Reference-constrained Geometric Optimization

Anonymous cvm submission

Paper ID 172

Abstract

Many geometric optimization problems contain a reference constraint that restricts the optimized vertices on a reference mesh. This constraint is highly nonlinear and non-convex. The existing methods usually suffer from a breach of condition or low optimization quality. In this paper, we present a novel framework for geometric optimization problems with reference constraints. Central to our framework is using planar parameterizations that makes it unnecessary to deal with the reference constraint. As a consequence, our optimization is performed in the parameterized domains. The parameterizations should possess very low isometric distortions to maintain the optimization results when mapping them from the plane back to the reference surface. Therefore, we iteratively select one patch from the view of developability to ensure low isometric distortion, compute a planar parameterization, perform geometric optimization in the parameterized domain, and map the optimized result back to the reference mesh. We demonstrate the efficacy of our method through a variety of geometric processing tasks. Compared to existing methods, we achieve much better optimization results while satisfying the reference constraints.

1. Introduction

Geometric optimization plays a fundamental role in many computer graphics and geometric processing tasks, such as surface parameterizations, mesh deformation, mesh quality improvement. These tasks are usually formulated as constrained optimization problems. This paper focuses on the reference-constrained geometric optimization problem:

$$\begin{aligned} \min_{\mathcal{P}} \quad & E(\mathcal{P}) \\ \text{s.t.} \quad & \mathbf{p}_i \in \mathcal{R}, \forall \mathbf{p}_i \in \mathcal{P}, \end{aligned} \tag{1}$$

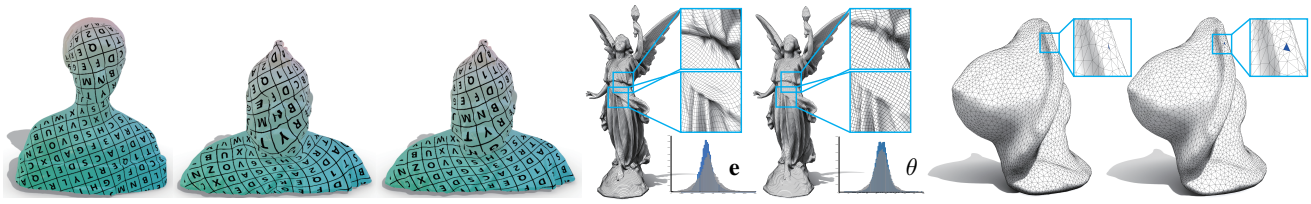
where $\mathcal{P}$ is the set of vertices to be optimized and is usually initialized as the vertices of a mesh $\mathcal{M}$, $\mathcal{R}$ is a reference mesh, and E denotes the objective energy function. In the problem, the optimized vertex $\mathbf{p}_i$ is constrained on the reference mesh $\mathcal{R}$. Many applications can be formulated as this problem, such as compatible remeshing [1, 37, 26], surface mesh quality optimization [12, 38, 36], and peeling art design [16].

This problem is very challenging. The reasons are twofold. First, it is almost impossible to calculate the derivatives of the constraints, so the common numerical optimization algorithms cannot be applied. Second, the curvature of the reference mesh $\mathcal{R}$ increases the difficulty of handling the reference constraint.

To resolve this challenging problem, one common methodology first converts the hard reference constraint to a soft constraint that restricts the vertex updates in the tangential planes and then projects the updated vertices back on the reference mesh using the closest point projection [7, 2, 21]. However, this methodology has two main limitations. First, since the optimized energy combines the energy function E with the soft constraint, the optimization process does not only minimize E, thereby causing a side effect on the optimization of E. Second, for different $\mathcal{R}$ and E, using a fixed parameter to control the tradeoff between the energy function E and the soft constraint is almost impossible.

We propose a novel optimization framework for the reference-constrained geometric optimization problem (1). The key observation of our approach is that if the reference mesh $\mathcal{R}$ is planar, the reference constraint can be converted to a linear constraint, thereby significantly reducing the difficulty of solving the problem. However, the reference mesh $\mathcal{R}$ is not planar in general. To this end, the parameterized domain is introduced to perform the geometric optimization of E. This is our key idea.

Nevertheless, parameterizations are often computed for disk-topology surfaces. Thus, each time one disk-topology patch is selected from the reference mesh $\mathcal{R}$ for being optimized. Besides, parameterizations should possess low isometric distortions to maintain the optimized results when mapping them back to the reference mesh $\mathcal{R}$. To ensure low isometric distortion, the disk-topology patch is selected from the view of developability. In a world, we iteratively perform the following four steps until convergence: (1) select a disk-topology patch, (2) parameterize the patch, (3) perform geometric optimization in plane while fixing the boundary to ease the fourth step, (4) map the optimized result back to the reference mesh.



(a) Compatible remeshing **(b) Quadrilateral mesh optimization** **(c) Minimum angle improvement**

Figure 1. Our optimization framework is applied to three reference-constrained geometric optimization problems (See more details in Section 4). For compatible remeshing, the initial compatible meshes (left and middle) contains a high distortion that is optimized to generate the resulting compatible meshes (left and right). For quadrilateral mesh optimization or minimum angle improvement, the left mesh is the input , the right is the output. For quadrilateral mesh optimization, the left histogram shows the distribution of edge length, the right histogram shows the distribution of the corner angle. From the zoomed-in views, our algorithm significantly improves the mesh quality.

Our method is capable of focusing on optimizing the target energy function without parameter tuning. We demonstrate the feasibility and efficacy of our method in various applications, including compatible remeshing, quadrilateral mesh optimization, minimum angle improvement (Figure 1). Compared to state-of-the-art methods, our method achieves better quality.

2. Related Work

Geometric optimization Geometric processing tasks are usually formulated as nonlinear minimization problems. Many numerical optimization approaches have been developed to solve these problems. Most algorithms make effort to find suitable descent directions to efficiently reduce the energy, such as the local-global solvers [23, 32, 19, 2], quasi-Newton methods [41, 31], and second-order methods [29, 5, 35, 8, 30]. Besides the aforementioned solvers, there are techniques [18, 13, 21] aimed at accelerating the existing solvers. The above methods focus on solvers.

This paper considers one type of geometric optimization problem that constrains the optimized vertices on a reference mesh. The core problems of many applications fall into this category, such as compatible remeshing [1, 37, 26], wire mesh design [7, 21], mesh quality improvement [9], and quad-mesh based design [10, 17]. As the constraints are not differentiable, one common solution is to relax the hard constraints to the soft ones, and then optimize the combined energy including optimization objectives and the soft constraints. As shown in the aforementioned section, the solution suffers from sophisticated parameters adjustment and ambiguous optimization energy. We propose a parameterization-based geometric processing method that decouples the optimization and the hard constraints.

Parameterization-based geometric processing Parameterizations are useful tools and have been applied to many problems, such as inter-surface mapping [1, 25], compatible remeshing [37], triangular mesh remeshing [4, 34]. We use parameterizations for solving the reference-constrained geometric optimization problem. To adapt to the disk topology requirement of computing parameterizations, we perform the optimization one patch by one patch.

Mesh cutting for parameterizations The parameterizations of the selected patches should have low isometric distortions. Generally, three common strategies are proposed to generate cuts for low parameterizations. First, the point-to-cut methods [40, 3, 27, 28] first detect feature points and then connect those points with a short path to construct cuts. Second, the coupling optimization methods [15, 22] simultaneously optimizes the parameterization distortion and the cut length. Those two strategies usually generate a single patch, suffering from slightly high distortion or costly computation. Third, segmentation-based methods [11, 14, 24, 39] adopt some auxiliary guidance, such as crease lines and developability, to partition the input mesh into multiple patches. Since only a fraction of the input mesh needs to be processed, low distortion is easily available. Moreover, when the patch is developable, an isometric parameterization can be easily achieved. We follow the third strategy to construct the patches. Especially, similar to the D-Chart algorithm [11], we employ a region-growing approach to generate the nearly developable patches.

3. Method

3.1. Problem overview

Inputs and goals Given a reference mesh $\mathcal{R}$ and a triangular mesh $\mathcal{M}$ whose vertices lie on $\mathcal{R}$, we seek to compute a new shape represented by a discrete set of vertices $\mathcal{P}$. Our goal is to minimize an user-specified energy while being subject to the hard constraint, i.e., the vertices $\mathcal{P}$ should lie on $\mathcal{R}$. We call this problem a *reference-constrained geometric optimization problem*. The vertices $\mathcal{P}$ is initialized by the vertices of $\mathcal{M}$.

Formulation The reference-constrained geometric optimization problem can be formulated as follows:

$$\begin{aligned} \min_{\mathcal{P}} \quad & E(\mathcal{P}) \\ \text{s.t.} \quad & \mathbf{p}_i \in \mathcal{R}, \forall \mathbf{p}_i \in \mathcal{P}, \end{aligned} \tag{2}$$

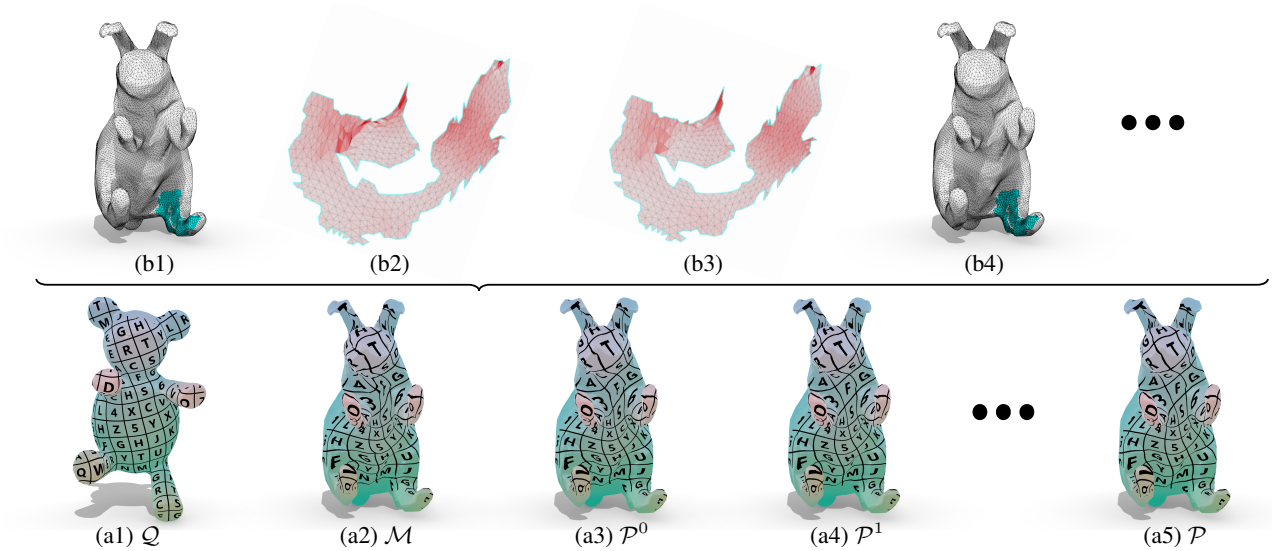


Figure 2. Workflow of our method. The initial compatible meshes are $\mathcal{Q}$ and $\mathcal{M}$. $\mathcal{P}^n$ represents the optimized mesh at the n^{th} iteration. The final result is $\mathcal{P}$. During each iteration, we perform an inner loop (upper row). At each inner iteration, we first select a nearly-developable patch (b1) that is bijectively parameterized to obtain a 2D patch (b2), then perform the geometric optimization on the parameterized domain to produce a new mesh (b3), finally map the optimized mesh back to the reference mesh (b4). We repeat these the inner iterations until all triangles in $\mathcal{M}$ are touched.

where $E(\mathcal{P})$ is the energy function that is always related to optimization objectives, design intent or fabrication requirements, such as low distortion, regularity and planarity.

Challenges The existing algorithms always employ an iterative two-step strategy that alternately performs optimization with relaxed soft constraints and projects back to the reference shape [7, 2, 21]. Since the projection operator is not differentiable and the weight between the optimization objectives and the soft constraint is difficult to tune, it often leads to inefficiency and unsatisfying results.

Methodology Our key observation is as follows: when the reference $\mathcal{R}$ is a simple plane, the hard constraint can be ignored by explicitly representing $\mathcal{P}$ as the points on the plane $\mathcal{R}$. When it comes to the general cases, i.e., $\mathcal{R}$ is a free surface, the closed-form expression for the points on $\mathcal{R}$ is unavailable. Fortunately, a planar parameterization is an useful tool to transform a surface onto the plane. With the assistance of the parameterizations, we can decouple the optimization with the constraint and ignore the hard constraint in the optimization phase. Thus, we propose a parameterization-based method to solve this optimization problem (2) (Section 3.2).

3.2. Parameteriaztion-based solution

Our algorithm is shown in Algorithm 1. Figure 2 shows an example to illustrate our pipeline. Specifically, we iteratively select nearly-developable patch (Section 3.3), parameterize the patch bijectively, perform the geometric optimization on the parameterized domain, map the optimized mesh back to $\mathcal{M}$ by barycentric interpolation, and map the optimized $\mathcal{M}$ back to the reference mesh $\mathcal{R}$ by closest point projection (Section 3.4).

ALGORITHM 1: Parameteriaztion-based geometric optimization

Input : Triangular mesh $\mathcal{M} \in \mathbb{R}^3$, Reference mesh $\mathcal{R} \in \mathbb{R}^3$
Output : Optimized shape $\mathcal{P} \in \mathbb{R}^3$
Initialize $\mathcal{P}$ with the vertices of $\mathcal{M}$.
while *the positions of* $\mathcal{P}$ *not convergence* **do**
  while *exist untouch region of* $\mathcal{M}$ **do**
    $P \leftarrow \texttt{PatchSelection}(\mathcal{M})$ (3.3);
    $P^p \leftarrow \texttt{PatchParameterization}(P)$;
    $\hat{P} \leftarrow \texttt{GeometricOptimization}(P^p)$ (3.4);
    $\mathcal{P} \leftarrow \texttt{MappingBackToReference}(\hat{P})$;
  end
end

3.3. Patch selection

Requirements The selected patch needs to satisfy two conditions. First, the patch can be parameterized to the plane with low isometric distortion, in which case the optimized patch recovered by barycentric interpolation approximates the source surface. Second, the patch should be disk-topology, which is suitable for parameterizations.

Developable surfaces We achieve the low isometric distortion goal by selecting the nearly-developable patch, simi-

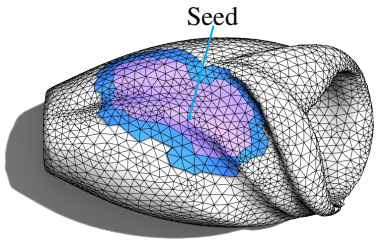


Figure 3. The patch growing from the seed consists of two parts: (1) the interior region (purple) is visited and (2) the 2-ring boundary region (blue) is untouched. The seed of next patch is randomly selected from the untouched triangles.

lar to [11]. Cone, cylinder, plane are three common types of the developable surface. They can be represented by a proxy (n_p, θ_p) where n_p is unit vector representing the axis direction and θ_p represents the angle between the surface normal and the axis. Given a triangle t with normal n_t, we define the following cost term to measure how well the triangle fits into the cone patch P:

$$E(P,t) = (< n_p, n_t > -cos\theta_p)^2 \tag{3}$$

For a given patch P, a simply connected region consisting of a set of triangles, its proxy can be computed by solving the following constrained optimization problem:

$$\min_{n_p,\theta_p} \sum_{t\in P} A_t E(P,t), \quad s.t. \quad \|n_p\|^2 = 1. \tag{4}$$

where A_t is the area of the triangle t.

Patch growing process The cone patch selection process is as follows:

1. Randomly choose a triangle t as the seed of the patch, instead of directly employing $(n_t, 0)$ as the proxy, we compute the proxies for three candidate patches—the 1-ring faces around each of the three vertices, and select the proxy with minimal cost as our initialization. Push the adjacent triangles of the seed into a priority queue Q, which is sorted according to the fitting cost in ascent order.

2. If the Q is not empty, pop the top triangle with minimal cost; otherwise, the growing process terminates.

3. If the minimal cost is less than a threshold ϵ_d, add it into the patch, push its adjacent triangles into Q, and go to step 2; otherwise, stop the algorithm.

The threshold ϵ_d controls the developability of the patch. The choice of this parameter is discussed in Section 5.1.

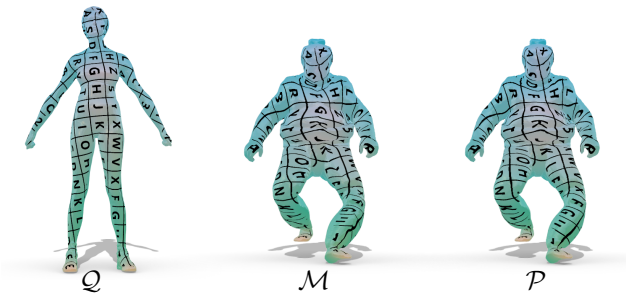


Figure 4. Compatible remeshing. The initial compatible meshes are $\mathcal{Q}$ and $\mathcal{M}$. After reducing distortion by solving (5), $\mathcal{M}$ is optimized to $\mathcal{P}$.

3.4. Optimization on patches

Once a nearly-developable patch of $\mathcal{M}$ is generated, we construct the bijective parameterizations for this patch by [33]. Then, we perform the geometric optimization on the parameterized domain. After the optimization finishes, the optimized vertices are mapped back to $\mathcal{M}$ by barycentric coordinates interpolation. Finally, the shape $\mathcal{M}$ are mapped to the reference mesh $\mathcal{R}$ by closest point projection.

Details There is no well-defined mapping between the unselected part of $\mathcal{M}$ and the 2D region outside the boundary of the parameterized domain. Moreover, the selected patch after optimization should be compatible with the rest part of $\mathcal{M}$. Thus, we keep the boundary of the selected patch fixed in the optimization for convenience.

Since the boundary is fixed during the optimization, we set the patch faces other than the 2-ring faces of the boundary as the touched state (Figure 3). Each iteration we randomly select an untouched face as the seed of the new cone patch. The inner loop of the Algorithm 1 terminates until all the faces have been touched.

4. Applications

Based on our parameterization-based solution, we implement three common reference-constrained geometric processing tasks:

- Compatible remeshing.
- Quadrilateral mesh optimization.
- Minimum angle improvement.

The only one step in the Algorithm 1 that needs to be modified to adapt to different applications is performing geometric optimization in the parameterized domain. The other steps remain the same in various applications. Thus, we focus on the altered step in the subsequent sections.

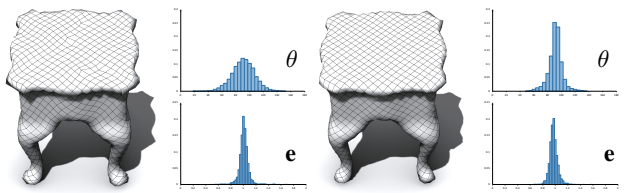


Figure 5. Quadrilateral mesh optimization. Left: the input quad mesh. Right: the resulting quad mesh after optimizing problem (7). The corner angles are concentrated at 90° and the edge lengths are more uniform after optimization.

4.1. Compatible remeshing

Given two oriented and topologically equivalent surfaces respectively represented by $\mathcal{Q}$ and $\mathcal{R}$, we aim to seek a mesh containing a set of vertices $\mathcal{P}$ that simultaneously satisfies two requirements: possesses the same topology as $\mathcal{Q}$ and represents a similar shape as $\mathcal{R}$. The latter requires that the vertices of $\mathcal{P}$ lie on $\mathcal{R}$. To enable high-quality correspondences between the two surface, the mapping distortion between $\mathcal{Q}$ and $\mathcal{P}$ should be low. The bijection lifting method [1] is employed to generate an initial $\mathcal{P}$, a mesh denoted by $\mathcal{M}$ meets the above two requirements but possesses poor correspondences to $\mathcal{Q}$. The mapping quality can be improved by solving the following optimization problem:

$$\begin{aligned} \min_{P^p} \quad & E(\mathcal{Q}, P^p) = \sum_{f \in P^p} D(J_f) \\ \text{s.t.} \quad & \text{the boundary of } P^p \text{ is fixed,} \end{aligned} \tag{5}$$

where P^p is the parameterized mesh of a selected patch P, J_f is the Jacobian matrix of the mapping between the triangle f of P^p and its corresponding triangle in $\mathcal{Q}$. $D(J_f)$ is the symmetric Dirichlet energy [31]:

$$D(J_f) = \begin{cases} \|J_f\|_F^2 + \|J_f^{-1}\|_F^2, & \text{if } \det(J_f) > 0 \\ +\infty, & \text{otherwise.} \end{cases} \tag{6}$$

where $\|\cdot\|_F$ denotes the Frobenius norm. As shown in Figure 4, the mapping distortion is effectively reduced after optimization. Note that the optimization is performed on each selected patch. We transfer texture and color to visualize the correspondence between compatible meshes, similar to [37].

4.2. Quadrilateral mesh optimization

A high-quality quadrilateral mesh should satisfy the following three geometric constraints in our view: (1) all the edge lengths are almost equal; (2) all four corner angles in each face are approximating 90°; (3) the quadrilateral mesh approximates the given reference surface $\mathcal{R}$. Instead of optimizing a combined energy that explicitly measures the deviation of edge length and angle like previous methods, we drive each corner triangle to approach an isosceles right triangle. Specifically, we introduce two virtual diagonal edges into each quadrilateral that virtually split the quadrilateral into four triangles. The resulting simplicial complex mesh P^p is optimized by driving each triangle to approach a specified reference triangle $\hat{t}$. The optimization problem is as follows:

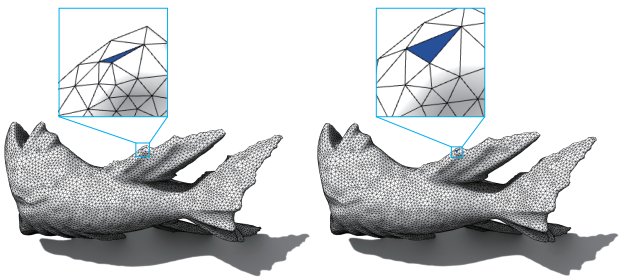


Figure 6. Minimum angle improvement. Left: the input triangle surface with the minimum angle 14.88°. Right: the optimized mesh with the minimum angle 28.72°.

$$\begin{aligned} \min_{P^p} \quad & E(P^p) = \sum_{f \in P^p} D(J_f) \\ \text{s.t.} \quad & \text{the boundary of } P^p \text{ is fixed,} \end{aligned} \tag{7}$$

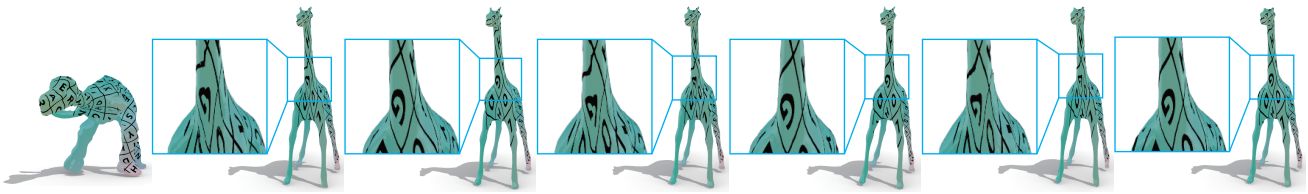
where J_f is the Jacobian matrix of the mapping between triangle f and $\hat{t}$, the reference $\hat{t}$ is an isosceles right triangle with leg length l, and $D(J_f)$ is the same as (6). Figure 5 shows the comparisons on the statistics of the corner angle and edge length before and after optimization. The comparison shows that the corner angles are concentrated at 90° and the edge lengths are more uniform after optimization. The leg length l has a significant influence on the output, which is discussed in Section 5.1. Note that in the parameterizations step, we first convert the quadrilateral mesh to triangle mesh by connecting any diagonal in each quadrilateral and then parameterize the resulting triangle mesh.

4.3. Minimum angle improvement

Mesh quality has a significant impact on the downstream applications, such as CAE, simulation. Minimum angle improvement is an essential technique to improve mesh quality. In this application, the triangular mesh $\mathcal{M}$ and the reference mesh $\mathcal{R}$ are the same in the beginning. We minimize the exponential MIPS energy [6], which can efficiently penalize the worst elements. The reference for each triangle is an equilateral triangle, whose edge length is set as the average edge length. The optimization problem for each selected patch is formulated as follows:

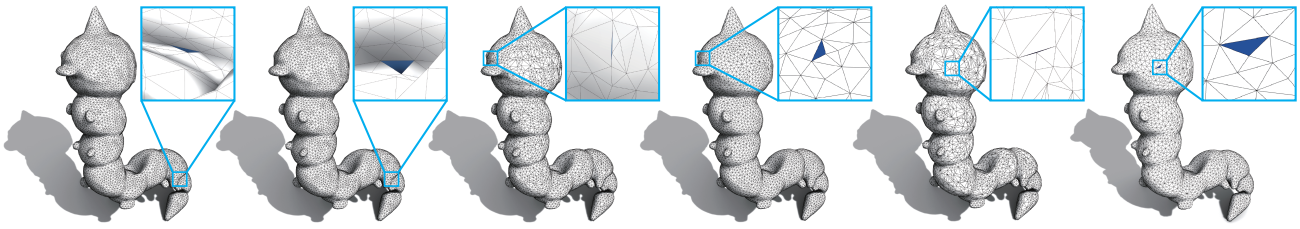
$$\begin{aligned} \min_{P^p} \quad & E(P^p) = \sum_{f \in P^p} \exp\left(\frac{\|J_f\|_F^2}{\det(J_f)}\right) \\ \text{s.t.} \quad & \text{the boundary of } P^p \text{ is fixed,} \end{aligned} \tag{8}$$

where J_f is the Jacobian matrix of the mapping between the triangle f in P^p and the equilateral triangle. We show an example in Figure 6.



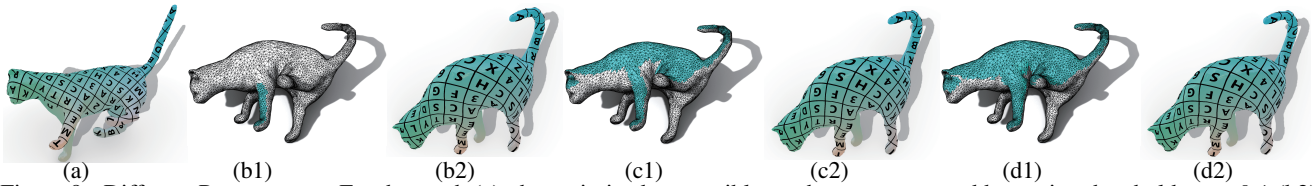
(a) (b1) (b2) (c1) (c2) (d1) (d2)

Figure 7. Different initial meshes $\mathcal{M}$ for compatible remeshing. Given a surface $\mathcal{Q}$ (a) and three different $\mathcal{M}$ (b1), (c1), and (d1), we obtain similar results (b2), (c2) and (d2), respectively.



(a1) (a2) (b1) (b2) (c1) (c2)

Figure 8. Different inputs for minimum angle improvement. The minimum angles of three different inputs (a1), (b1) and (c1) are $17.76°$, $0.81°$ and $1.49°$, respectively. After optimization, the results are (a2), (b2) and (c2) with the minimum angles $29.92°$, $22.33°$ and $17.16°$.



(a) (b1) (b2) (c1) (c2) (d1) (d2)

Figure 9. Different Parameter ϵ_d. For the mesh (a), the optimized compatible meshes are computed by setting threshold $\epsilon_d = 0.1$ (b2), $\epsilon_d = 0.25$ (c2), $\epsilon_d = 0.6$ (d2). One selected patch is shown in (b1), (c1) and (d1), respectively.

5. Experiments

We have tested our algorithm on various pairs of 3D models to evaluate its performance. Our method is implemented in C++, and all the experiments are performed on a desktop PC with a 3.60 GHz Intel Core i7-4790 and 16 GB of RAM. We adopt the solver proposed by [30] to solve the optimization problem. The linear systems are solved using the Intel® Math Kernel Library. Statistics and timings for all the demonstrated examples in three applications are reported in Table 1, 2, and 3, respectively.

Quality metrics The mapping distortion between the two compatible meshes before, after optimization are denoted as E^b, E^a, respectively. The correspondence quality improvement can be observed from both the texture and the reduction of the mapping distortion.

In quadrilateral mesh optimization, a quadrilateral mesh with uniform edge length and perpendicular corner edge directions is expected. We denote the relevant quality metrics of the edge length divided by the average length and corner angle as $e^{\dagger}_{*}$ and $\theta^{\dagger}_{*}$, respectively. The subscript $*$ includes the minimum min, maximum max, and standard deviation std. We use the superscript b and a to represented the metrics before and after the optimization.

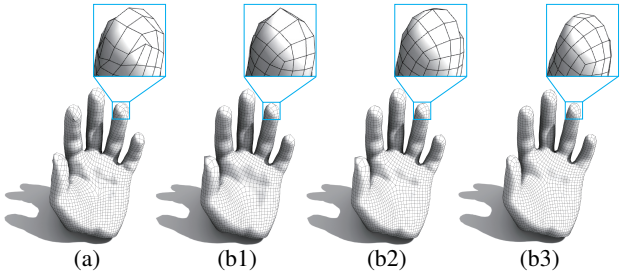
For minimum angle improvement, we compute the minimum angles of each face, and report its minimum θ_{min}, average θ_{avg}, and standard deviation θ_{std}.

5.1. Evaluations

Various initial meshes In the compatible remeshing application, three initial meshes $\mathcal{M}$ with different quality of correspondences are tested in Figure 7, where the two surface mesh $\mathcal{Q}$ and $\mathcal{R}$ are fixed. The first two initial meshes are generated by the functional maps method [20] and bijection lifting method [1], respectively. The third $\mathcal{M}$ is obtained by randomly relocating the initial mesh of [20], respectively. The mapping distortion is significantly reduced after the optimization in all the different inputs, which demonstrates the robustness of our algorithm.

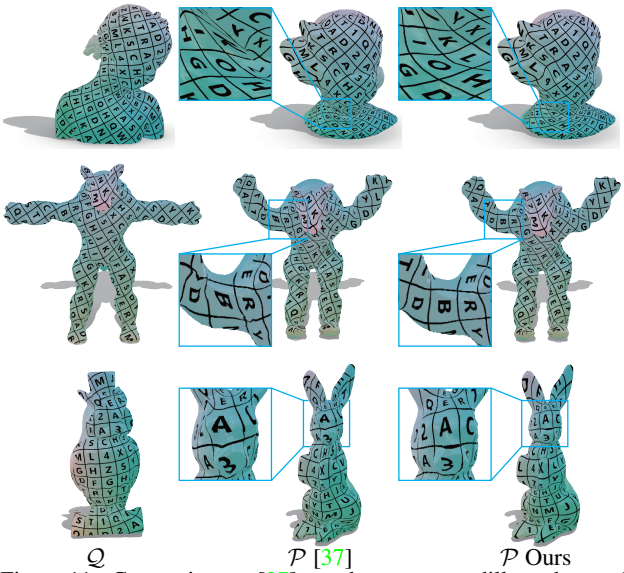
Various triangulations In the minimum angle improvement application, three types of tessellations representing the same surface are tested to demonstrates the robustness of our method. As shown in Figure 8, the minimum angles of all meshes are effectively improved.

Parameter ϵ_d The threshold ϵ_d controls the developability level of the selected patch. In general, the larger ϵ_d, the larger the size of the selected patch and the poorer developability. When ϵ_d is large, on one hand, only fewer patches are needed to cover the entire surface. On the other hand, poor



(a) (b1) (b2) (b3)

Figure 10. Different target edge lengths for quadrilateral mesh optimization. From the input quadrilateral mesh (a), the optimizing results are computed by setting target edge lengths $l = 0.2\bar{e}$ (b1), $l = \bar{e}$ (b2), $l = 5\bar{e}$ (b3).



$\mathcal{Q}$ $\mathcal{P}$ [37] $\mathcal{P}$ Ours

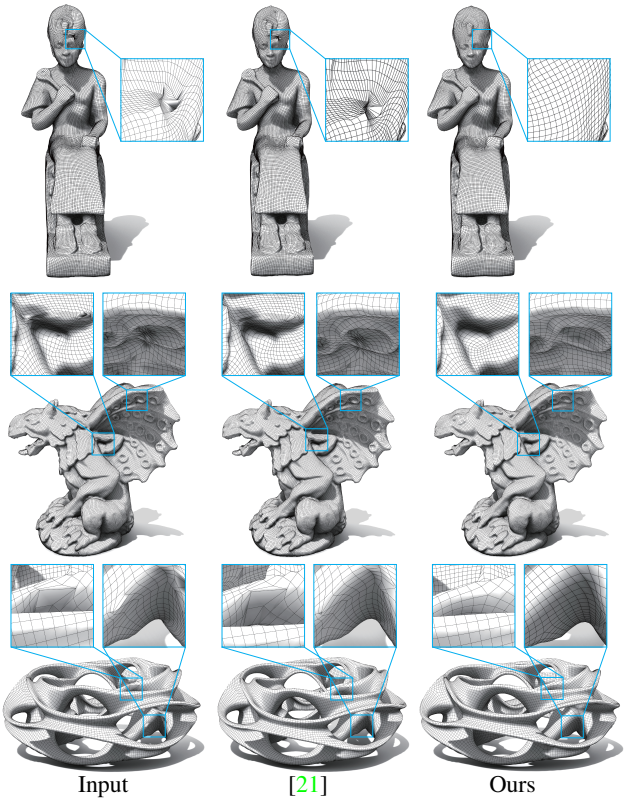
Figure 11. Comparison to [37] on a bust, an armadillo and an owl.

developability leads to slightly high parameterizations distortion. Thus, there is a tradeoff in the choice of ϵ_d (Figure 9). We set ϵ_d as 0.25 by default.

Various leg lengths In the quadrilateral mesh optimization application, the leg length l of the isosceles right triangle controls the output quadrilateral mesh edge length, and have an influence on the final result in our experiment. In Figure 10, three different ls are tested: (1) $l = 0.2\bar{e}$, (2) $l = \bar{e}$, (3) $l = 5\bar{e}$, where $\bar{e}$ is the average edge length of the input mesh. When l is set no more than $\bar{e}$, the regions around the singular vertices and the regions with high mean curvature can not be optimized effectively. When l is larger than $\bar{e}$, the optimized edges approach a large target. Since the boundary is fixed during the geometric optimization in 2D, approaching the large target leads to a more uniform mesh. In our experiment, we set $l = 5\bar{e}$ by default.

5.2. Comparisons

In the compatible remeshing application, given the initial mesh $\mathcal{M}$ generated by the bijection lifting method [1], [37] optimizes a combined energy including the as-rigid-as-possible energy and the soft constraints to reduce the mapping distortion. Figure 11 shows the comparison on three models between our method and [37]. Judging from the texture mapping, our method achieves lower distortion than [37].



Input [21] Ours

Figure 12. Comparison to [21] on an edypt women, a jercy and a garland.

The wire mesh design problem [7] shares a similar goal as ours to optimize the quad mesh. It treats the reference constraint as a soft energy. [21] uses an Anderson acceleration based solver for the wire mesh design problem. We compare our method with [21] on three models in Figure 12. As shown in Figure 12, our results are more uniform than the results of [21]. In practice, [21] surfers from poor performance when the quality of the input mesh is bad.

5.3. Testing on various models

More models are tested to demonstrated the robustness and effectiveness of our algorithm. The results of the three applications are shown in Figure 13, 14, and 15, respectively. The statistics are included in Table 1, 2, and 3, respectively.

6. Conclusion

In this paper, we present a novel framework for geometric optimization problems with reference constraints. To decouple the geometric optimization from the hard constraints,

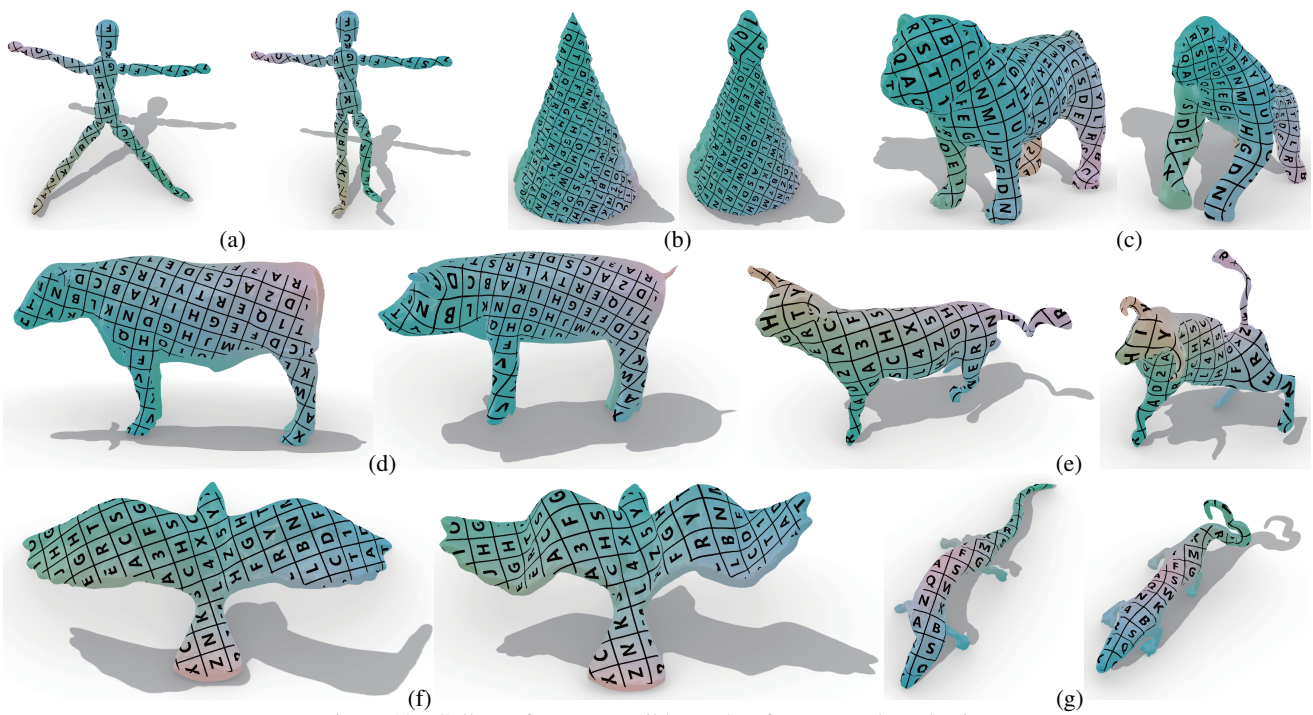


Figure 13. Gallery of our compatible meshes for seven selected pairs.

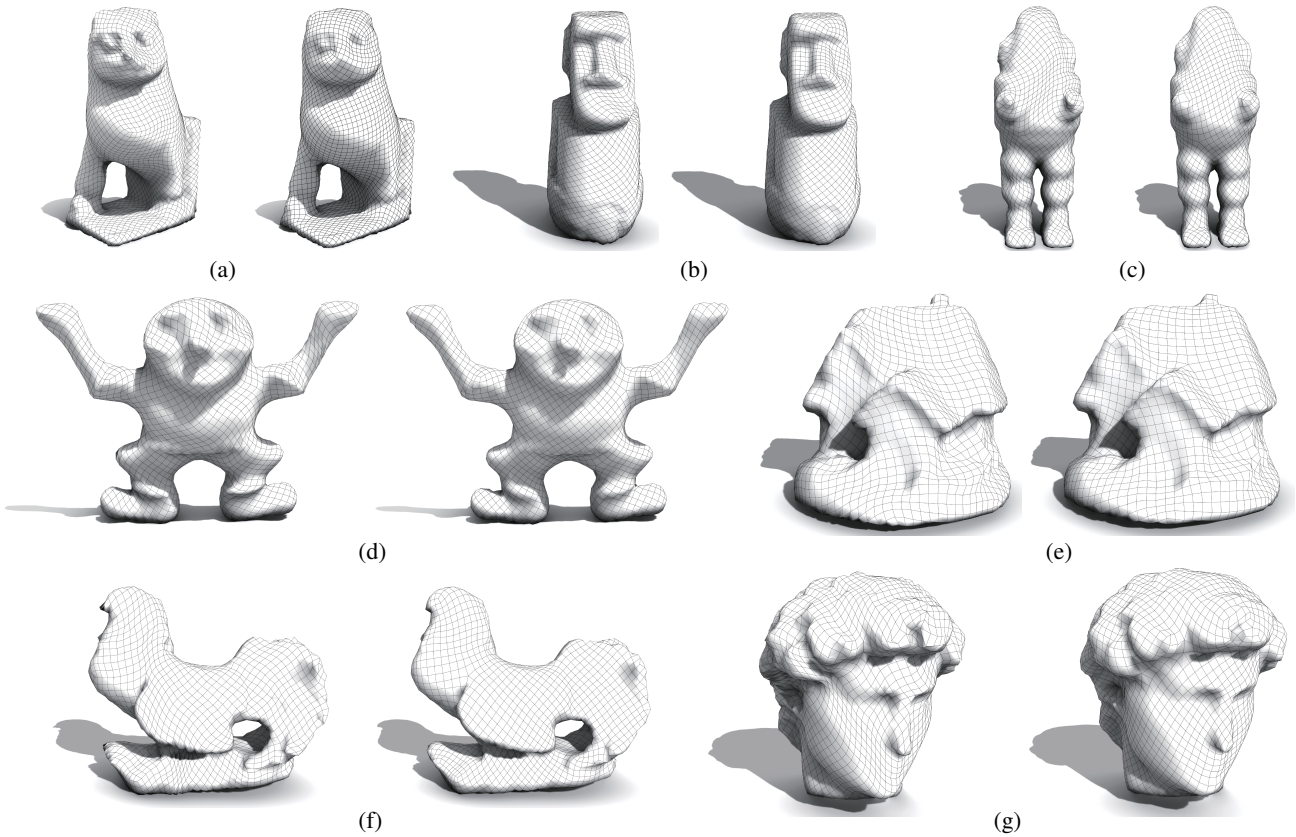


Figure 14. Gallery of quadrilateral mesh optimization for seven models. For each pair of models, the left model is the input and the right is the output.

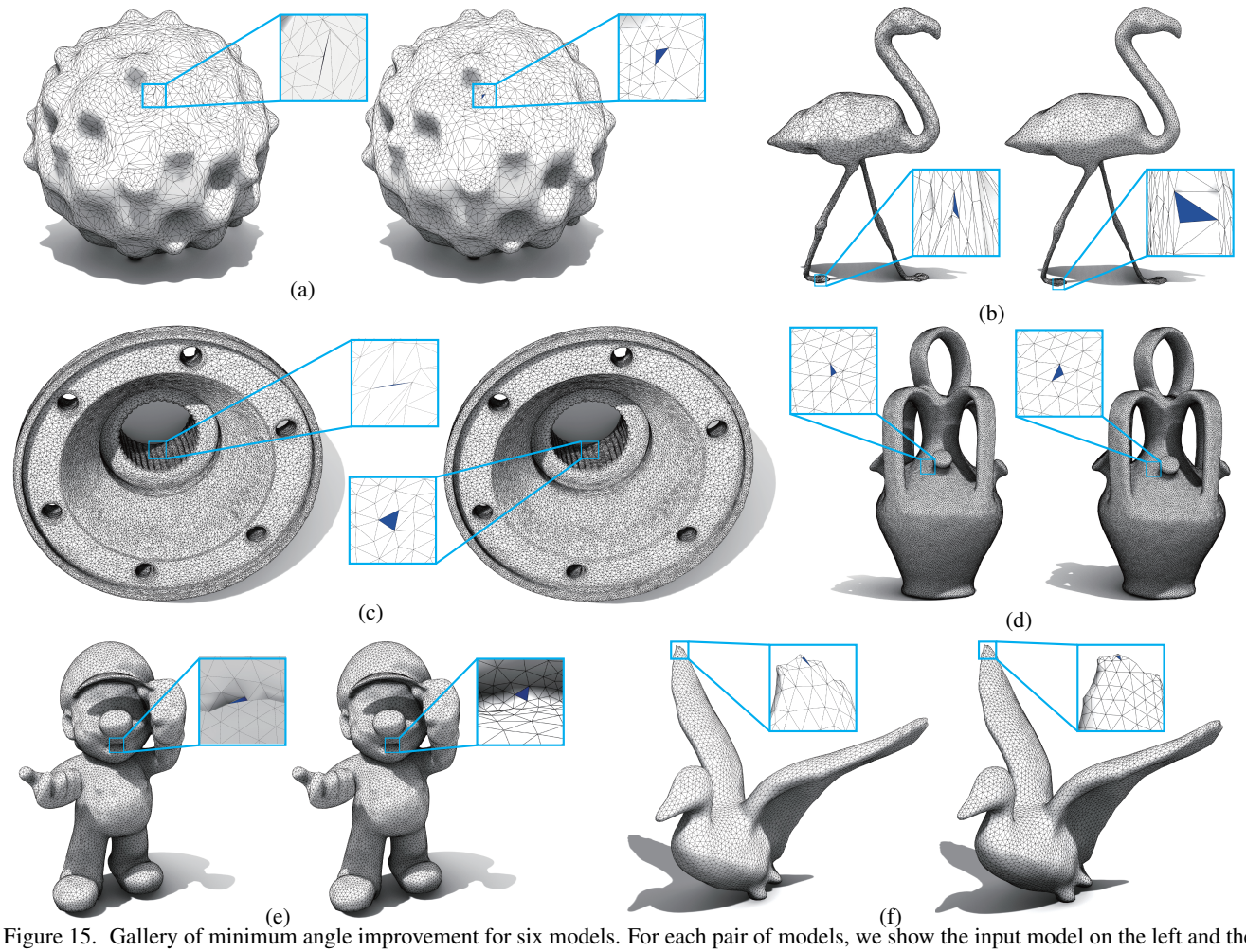


Figure 15. Gallery of minimum angle improvement for six models. For each pair of models, we show the input model on the left and the output on the right.

we iteratively select the nearly-developable patch, parameterize the patch, perform the geometric optimization on the parameterized domain, and map the optimized mesh back to the reference surface. Besides, our parameterization-based solutions are successfully employed in three common applications: compatible remeshing, quadrilateral mesh optimization, minimum angle improvement. A lot of experiments show that our algorithm outperforms the previous methods.

Fixed topology Since only the positions of the vertices are optimized and the topology keeps fixed during the optimization, it limits the space for optimization and may lead to limited improvement in the quadrilateral mesh optimization and the minimum angle improvement applications. Thus, simultaneously changing the topology and relocating the position is a possible way to obtain a result with higher quality.

High computational cost Our method is not dominant in time. Since we process the mesh piece by piece, it costs more time than the previous methods. Due to the locality of our method, one interesting future work is to explore parallel strategies to speed up our algorithm.

References

[1] N. Aigerman, R. Poranne, and Y. Lipman. Lifted bijections for low distortion surface mappings. *ACM Trans. Graph. (SIGGRAPH)*, 33(4):69:1–69:12, 2014. 1, 2, 5, 6, 7

[2] S. Bouaziz, M. Deuss, Y. Schwartzburg, T. Weise, and M. Pauly. Shape-up: Shaping discrete geometry with projections. *Comput. Graph. Forum*, 31(5), 2012. 1, 2, 3

[3] S. Chai, X.-M. Fu, X. Hu, Y. Yang, and L. Liu. Sphere-based Cut Construction for Planar Parameterizations. *Computer & Graphics (SMI 2018)*, 74:66–75, 2018. 2

[4] S. Chai, X.-M. Fu, and L. Liu. Voting for Distortion Points in Geometric Processing. *IEEE. T. Vis. Comput. Gr.*, 2019. 2

[5] X.-M. Fu and Y. Liu. Computing inversion-free mappings by simplex assembly. *ACM Trans. Graph. (SIGGRAPH ASIA)*, 35(6):216:1–216:12, 2016. 2

Model	$\#V_{in}/\#F_{in}$	E^b	E^a	N_{in}/N_{out}	time(s)
Fig.1 (a)	25026 / 50048	2.27	2.21	37 / 8	401.98
Fig.2 (a5)	14905 / 29806	2.73	2.57	60 / 8	467.63
Fig.4	44026 / 88048	2.76	2.49	119 / 10	2513.43
Fig.7 (b2)	9494 / 18984	4.31	3.89	53 / 10	701.42
Fig.7 (c2)	9494 / 18984	7.05	6.58	53 / 12	1025.68
Fig.7 (d2)	9494 / 18984	4.64	4.09	41 / 10	468.94
Fig.9 (b2)	8808 / 17612	2.76	2.69	81/ 10	448.05
Fig.9 (c2)	8808 / 17612	2.76	2.64	39 / 7	255.21
Fig.9 (d2)	8808 / 17612	2.76	2.72	16 / 6	255.21
Fig.11 Top	15516 / 31028	2.60	2.44	49 / 11	727.22
Fig.11 Mid	9436 / 18868	2.20	2.05	112 / 13	1483.32
Fig.11 Bot	12754 / 25504	2.84	2.68	37 / 10	420.58
Fig.13 (a)	10002 / 20000	2.23	2.05	115 / 6	961.50
Fig.13 (b)	20916 / 41828	2.26	2.24	39 / 4	428.90
Fig.13 (c)	13195 / 27826	2.38	2.21	38 / 7	214.82
Fig.13 (d)	26225 / 52446	2.32	2.21	62 / 8	701.71
Fig.13 (e)	12272 / 24540	2.38	2.25	65 / 9	400.91
Fig.13 (f)	18170 / 36336	2.09	2.03	36 / 8	301.59
Fig.13 (g)	13002 / 26000	2.29	2.12	70 / 10	502.57

Table 1. Statistics for compatible remeshing. The table records the number of vertices and faces of the input mesh ("$\#V_{in}/\#F_{in}$ "), the energy of the initial mapping ("E^b") and the result mapping ("E^a "), the number of average inner iterations and outer iterations ("$\#N_{in}/\#N_{out}$ "), and the running time in seconds for all the models.

[6] X.-M. Fu, Y. Liu, and B. Guo. Computing locally injective mappings by advanced MIPS. *ACM Trans. Graph. (SIGGRAPH)*, 34(4):71:1–71:12, 2015. 5

[7] A. Garg, A. O. Sageman-Furnas, B. Deng, Y. Yue, E. Grinspun, M. Pauly, and M. Wardetzky. Wire mesh design. *ACM Trans. Graph. (SIGGRAPH)*, 33(4):66:1–66:12, 2014. 1, 2, 3, 7

[8] B. Golla, H.-P. Seidel, and R. Chen. Piecewise linear mapping optimization based on the complex view. 37(7):233–243, 2018. 2

[9] K. Hu, D. Yan, D. Bommes, P. Alliez, and B. Benes. Error-bounded and feature preserving surface remeshing with minimal angle improvement. *IEEE Transactions on Visualization and Computer Graphics*, 23(12):2560–2573, 2017. 2

[10] C. Jiang, C. Wang, F. Rist, J. Wallner, and H. Pottmann. Quad-mesh based isometric mappings and developable surfaces. *ACM Transactions on Graphics*, 39(4), 2020. 2

[11] D. Julius, V. Kraevoy, and A. Sheffer. D-Charts: Quasi-Developable Mesh Segmentation. In *Comput. Graph. Forum*, volume 24, pages 581–590, 2005. 2, 4

[12] P. M. Knupp. Achieving finite element mesh quality via optimization of the jacobian matrix norm and associated quantities. part i-a framework for surface mesh optimization. *International Journal for Numerical Methods in Engineering*, 48(3):401–420, 2000. 1

[13] S. Z. Kovalsky, M. Galun, and Y. Lipman. Accelerated Quadratic Proxy for Geometric Optimization. *ACM Trans. Graph. (SIGGRAPH)*, 35(4):134:1–134:11, 2016. 2

[14] B. Lévy, S. Petitjean, N. Ray, and J. Maillot. Least squares conformal maps for automatic texture atlas generation. *ACM Trans. Graph. (SIGGRAPH)*, 21(3):362–371, 2002. 2

[15] M. Li, D. M. Kaufman, V. G. Kim, J. Solomon, and A. Sheffer. OptCuts: Joint Optimization of Surface Cuts and Parameterization. *ACM Trans. Graph. (SIGGRAPH ASIA)*, 37(6), 2018. 2

[16] H. Liu, X.-T. Zhang, X.-M. Fu, Z.-C. Dong, and L. Liu. Computational peeling art design. *ACM Transactions on Graphics(SIGGRAPH)*, 38(4):64:1–64:12, 2019. 1

[17] H.-Y. Liu, Z.-Y. Liu, Z.-Y. Zhao, L. Liu, and X.-M. Fu. Practical fabrication of discrete chebyshev nets. *Computer Graphics Forum*, 39(7), 2020. 2

[18] L. Liu, C. Ye, R. Ni, and X.-M. Fu. Progressive Parameterizations. *ACM Trans. Graph. (SIGGRAPH)*, 37(4):41:1–41:12, 2018. 2

[19] L. Liu, L. Zhang, Y. Xu, C. Gotsman, and S. J. Gortler. A local/global approach to mesh parameterization. *Comput. Graph. Forum (SGP)*, 27(5):1495–1504, 2008. 2

[20] S. Melzi, J. Ren, E. Rodolà, A. Sharma, P. Wonka, and M. Ovsjanikov. Zoomout: Spectral upsampling for efficient shape correspondence. *ACM Trans. Graph.*, 38(6), 2019. 6

[21] Y. Peng, B. Deng, J. Zhang, F. Geng, W. Qin, and L. Liu. Anderson acceleration for geometry optimization and physics simulation. *ACM Trans. Graph.*, 37(4), July 2018. 1, 2, 3, 7

[22] R. Poranne, M. Tarini, S. Huber, D. Panozzo, and O. Sorkine-Hornung. Autocuts: Simultaneous Distortion and Cut Optimization for UV Mapping. *ACM Trans. Graph. (SIGGRAPH ASIA)*, 36(6):215:1–215:11, 2017. 2

[23] M. Rabinovich, R. Poranne, D. Panozzo, and O. Sorkine-Hornung. Scalable Locally Injective Maps. *ACM Trans. Graph.*, 36(2), 2017. 2

[24] P. V. Sander, S. J. Gortler, J. Snyder, and H. Hoppe. Signal-specialized Parametrization. In *Proceedings of the 13th Eurographics Workshop on Rendering*, pages 87–98, 2002. 2

[25] P. Schmidt, J. Born, M. Campen, and L. P. Kobbelt. Distortion-minimizing injective maps between surfaces. *ACM Transactions on Graphics (TOG)*, 2019. 2

[26] P. Schmidt, M. Campen, J. Born, and L. Kobbelt. Inter-surface maps via constant-curvature metrics. *ACM Transactions on Graphics*, 39(4), 2020. 1, 2

[27] A. Sheffer. Spanning tree seams for reducing parameterization distortion of triangulated surfaces. In *Shape Modeling International*, pages 61–66, 2002. 2

[28] A. Sheffer and J. C. Hart. Seamster: inconspicuous low-distortion texture seam layout. In *Proceedings of the conference on Visualization'02*, pages 291–298, 2002. 2

[29] A. Shtengel, R. Poranne, O. Sorkine-Hornung, S. Z. Kovalsky, and Y. Lipman. Geometric Optimization via Composite Majorization. *ACM Trans. Graph. (SIGGRAPH)*, 36(4):38:1–38:11, 2017. 2

[30] B. Smith, F. D. Goes, and T. Kim. Analytic eigensystems for isotropic distortion energies. *ACM Transactions on Graphics (TOG)*, 38(1):1–15, 2019. 2, 6

Model	$\#V_{in}/\#F_{in}$	$e^b_{min}/e^b_{max}/e^b_{std}$	$e^a_{min}/e^a_{max}/e^a_{std}$	$\theta^b_{min}/\theta^b_{max}/\theta^b_{avg}/\theta^b_{std}$	$\theta^a_{min}/\theta^a_{max}/\theta^a_{avg}/\theta^a_{std}$	N_{in}/N_{out}	time(s)
Fig.5	2086/4186	0.475/1.697/0.081	0.667/1.625/0.071	19.32/146.32/88.46/16.53	48.81/132.55/88.76/9.90	37 / 7	80.66
Fig.10 (b1)	3344/6594	0.154/4.615/0.010	0.158/3.158/0.008	12.94/175.40/89.75/13.28	24.71/170.35/89.86/8.71	36 / 6	100.65
Fig.10 (b2)	3344/6594	0.154/4.615/0.010	0.158/2.895/0.007	12.94/175.40/89.75/13.28	15.77/162.37/89.86/8.98	35 / 7	123.68
Fig.10 (b3)	3344/6594	0.154/4.615/0.010	0.078/5.000/0.009	12.94/175.40/89.75/13.28	10.60/162.37/89.87/10.91	33 / 7	120.40
Fig.12 Top	42010/84012	0.066/3.200/0.001	0.472/3.010/0.001	4.58/179.96/89.88/10.70	36.86/141.22/89.92/6.56	79 / 10	2103.44
Fig.12 Mid	70439/140874	0.111/3.889/0.002	0.333/3.000/0.002	2.64/179.42/89.62/18.32	8.39/151.65/89.79/9.70	261 / 18	7034.68
Fig.12 Bot	28882/57848	0.145/5.000/0.005	0.500/4.050/0.004	4.39/179.25/89.81/15.83	33.07/133.48/89.86/7.92	104 / 14	3469.21
Fig.14 (a)	3591/7182	0.495/1.505/0.040	0.604/1.594/0.075	28.12/155.66/89.36/16.99	43.78/125.21/89.56/6.25	30 / 6	75.36
Fig.14 (b)	2669/5334	0.265/1.633/0.002	0.583/1.417/0.004	18.53/157.65/89.31/18.27	50.6/127.22/89.58/6.93	19 / 8	58.24
Fig.14 (c)	2209/4414	0.632/1.823/0.047	0.652/1.500/0.061	30.71/147.52/89.21/18.65	55.80/126.79/89.44/7.49	35 / 5	98.72
Fig.14 (d)	2200/4396	0.508/1.564/0.190	0.628/1.764/0.340	33.37/160.48/89.13/21.58	42.48/128.4/89.41/9.40	33 / 9	83.98
Fig.14 (e)	2705/5406	0.281/3.253/0.470	0.674/1.927/0.450	10.56/175.15/88.89/27.74	50.46/127.16/89.34/10.22	45 / 8	90.39
Fig.14 (f)	2996/5996	0.389/1.951/0.204	0.560/1.725/0.248	21.58/166.36/88.78/20.84	41.10/134.96/89.19/9.56	52/6	40.82
Fig.14 (g)	3030/6056	0.508/1.864/0.003	0.509/2.000/0.007	15.91/157.95/89.08/26.01	39.44/136.68/89.51/11.62	48 / 9	71.50

Table 2. Statistics for quad mesh optimization. The table records the number of vertices and faces of the input mesh ("$\#V_{in}/\#F_{in}$ "), related measures, the number of average inner iterations and outer iterations ("$\#N_{in}/\#N_{out}$ "), and the running time in seconds for all the models.

Model	$\#V_{in}/\#F_{in}$	$\theta^b_{min}/\theta^b_{avg}/\theta^b_{std}$	$\theta^a_{min}/\theta^a_{avg}/\theta^a_{std}$	N_{in}/N_{out}	time(s)
Fig.1 (c)	4460/8916	9.15/50.28/10.06	39.09/52.43/7.89	27 / 9	98.54
Fig.6	10887/21770	14.88/50.82/9.50	28.72/ 52.03/8.29	36 / 7	125.22
Fig.8 (a2)	7536/15068	17.76/51.65/8.73	29.92/52.62/7.78	69 / 5	92.11
Fig.8 (b2)	5626/11848	0.81/45.73/19.09	22.33/47.19/18.36	66 / 6	98.33
Fig.8 (c2)	4457/8910	1.49/39.86/25.26	17.16/42.57/18.36	66 / 9	150.86
Fig.15 (a)	5724 / 11444	2.72/35.72/25.81	14.27/45.04/16.06	19 / 8	145.74
Fig.15 (b)	9279/18554	8.14/41.97/25.86	16.24/41.72/19.70	46 / 10	334.957
Fig.15 (c)	49988/100000	1.48/37.69/24.15	20.85/44.91/15.99	90 / 10	1042.51
Fig.15 (d)	41158/82332	17.33/53.67/6.81	35.34/53.70/6.83	79 /10	1042.51
Fig.15 (e)	12694/25392	9.47/51.58/8.74	35.78/52.61/7.73	78 / 10	200.69
Fig.15 (f)	6306/12608	16.97/52.53/8.30	32.36/53.43/7.09	29 / 7	93.95

Table 3. Statistics for minimum angle improvement. The table records the number of vertices and faces of the input mesh ("$\#V_{in}/\#F_{in}$ "), related measures, the number of average inner iterations and outer iterations ("$\#N_{in}/\#N_{out}$ "), and the running time in seconds for all the models.

[31] J. Smith and S. Schaefer. Bijective Parameterization with Free Boundaries. *ACM Trans. Graph. (SIGGRAPH)*, 34(4):70:1–70:9, 2015. 2, 5

[32] O. Sorkine and M. Alexa. As-rigid-as-possible surface modeling. In *Symp. Geom. Proc.*, pages 109–116, 2007. 2

[33] J. P. Su, C. Ye, L. Liu, and X. M. Fu. Efficient bijective parameterizations. *ACM Transactions on Graphics*, 39(4), 2020. 4

[34] V. Surazhsky, P. Alliez, and C. Gotsman. Isotropic remeshing of surfaces: A local parameterization approach. In *In Proceedings of 12th International Meshing Roundtable*, pages 215–224, 2003. 2

[35] J. Teran, E. Sifakis, G. Irving, and R. Fedkiw. Robust quasistatic finite elements and flesh simulation. Association for Computing Machinery, 2005. 2

[36] Y. Yang, X.-M. Fu, and L. Liu. Computing surface polycube-maps by constrained voxelization. *Comput. Graph. Forum*, 38(7), 2019. 1

[37] Y. Yang, W.-X. Zhang, Y. Liu, L. Liu, and X.-M. Fu. Error-bounded compatible remeshing. *ACM Trans. Graph.*, 39(4), 2020. 1, 2, 5, 7

[38] Y. Zhang, C. Bajaj, and G. Xu. Surface smoothing and quality improvement of quadrilateral/hexahedral meshes with geometric flow. *Communications in Numerical Methods in Engineering*, 25(1):1–18, 2009. 1

[39] K. Zhou, J. Synder, B. Guo, and H.-Y. Shum. Iso-charts: Stretch-driven Mesh Parameterization Using Spectral Analysis. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, pages 45–54, 2004. 2

[40] T. Zhu, C. Ye, X.-M. Fu, and S. Chai. Greedy cut construction for parameterizations. *Computer Graphics Forum*, 39(2), 2020. 2

[41] Y. Zhu, R. Bridson, and D. M. Kaufman. Blended cured quasi-newton for distortion optimization. *ACM Trans. Graph. (SIGGRAPH)*, 37(4):40:1–40:14, 2018. 2