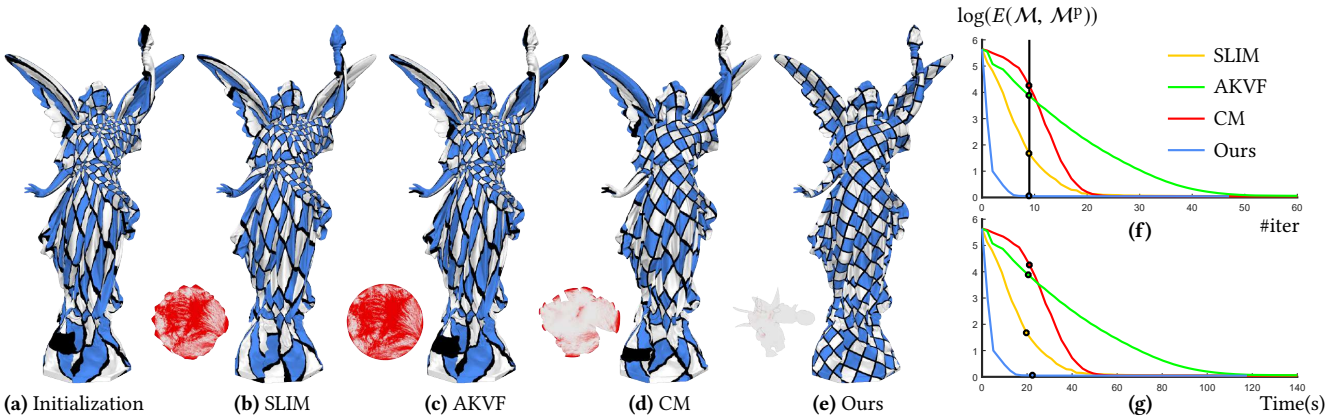
Progressive Parameterizations

LIGANG LIU, University of Science and Technology of China, China
CHUNYANG YE, University of Science and Technology of China, China
RUIQI NI, University of Science and Technology of China, China
XIAO-MING FU*, University of Science and Technology of China, China



(a) Initialization **(b)** SLIM **(c)** AKVF **(d)** CM **(e)** Ours

Fig. 1. Parameterizing a disk topology mesh (1792000 triangles) with low isometric distortion and no foldovers by optimizing the symmetric Dirichlet energy ($E(\mathcal{M}, \mathcal{M}^{\mathrm{p}})$ in Problem (5)). Starting from the same bijective initialization [Tutte 1963] (a), the energy is measured during the optimization process. The figure (b - e) shows a snapshot of the state each method achieved at the 9^{th} iteration marked on the graph (f). The corresponding time in seconds is marked on the graph (g). The color of the triangles from the parameterized meshes encodes the symmetric Dirichlet energy metric, with white being optimal. Our approach achieves much better efficiency than the competitors including SLIM [Rabinovich et al. 2017], AKVF [Claici et al. 2017], and CM [Shtengel et al. 2017]. In order to achieve our result in (e), SLIM, AKVF and CM require 144, 81, and 17 more iterations and 314.60, 184.45, and 38.65 more seconds, respectively.

We propose a novel approach, called *Progressive Parameterizations*, to compute foldover-free parameterizations with low isometric distortion on disk topology meshes. Instead of using the input mesh as a reference to define the objective function, we introduce a progressive reference that contains bounded distortion to the parameterized mesh and is as close as possible to the input mesh. After optimizing the bounded distortion energy between the progressive reference and the parameterized mesh, the parameterized mesh easily approaches the progressive reference, thereby also coming close to the input. By iteratively generating the progressive reference and optimizing the bounded distortion energy to update the parameterized mesh, our algorithm achieves high-quality parameterizations with strong practical reliability and high efficiency. We have demonstrated that our algorithm succeeds on a massive test data set containing over 20712 complex disk topology meshes. Compared to the state-of-the-art methods, our method has achieved higher computational efficiency and practical reliability.

*The corresponding author

Authors' addresses: Ligang Liu, University of Science and Technology of China, China, lgliu@ustc.edu.cn; Chunyang Ye, University of Science and Technology of China, China, yechyang@mail.ustc.edu.cn; Ruiqi Ni, University of Science and Technology of China, China, nrq@mail.ustc.edu.cn; Xiao-Ming Fu, University of Science and Technology of China, China, fuxm@ustc.edu.cn.

0730-0301/2018/8-ART41 $15.00
https://doi.org/10.1145/3197517.3201331

CCS Concepts: • **Computing methodologies** → **Shape modeling**;

Additional Key Words and Phrases: parameterizations, isometric mapping, low distortion, foldover-free, flip-free

ACM Reference Format:
Ligang Liu, Chunyang Ye, Ruiqi Ni, and Xiao-Ming Fu. 2018. Progressive Parameterizations. *ACM Trans. Graph.* 37, 4, Article 41 (August 2018), 12 pages. https://doi.org/10.1145/3197517.3201331

1 INTRODUCTION

Surface parameterization is a fundamental problem in computer graphics and geometric processing; thus, it has been widely used in many applications, such as texture mapping, remeshing, inter-surface mapping, and shape analysis (cf. extensive surveys [Floater and Hormann 2005; Hormann et al. 2007; Sheffer et al. 2006]).

These tasks rely on the computation of a *foldover-free*, *low-distortion* parameterization. Linear methods, such as Tutte's method and its variants [Aigerman and Lipman 2015; Floater 2003; Tutte 1963], provide injectivity-guaranteed parameterization, but usually exhibit extremely large distortion for complex inputs. Nonlinear methods formulate parameterization as an optimization problem by minimizing an energy functions with some constraints that preserve the orientations of triangles. Generally, the objective function includes a low-distortion term that is large when the triangle is highly distorted. It may also include a foldover-preventing term that goes to infinity when a triangle flips or degenerates. These objective functions are highly non-convex and non-linear; thus, they are numerically difficult to optimize, particularly for large-scale inputs.

From a geometric standpoint, these energy functions in the parameterization problem (and other geometric processing problems) are defined on triangle vertices reflecting the topology of the underlying triangular mesh and thus are highly structured. Instead of using generic optimization methods, many geometric optimization algorithms usually take advantage of the particular structure of the energies to solve the optimization. There are many specialized optimization methods, such as the local/global methods [Liu et al. 2008; Sorkine and Alexa 2007], bounded distortion methods [Aigerman et al. 2014; Kovalsky et al. 2015; Lipman 2012], representation based methods [Chien et al. 2016b; Fu and Liu 2016; Sheffer et al. 2005], etc. However, these methods are *not* guaranteed to produce a valid foldover-free parameterization.

From an optimization standpoint, the commonly used methods that guarantee a valid solution are those that begin with a feasible (folder-free) initialization, e.g., Tutte's method [Tutte 1963], and ensure that it never leaves the feasible region, i.e., maintaining foldover-free constraints, during the optimization iterations. There are many recent works such as the block coordinate descent methods [Fu et al. 2015; Hormann and Greiner 2000], the quasi-Newton method [Smith and Schaefer 2015], preconditioning methods [Claici et al. 2017; Kovalsky et al. 2016], the reweighting descent method [Rabinovich et al. 2017], or the composite majorization method [Shtengel et al. 2017]. However, the Tutte's parameterization may introduce extremely large distortion for some triangles. Thus, these approaches may demand a large number of iterations to obtain a good enough result and to converge in many cases.

In contrast to existing methods, which focus on developing sophisticated techniques for solving poorly behaved optimizations, we study the problem from a *different* view. We observe that the distortion on each triangle is measured by some (isometric) distortion metrics on the Jacobian of the affine map between the triangle of the input mesh, called the *reference triangle*, and the corresponding triangle in the target (i.e., the initial Tutte's parameterization at the first iteration). Our *key insight* is that instead of taking the triangles in the input mesh (the *ideal reference triangles*) as the reference ones, we take some intermediate reference triangles to define the distortion energy. If we carefully choose the appropriate intermediate reference triangles such that the distortion metric of each individual triangle can be *bounded*, then the minimization can be quickly solved. We do this in an iterative manner and choose the reference triangles that *progressively* approximate the ideal triangles during the iterations. If the reference triangles approximate the ideal triangles well, then we can obtain a good solution to the original optimization problem. In our method, we generate a series of intermediate parameterizations according to the progressive references; we call them *progressive parameterizations*.

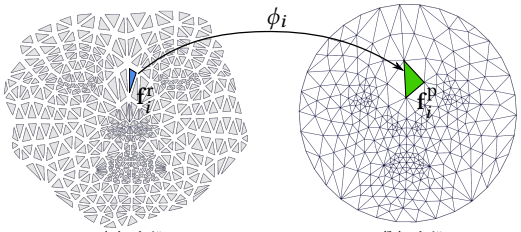
As far as we know, our approach is the first to solve the parameterization problem via iteratively altering the objective distortion energy instead of performing complicated optimization skills. Our approach is simple and performs better and faster than current state-of-the-art methods. We have conducted a large number of experiments and comparisons, which show the feasibility and effectiveness of our proposed approach.

2 RELATED WORK

Foldover-free parameterizations. Parameterizations of disk topology meshes have attracted considerable research attention in the past thirty years (cf. the surveys in [Floater and Hormann 2005; Hormann et al. 2007; Sheffer et al. 2006]). Here, we review only the most relevant prior works. Many methods focus on creating foldover-free parameterizations, such as the Tutte's embedding and its variants [Aigerman et al. 2017; Aigerman and Lipman 2015, 2016; Bright et al. 2017; Floater 2003; Tutte 1963], bounded distortion methods [Aigerman et al. 2014; Kovalsky et al. 2015; Lipman 2012], fixed boundary methods [Weber and Zorin 2014], representation-based methods [Chien et al. 2016b; Fu and Liu 2016; Sheffer et al. 2005], and maintenance-based methods [Claici et al. 2017; Degener et al. 2003; Fu et al. 2015; Hormann and Greiner 2000; Jiang et al. 2017; Kovalsky et al. 2016; Rabinovich et al. 2017; Schüller et al. 2013; Shtengel et al. 2017; Smith and Schaefer 2015]. The Tutte's embedding and its variants are theoretically guaranteed to generate bijective parameterizations that do not flip. However, their isometric distortions are substantially large. Setting an appropriate bound that will ensure no foldovers in the bounded distortion methods remains a subject for study. The representation-based methods are not theoretically guaranteed to eliminate all foldovers. The fixed boundary method can always produce foldover-free results, but it requires a prescribed boundary and may introduce high distortion if the given boundary is inappropriate. The maintenance-based methods first initialize using the Tutte's embedding and then optimize a foldover-prevented energy, like the AMIPS energy [Fu et al. 2015] or the symmetric Dirichlet energy [Smith and Schaefer 2015], to reduce the isometric distortion while ensuring no foldovers. However, the designed numerical solvers may converge slowly due to the large isometric distortion of the initializations. To guarantee no foldovers, we also use the maintenance-based method. In contrast to the existing methods, we present the progressive reference triangles as a way to define a bounded energy that can relieve the large distortion effects and improve efficiency.

Planar affine transformation interpolation. Many methods generate the planar shape sequence between the given source and target models, like the ARAP interpolation [Alexa et al. 2000], controllable conformal interpolation [Weber and Gotsman 2010], or bounded distortion interpolation [Chen et al. 2013; Chien et al. 2016a]. These techniques can be used to generate the intermediate reference; however, the shape reconstruction often requires solving a linear system, which is costly for our method. Furthermore, foldovers may appear in the reconstructed shape, which are not appropriate guides for computing parameterization. Therefore, we only individually interpolate the intermediate triangles without shape reconstruction.

Since affine transformations are usually used to define distortion functions, we interpolate the affine transformations to construct the intermediate references. This process does not utilize triangle edges or angles. Many methods are proposed to conduct the planar affine transformation interpolation, such as the ARAP method [Alexa et al. 2000], and the exponential map method [Alexa 2002; Grassia 1998; Rossignac and Vinacua 2011]. Our method employs the exponential function to conduct the interpolation.



(a) $\mathcal{M}^{\mathrm{r}}$ **(b)** $\mathcal{M}^{\mathrm{p}}$

Fig. 2. Reference-based parameterization. (a) Individual reference triangles. (b) Parameterized mesh.

Cut path construction. Taking a closed triangular mesh as input, some approaches [Gu et al. 2002; Sheffer 2002; Sheffer and Hart 2002] first optimize the cut paths that tailor the mesh to the disk topology. Then, they compute a low isometric distortion parameterization. In contrast to the above methods, the cuts and isometric distortions are optimized simultaneously [Poranne et al. 2017]. Furthermore, multiple charts are generated to produce low isometric distortion results, such as [Lévy et al. 2002; Sorkine et al. 2002; Zhou et al. 2004]. On the one hand, our method can be used to parameterize single patch meshes for these methods. On the other hand, our method can provide as isometric as possible results, even with poor cuts. To verify the practical robustness and reliability, we construct a large testing data set of 20712 disk topology models using the formerly mentioned methods and designed poor cuts in Section 4.2.

3 PROGRESSIVE PARAMETERIZATIONS

Mesh parameterization. We denote the input 3D triangular mesh with disk topology as $\mathcal{M} = \{\mathcal{V}, \mathcal{F}\}$ with N_{v} vertices $\mathcal{V} = \{\mathbf{v}_i, i = 1, \cdots, N_{\mathrm{v}}\}$ and N_{f} triangles $\mathcal{F} = \{\mathbf{f}_i, i = 1, \cdots, N_{\mathrm{f}}\}$. The parameterization of $\mathcal{M}$ is a continuous piecewise affine map $\Phi : \mathcal{M} \to \mathcal{M}^{\mathrm{p}}$ from $\mathcal{M}$ to a planar triangular mesh $\mathcal{M}^{\mathrm{p}}$, where $\mathcal{M}^{\mathrm{p}} = \{\mathcal{V}^{\mathrm{p}}, \mathcal{F}^{\mathrm{p}}\}$ with vertices $\mathcal{V}^{\mathrm{p}} = \{\mathbf{v}_i^{\mathrm{p}}, i = 1, \cdots, N_{\mathrm{v}}\}$ and triangles $\mathcal{F}^{\mathrm{p}} = \{\mathbf{f}_i^{\mathrm{p}}, i = 1, \cdots, N_{\mathrm{f}}\}$. We denote the triangles with three corresponding vertices as $\mathbf{f}_i = \triangle \mathbf{v}_{i,0}\mathbf{v}_{i,1}\mathbf{v}_{i,2}$ and $\mathbf{f}_i^{\mathrm{p}} = \triangle \mathbf{v}_{i,0}^{\mathrm{p}}\mathbf{v}_{i,1}^{\mathrm{p}}\mathbf{v}_{i,2}^{\mathrm{p}}$.

3.1 Reference-guided distortion metric

Reference-guided distortion metric. The distortion of triangle $\mathbf{f}_i^{\mathrm{p}}$ is measured with a distortion metric that reflects angle and/or area changes between the reference triangle, denoted as $\mathbf{f}_i^{\mathrm{r}} = \triangle \mathbf{v}_{i,0}^{\mathrm{r}}\mathbf{v}_{i,1}^{\mathrm{r}}\mathbf{v}_{i,2}^{\mathrm{r}}$, and $\mathbf{f}_i^{\mathrm{p}}$ (Fig. 2). The distortion metric, denoted as $D(\mathbf{f}_i^{\mathrm{r}}, \mathbf{f}_i^{\mathrm{p}})$, depends solely on the shapes of $\mathbf{f}_i^{\mathrm{r}}$ and $\mathbf{f}_i^{\mathrm{p}}$ and is invariant to rotations and translations. The set of reference triangles is denoted as $\mathcal{M}^{\mathrm{r}} = \{\mathbf{f}_i^{\mathrm{r}}, i = 1, \cdots, N_f\}$. Note that $\mathcal{M}^{\mathrm{r}}$ is just a set of individual triangles in the plane (Fig. 2 (a)), instead of a connected triangular mesh.

Jacobian of affine maps. We denote $\phi_i : \mathbf{f}_i^{\mathrm{r}} \to \mathbf{f}_i^{\mathrm{p}}$ as the affine map between the two triangles (Fig. 2). This can be represented as $\phi_i(\mathbf{x}) = J_i\mathbf{x} + \mathbf{b}_i, \forall \mathbf{x} \in \mathbf{f}_i^{\mathrm{r}}$, where $\mathbf{b}_i$ is a translation vector and J_i is a 2×2 Jacobian matrix determined by $\mathbf{f}_i^{\mathrm{r}}$ and $\mathbf{f}_i^{\mathrm{p}}$. It is computed as:

$$J_i(\mathbf{f}_i^{\mathrm{r}}, \mathbf{f}_i^{\mathrm{p}}) = \begin{bmatrix} \mathbf{v}_{i,0}^{\mathrm{p}} - \mathbf{v}_{i,1}^{\mathrm{p}} & \mathbf{v}_{i,0}^{\mathrm{p}} - \mathbf{v}_{i,2}^{\mathrm{p}} \end{bmatrix} \begin{bmatrix} \mathbf{v}_{i,0}^{\mathrm{r}} - \mathbf{v}_{i,1}^{\mathrm{r}} & \mathbf{v}_{i,0}^{\mathrm{r}} - \mathbf{v}_{i,2}^{\mathrm{r}} \end{bmatrix}^{-1}. \quad (1)$$

If the reference triangle $\mathbf{f}_i^{\mathrm{r}}$ is given, $J_i(\mathbf{f}_i^{\mathrm{r}}, \mathbf{f}_i^{\mathrm{p}})$ is a linear function of $\mathbf{f}_i^{\mathrm{p}}$, i.e., a function of $\mathbf{v}_{i,0}^{\mathrm{p}}, \mathbf{v}_{i,1}^{\mathrm{p}}, \mathbf{v}_{i,2}^{\mathrm{p}}$.

Isometric distortion metric. The distortion metric $D(\mathbf{f}_i^{\mathrm{r}}, \mathbf{f}_i^{\mathrm{p}})$ is generally formulated in terms of $J_i(\mathbf{f}_i^{\mathrm{r}}, \mathbf{f}_i^{\mathrm{p}})$ in geometric processing. $D(\mathbf{f}_i^{\mathrm{r}}, \mathbf{f}_i^{\mathrm{p}})$ is known as an isometric distortion metric if it is minimal only for rotations. A variety of isometric distortion metrics have been developed in the literature. We adopt a popular choice, which is the symmetric Dirichlet energy metric [Smith and Schaefer 2015]:

$$D(\mathbf{f}_i^{\mathrm{r}}, \mathbf{f}_i^{\mathrm{p}}) = D(J_i(\mathbf{f}_i^{\mathrm{r}}, \mathbf{f}_i^{\mathrm{p}})) = \begin{cases} \frac{1}{4}\left(\|J_i\|_F^2 + \|J_i^{-1}\|_F^2\right), & \text{if } \det J_i > 0, \\ +\infty, & \text{otherwise,} \end{cases} \quad (2)$$

where $\|\cdot\|_F$ denotes the Frobenius norm. The above distortion metric is infinite for degenerate or flipped triangles. Note that we can adopt any other flip-preventing distortion metrics, such as the isometric AMIPS [Fu et al. 2015], which are also infinite for degenerate or flipped triangles.

We utilize the singular values of J_i to compute $D(\mathbf{f}_i^{\mathrm{r}}, \mathbf{f}_i^{\mathrm{p}})$. Here, we denote $J_i = U_i S_i V_i^T$ as the singular value decomposition (SVD) of J_i, where U_i and V_i are the orthogonal matrices, and $S_i = \mathrm{diag}(\sigma_i, \tau_i)$ is a diagonal matrix with singular values on the diagonal.

$$D(\mathbf{f}_i^{\mathrm{r}}, \mathbf{f}_i^{\mathrm{p}}) = \begin{cases} \frac{1}{4}\left(\sigma_i^2 + \sigma_i^{-2} + \tau_i^2 + \tau_i^{-2}\right), & \text{if } \det J_i > 0, \\ +\infty, & \text{otherwise,} \end{cases} \quad (3)$$

when $\sigma_i = \tau_i = 1$, J_i is a rotation transformation, denoted as $J_i^0 = U_i I V_i^T$ (I is the identity matrix), and $D(\mathbf{f}_i^{\mathrm{r}}, \mathbf{f}_i^{\mathrm{p}})$ reaches the minimum of 1.

3.2 Formulation and challenges

Minimization formulation. The distortion of the parameterization $\mathcal{M}^{\mathrm{p}}$ is the sum of the distortions of its individual triangles. $\mathcal{M}^{\mathrm{p}}$ is obtained by solving the following minimization:

$$\begin{aligned} \min_{\mathcal{M}^{\mathrm{p}}} \quad & E(\mathcal{M}^{\mathrm{r}}, \mathcal{M}^{\mathrm{p}}) = \sum_{i=1}^{N_{\mathrm{f}}} \omega_i D(\mathbf{f}_i^{\mathrm{r}}, \mathbf{f}_i^{\mathrm{p}}), \\ \text{s.t.} \quad & \det J_i(\mathbf{f}_i^{\mathrm{r}}, \mathbf{f}_i^{\mathrm{p}}) > 0, \quad i = 1, \cdots, N_{\mathrm{f}}, \end{aligned} \quad (4)$$

where ω_i are weights.

Existing methods for selecting reference triangles. In all existing works, the triangles from $\mathcal{M}$ are chosen as reference triangles. Specifically, each triangle $\mathbf{f}_i$ is isometrically mapped onto a triangle on the plane with its own local coordinate frame. This mapped triangle is chosen as the ideal reference triangle. Without the loss of generality, we also denote the ideal reference triangle as $\mathbf{f}_i$. Thus, the problem (4) becomes:

$$\begin{aligned} \min_{\mathcal{M}^{\mathrm{p}}} \quad & E(\mathcal{M}, \mathcal{M}^{\mathrm{p}}) = \sum_{i=1}^{N_{\mathrm{f}}} \omega_i D(\mathbf{f}_i, \mathbf{f}_i^{\mathrm{p}}), \\ \text{s.t.} \quad & \det J_i(\mathbf{f}_i, \mathbf{f}_i^{\mathrm{p}}) > 0, \quad i = 1, \cdots, N_{\mathrm{f}}, \end{aligned} \quad (5)$$

where ω_i is set as the area of $\mathbf{f}_i$, $i = 1, \cdots, N_f$, in existing methods.

Optimization and initialization. The objective function in (4) or (5) is highly nonlinear with nonconvex and nonlinear constraints. Several recent works have proposed various methods to solve it [Claici et al. 2017; Rabinovich et al. 2017; Shtengel et al. 2017; Smith and

Schaefer 2015]. To guarantee a foldover-free parameterization, the most common methodology starts from a flipless initialization and then reduces the distortion energy while ensuring that the solution is always kept within the feasible region. One known method to compute flipless initial parameterizations is Tutte's embedding method. It computes the parameterization by solving a linear system wherein each vertex is represented as a positive-weighted average of its one-ring neighbors. This is guaranteed to obtain flipless parameterization in cases where the mesh boundary is fixed to a convex shape. In our implementation, we also employ it to obtain a foldover-free initialization.

Challenges. However, the energy is numerically difficult to optimize, leading to numerous iterations and high computational cost. Our key observation is that the initialization generated by the Tutte's method usually contains extremely large distortion in some triangles due to the fixed boundary constraints. If the solver starts solving (5) with this initialization, these large distortions may lead to insufficient energy descent in the first several iterations, causing very slow convergence or even early entrapment by a local minimum. This is the main challenge in achieving high-quality parameterizations with high efficiency and practical reliability. To this end, we have developed an efficient and reliable method to solve (5).

3.3 Our method

Key idea. Our key idea is to reduce the distortions of highly distorted triangles up to some bound $K > 1$. We observe that if the distortions of all triangles are bounded by K, then only a few iterations in the optimization of (4) are necessary to obtain a good result; thus, we can significantly reduce iterations, resulting in fast convergence. By observing that the distortion $D(\mathbf{f}_i^{\mathrm{r}}, \mathbf{f}_i^{\mathrm{p}})$ is dependent on the reference triangle $\mathbf{f}_i^{\mathrm{r}}$, we seek to find appropriate reference triangles to bound $D(\mathbf{f}_i^{\mathrm{r}}, \mathbf{f}_i^{\mathrm{p}})$.

Construction of new reference triangles. Given a triangle $\mathbf{f}_i^{\mathrm{p}}$ that has a large distortion, i.e, $D(\mathbf{f}_i, \mathbf{f}_i^{\mathrm{p}}) > K$, we aim to bound its distortion. Our goal is to find a triangle in between $\mathbf{f}_i$ and $\mathbf{f}_i^{\mathrm{p}}$ as the reference $\mathbf{f}_i^{\mathrm{r}}$ for each i. To this end, we first interpolate a new Jacobian $J_i(t)$ between its Jacobian $J_i^1 := J_i = U_i S_i V_i^T$ and its optimum $J_i^0 = U_i I V_i^T$ using a parameter $t \in [0, 1]$ ($J_i(0) = J_i^0, J_i(1) = J_i$) to contain the bounded distortion, i.e., $D(J_i(t)) \le K$. Then, the vertices of the new reference triangle $\mathbf{f}_i^{\mathrm{r}} = \triangle \mathbf{v}_{i,0}^{\mathrm{r}} \mathbf{v}_{i,1}^{\mathrm{r}} \mathbf{v}_{i,2}^{\mathrm{r}}$ are constructed as:

$$\mathbf{v}_{i,j}^{\mathrm{r}}(t) = J_i^{-1}(t)\, \mathbf{v}_{i,j}^{\mathrm{p}}, \;\; j = 0, 1, 2. \tag{6}$$

There are many possible ways to interpolate $J_i(t)$ between J_i^0 and J_i^1 (see more discussions in Section 4.1). We adopt the methods proposed in [Alexa 2002; Grassia 1998; Rossignac and Vinacua 2011] to compute $J_i(t)$ via interpolating the exponential function:

$$J_i(t) = U_i \,\mathrm{diag}(\sigma_i^t, \tau_i^t) V_i^T, \quad t \in [0, 1]. \tag{7}$$

The bounded distortion constraint requires:

$$D(J_i(t)) = \frac{1}{4}\left(\sigma_i^{2t} + \sigma_i^{-2t} + \tau_i^{2t} + \tau_i^{-2t}\right) \le K, \; t \in [0, 1]. \tag{8}$$

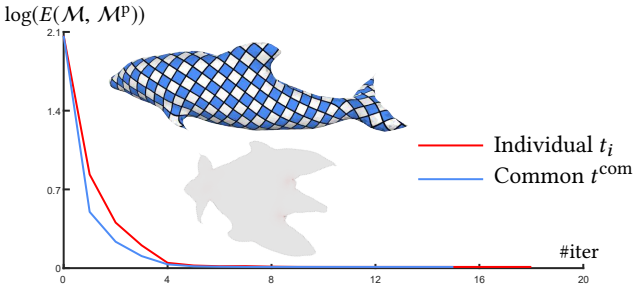


Fig. 3. Different choices of t. The middle figure shows the generated parameterization on a fish model (16428 triangles) using the common t^{com}, which is similar to the obtained parameterization via the individual t_i. The graph shows the distortion energy as a function of the number of iterations.

As the gradient of $D(J_i(t))$ with respect to $t \in [0, 1]$ is always positive, $D(J_i(t))$ is a strictly increasing function. Thus, for each individual triangle, we compute the maximum parameter t_i by solving the equation via the Newton-Raphson method:

$$\frac{1}{4}\left(\sigma_i^{2t_i} + \sigma_i^{-2t_i} + \tau_i^{2t_i} + \tau_i^{-2t_i}\right) = K. \tag{9}$$

For the triangles that already have small distortions that are less than K, i.e., $D(J_i(t)) \le K$, we set $t_i = 1$.

Note that t_i is different for various triangles $\mathbf{f}_i^{\mathrm{p}}$. To guarantee consistency, we choose a common parameter to serve as the minimum of t_i for all triangles:

$$t^{\mathrm{com}} = \min_{1 \le i \le N_{\mathrm{f}}} \{t_i\}. \tag{10}$$

Then, we use Eq. (7) to compute $J_i(t^{\mathrm{com}})$. This is also used to determine the new reference triangle $\mathbf{f}_i^{\mathrm{r}}$ via Eq. (6). Fig. 3 shows an experimental comparison between the common t^{com} and the individual t_i. This comparison shows that the former is slightly more efficient than the latter.

Updating parameterizations. With the new reference triangles $\mathbf{f}_i^{\mathrm{r}}$ (and thus the new $\mathcal{M}^{\mathrm{r}}$), we obtain a new parameterization $\mathcal{M}_{\mathrm{new}}^{\mathrm{p}}$ by solving (4). It is worth pointing out that we set the uniform weights $\omega_i = \frac{1}{N_{\mathrm{f}}}$ in (4) instead of the area of the triangle that is used in (5). As our goal is to reduce the distortions on the triangles, we expect that the distortion in each individual triangle is reduced equally. Otherwise, the triangles with large distortions but small areas may suffer from distortions that cannot be sufficiently reduced during the optimization. This equal treatment ensures that the number of convergence iterations in our method is resistant to various tessellations in one surface with almost the same cuts (see Fig. 9). Besides, some constructed reference triangles do possess very small areas, which causes numerical instability.

3.4 Algorithm

After achieving $\mathcal{M}_{\mathrm{new}}^{\mathrm{p}}$, we update $\mathcal{M}^{\mathrm{p}} = \mathcal{M}_{\mathrm{new}}^{\mathrm{p}}$. Then, the above procedure can be conducted iteratively. Algorithm 1 gives the pseudocode of our method. The termination condition also indicates $\mathcal{M}^{\mathrm{r}} = \mathcal{M}$. The details of the algorithm, including the solvers of (4) and (5), etc. , are discussed in the next sections. An example is shown in Fig. 4 to illustrate the iteration process.

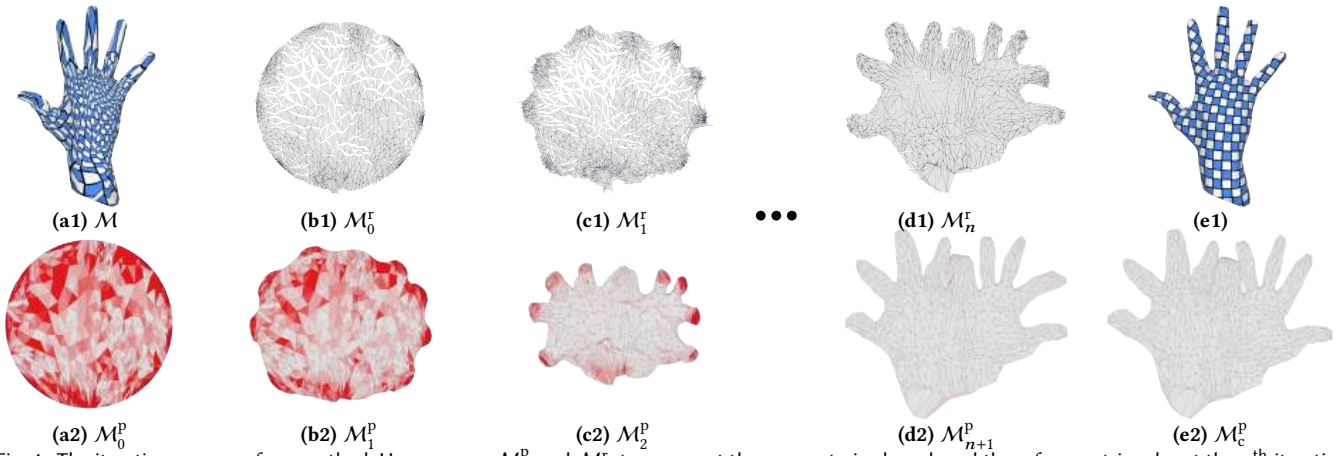


Fig. 4. The iteration process of our method. Here, we use $\mathcal{M}_n^{\mathrm{p}}$ and $\mathcal{M}_n^{\mathrm{r}}$ to represent the parameterized mesh and the reference triangles at the n^{th} iteration, respectively. Since $\mathcal{M}_{n+1}^{\mathrm{p}}$ does not satisfy the condition of the while loop of Algorithm 1, our method exits it and starts solving (5). The texture mapping is shown in (e1) using the convergence parameterization $\mathcal{M}_{\mathrm{c}}^{\mathrm{p}}$ (e2).

Algorithm 1: Progressive parameterization

Input : 3D triangular mesh $\mathcal{M} = \{\mathbf{f}_i, i = 1, \cdots, N_{\mathrm{f}}\}$
Output: 2D parameterization $\mathcal{M}^{\mathrm{p}} = \{\mathbf{f}_i^{\mathrm{p}}, i = 1, \cdots, N_{\mathrm{f}}\}$
Initialization: $\mathbf{f}_i^{\mathrm{p}} \leftarrow$ `TutteEmbedding` of $\mathcal{M}$;
while $\exists i, D(\mathbf{f}_i, \mathbf{f}_i^p) > K$ **do**
  $t^{\mathrm{com}} \leftarrow$ `CommonParameter` via Eq. (10);
  $\mathbf{f}_i^{\mathrm{r}} \leftarrow$ `UpdateReferences` via Eq. (6);
  $\mathbf{f}_i^{\mathrm{p}} \leftarrow$ `UpdateParameterization` via (4);
end
$\mathbf{f}_i^{\mathrm{p}} \leftarrow$ `FinalOptimization` via (5);

The reference triangles are interpolated between the ideal reference triangles and the parameterized triangles. Once the parameterized triangles are updated by solving (4), we re-construct the new reference triangles between the updated parameterized triangles and the ideal triangles. The new reference triangles progressively approach the ideal triangles until $\mathcal{M}^{\mathrm{r}} = \mathcal{M}$.

3.5 Hybrid solver

Solver choices. To solve (4) and (5), the existing solvers, e.g., SLIM [Rabinovich et al. 2017] or CM [Shtengel et al. 2017], can be utilized; however, they have their own characteristics. The SLIM solver conducts a reweighting scheme that could penalize the maximum distortion effectively, but it has a poor convergence rate. Thus, the SLIM solver is suitable for reducing serious distortion at the beginning of the distortion minimization. The CM solver contains an approximated Hessian matrix to make it converge quickly; however, it has no ability to reduce large distortion within a few iterations at the beginning of the optimization. Therefore, the CM solver can be used for quick convergence after significant distortion has been eliminated. Consequently, we utilize the SLIM solver in first several iterations, and then use the CM solver in the remaining optimization.

Conditions for solver change. For all of the examples in our experiments, when 99% of the triangles satisfy $D(\mathbf{f}_i, \mathbf{f}_i^{\mathrm{p}}) \le K$ or the relative error of $E(\mathcal{M}, \mathcal{M}^{\mathrm{p}})$ is less than 10^{-1}, we stop the SLIM solver and start the CM solver. The relative error of $E(\mathcal{M}, \mathcal{M}^{\mathrm{p}})$ at the n^{th} iteration is defined as:

$$e_n = \frac{|E(\mathcal{M}, \mathcal{M}_{n-1}^{\mathrm{p}}) - E(\mathcal{M}, \mathcal{M}_n^{\mathrm{p}})|}{E(\mathcal{M}, \mathcal{M}_n^{\mathrm{p}})}, \; n > 1, \tag{11}$$

where $\mathcal{M}_n^{\mathrm{p}}$ is the parameterized mesh at the n^{th} iteration, and e_n is set as a large positive value (e.g., 1000). When $e_n < 10^{-1}$, this indicates that the SLIM solver was not able to rapidly reduce the energy any further. If 99% of the triangles are with $D(\mathbf{f}_i, \mathbf{f}_i^{\mathrm{p}}) \le K$, then most of the triangles contain bounded distortions; therefore the large distortion penalization effect of the SLIM solver is not notable. As a result, once one of them is satisfied, we replace the SLIM solver with the CM solver. If the initializations satisfy $D(\mathbf{f}_i, \mathbf{f}_i^{\mathrm{p}}) \le K, \forall i$, our method is exactly the same as the CM method, meaning that our hybrid solver is more efficient than the individual ones (see the comparisons in Fig. 11). Judging from the extensive evaluations in Section 4.2, our solver's change conditions perform well in practice.

3.6 Implementation details and discussions

Initialization. The parameterization mesh $\mathcal{M}^{\mathrm{p}}$ is initialized with Tutte's embedding algorithm, which maps $\mathcal{M}$ to a unit circle with uniform weights. Then, $\mathcal{M}^{\mathrm{p}}$ is scaled so that it has the same area as $\mathcal{M}$. The scale is $\sqrt{|\mathcal{M}|/\pi}$, where $|\mathcal{M}|$ indicates the area of $\mathcal{M}$, and π is the area of the unit circle. One may use the cotangent weights for Tutte's embedding algorithm when they happen to be positive. In Fig. 5, we parameterize a model using them. Their slight difference indicates that our method is insensitive to these two weights.

Note that the reference code of [Shtengel et al. 2017] tries to scale the initial $\mathcal{M}^{\mathrm{p}}$ by reducing $E(\mathcal{M}, \mathcal{M}^{\mathrm{p}})$. This scale would increase the speed for some models; however, it is not robust. If the initial $\mathcal{M}^{\mathrm{p}}$ contains nearly degenerate triangles, then the scale would be extremely large (e.g., approximately 5×10^5 for one model in our experiments). This is due to the substantially large distortions on nearly degenerate triangles. Then, the $\mathcal{M}^{\mathrm{p}}$ may become so large that the solvers could hardly decrease the energy.

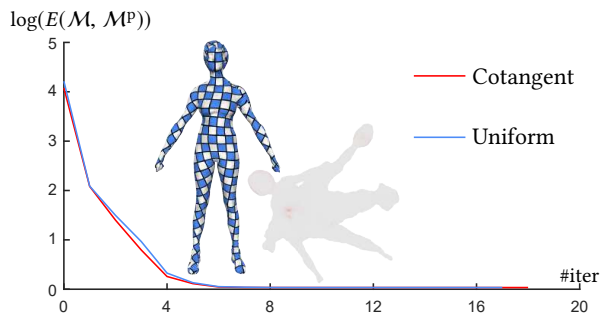


Fig. 5. The different weights for Tutte's embedding algorithm. The female model contains 19012 triangles. The graph shows the $\log(E(\mathcal{M}, \mathcal{M}^{\mathrm{p}}))$ vs. the iteration number.

Step size. In each iteration of the SLIM or CM solvers, we first compute a maximal non-inverting step size $\alpha_{\max}$, according to the method of [Smith and Schaefer 2015]. Then, we perform a standard Armijo backtracking algorithm to determine the step size, which is initialized as $\alpha := \min\{1, 0.9\alpha_{\max}\}$. The parameterized triangles are all ensured to be foldover-free in each iteration of our algorithm.

Convergence conditions of (5). The optimization of (5) terminates when one of following conditions is satisfied: (i) $e_n < 10^{-6}$; (ii) the norm of the gradient is less than 10^{-6}.

Distortion bound K. K indicates the maximum allowable distortion distance between $\mathbf{f}_i^{\mathrm{r}}$ and $\mathbf{f}_i^{\mathrm{p}}$. A small value of K shows that $D(\mathbf{f}_i, \mathbf{f}_i^{\mathrm{p}})$ will be reduced slightly, which causes many more iterations. A large one may cause great difficulties in distortion descent, as $\mathbf{f}_i$ is always treated as the reference triangle. In our experiments, we select a moderate value and set $K = 60$ for all models.

Practical termination conditions. As stated before, the condition of the while loop of Algorithm 1 is set as: $\exists i, D(\mathbf{f}_i, \mathbf{f}_i^{\mathrm{p}}) > K$. However, this condition is so strict that our method may require additional meaningless iterations to satisfy it. Therefore, we have relaxed it in practice. In our implementation, if 99% of the triangles satisfy $D(\mathbf{f}_i, \mathbf{f}_i^{\mathrm{p}}) \leq K$ or $e_n < 10^{-2}$, we exit the while loop and start solving (5). The first condition shows that almost no triangles exhibit large distortions. The second one indicates that the new reference did not significantly reduce the distortion.

One iteration in solving (4). In principle, we should solve (4) until convergence for a constructed reference. Nevertheless, the employed solvers (i.e., the SLIM and CM solvers) make a very large step size when the optimized distortion on each triangle is below K. Thus, given a constructed reference, we update $\mathcal{M}^{\mathrm{p}}$ by one step of energy descent. In Fig. 6, we show a comparison between the optimizations using convergence and one iteration for (4). Although the convergence optimization achieves lower distortion energy than the single iteration optimization, its iterations are much higher, leading to much greater computational cost for the whole algorithm.

Other requirements of parameterizations. Our method generates low isometric distortion parameterizations. However, if other distortion requirements are emphasized by users, we can add a post-processing step to satisfy them. For example, if low conformal distortion is required, the conformal AMIPS energy [Fu et al. 2015] can

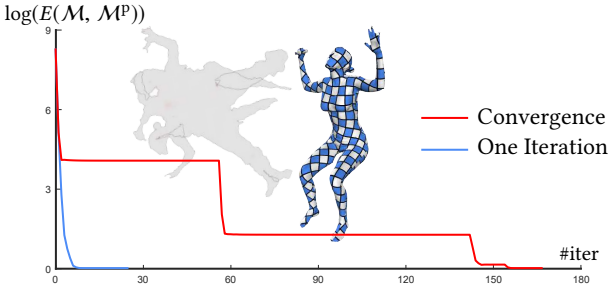


Fig. 6. A comparison of two optimization strategies for solving (4) with a constructed reference. The comparison is conducted by computing parameterizations on a mesh of a woman (43816 triangles). Since the final results are similar, we show the obtained result using the single iteration scheme in the middle. The graph plots $\log(E(\mathcal{M}, \mathcal{M}^{\mathrm{p}}))$ vs. iteration number.

be minimized. When the maximum isometric distortion requires penalization, the exponential symmetric Dirichlet energy [Rabinovich et al. 2017] can be applied.

Our method can be extended to support other constraints, e.g., bijection. For example, we could incorporate our method into [Jiang et al. 2017] to replace their solver for computing the bijective results with higher efficiency.

4 EXPERIMENTS

Our method generates foldover-free and low isometric distortion parameterizations with strong practical robustness and high efficiency. We select the scalable locally injective maps (SLIM) [Rabinovich et al. 2017], the approximate Killing vector field method (AKVF) [Claici et al. 2017], and the composite majorization method (CM) [Shtengel et al. 2017] as the competitors. The competing methods are reimplemented for fair comparisons. The accuracy is verified by performing comparisons between the reference codes provided by the authors and our reimplementations. The comparison results ensure that our implementation is correctly confirmed. The competitors also optimize symmetric Dirichlet energy $E(\mathcal{M}, \mathcal{M}^{\mathrm{p}})$ and use the same initializations and convergence conditions as our method. Except for the convergence conditions introduced in Section 3.6, our standard practice is to terminate the algorithm when the iteration number reaches its maximum, which is set at 5000 in our experiments. The number of iterations in our method is the same as the number of times of the linear solve. We employ the PARDISO solver [Petra et al. 2014a,b; Schenk et al. 2007] for the linear solve in our method and the competitors. Our experiments were performed on a desktop PC with a 4.0 GHz Intel Core i7-4790K and 8 GB of RAM. Table 1 summarizes the comparison results. The C++ implementation for optimizing the symmetric Dirichlet energy and the benchmark data set are publicly accessible at http://staff.ustc.edu.cn/~fuxm/projects/ProgressivePara/.

Distortion metrics. To determine the quality of the parameterizations and compare them with other methods, we use the convergence energy $E(\mathcal{M}, \mathcal{M}_c^{\mathrm{p}})$, where $\mathcal{M}_c^{\mathrm{p}}$ is the convergence parameterized mesh. To make $E(\mathcal{M}, \mathcal{M}_c^{\mathrm{p}})$ comparable for different models and various methods, it is weighted using the inverse of the area of mesh $\mathcal{M}$. This is denoted as $E_{\mathrm{c}} := E(\mathcal{M}, \mathcal{M}_c^{\mathrm{p}})/|\mathcal{M}|$, where $|\mathcal{M}|$ is

the area of $\mathcal{M}$. Also, the color of the triangles from the parameterized meshes are encoded via the symmetric Dirichlet energy metric $D(\mathbf{f}_i, \mathbf{f}_i^{\mathrm{p}})$, with white being optimal.

Iteration count and running time. We report the convergence iteration count N_{c} and convergence time t_{c} for every mesh and method. We do not optimize $E(\mathcal{M}, \mathcal{M}^{\mathrm{p}})$ in every iterations, but $\log(E(\mathcal{M}, \mathcal{M}^{\mathrm{p}}))$ is still plotted in the energy vs. time and energy vs. iteration number graphs. To make the running time comparable between various methods, we exclude the time required to construct the initial parameterization, which is required only once.

To determine the speed of achieving a low distortion result, we compute the number of iterations η as follows:

$$\begin{aligned} \min \quad & n \\ \text{s.t.} \quad & \frac{E(\mathcal{M}, \mathcal{M}_n^{\mathrm{p}}) - E_{\mathrm{c}}}{E_{\mathrm{c}}} \le \varepsilon, \quad 1 \le n \le N_{\mathrm{c}}, \end{aligned} \tag{12}$$

where ε is an indicator of similarity between $\mathcal{M}_n^{\mathrm{p}}$ and $\mathcal{M}_c^{\mathrm{p}}$ (we set $\varepsilon = 0.01$). A small η indicates that the approach achieves a result that is close to the convergence result with a small number of iterations (smaller η is better).

4.1 Evaluations and comparisons

Other reference triangle construction methods. Other construction methods of reference triangles could be used in our framework. We have considered two other approaches for generating $\sigma_i(t)$ and $\tau_i(t)$ to define $J_i(t) = U_i \operatorname{diag}(\sigma_i(t), \tau_i(t)) V_i^T$.

- The first one is based on a linear function:

$$\sigma_i(t) = t\sigma_i + (1-t), \quad \tau_i(t) = t\tau_i + (1-t). \tag{13}$$

  Using this interpolation, $D(J_i(t))$ is also strictly increasing with respect to t when $t \in [0, 1]$ and $D(\mathbf{f}_i, \mathbf{f}_i^{\mathrm{p}}) > K > 1$. Thus, we also use the Newton-Raphson method to compute t_i for $\mathbf{f}_i$ and choose the smallest one for all of the triangles.
- The second one is computed by projection.

$$\begin{aligned} \min_{\sigma_i(t), \tau_i(t)} \quad & (\sigma_i(t) - \sigma_i)^2 + (\tau_i(t) - \tau_i)^2 \\ \text{s.t.} \quad & \frac{1}{4}\left(\sigma_i^2(t) + \sigma_i^{-2}(t) + \tau_i^2(t) + \tau_i^{-2}(t)\right) \le K, \\ & \sigma_i(t) > 0, \ \tau_i(t) > 0. \end{aligned} \tag{14}$$

  The resulting $\sigma_i(t)$ and $\tau_i(t)$ are the projections of σ_i and τ_i onto the bounded distortion space. However, solving (14) is costly and difficult. Thus, by taking a convex subspace from the constrained space, we generate the projection-based interpolation method.

$$\begin{aligned} \min_{\sigma_i(t), \tau_i(t)} \quad & (\sigma_i(t) - \sigma_i)^2 + (\tau_i(t) - \tau_i)^2 \\ \text{s.t.} \quad & \left(\sigma_i^2(t) + \sigma_i^{-2}(t)\right) \le 2K, \ \sigma_i(t) > 0 > 0, \\ & \left(\tau_i^2(t) + \tau_i^{-2}(t)\right) \le 2K, \ \tau_i(t) > 0. \end{aligned} \tag{15}$$

  Here, the constraints $\left(x^2 + x^{-2}\right) \le 2K, \ x > 0$ are equivalent to $(K - \sqrt{K^2-1})^{\frac{1}{2}} \le x \le (K + \sqrt{K^2-1})^{\frac{1}{2}}$, which are convex. Therefore, problem (15) is a convex quadratic programming, and it is very easy to solve $\sigma_i(t)$ and $\tau_i(t)$.

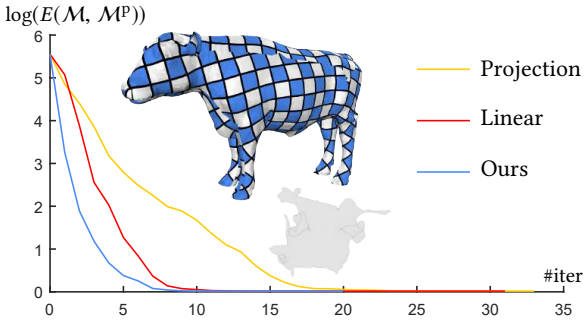


Fig. 7. Comparison with other affine transformation interpolation methods using a cow mesh (46504 triangles). We show the final parameterization generated by our exponential scheme in the middle. The graph plots $\log(E(\mathcal{M}, \mathcal{M}^{\mathrm{p}}))$ vs. iteration number.

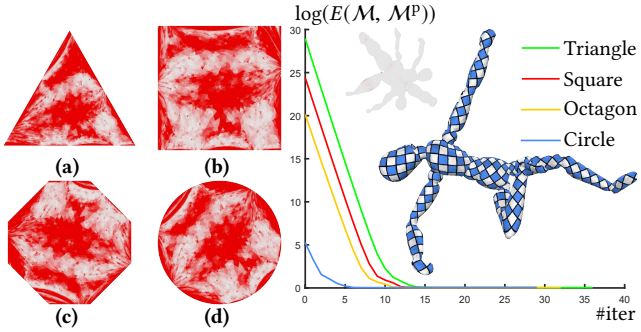


Fig. 8. Four different initializations for parameterizing a woodman model (25902 triangles): (a) triangle, (b) square, (c) octagon, and (d) circle. The graph shows $\log(E(\mathcal{M}, \mathcal{M}^{\mathrm{p}}))$ vs. number of iterations.

We show a comparison with these interpolation methods in Fig. 7. Our method, which is based on exponential function, is slightly more efficient. At the beginning of the optimization process, the linear interpolation method generates a very small value for t_i as compared to exponential interpolation, which is due to the large distortion. A small t_i value makes the interpolated reference triangles too close to the current parameterized triangles, causing some redundant iterations and slowing down the convergence. In our experiments, the common t^{com} strategy performs slightly better than the different t_i strategy in most cases. The projection method projects affine transformations onto the boundary of the constrained space, just as our interpolation method uses the different t_i strategy. Thus, it may spend more time.

Various initializations. In Fig. 8, we parameterize the same model with four initializations by mapping the input mesh onto different convex polygons via Tutte's embedding method [Tutte 1963]. The plotted graph indicates that our method converges after a similar number of iterations (36 for the triangle, 29 for the square, 32 for the octagon and 29 for the circle). This shows that our method is insensitive to various initializations.

Various tessellations. The uniform weight in $E(\mathcal{M}^{\mathrm{r}}, \mathcal{M}^{\mathrm{p}})$ makes the iteration number of our method similar for one surface that has nearly identical cuts but different tessellations. In Fig. 9, five types of tessellations representing one surface are parameterized. Our method converges after almost the same number of iterations

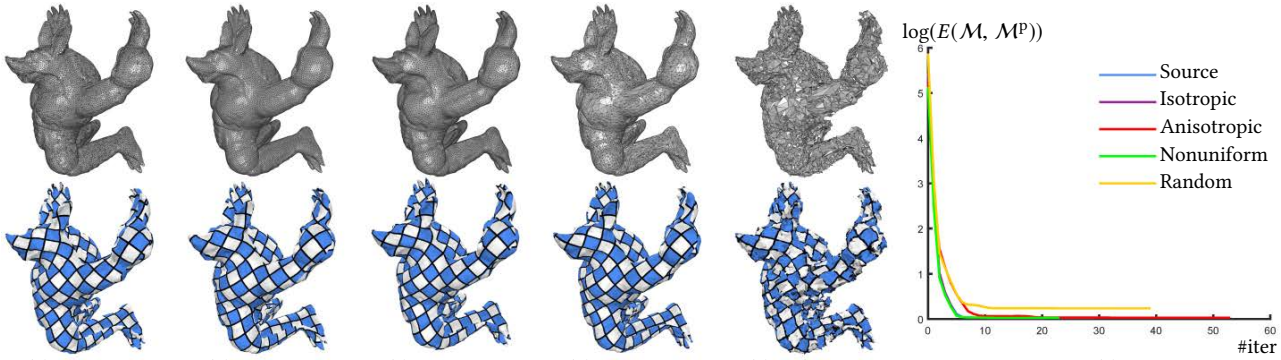


Fig. 9. Various tessellations. Five types of tessellations are included: (a) source; (b) isotropic; (c) anisotropic; (d) nonuniform; (e) random. The rightmost graph (f) plots $\log(E(\mathcal{M}, \mathcal{M}^{\mathrm{p}}))$ vs. number of iterations. The anisotropic mesh is generated by the LCT method [Fu et al. 2014].

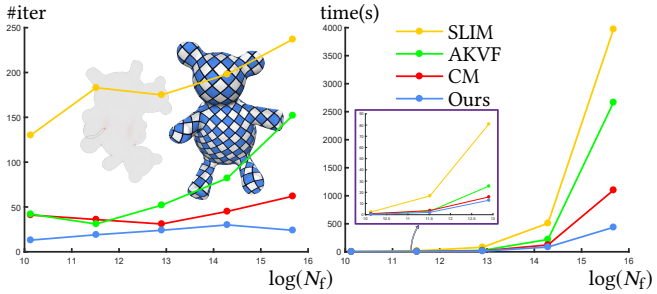


Fig. 10. Scalability test for four approaches including SLIM, AKVF, CM, and our method. A Teddy surface is represented by the meshes of increasing resolutions (25082, 100328, 401312, 1605248, and 6420992 triangles) is employed. The left graph shows the number of iterations vs. $\log(N_{\mathrm{f}})$, and the right graph plots running time (in seconds) vs. $\log(N_{\mathrm{f}})$.

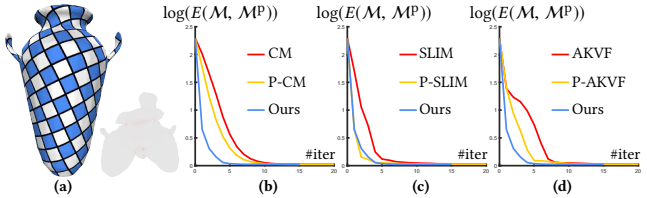


Fig. 11. Improvement of existing methods using our new reference. The parameterization is conducted on a vase model (395328 triangles). (a) the resulting parameterization. The graphs (b - d) show $\log(E(\mathcal{M}, \mathcal{M}^{\mathrm{p}}))$ vs. number of iterations.

and generates low isometric distortion parameterizations for all models. We report the statistics for SLIM, AKVF, and CM in Table 1. Especially with regard to the anisotropic one, the competitors use many more iterations (4253 for SLIM, 1151 for CM, and only 52 for ours) and much more time to converge (223.554 seconds for SLIM, 64.309 seconds for CM, and only 3.215 seconds for ours). AKVF does not converge within 5000 iterations.

Scalability. A progressive Teddy model with between 25K and 6421K faces is parameterized to compare with the competitors (Fig. 10). For our method and CM, N_{c} is almost constant for all of the models; however, our method uses about 53.46% of the iterations required by CM (left graph in Fig. 10). The SLIM and AKVF methods require even more iterations as the resolution increases. The right graph in Fig. 10 indicates that our method scales much better than the competitors with regard to high resolution meshes.

Improving existing algorithms. The existing approaches could be improved by using our progressive reference to increase efficiency and practical robustness, denoted as P-CM, P-SLIM, and P-AKVF. An example is shown in Fig. 11. From Fig. 11 (b), although the technique improves the CM, the energy descent rate in the first several iterations is still very slow. From Fig. 11 (c), it can be seen that our technique makes the SLIM decrease the energy very quickly at the beginning. However, its convergence rate is very poor, i.e., the P-SLIM does not terminate even after the 20^{th} iterations while our hybrid solver stops at the 15^{th} iteration. The AKVF is also improved significantly by our method (Fig. 11 (d)).

Efficiency comparison. We use one model in Fig. 1 and five models in Fig. 12 to compare the levels of efficiency. Our approach outperforms the competitors. As seen in the plotted graphs, the SLIM usually suppresses the maximum distortion seriously at the beginning; however, it necessitates many more iterations to converge. The AKVF method is not robust to various cuts. For example, since the cut does not reach the two fingers of the woman model (Fig. 12 (b)), it converges slowly. Although the CM method converges quickly, the several iterations required in the beginning are still not able to adequately reduce distortion. Due to the bounded distortions between the constructed reference and the parameterized triangles, our method achieves a very low energy after an almost constant number of iterations for various models (see the η value in Table 1).

4.2 Benchmark

To verify that our method is practically robust and efficient for various complex disk topology meshes, we construct a large testing benchmark containing 20712 meshes. This benchmark starts from well cut meshes $\mathcal{D}_1$, continues with moderately bad ones $\mathcal{D}_2$, and ends with extremely challenging examples $\mathcal{D}_3$. The construction of $\mathcal{D}_1$, $\mathcal{D}_2$, and $\mathcal{D}_3$ use the genus-zero closed meshes, which are provided by [Hu et al. 2018] and already contain various triangulations for one surface. We will release the complete data set in the future.

Well cut mesh set $\mathcal{D}_1$. We use the former methods [Gu et al. 2002; Sheffer 2002; Sheffer and Hart 2002] to generate $\mathcal{D}_1$. To test the

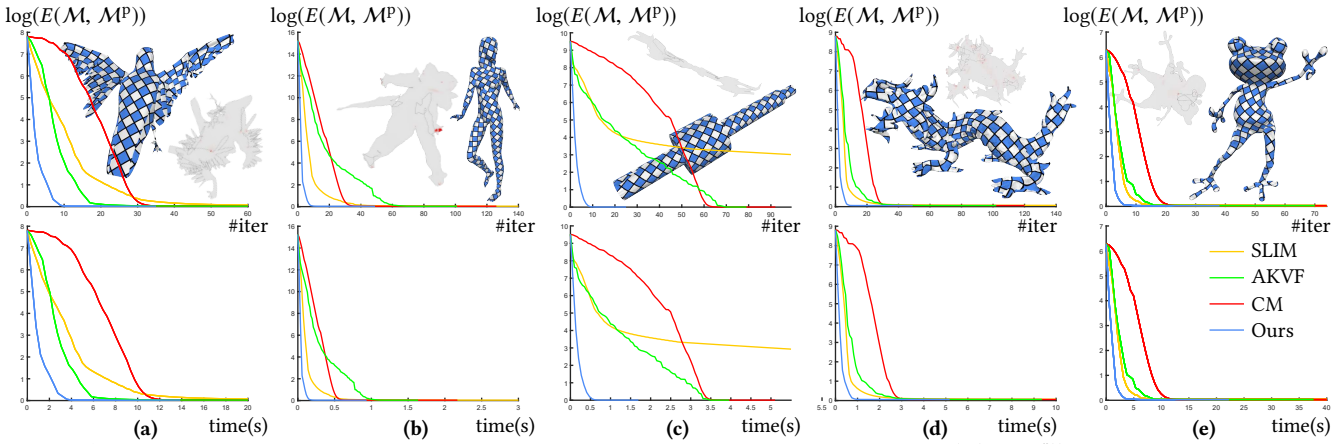


Fig. 12. Efficiency comparisons. Six models are used for comparisons. The graphs in the first row show the $\log(E(\mathcal{M}, \mathcal{M}^{\mathrm{p}}))$ vs. number of iterations, and the ones in the second row plot $\log(E(\mathcal{M}, \mathcal{M}^{\mathrm{p}}))$ vs. running time in seconds

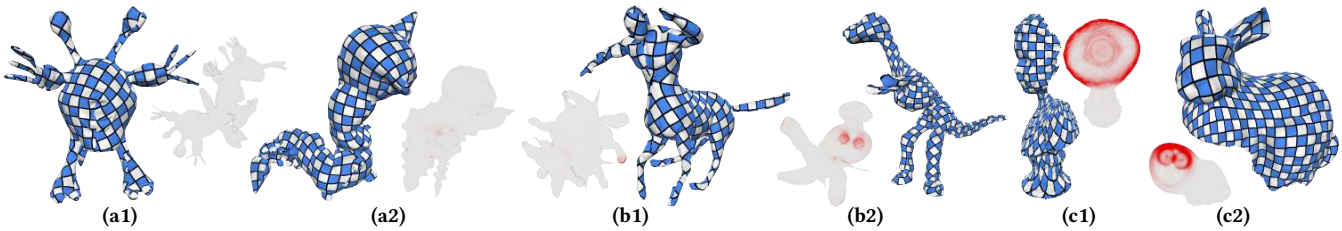


Fig. 13. Parameterizations of six models from the benchmark. The left, middle, and right two models are from $\mathcal{D}_1$, $\mathcal{D}_2$, and $\mathcal{D}_3$, respectively.

scalability, we subdivide the small sized meshes to augment the data set. In total, there are 10273 meshes in $\mathcal{D}_1$.

Bad cut collection $\mathcal{D}_2$. Given a genus-zero closed mesh $\mathcal{M}$, we first select n as far as possible vertices $\mathcal{Q}$, and then construct the minimal spanning tree following the model of [Sheffer 2002] between these vertices to define the cut. We initially set $\mathcal{Q} = \emptyset$, and add the vertex with the least distance to the lower left front corner of the bounding box of $\mathcal{M}$ into $\mathcal{Q}$. Then, we iteratively add the vertex $\mathbf{v}_k$ with the largest $\sum_{\mathbf{q}_i \in \mathcal{Q}} \mathrm{Dist}(\mathbf{v}_k, \mathbf{q}_i)$ into the $\mathcal{Q}$ until $\mathcal{Q}$ has n elements. Here, $\mathrm{Dist}(\mathbf{x}_1, \mathbf{x}_2)$ is the Euclidean distance between $\mathbf{x}_1$ and $\mathbf{x}_2$. In our experiments, we choose $n = 5, 9, 13$ to construct $\mathcal{D}_2$, which contains 6189 single patch meshes all together.

Extremely challenging cases $\mathcal{D}_3$. We first randomly select one face as the seed on the input genus-zero closed mesh $\mathcal{M}$. Then, we generate a patch $\mathcal{P}$ by propagating this seed using a width-first search strategy once the ratio between the number of faces on $\mathcal{P}$ and $\mathcal{M}$ is greater than β. Finally, we add $\mathcal{M} \setminus \mathcal{P}$ into $\mathcal{D}_3$ if it is a manifold. We select $\beta = 30\%, 50\%, 70\%$ in our implementation. To further increase the challenge, we select half of $\mathcal{D}_3$ to perform the data augmentation. For each vertex $\mathbf{v}_k$ from one selected model $\mathcal{M}$, we position it at a random point within the sphere, where the center is the original position of $\mathbf{v}_k$ and the radius is the length of one random edge in one-ring of $\mathbf{v}_k$. Then, we replace $\mathcal{M}$ with the randomly perturbed mesh. In short, there are 4250 models in $\mathcal{D}_3$.

Benchmark results. Since the performance gains in our method could originate from either the hybrid solver or the progressive references, we have added two additional methods to pinpoint the primary source of these performance gains. One method is to solve (5)

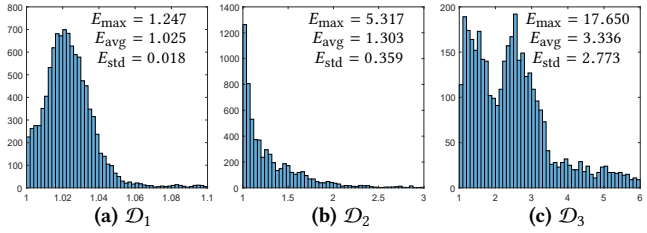


Fig. 14. Distributions of $E_{\mathrm{c}}^{\mathrm{Ours}}$. The maximum, average, and standard deviation of $E_{\mathrm{c}}^{\mathrm{Ours}}$ over all the meshes in one data set are denoted as $E_{\max}$, E_{avg}, and E_{std}, respectively.

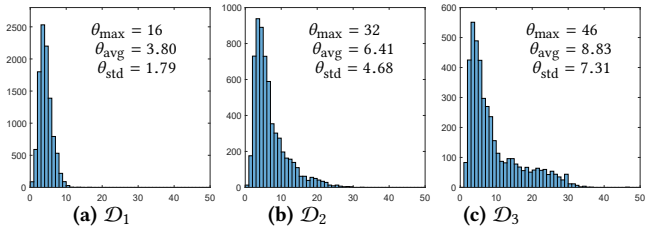
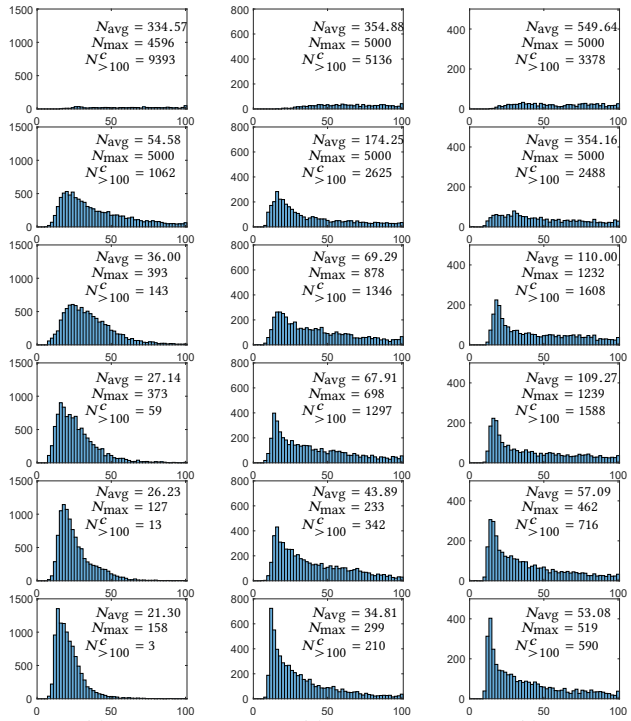


Fig. 15. Distributions of θ.

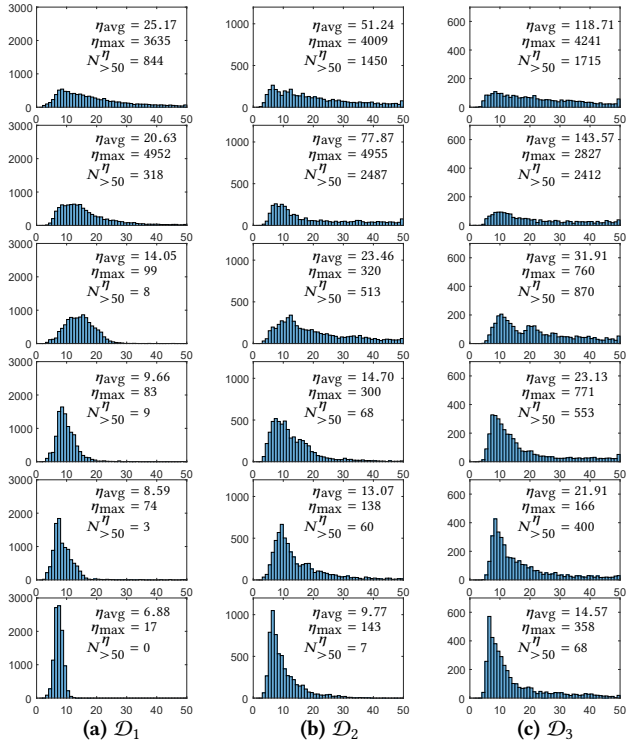
via our hybrid solver, denoted as HYBRID. The other method is the above mentioned P-CM that combines the CM solver and our progressive references. We then ran the three competitors, the two additional methods, and our method on the three data collections. The low isometric distortion parameterizations for the six models using our method are shown in Fig. 13.

We denote the convergence energies of SLIM, AKVF, CM, HYBRID, P-CM and ours as $E_{\mathrm{c}}^{\mathrm{SLIM}}$, $E_{\mathrm{c}}^{\mathrm{AKVF}}$, $E_{\mathrm{c}}^{\mathrm{CM}}$, $E_{\mathrm{c}}^{\mathrm{HYBRID}}$, $E_{\mathrm{c}}^{\mathrm{P\text{-}CM}}$, and



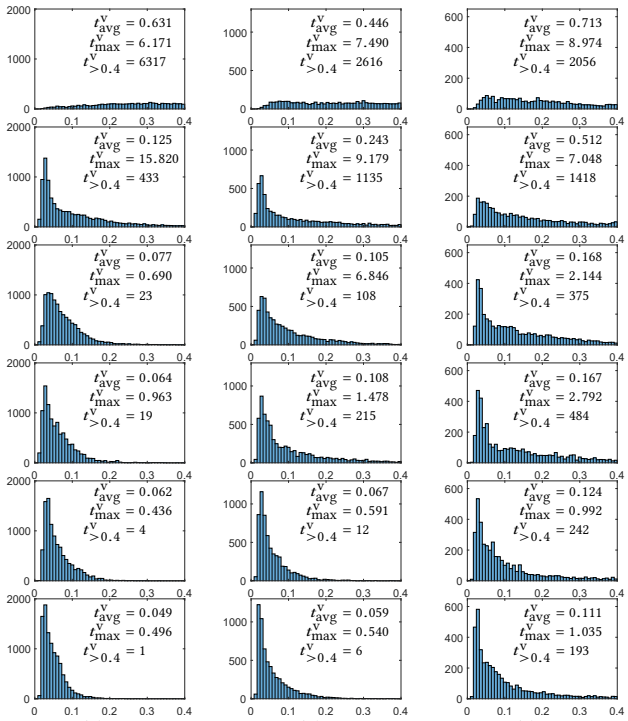
(a) $\mathcal{D}_1$ **(b)** $\mathcal{D}_2$ **(c)** $\mathcal{D}_3$

Fig. 16. Distributions of N_c. From the top, each successive row represents the results of SLIM, AKVF, CM, HYBRID, P-CM, and our method. The number of models within $N_c > 100$ in one collection is denoted as $N^c_{>100}$.



(a) $\mathcal{D}_1$ **(b)** $\mathcal{D}_2$ **(c)** $\mathcal{D}_3$

Fig. 17. Distributions of η. From the top, each successive row represents the results of SLIM, AKVF, CM, HYBRID, P-CM, and our method. The number of models within $\eta > 50$ in one collection is denoted as $N^\eta_{>50}$.



(a) $\mathcal{D}_1$ **(b)** $\mathcal{D}_2$ **(c)** $\mathcal{D}_3$

Fig. 18. Distributions of t^v. From the top, each successive row represents the results of SLIM, AKVF, CM, HYBRID, P-CM, and our method. The number of models within $t^v > 0.4ms$ in one collection is denoted as $t^v_{>0.4}$.

E_c^{Ours}, respectively. The averages of $\frac{E_c^{SLIM}}{E_c^{Ours}}$, $\frac{E_c^{AKVF}}{E_c^{Ours}}$, $\frac{E_c^{CM}}{E_c^{Ours}}$, $\frac{E_c^{HYBRID}}{E_c^{Ours}}$, and $\frac{E_c^{P\text{-}CM}}{E_c^{Ours}}$ over all the meshes are 1.0006, 1.0002, 1.0000, 1.0000 and 1.0000, respectively. Their standard deviations are 1.47×10^{-3}, 8.85×10^{-4}, 7.74×10^{-4}, 5.65×10^{-4}, and 8.13×10^{-4}, respectively. Thus, the result quality produced by all methods is almost the same. The histograms of our results from the three collections are shown in Fig. 14.

The number of the times of constructing reference triangles is denoted as θ. Its maximum, average, and standard deviation over all the meshes in one data set are denoted as θ_{max}, θ_{avg}, and θ_{std}, respectively. The histograms in Fig. 15 show the distributions of θ.

We count all N_c and η in all the meshes, and denote their average and maximum as N_{avg}, η_{avg} and N_{max}, η_{max}. Since the running time is affected by the size of the mesh, it is weighted with the inverse of vertex number, which denoted as $t^v := t_c/N_v$. The average and maximum of t^v are denoted as t^v_{avg} and t^v_{max}. We illustrate N_c (Fig. 16), η (Fig. 17), and t^v (Fig. 18) via histograms. From the statistics, it is apparent that P-CM outperforms HYBRID, indicating that our progressive reference generates higher performance gains than the hybrid solver. Our method outperforms the other methods according to all of these metrics. Besides, our method uses at most 20 iterations to achieve a result that is 1% different from the convergence result for the well cut models (see our maximum η in both Fig. 17 and Table 1).

Table 1. Statistics and timings for parameterizations. Although not all the results of the competitors are shown in each figure, we have reported their statistics here. A number in boldface emphasizes the best result for each comparison. We have omitted the experiments that did not converge within 5000 iterations.

Model		Ours		CM		AKVF		SLIM	
Name	N_v/N_f ($\times 10^3$)	$N_c/\eta/t_c$(s)	E_c	$N_c/\eta/t_c$(s)	E_c	$N_c/\eta/t_c$(s)	E_c	$N_c/\eta/t_c$(s)	E_c
Fig. 1	900/1792	48/9/**123.249**	1.047	92/26/229.509	1.047	198/65/486.879	1.047	384/40/889.649	1.048
Fig. 3	8/16	15/6/**0.201**	1.011	20/11/0.247	1.011	91/65/1.014	1.011	450/123/4.899	1.011
Fig. 6	23/44	25/9/**0.942**	1.023	48/22/1.555	1.023	81/46/2.430	1.023	507/46/14.815	1.024
Fig. 7	24/47	20/9/**0.831**	1.017	35/20/1.319	1.017	50/25/1.739	1.017	344/33/11.347	1.017
Fig. 8 (a)	13/26	36/17/**0.781**	1.035	100/70/1.948	1.036	80/61/1.441	1.035	–/–/–	–
Fig. 8 (b)	13/26	29/15/**0.626**	1.035	81/59/1.584	1.035	72/43/1.291	1.035	390/45/6.725	1.036
Fig. 8 (c)	13/26	32/8/**0.694**	1.035	77/16/1.502	1.035	70/45/1.257	1.035	224/41/3.869	1.036
Fig. 8 (d)	13/26	29/8/**0.631**	1.035	80/16/1.564	1.035	59/41/1.057	1.035	180/24/3.111	1.036
Fig. 9 (a)	32/63	29/8/**1.792**	1.028	58/21/3.305	1.028	260/146/13.713	1.028	557/301/28.443	1.033
Fig. 9 (b)	29/56	21/8/**1.351**	1.024	38/18/2.155	1.024	36/19/1.902	1.024	240/14/12.027	1.024
Fig. 9 (c)	31/60	52/20/**3.215**	1.032	1151/24/64.309	1.032	–/–/–	–	4253/3685/223.554	1.066
Fig. 9 (d)	14/27	23/9/**0.522**	1.027	42/19/0.857	1.027	157/119/2.850	1.027	833/203/14.331	1.027
Fig. 9 (e)	16/31	39/12/**1.004**	1.267	92/24/2.155	1.267	182/106/3.967	1.267	658/209/13.912	1.269
Fig. 10 - 1	13/25	13/7/**0.299**	1.029	41/9/0.859	1.029	42/6/0.835	1.029	130/9/2.374	1.030
Fig. 10 - 2	51/100	19/7/**2.062**	1.029	36/9/3.679	1.028	31/6/3.032	1.029	183/10/16.744	1.029
Fig. 10 - 3	202/401	24/7/**12.864**	1.028	31/9/15.823	1.028	52/7/25.494	1.028	175/12/80.862	1.028
Fig. 10 - 4	805/1605	30/8/**84.506**	1.028	45/10/122.659	1.028	82/12/220.026	1.028	198/12/508.401	1.028
Fig. 10 - 5	3215/6421	24/8/**436.976**	1.028	62/10/1100.326	1.028	152/15/2666.282	1.028	237/15/3972.247	1.028
Fig. 11	199/395	15/7/**7.535**	1.024	29/13/13.693	1.024	44/15/20.040	1.024	139/14/59.785	1.025
Fig. 12 (a)	195/382	35/11/**13.868**	1.021	76/37/27.422	1.021	89/34/30.411	1.021	895/142/297.864	1.022
Fig. 12 (b)	12/22	49/12/**0.935**	1.036	126/38/2.170	1.036	101/70/1.636	1.036	587/87/9.099	1.036
Fig. 12 (c)	31/60	25/11/**1.695**	1.005	92/66/5.104	1.005	79/71/3.959	1.005	2677/2310/132.544	1.005
Fig. 12 (d)	58/111	49/10/**5.176**	1.057	120/34/11.655	1.057	102/46/9.357	1.057	1002/131/89.029	1.059
Fig. 12 (e)	259/512	38/10/**22.238**	1.030	93/24/51.295	1.030	70/23/37.605	1.030	477/32/248.781	1.031
Fig. 13 (a1)	227/447	31/10/**17.569**	1.020	57/33/29.969	1.020	47/22/23.822	1.020	242/17/118.182	1.020
Fig. 13 (a2)	13/26	65/10/**1.409**	1.041	121/15/2.435	1.041	187/89/3.446	1.041	1182/107/20.919	1.042
Fig. 13 (b1)	153/302	41/9/**13.229**	1.044	84/23/25.343	1.044	54/22/15.423	1.044	298/28/80.940	1.045
Fig. 13 (b2)	10/19	19/9/**0.289**	1.182	37/19/0.502	1.182	52/24/0.652	1.182	136/18/1.590	1.182
Fig. 13 (c1)	22/44	14/9/**0.688**	1.823	29/21/1.216	1.823	84/49/3.309	1.823	100/24/3.745	1.823
Fig. 13 (c2)	34/67	23/11/**1.916**	1.633	52/25/3.797	1.633	67/43/4.781	1.633	191/15/12.621	1.633

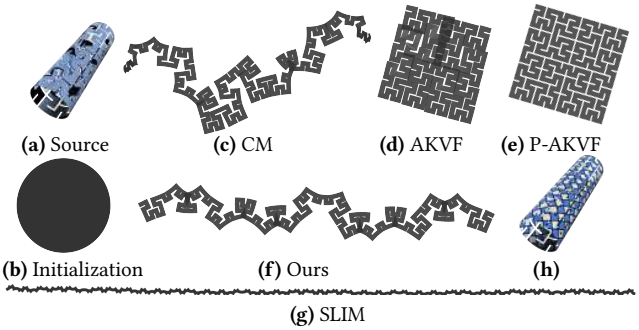


Fig. 19. A stress test on a Hilbert-curve-shaped developable surface (a). Starting from the same bijective initialization (b), the symmetric Dirichlet energy is optimized. The figures (c - g) show snapshots of the state each method achieves at the 15^{th} iteration. The texture mapping in (a) is based on the initial parameterization (b). We also show the texture mapping in (h) using the parameterization result (e).

5 CONCLUSION

Our method provides a novel and simple way to generate low isometric distortion parameterizations without foldovers. Due to the use of a new reference producing a bounded objective function, our method explicitly avoids optimizing the extremely large distortion during the whole optimization process, thereby exhibiting strong practical reliability and high efficiency. We have demonstrated the practical robustness and advantages of our method on a large data set containing 20712 models.

Hybrid solver. Our hybrid solver first uses the SLIM solver and then the CM solver. The speed at which our method converges depends on the convergence rate of the CM solver. In Fig. 19, we perform a stress test, which is similar to [Rabinovich et al. 2017; Smith and Schaefer 2015], on a Hilbert-curve-shaped developable surface. Although our method significantly reduces the distortion during the first several iterations, it uses 119 iterations to reach the global minimum. The SLIM does not reach the global minimum within 5,000 iterations. The CM uses 132 iterations, but the AKVF only requires 22 iterations. If our progressive reference is incorporated into the AKVF, i.e., P-AKVF, we are able to achieve the global minimum in a total of 15 iterations. This indicates that our progressive reference is indeed useful. In the future, an interesting topic for research would be to develop a more suitable and efficient solver for our progressive reference.

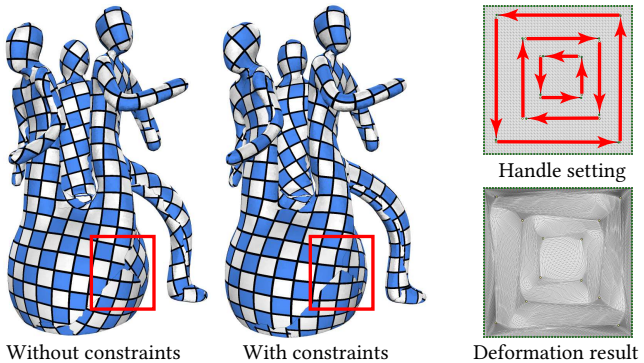


Fig. 20. Two geometric applications with constraints. (a)Global seamless parameterization with rotations on the seams that have been extracted from a cross field. (b) Mesh deformation.

Other applications with constraints. To handle the prescribed constraints, we treat them as soft constraints [Rabinovich et al. 2017; Shtengel et al. 2017]. In this manner, the constraints are gradually satisfied by increasing the weights of the soft constraint energy terms during the optimization. In Fig. 20, we show an example of global seamless parameterization [Bommes et al. 2009] and an example of mesh deformation. In these two examples, we choose the CM as the competitor. For the global seamless parameterization, the N_c and t_c of the CM and our method are (48, 4.744s) and (35, 3.884s), respectively. For the mesh deformation, the N_c and t_c of the CM and our method are (97, 2.395s) and (71, 1.802s), respectively. In these two applications, the performance gain of our method is not as high as in the parameterizations. Although high distortion is present in the deformation, it is not extreme, and our progressive references exhibit a slight acceleration. Therefore, pairing our method with soft constraints does not yield much higher performance gains than other methods with boundary or positional constraints.

Theoretical guarantee to reduce $E(\mathcal{M}, \mathcal{M}^p)$. As only $E(\mathcal{M}^r, \mathcal{M}^p)$ is optimized to drive $\mathcal{M}^p$ to approach $\mathcal{M}$ in some iterations, we have no theoretical guarantee to consistently reduce $E(\mathcal{M}, \mathcal{M}^p)$. For one triangle $\mathbf{f}_i$, if the optimization of $E(\mathcal{M}^r, \mathcal{M}^p)$ satisfies the following conditions, $D(\mathbf{f}_i, \mathbf{f}_i^p)$ decreases.

$$a_1^2 + a_1^{-2} \le b_1^2 + b_1^{-2}, \quad a_2^2 + a_2^{-2} \le b_2^2 + b_2^{-2}. \tag{16}$$

b_1 and b_2 ($b_1 > b_2$) are the singular values of $J_i(\mathbf{f}_i^r, \mathbf{f}_i^p)$ before the optimization. After the optimization, they are denoted as a_1 and

a_2 ($a_1 > a_2$). We have provided a proof in the supplementary material. Because $E(\mathcal{M}^{\mathrm{r}}, \mathcal{M}^{\mathrm{p}})$ includes the distortions of all triangles, its optimization cannot ensure that all triangles satisfy the constraints (16). Even then, as $\mathbf{f}_i^{\mathrm{r}}$ is between $\mathbf{f}_i$ and $\mathbf{f}_i^{\mathrm{p}}$ and our optimization of $E(\mathcal{M}^{\mathrm{r}}, \mathcal{M}^{\mathrm{p}})$ tries to reduce the isometric distortion between $\mathbf{f}_i^{\mathrm{r}}$ and $\mathbf{f}_i^{\mathrm{p}}$ as much as possible, our method still succeeds in making $\mathbf{f}_i^{\mathrm{p}}$ approach $\mathbf{f}_i$ in practice. Therefore, our method achieves high-quality parameterizations with strong practical robustness.

Progressive idea. The idea of applying a progressive process to solve problems is actually more general. For example, it may be a more practical way to make a progressive movement to the desired positions in handle-based deformation [Schüller et al. 2013]. Thus, adopting the progressive strategy for other applications would be an intriguing direction for future research.

ACKNOWLEDGEMENTS

The authors would like to thank Michael Rabinovich, Anna Shtengel, and Sebastian Claici for sharing their code and the anonymous reviewers for their constructive suggestions and comments. This work is supported by the National Natural Science Foundation of China (61672482,61672481), the One Hundred Talent Project of the Chinese Academy of Sciences, the Fundamental Research Funds for the Central Universities (WK0010460006), and the Anhui Provincial Natural Science Foundation (1808085QF208).

REFERENCES

Noam Aigerman, Shahar Z. Kovalsky, and Yaron Lipman. 2017. Spherical Orbifold Tutte Embeddings. *ACM Trans. Graph. (SIGGRAPH)* 36, 4 (2017), 90:1–90:13.

Noam Aigerman and Yaron Lipman. 2015. Orbifold Tutte Embeddings. *ACM Trans. Graph. (SIGGRAPH ASIA)* 34, 6 (2015), 190:1–190:12.

Noam Aigerman and Yaron Lipman. 2016. Hyperbolic Orbifold Tutte Embeddings. *ACM Trans. Graph. (SIGGRAPH ASIA)* 35, 6 (2016), 190:1–190:12.

Noam Aigerman, Roi Poranne, and Yaron Lipman. 2014. Lifted Bijections for Low Distortion Surface Mappings. *ACM Trans. Graph. (SIGGRAPH)* 33, 4 (2014), 69:1–69:12.

Marc Alexa. 2002. Linear Combination of Transformations. *ACM Trans. Graph.* 21, 3 (2002), 380–387.

Marc Alexa, Daniel Cohen-Or, and David Levin. 2000. As-Rigid-As-Possible Shape Interpolation. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '00)*. 157–164.

David Bommes, Henrik Zimmer, and Leif Kobbelt. 2009. Mixed-Integer Quadrangulation. *ACM Trans. Graph. (SIGGRAPH)* 28, 3 (2009), 77:1–77:10.

Alon Bright, Edward Chien, and Ofir Weber. 2017. Harmonic Global Parametrization with Rational Holonomy. *ACM Trans. Graph. (SIGGRAPH)* 36, 4 (2017), 89:1–89:15.

Renjie Chen, Ofir Weber, Daniel Keren, and Mirela Ben-Chen. 2013. Planar Shape Interpolation with Bounded Distortion. *ACM Trans. Graph. (SIGGRAPH)* 32, 4 (2013), 108:1–108:12.

Edward Chien, Renjie Chen, and Ofir Weber. 2016a. Bounded Distortion Harmonic Shape Interpolation. *ACM Trans. Graph. (SIGGRAPH)* 35, 4 (2016), 105:1–105:15.

Edward Chien, Zohar Levi, and Ofir Weber. 2016b. Bounded Distortion Parametrization in the Space of Metrics. *ACM Trans. Graph. (SIGGRAPH ASIA)* 35, 6 (2016), 215:1–215:16.

S Claici, M Bessmeltsev, S Schaefer, and J Solomon. 2017. Isometry-Aware Preconditioning for Mesh Parameterization. *Comput. Graph. Forum (SGP)* 36, 5 (2017), 37–47.

P. Degener, J. Meseth, and R. Klein. 2003. An Adaptable Surface Parameterization Method. In *Int. Meshing Roundtable*. 201–213.

Michael S. Floater. 2003. One-to-one Piecewise Linear Mappings over Triangulations. *Math. Comput.* 72 (2003), 685–696.

Michael S. Floater and Kai Hormann. 2005. Surface Parameterization: A Tutorial and Survey. In *In Advances in Multiresolution for Geometric Modelling*. Springer, 157–186.

Xiao-Ming Fu and Yang Liu. 2016. Computing Inversion-Free Mappings by Simplex Assembly. *ACM Trans. Graph. (SIGGRAPH ASIA)* 35, 6 (2016).

Xiao-Ming Fu, Yang Liu, and Baining Guo. 2015. Computing Locally Injective Mappings by Advanced MIPS. *ACM Trans. Graph. (SIGGRAPH)* 34, 4 (2015), 71:1–71:12.

Xiao-Ming Fu, Yang Liu, John Snyder, and Baining Guo. 2014. Anisotropic Simplicial Meshing Using Local Convex Functions. *ACM Trans. Graph. (SIGGRAPH ASIA)* 33, 6 (2014), 182:1–182:11.

F Sebastian Grassia. 1998. Practical Parameterization of Rotations Using the Exponential Map. *Journal of graphics tools* 3, 3 (1998), 29–48.

Xianfeng Gu, Steven J. Gortler, and Hugues Hoppe. 2002. Geometry Images. *ACM Trans. Graph. (SIGGRAPH)* 21, 3 (2002), 355–361.

K. Hormann and G. Greiner. 2000. MIPS: An Efficient Global Parametrization Method. In *Curve and Surface Design: Saint-Malo 1999*. Vanderbilt University Press, 153–162.

Kai Hormann, Bruno Lévy, and Alla Sheffer. 2007. Mesh Parameterization: Theory and Practice. In *ACM SIGGRAPH 2007 Courses (SIGGRAPH '07)*.

Xin Hu, Xiao-Ming Fu, and Ligang Liu. 2018. Advanced Hierarchical Spherical Parameterizations. *IEEE. T. Vis. Comput. Gr.* 24, 6 (2018), 1930–1941.

Zhongshi Jiang, Scott Schaefer, and Daniele Panozzo. 2017. Simplicial Complex Augmentation Framework for Bijective Maps. *ACM Trans. Graph. (SIGGRAPH ASIA)* 36, 6 (2017), 186:1–186:9.

Shahar Z. Kovalsky, Noam Aigerman, Ronen Basri, and Yaron Lipman. 2015. Large-scale Bounded Distortion Mappings. *ACM Trans. Graph. (SIGGRAPH ASIA)* 34, 6 (2015), 191:1–191:10.

Shahar Z. Kovalsky, Meirav Galun, and Yaron Lipman. 2016. Accelerated Quadratic Proxy for Geometric Optimization. *ACM Trans. Graph. (SIGGRAPH)* 35, 4 (2016), 134:1–134:11.

Bruno Lévy, Sylvain Petitjean, Nicolas Ray, and Jérome Maillot. 2002. Least Squares Conformal Maps for Automatic Texture Atlas Generation. *ACM Trans. Graph. (SIGGRAPH)* 21, 3 (2002), 362–371.

Yaron Lipman. 2012. Bounded Distortion Mapping Spaces for Triangular Meshes. *ACM Trans. Graph. (SIGGRAPH)* 31, 4 (2012), 108:1–108:13.

Ligang Liu, Lei Zhang, Yin Xu, Craig Gotsman, and Steven J. Gortler. 2008. A Local/Global Approach to Mesh Parameterization. *Comput. Graph. Forum (SGP)* 27, 5 (2008), 1495–1504.

Cosmin G. Petra, Olaf Schenk, and Mihai Anitescu. 2014a. Real-time Stochastic Optimization of Complex Energy Systems on High-performance Computers. *IEEE Computing in Science & Engineering* 16, 5 (2014), 32–42.

Cosmin G. Petra, Olaf Schenk, Miles Lubin, and Klaus Gärtner. 2014b. An Augmented Incomplete Factorization Approach for Computing the Schur Complement in Stochastic Optimization. *SIAM Journal on Scientific Computing* 36, 2 (2014), C139–C162.

Roi Poranne, Marco Tarini, Sandro Huber, Daniele Panozzo, and Olga Sorkine-Hornung. 2017. Autocuts: Simultaneous Distortion and Cut Optimization for UV Mapping. *ACM Trans. Graph. (SIGGRAPH ASIA)* 36, 6 (2017), 215:1–215:11.

Michael Rabinovich, Roi Poranne, Daniele Panozzo, and Olga Sorkine-Hornung. 2017. Scalable Locally Injective Maps. *ACM Trans. Graph.* 36, 2 (2017).

Jarek Rossignac and Álvar Vinacua. 2011. Steady Affine Motions and Morphs. *ACM Trans. Graph.* 30, 5 (2011), 116.

Olaf Schenk, Andreas Wächter, and Michael Hagemann. 2007. Mtching-based Preprocessing Algorithms to the Solution of Saddle-point Problems in Large-scale Nonconvex Interior-point Optimization. *Computational Optimization and Applications* 36, 2 (2007), 321–341.

Christian Schüller, Ladislav Kavan, Daniele Panozzo, and Olga Sorkine-Hornung. 2013. Locally Injective Mappings. *Comput. Graph. Forum (SGP)* 32, 5 (2013), 125–135.

Alla Sheffer. 2002. Spanning Tree Seams for Reducing Parameterization Distortion of Triangulated Surfaces. In *Shape Modeling International*. 61–66.

Alla Sheffer and John C Hart. 2002. Seamster: Inconspicuous Low-distortion Texture Seam Layout. In *Proceedings of the conference on Visualization'02*. 291–298.

Alla Sheffer, Bruno Lévy, Maxim Mogilnitsky, and Alexander Bogomyakov. 2005. ABF++: Fast and Robust Angle Based Flattening. *ACM Trans. Graph.* 24, 2 (2005), 311–330.

Alla Sheffer, Emil Praun, and Kenneth Rose. 2006. Mesh Parameterization Methods and Their Applications. *Found. Trends. Comput. Graph. Vis.* 2, 2 (2006), 105–171.

Anna Shtengel, Roi Poranne, Olga Sorkine-Hornung, Shahar Z. Kovalsky, and Yaron Lipman. 2017. Geometric Optimization via Composite Majorization. *ACM Trans. Graph. (SIGGRAPH)* 36, 4 (2017), 38:1–38:11.

Jason Smith and Scott Schaefer. 2015. Bijective Parameterization with Free Boundaries. *ACM Trans. Graph. (SIGGRAPH)* 34, 4 (2015), 70:1–70:9.

Olga Sorkine and Marc Alexa. 2007. As-Rigid-As-Possible Surface Modeling. In *Symposium on Geometry Processing*. 109–116.

Olga Sorkine, Daniel Cohen-Or, Rony Goldenthal, and Dani Lischinski. 2002. Bounded-distortion Piecewise Mesh Parameterization. In *Proceedings of the Conference on Visualization '02*. 355–362.

W. T. Tutte. 1963. How to Draw A Graph. In *Proceedings of the London Mathematical Society*, Vol. 13. 747–767.

Ofir Weber and Craig Gotsman. 2010. Controllable Conformal Maps for Shape Deformation and Interpolation. *ACM Trans. Graph. (SIGGRAPH)* 29, 4 (2010), 78:1–78:11.

Ofir Weber and Denis Zorin. 2014. Locally Injective Parametrization with Arbitrary Fixed Boundaries. *ACM Trans. Graph. (SIGGRAPH)* 33, 4 (2014), 75:1–75:12.

Kun Zhou, John Synder, Baining Guo, and Heung-Yeung Shum. 2004. Iso-charts: Stretch-driven Mesh Parameterization Using Spectral Analysis. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*. 45–54.