Computational Visual Media
https://doi.org/10.1007/s41095-022-0269-5 Vol. 9, No. 2, June 2023, 265–277

Research Article

Practical construction of globally injective parameterizations with positional constraints

Qi Wang[1], Wen-Xiang Zhang[1], Yuan-Yuan Cheng[1], Ligang Liu[1], and Xiao-Ming Fu[1] (✉)

Abstract We propose a novel method to compute globally injective parameterizations with arbitrary positional constraints on disk topology meshes. Central to this method is the use of a scaffold mesh that reduces the globally injective constraint to a locally flip-free condition. Hence, given an initial parameterized mesh containing flipped triangles and satisfying the positional constraints, we only need to remove the flips of a overall mesh consisting of the parameterized mesh and the scaffold mesh while always meeting positional constraints. To successfully apply this idea, we develop two key techniques. Firstly, an initialization method is used to generate a valid scaffold mesh and mitigate difficulties in eliminating flips. Secondly, edge-based remeshing is used to optimize the regularity of the scaffold mesh containing flips, thereby improving practical robustness. Compared to state-of-the-art methods, our method is much more robust. We demonstrate the capability and feasibility of our method on a large number of complex meshes.

Keywords globally injective parameterization; constrained parameterization; bijection; flip-free; scaffold mesh

1 Introduction

Computing parameterizations of disk topology meshes is a fundamental problem in computer graphics [1–5]. It is a basic requirement for parameterizations to be globally injective, i.e., the parameterized boundary should be self-intersection-free, and the parameterized triangles should be flip-free. Furthermore, in many applications, such as u–v editing, a set of vertices should be constrained to desired positions.

Generating desired parameterizations is difficult, and existing methods provide no theoretical guarantee that they develop an initial parameterization satisfying the above constraints. Techniques for constructing flip-free parameterizations with positional constraints often do not handle the intersection-free condition [6–9]. Thus, we need to optimize the intersection-free boundary or the positional constraints from the initialization. However, the non-linear intersection-free constraint is not straightforward to handle, as the intersecting boundary edge pairs usually change during optimization. We note that computing intersection-free and flip-free parameterizations without positional constraints is a popular research topic [10–13], while optimizing the positional constraints as a soft energy may lead to self-locking situations [14, 15].

Only one existing method handles all three types of constraints [16]. They take as input an initial mapping satisfying the positional constraints and violating the other two constraints, and then try to remove the intersecting boundaries and flipped triangles by optimizing dedicated penalty energies while keeping the positional constraints. In practice, complicated self-intersecting boundaries often cause the approach to fail to obtain the desired results (see Figs. 1 and 13).

In this paper, we propose a practical and robust method to compute globally injective parameterizations with hard positional constraints. The key idea is to use a scaffold mesh to convert the globally injective constraint into the flip-free

1 School of Mathematical Sciences, University of Science and Technology of China, Hefei 230026, China. E-mail: Q. Wang, wq2014@mail.ustc.edu.cn; W.-X. Zhang, zwx111@mail.ustc.edu.cn; Y.-Y. Cheng, chyy@mail.ustc.edu.cn; L. Liu, lgliu@ustc.eu.cn; X.-M. Fu, fuxm@ustc.edu.cn (✉).

Manuscript received: 2021-11-10; accepted: 2022-01-06

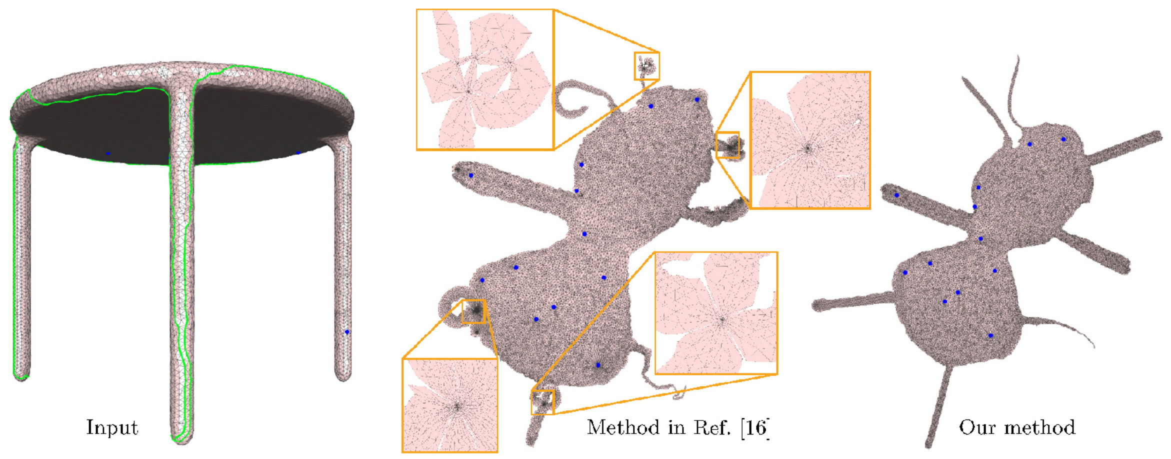


Fig. 1 For the Desk model, the method of Ref. [16] fails to generate a globally injective parameterization which satisfies the positional constraints (blue dots), whereas our method succeeds. Green lines indicate boundary edges of the input model.

constraint (Fig. 3), following Refs. [11, 17]. The scaffold mesh encloses the parameterized mesh, and its inner boundary conforms to the boundary of the parameterized mesh. If the overall mesh composed of the scaffold mesh and the parameterized mesh has no flipped triangles and its outer boundary is intersection-free, the boundary of the parameterized mesh is intersection-free [18]. Then, previous elegant methods for computing flip-free mappings with positional constraints can be used.

Two factors should be considered to ensure robustness in practice. Firstly, if the boundary of the parameterized mesh has intersections, it is non-trivial to generate a scaffold mesh whose inner boundary conforms to the boundary of the parameterized mesh. Secondly, after generating the scaffold mesh, the flip-free condition is applied to the overall mesh, so the scaffold and the parameterized mesh affect each other.

To handle the former, we propose a three-step process to initialize the parameterizations. After obtaining a globally injective parameterization without positional constraints, we initialize a valid scaffold mesh using constrained Delaunay triangulation (CDT) and then enforce the positional constraints on the initial parameterized mesh.

To handle second factor, two techniques are used: (i) in the initialization process, the initial overall mesh is similarly transformed to reduce the distances between the current positions of the constrained vertices and their target positions, and (ii) after the flip-free mapping computation algorithm terminates, if there are still flips, the scaffold mesh is remeshed, following Ref. [11]. Since the boundary of our parameterized mesh is not intersection-free, the CDT algorithm used by Ref. [11] cannot be used. Instead, the edge-based remeshing algorithm [19] is applied.

We demonstrate the practical robustness of our method on a large number of models (Figs. 2 and 11). Our approach is much more robust than previous methods in practice. For example, on the dataset provided by Ref. [16], our success rate is 99.8%, whereas the state-of-the-art method [16] only has a success rate of 85.8%.

2 Related work

2.1 Flip-free parameterization

Many methods have been proposed to realize flip-free parameterizations [5–7, 20–22].

On the one hand, some of them first initialize a

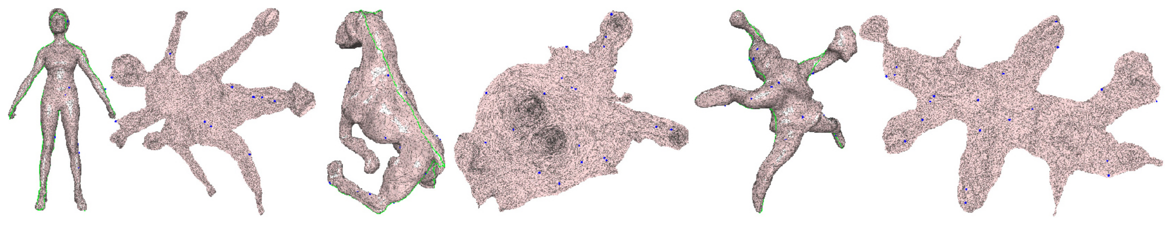


Fig. 2 Our method succeeds in constructing globally injective parameterizations with positional constraints for three complex models.

flip-free parameterization through Tutte's embedding or other variants [23–27] and then optimize flip-prevented distortion energies, such as the MIPS energy [28], the AMIPS energy [6], or the symmetric Dirichlet energy [10, 29]. Many methods have been proposed to optimize the non-linear energies, including first-order methods [6, 22, 30], quasi-Newton algorithms [31], and Newton-type methods [32–34].

On the other hand, some methods take an inverted configuration as input and try to remove the flipped elements afterwards. These include a bounded distortion mapping method [9], projection-based methods [8, 35], an assembly-based method [36], penalty-based methods [37, 38], and area-based methods [39, 40]. However, until now, there has been no theoretical guarantee that all flips will be totally removed.

2.2 Globally injective parameterization

As well as the flip-free constraint, globally injective parameterizations also require intersection-free boundaries [18]. Tutte's embedding or other variants can theoretically guarantee global injectivity, but they lead to significant distortion. To solve the problem, some methods start from a globally injective mapping that may have high distortion, and iteratively optimize the mapping, while checking for overlaps after each step, and using a barrier function [10] to prevent overlaps. However, finding candidate pairs of overlapping elements and deciding how to handle them are time-intensive, as discussed in Refs. [11, 12]. Another class of methods using a scaffold mesh to guarantee global injectivity will be described next.

2.3 Scaffolds for mapping construction

Scaffold meshes have already been used in previous work, e.g., for surface tracking [41], surface parameterization [11, 17], and collision handling [42]. As noted above, the main challenge in obtaining global injectivity is to avoid collisions between any pair of non-adjacent boundary edges. Since the number of collisions is hard to estimate in each iteration, collision detection and elimination can take a lot of time, as mentioned in Refs. [11, 12, 43]. After introducing a scaffold mesh, the globally injective constraint can be converted to a flip-free constraint [11, 18], which sidesteps the need to explicitly detect and avoid collisions. As long as the overall mesh consisting of the parameterized mesh and the scaffold mesh is flip-free, the parameterized mesh is sure to be flip-free and intersection-free. Our algorithm also uses a scaffold mesh to achieve global injectivity.

2.4 Constrained parameterization

Constructing flip-free parameterizations with positional constraints has attracted considerable research attention in recent years [7, 14, 16, 36], but without the intersection-free constraint being considered. Many methods can be successfully applied to fixed-boundary mapping problems, thus leading to an intersection-free mapping as long as the initial boundary has no intersections [8, 38, 40]. However, existing methods provide no theoretical guarantee that they generate an initial parameterization satisfying the intersection-free condition, the flip-free condition, and arbitrary positional constraints. In fact, given an intersection-free and flip-free parameterization, optimizing the positional constraints while always satisfying the globally injective condition may result in self-locking situations, as observed by Refs. [14, 15]. Du et al. [16] recover a globally injective parameterization satisfying arbitrary positional constraints from a non-injective initial mapping while meeting the positional constraints. However, their method fails for many models (see Figs. 1 and 13). We also consider these constraints, but convert the intersection-free condition to a flip-free condition, leading to a higher success rate.

3 Method

3.1 Problem and formulation

3.1.1 Inputs and goals

Given an input disk topology mesh $\mathcal{M}$, we aim to compute a parameterized mesh $\mathcal{P} \in \mathbb{R}^2$ satisfying the following requirements:

- *Intersection-free boundary*: the boundary of $\mathcal{P}$ should have no self-intersections.
- *Flip-free triangles*: the signed area of each triangle of $\mathcal{P}$ should be positive.
- *Positional constraints*: $\boldsymbol{v}_i = \boldsymbol{v}_i^h, \forall \boldsymbol{v}_i \in \mathcal{H}$, where $\mathcal{H}$ is a set of constrained vertices, and $\boldsymbol{v}_i^h$ is the target position of vertex $\boldsymbol{v}_i$.
- *Low distortion*: the piecewise affine mapping $f : \mathcal{M} \to P$ should exhibit low distortion.

The intersection-free constraint and the flip-free condition form the bijection requirement. The constrained vertex set $\mathcal{H}$ and the target positions are further inputs. $\mathcal{H}$ may include interior or boundary vertices.

3.1.2 Formulation

Computing globally injective parameterizations with positional constraints and the low distortion requirement can be formulated as a constrained optimization problem. Let $E_{\text{distortion}}(\mathcal{P})$ measure the parameterization distortion, $\mathcal{B}_p$ be a set containing all non-adjacent boundary edge pairs, $\mathcal{F}$ be the set of triangles in $\mathcal{P}$, and $\mathcal{A}(\boldsymbol{f}_i)$ denote the signed area of $\boldsymbol{f}_i$. Then we wish to minimise $E_{\text{distortion}}$ subject to the constraints:

$$\begin{aligned} \min_{\mathcal{P}} \quad & E_{\text{distortion}}(\mathcal{P}) \\ \text{s.t.} \quad & \boldsymbol{b}_i \cap \boldsymbol{b}_j = \emptyset, \quad \forall (\boldsymbol{b}_i, \boldsymbol{b}_j) \in \mathcal{B}_p, \\ & \mathcal{A}(\boldsymbol{f}_i) > 0, \quad \forall \boldsymbol{f}_i \in \mathcal{F}, \\ & \boldsymbol{v}_i = \boldsymbol{v}_i^h, \quad \forall \boldsymbol{v}_i \in \mathcal{H} \end{aligned} \tag{1}$$

3.1.3 Key idea

As explained, no existing method can generate a parameterized mesh $\mathcal{P}$ satisfying the aforementioned three constraints while providing a theoretical guarantee. Thus, we need to optimize the initial $\mathcal{P}$ to meet the constraints. In practice, the intersection-free constraint is very complicated to achieve, increasing the optimization difficulty. Our key idea is to use a scaffold mesh $\mathcal{S}$ to convert the intersection-free constraint into a flip-free constraint (see Fig. 3).

3.1.4 Scaffold meshes

To achieve the constraint conversion, $\mathcal{S}$ should satisfy the following conditions:

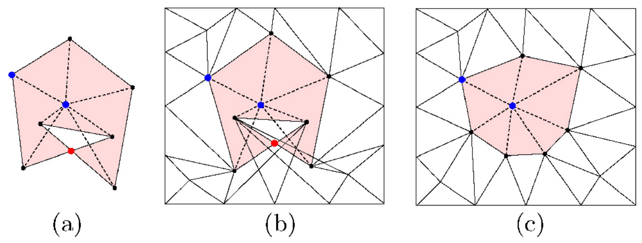


Fig. 3 Key idea. (a) It is difficult to remove both the intersections on the boundary and the flipped triangles of the parameterized mesh while satisfying the positional constraints (blue vertices). (b) Taking the parameterized mesh and the scaffold mesh (white triangles) as an overall mesh, we only need to eliminate its flipped triangles. (c) Removing flipped triangles from the overall mesh while fixing the boundary of the scaffold mesh achieves a globally injective parameterization.

- *Conforming condition*: the inner boundary of $\mathcal{S}$ should be the same as the boundary of $\mathcal{P}$.
- *Bounding condition*: $\mathcal{S}$ should tightly bound $\mathcal{P}$.
- *Boundary condition*: the outer boundary of $\mathcal{S}$ should be intersection-free.

When $\mathcal{S}$ and $\mathcal{P}$ contain no flipped triangles and $\mathcal{S}$ satisfies the conditions above, the parameterization is globally injective [18]. In practice, the boundary condition for $\mathcal{S}$ is realized as positional constraints: the initial $\mathcal{S}$ is constructed to have an intersection-free boundary, and its outer boundary vertices are fixed during the optimization process.

3.1.5 Reformulation

We reformulate Eq. (1) as Eq. (2):

$$\begin{aligned} \min_{\mathcal{P}} \quad & E_{\text{distortion}}(\mathcal{P}) \\ \text{s.t.} \quad & \mathcal{A}(\boldsymbol{f}_i) > 0, \quad \forall \boldsymbol{f}_i \in \mathcal{F}, \\ & \boldsymbol{v}_i = \boldsymbol{v}_i^h, \quad \forall \boldsymbol{v}_i \in \mathcal{H}, \\ & \mathcal{A}(\boldsymbol{f}_i^s) > 0, \quad \forall \boldsymbol{f}_i^s \in \mathcal{F}^s, \\ & \boldsymbol{v}_i^s \text{ is fixed}, \quad \forall \boldsymbol{v}_i^s \in \mathcal{V}_b^s \end{aligned} \tag{2}$$

where $\mathcal{F}^s$ is the set of triangles of $\mathcal{S}$ and $\mathcal{V}_b^s$ contains all outer boundary vertices of $\mathcal{S}$.

3.2 Algorithm workflow

3.2.1 Challenges

Using the scaffold mesh $\mathcal{S}$, only the flip-free constraint and the positional constraint remain.

Therefore, we can use existing flip-free parameterization methods to obtain the desired result. However, two challenges remain for the algorithm to be robust in practice. Firstly, when the boundary of $\mathcal{P}$ is self-intersecting, constructing $\mathcal{S}$ to satisfy the conforming condition is non-trivial. Thus, it is difficult to generate an initial $\mathcal{S}$ to make the optimization process efficient and robust. Secondly, since the flip-free constraint is enforced on both $\mathcal{S}$ and $\mathcal{P}$ in Eq. (2), the quality of $\mathcal{S}$ affects the success rate of the optimization process.

3.2.2 Pipeline

To overcome these two challenges, we propose the following algorithm workflow (see Fig. 4):

1. Generate a globally injective parameterization (the parameterized mesh is denoted $\mathcal{P}_0$) without the positional constraints, using Ref. [11].
2. Transform $\mathcal{P}_0$ similarly to minimize the distance between the current positions of constrained vertices and the target positions.

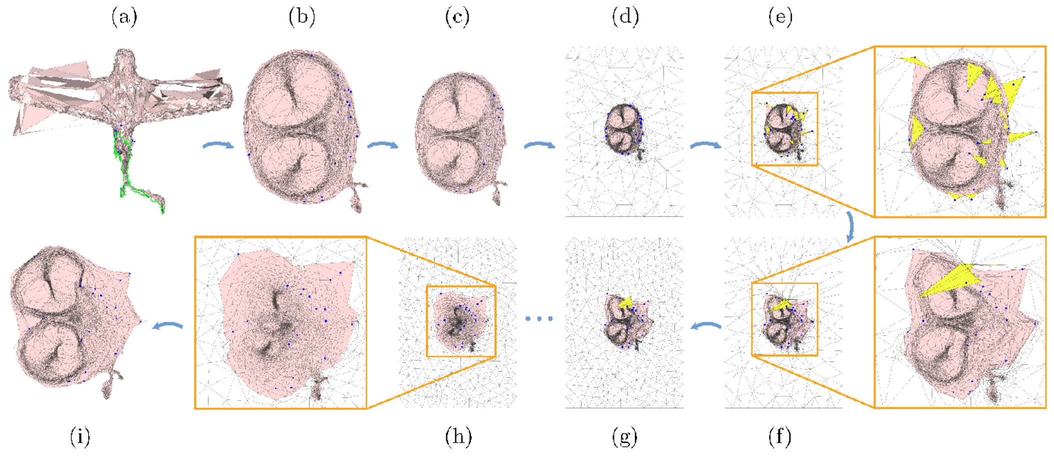


Fig. 4 Workflow of our method. Given an input triangle mesh (a), an initial globally injective parameterization $\mathcal{P}_0$ is computed using Ref. [11] (b). We transform $\mathcal{P}_0$ similarly to reduce distances between the current positions of constrained vertices and the target positions (c) and construct the scaffold mesh $\mathcal{S}$ using the CDT algorithm (d). Afterwards, we move the constrained vertices to their target positions, which may flip triangles (e), and then remove flips using Ref. [8] until termination (f). If there are flipped triangles in (f), we first perform an edge-based remeshing algorithm on $\mathcal{S}$ to improve its mesh quality (g) and then remove flips. This remeshing–removing process is iteratively conducted until no flips exist (h). Finally, we reduce the distortion while fixing the constrained vertices using Ref. [11] (i). ($\mathcal{P}_0$ in (c) and (d) has the same size; it is scaled in (d) for visualization.)

3. Construct $\mathcal{S}$ using a constrained Delaunay triangulation (CDT) algorithm, whose constrained edge set includes all boundary edges of $\mathcal{P}_0$ and the edges of a large bounding box $(BB)_0$ around $\mathcal{P}_0$.
4. Move the constrained vertices directly to their target positions.
5. Remove flipped triangles using Ref. [8]. If there are no flipped triangles, we first reduce the distortion using Ref. [11] while fixing the constrained vertices and then terminate the algorithm.
6. After performing edge-based remeshing using Ref. [19] on $\mathcal{S}$ to improve its quality, we return to Step 5, unless the maximum number of remeshing–removing iterations $n_{\max}$ has been reached, when we terminate the algorithm.

In Step 3, since the boundary of $\mathcal{P}_0$ is intersection-free, the CDT algorithm can be applied. In Step 6, we set $n_{\max} = 10$ in all experiments. Figure 5 shows several intermediate results when removing flips in Step 5. In Figs. 6–8, we show the effects of the distance reduction step (Step 2), the remeshing step (Step 6), and the distortion optimization step (Step 5), respectively.

3.3 Implementation details

3.3.1 Reducing distances

In order to reduce the influence of the initial scaffold mesh $\mathcal{S}$, accelerate the convergence speed of the optimization, and improve the success rate of the optimization process, we propose use of a distance reduction procedure (Step 2).

Let $\mathcal{V}^O = \{\boldsymbol{v}_1^o, \ldots, \boldsymbol{v}_m^o\}$ and $\mathcal{V}^{\mathcal{H}} = \{\boldsymbol{v}_1^h, \ldots, \boldsymbol{v}_m^h\}$ be the original positions of all constrained vertices on $\mathcal{P}_0$ and their corresponding target positions, respectively. The problem is converted to finding the point set $\mathcal{V}^*$ conformal to $\mathcal{V}^O$ and closest to $\mathcal{V}^{\mathcal{H}}$, i.e., to solve

$$\{\boldsymbol{v}_1^*, \ldots, \boldsymbol{v}_m^*\} = \underset{\mathcal{V}^*}{\operatorname{argmin}} \sum_{j=1}^{m} \|\boldsymbol{v}_j^* - \boldsymbol{v}_j^h\|_2^2 \qquad (3)$$

According to Ref. [44], the centroids of $\mathcal{V}^{\mathcal{H}}$ and $\mathcal{V}^*$ should coincide when reaching the minimum of Eq. (3). After $\mathcal{V}^{\mathcal{H}}$ is centered, since $\mathcal{V}^*$ is conformal to $\mathcal{V}^O$, Eq. (3) can be presented as [45]:

$$\underset{\boldsymbol{v}_{1x}^*, \boldsymbol{v}_{1y}^*}{\operatorname{argmin}} \| \underbrace{\begin{bmatrix} I_{2\times 2} \\ s_2 R_{\theta_2} \\ \vdots \\ s_m R_{\theta_m} \end{bmatrix}}_{A} \underbrace{\boldsymbol{v}_1^*}_{\boldsymbol{x}} - \underbrace{\begin{bmatrix} \boldsymbol{v}_1^h \\ \boldsymbol{v}_2^h \\ \vdots \\ \boldsymbol{v}_m^h \end{bmatrix}}_{\boldsymbol{b}} \|_2^2 \qquad (4)$$

where s_i and R_{θ_i} represent the scaling factor and the rotation matrix mapping the first point to the ith point in the original centered set $\mathcal{V}^O$ respectively. We can get the optimal solution of Eq. (4) by solving the linear equation $\boldsymbol{x} = (A^{\mathrm{T}}A)^{-1}A^{\mathrm{T}}\boldsymbol{b}$. Once the optimal position of $\boldsymbol{v}_1^*$ has been found, we can get the positions of all vertices on $\mathcal{P}_0$. For the input model in Fig. 6, if the step of reducing distance is not performed, more remeshing–removing iterations

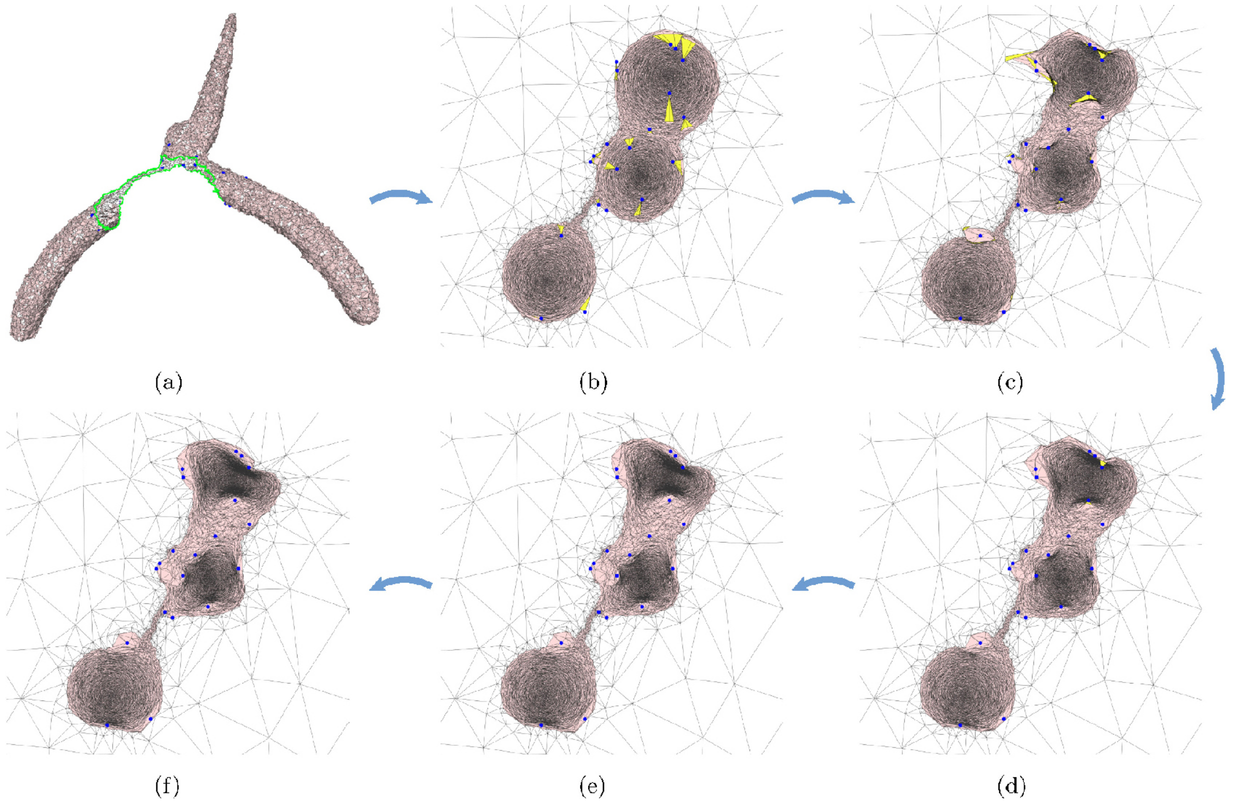


Fig. 5 Intermediate results when removing flips using Ref. [8]. (a) Input mesh. (b) Initialization. (c)–(f) Results after 4, 10, 25, and 41 iterations respectively. There are no flipped triangles in (f), so the remeshing step is not needed for this example.

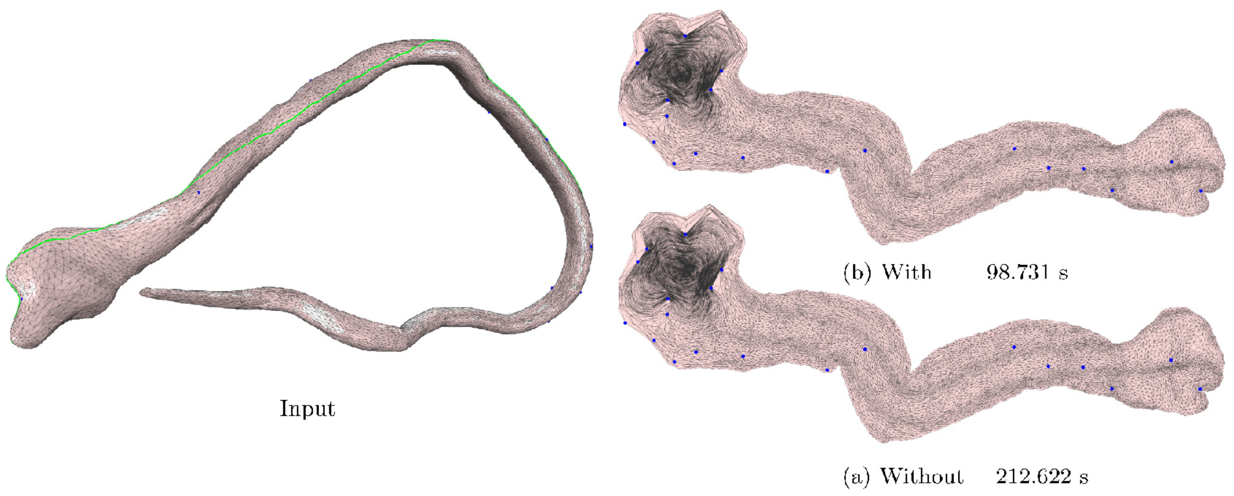


Fig. 6 Distance reduction step. Results without (a) and with (b) the distance reduction step, for the same positional constraints.

are needed to obtain the desired result satisfying all constraints, which greatly increases the running time.

3.3.2 *Generating initial scaffolds*

We enlarge the bounding box of $\mathcal{P}_0$ by 3 times to generate $(BB)_0$ and sample $\boldsymbol{n}$ points uniformly on each edge of $(BB)_0$. Then we use the Triangle library [46] to compute the CDT to construct the initial $\mathcal{S}$, whose constrained edge set includes all boundary edges of $\mathcal{P}_0$ and $(BB)_0$. In CDT processing, we set the maximum triangle area to be ρl^2, where l is the average length of all boundary edges of $\mathcal{P}_0$. In our experiments, we set $\boldsymbol{n} = 15$ and $\rho = 200$ (see further discussion in Fig. 10).

3.3.3 *Remeshing*

Since the quality of $\mathcal{S}$ affects the success rate of the optimization process, edge-based remeshing of $\mathcal{S}$ is

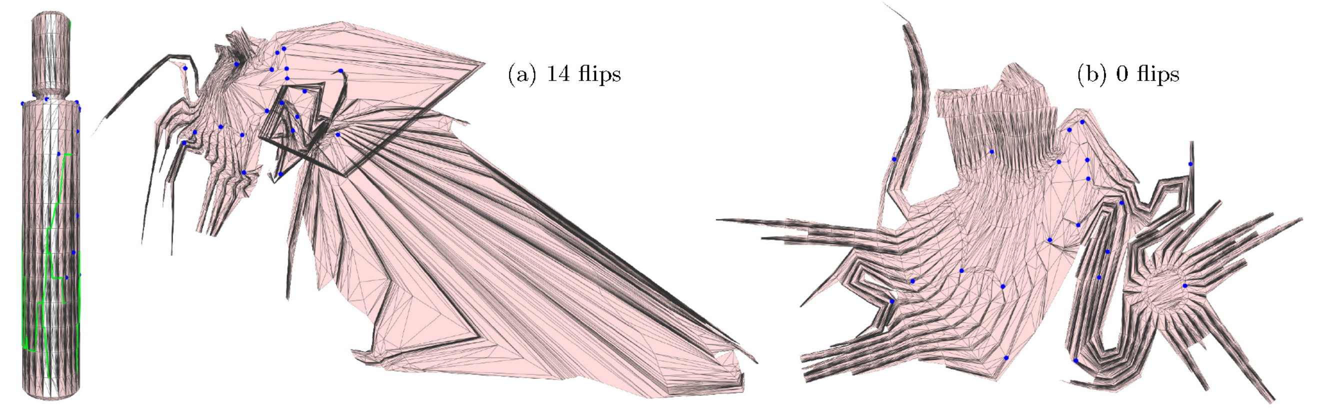


Fig. 7 Remeshing step. The results without (a) and with (b) the remeshing step, for the same positional constraints. Without remeshing, the algorithm fails to generate a globally injective result (a).

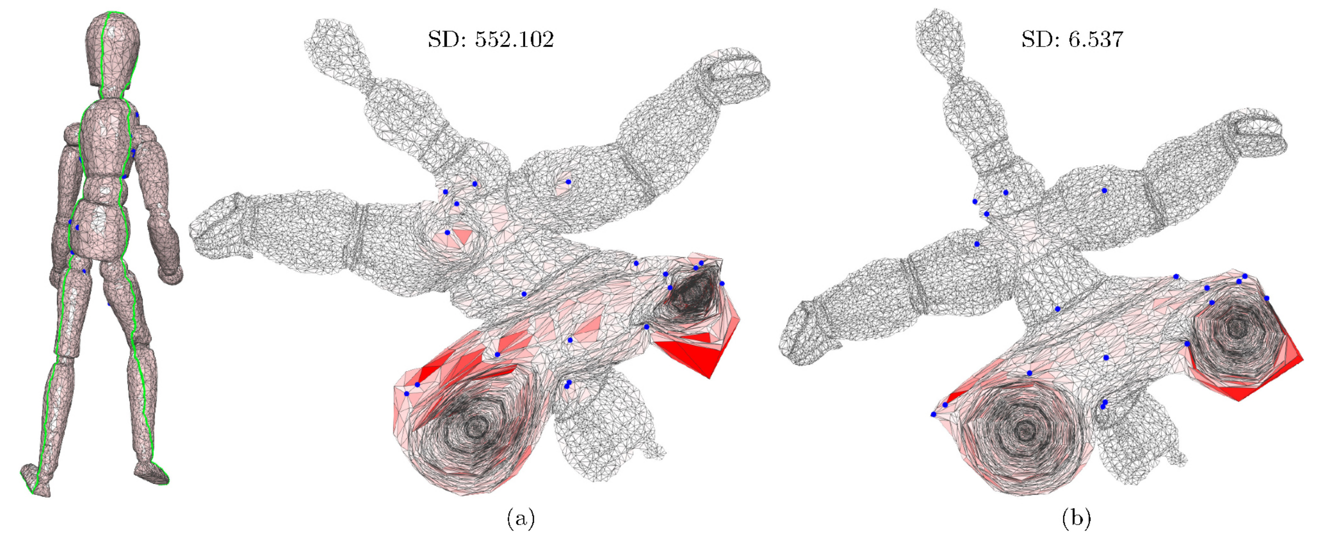


Fig. 8 Distortion optimization step. The results without (a) and with (b) the distortion optimization step, for the same positional constraints. SD is the symmetric Dirichlet distortion energy.

used to improve its quality. Local operations: edge collapse, edge split, edge flip, and vertex relocation, are applied iteratively. The initial target edge length L is set to the average length of the edges of $\mathcal{S}$. As the process goes on, if the number of mesh edges is too large, we increase the target edge length accordingly. The specific remeshing operations performed on $\mathcal{S}$ are as follows:

1. We first smooth all interior vertices 10 times. The new position of each vertex $\boldsymbol{v}_i$ is simply the barycenter $\boldsymbol{c}_i$ of its one-ring neighbors $\mathcal{N}_i$:
$$\boldsymbol{c}_i = \frac{\sum_{j\in\mathcal{N}_i} \omega_j \boldsymbol{v}_j}{\sum_{j\in\mathcal{N}_i} \omega_j}, \quad \omega_j = 1$$
2. We iteratively perform the following operations 10 times, following Ref. [19]:
   (a) Edge split: every interior edge $\boldsymbol{e}$ is split if it is longer than $7L/3$. If an interior edge is incident to two vertices of the inner boundary, it is also split.
   (b) Edge collapse: every interior edge $\boldsymbol{e}$ shorter than $2L/5$ is collapsed.
   (c) Edge flip: every interior edge $\boldsymbol{e}$ is flipped if this operation decreases the squared difference of the valences of the four vertices of the two incident triangles from their optimal value of 6.
   (d) Vertex smoothing: every interior vertex $\boldsymbol{v}$ is smoothed as in Step 1.

4 Experiments

4.1 Background

We have evaluated our algorithm on various models. Our method was implemented in C++, and all experiments are performed on a desktop PC with a 4.20 GHz Intel Core i7-7700K CPU and 16 GB of RAM. The linear systems were solved using the Intel Math Kernel Library.

4.2 Initial parameterizations

We tested three types of initial globally injective parameterizations for one model: (i) Tutte's embedding [27], (ii) the intermediate result of Ref. [11], (ii) the final result of Ref. [11]. Our method succeeded in each case (see Fig. 9). In practice, the initial parameterization affects whether the remeshing stage is needed, so the total running time differs. For example, our method has to perform remeshing operations to eliminate flips in Fig. 9(a), but this is not required in Figs. 9(b) and 9(c). The symmetric Dirichlet distortion energies of these three cases are 4.308, 4.308, and 4.308, respectively, indicating that the mapping quality of the result is insensitive to the initial parameterization, due to the final distortion reduction step.

4.3 Scaffolds

We ran our method with three different pairs of parameters $(\boldsymbol{n}, \rho)$ for scaffold mesh generation: see Fig. 10. The resulting total running time and the symmetric Dirichlet distortion energies are (4.934 s, 5.069), (5.559 s, 5.069), and (5.816 s, 5.069) for the cases in Figs. 10(a)–10(c), respectively, demonstrating that our method is insensitive to $\boldsymbol{n}$ and ρ.

4.4 Testing on a dataset

To verify the effectiveness and robustness of our method, we tested our algorithm on a dataset containing 1791 models, provided by Ref. [16]. For all models in the dataset, we used the final results of Ref. [11] as the initializations. The parameters $(\boldsymbol{n}, \rho)$ for generating scaffold meshes were fixed to (15, 200). We observe that our algorithm achieves global injectivity for most of the models: our success rate is 99.8%, just failing on three examples (see Fig. 14). These results demonstrate the practical robustness of our method. Six models are shown in Fig. 11.

4.5 Speed

In Fig. 12, we plot running time versus the number of vertices N_v for each algorithm step of our method. The running time tends to increase with N_v, except for the remeshing step: most models do not need

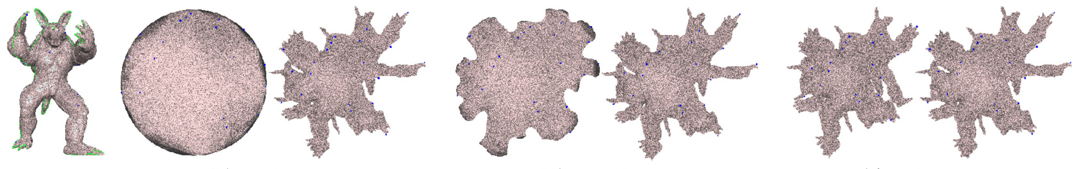


Fig. 9 Three different initial globally injective parameterizations, including Tutte's embedding [27] (a), the result after running the optimization of Ref. [11] for 5 iterations (b), and the result after running the optimization of Ref. [11] until convergence (c). The total running time of the algorithm is given below each result.

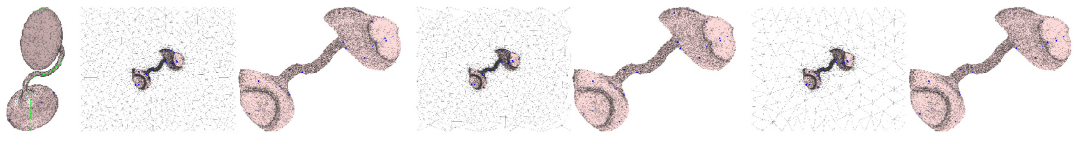


Fig. 10 Three different scaffold meshes.

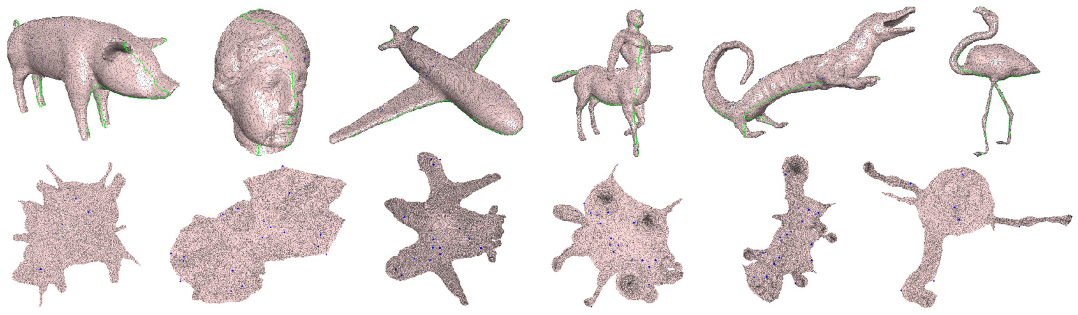


Fig. 11 Gallery: our method succeeds in generating globally injective parameterizations with hard positional constraints.

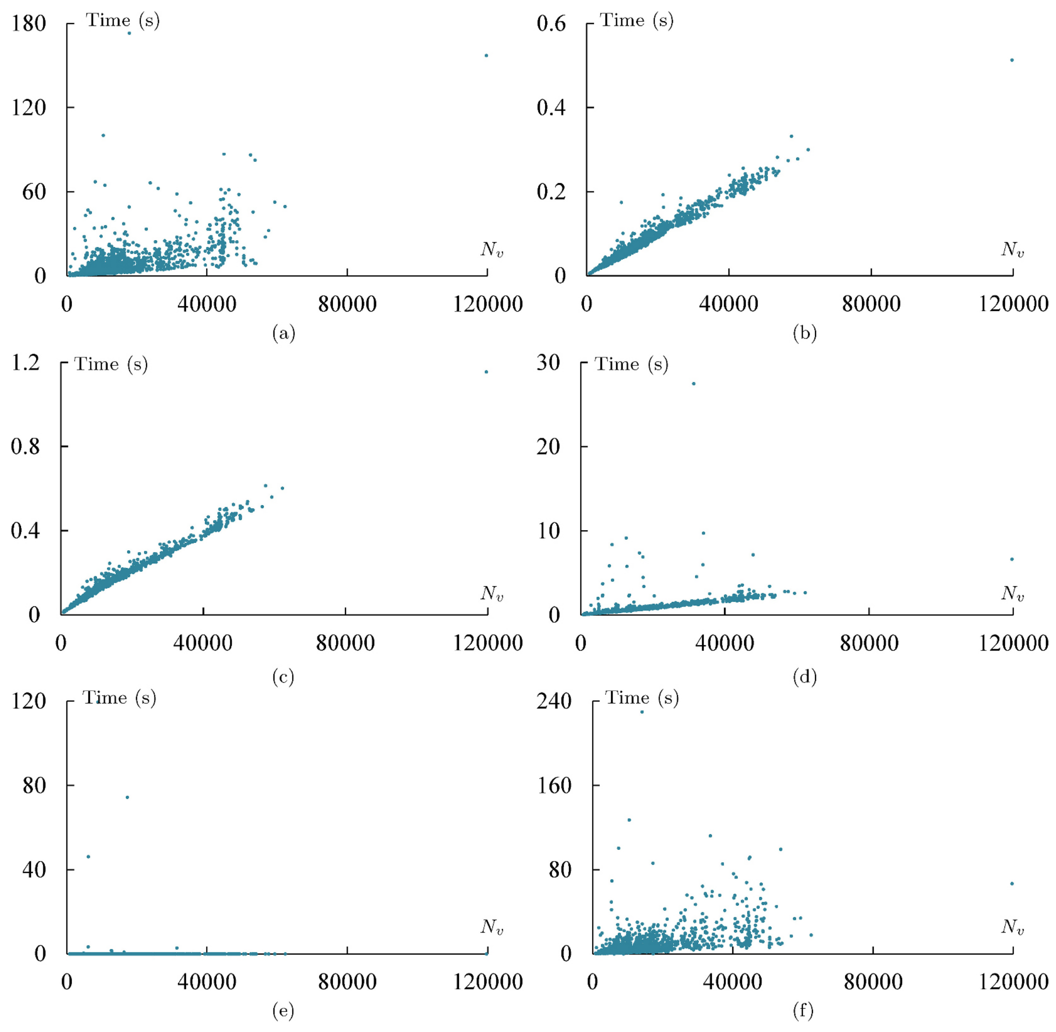


Fig. 12 Timings. The graphs plot the running time (s) vs. the number of vertices N_v for each step of our method: (a) generating the initial globally injective parameterization, (b) reducing distances, (c) constructing the scaffold mesh, (d) removing flips, (e) remeshing and removal process, (f) optimizing distortion energy. Each dot represents one mesh. Most meshes do not need to be remeshed, so most dots in (e) take 0 s.

this step. In particular, the running time almost line-arly increases with respect to N_v in the distance reduction step, the scaffold mesh construction step, and the flip removal step. The total running time is mainly affected by the processes of generating the initial parameterization and optimizing the distortion energy.

4.6 Comparisons

We compared our method to that of Ref. [16] using the authors' results. While our success rate reaches 99.8%, theirs is only 85.8%. We calculated the symmetric Dirichlet energy of 1535 results that both the method of Ref. [16] and ours obtained successfully. Denoting the ratio of their energy to ours as ρ_{SD}, then the average ρ_{SD} is 1.0011, with standard deviation 0.0102. Figure 13 shows comparisons on four examples. Our method is able to generate globally injective parameterizations while they fail, indicating that our method is more robust than theirs in practice.

5 Conclusions

5.1 Summary

In this paper, we have presented a novel algorithm for disk topology meshes to generate globally injective parameterizations with hard positional constraints. The key idea of our algorithm is to use a scaffold mesh to convert the global injectivity constraint into a flip-free constraint. We use this key idea to propose a new pipeline to compute globally injective parameterizations. We have tested our algorithm on a large dataset. The results show that our algorithm is more robust than previous methods.

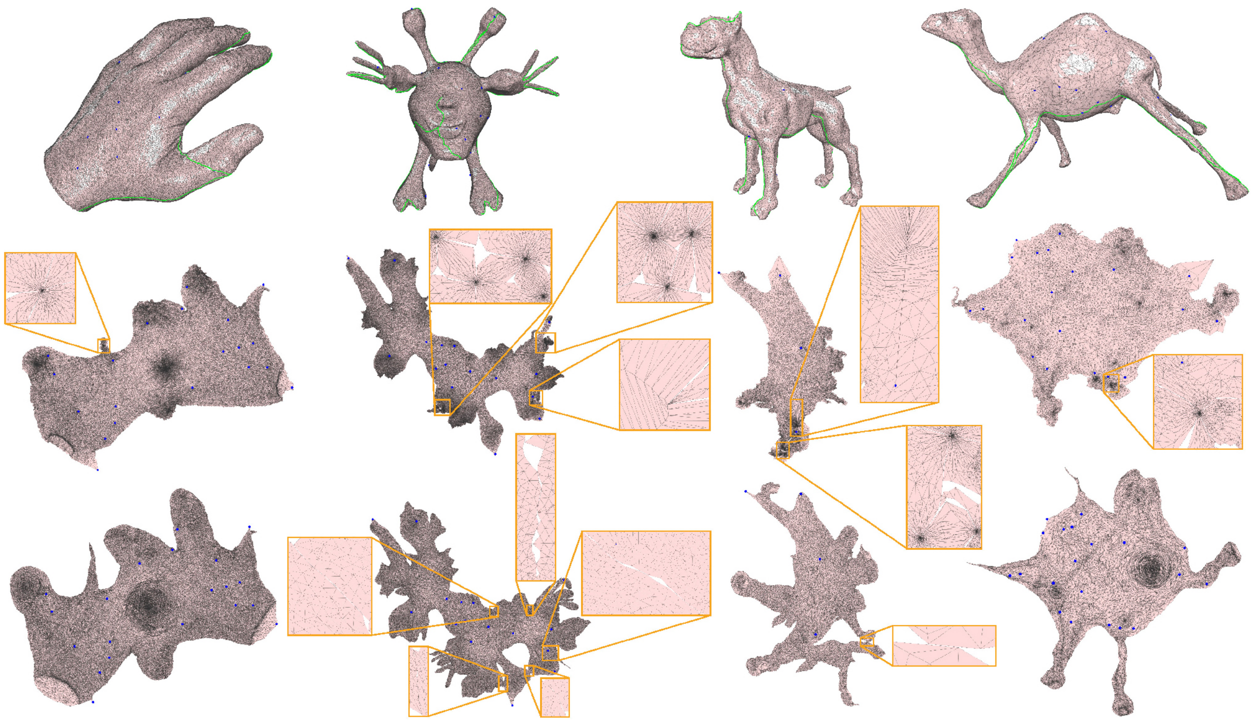


Fig. 13 Comparisons of our results with those of Ref. [16], while applying the same constraints to four meshes. Top row: input meshes. Middle row: the results of Ref. [16]; they contain self-intersections or flips. Bottom row: our results are globally injective.

5.2 Limitations

Although our algorithm succeeds more often than Ref. [16] on a large dataset containing 1791 models, there are still three models whose results violate hard constraints due to numerical problems, as shown in Fig. 14. Our success is not theoretically guaranteed for arbitrary models.

5.3 Future work

In future, developing a theoretically guaranteed method is an interesting direction. It is also worthwhile to study constrained globally injective mappings in 3D, since constructing scaffold meshes and removing flips in 3D are hard tasks.

Acknowledgements

We would like to thank the anonymous reviewers for their constructive suggestions and comments. This work was supported by the National Natural Science Foundation of China (61802359, 62025207) and USTC Research Funds of the Double First-Class Initiative (YD0010002003).

Declaration of competing interest

The authors have no competing interests to declare that are relevant to the content of this article.

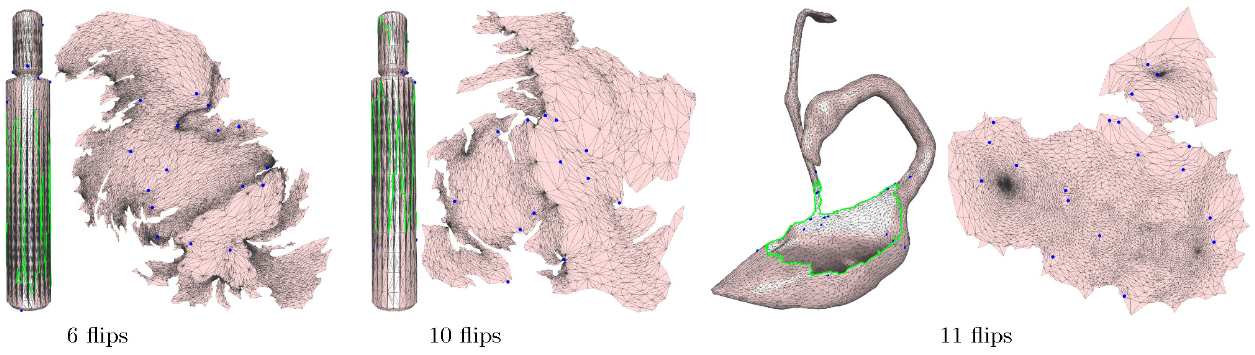


Fig. 14 Numerical issues. Our method cannot generate desired results for three examples due to numerical problems: several triangles degenerate to one point. The method of Ref. [26] can be used to resolve this problem.

References

[1] Floater, M.; Hormann, K. Parameterization of triangulations and unorganized points. In: *Tutorials on Multiresolution in Geometric Modelling. Mathematics and Visualization*. Iske, A.; Quak, E.; Floater, M. S. Eds. Springer Berlin Heidelberg, 287–316, 2002.

[2] Floater, M. S.; Hormann, K. Surface parameterization: A tutorial and survey. In: *Advances in Multiresolution for Geometric Modelling. Mathematics and Visualization*. Dodgson, N. A.; Floater, M. S.; Sabin, M. A. Eds. Springer Berlin Heidelberg, 157–186, 2005.

[3] Sheffer, A.; Praun, E.; Rose, K. Mesh parameterization methods and their applications. *Foundations and Trends® in Computer Graphics and Vision* Vol. 2, No. 2, 105–171, 2006.

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

[5] Fu, X. M.; Su, J. P.; Zhao, Z. Y.; Fang, Q.; Ye, C. Y.; Liu, L. G. Inversion-free geometric mapping construction: A survey. *Computational Visual Media* Vol. 7, No. 3, 289–318, 2021.

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

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

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

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

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

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

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

[13] Overby, M.; Kaufman, D.; Narain, R. Globally injective geometry optimization with non-injective steps. *Computer Graphics Forum* Vol. 40, No. 5, 111–123, 2021.

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

[15] Fang, Y.; Li, M. C.; Jiang, C.; Kaufman, D. M. Guaranteed globally injective 3D deformation processing. *ACM Transactions on Graphics* Vol. 40, No. 4, Article No. 75, 2021.

[16] Du, X. Y.; Kaufman, D. M.; Zhou, Q. N.; Kovalsky, S. Z.; Yan, Y. J.; Aigerman, N.; Ju, T. Optimizing global injectivity for constrained parameterization. *ACM Transactions on Graphics* Vol. 40, No. 6, Article No. 260, 2021.

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

[18] Lipman, Y. Bijective mappings of meshes with boundary and the degree in mesh processing. *SIAM Journal on Imaging Sciences* Vol. 7, No. 2, 1263–1283, 2014.

[19] Botsch, M.; Kobbelt, L. A remeshing approach to multiresolution modeling. In: Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing, 185–192, 2004.

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

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

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

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

[24] Aigerman, N.; Lipman, Y. Hyperbolic orbifold tutte embeddings. *ACM Transactions on Graphics* Vol. 35, No. 6, Article No. 217, 2016.

[25] Floater, M. One-to-one piecewise linear mappings over triangulations. *Mathematics of Computation* Vol. 72, No. 242, 685–696, 2003.

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

[27] Tutte, W. T. How to draw a graph. *Proceedings of the London Mathematical Society* Vol. s3-13, 743–767, 1963.

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

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

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

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

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

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

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

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

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

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

[38] Garanzha, V.; Kaporin, I.; Kudryavtseva, L.; Protais, F.; Ray, N.; Sokolov, D. Foldover-free maps in 50 lines of code. *ACM Transactions on Graphics* Vol. 40, No. 4, Article No. 102, 2021.

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

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

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

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

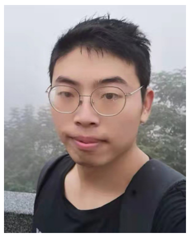
[43] Ainsley, S.; Vouga, E.; Grinspun, E.; Tamstorf, R. Speculative parallel asynchronous contact mechanics. *ACM Transactions on Graphics* Vol. 31, No. 6, Article No. 151, 2012.

[44] Horn, B. K. P. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America A* Vol. 4, No. 4, 629, 1987.

[45] Bouaziz, S.; Deuss, M.; Schwartzburg, Y.; Weise, T.; Pauly, M. Shape-up: Shaping discrete geometry with projections. *Computer Graphics Forum* Vol. 31, No. 5, 1657–1667, 2012.

[46] Shewchuk, J. R. Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator. In: *Applied Computational Geometry towards Geometric Engineering. Lecture Notes in Computer Science, Vol. 1148.* Lin, M. C.; Manocha, D. Eds. Springer Berlin Heidelberg, 203–222, 1996.

Qi Wang received her B.Sc. degree in 2018 from the University of Science and Technology of China. She is currently a Ph.D. candidate in the School of Mathematical Sciences, University of Science and Technology of China. Her research interests include geometric processing and computer graphics.



Wen-Xiang Zhang received his B.Sc. degree in 2018 from Shandong University. He is currently a Ph.D. candidate in the School of Mathematical Sciences, University of Science and Technology of China. His research interests include geometric processing and computer graphics.

Yuan-Yuan Cheng received her B.Sc. degree in 2019 from Nanjing University of Science and Technology. She is currently a Ph.D. candidate in the School of Mathematical Sciences, University of Science and Technology of China. Her research interests include geometric processing and computer graphics.

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

Xiao-Ming Fu received his B.Sc. degree in 2011 and his Ph.D. degree in 2016 from the University of Science and Technology of China. He is currently an associate professor in the School of Mathematical Sciences, University of Science and Technology of China. His research interests include geometric processing and computer-aided geometric design. His research work can be found at his research website: `http://staff.ustc.edu.cn/~fuxm`.