Volume xx (200y), Number z, pp. 1–15

Mesh Parameterization Driven by Unit Normal Flow

Hui Zhao [1 2] † Kehua Su [3] ‡ Chenchen Li [3] Boyu Zhang [9] § Lei Yang [7] Na Lei [5] Xiaoling Wang [8] Steven J. Gortler [6] ¶ Xianfeng Gu [4] ‖

[1] State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences [2] University of Chinese Academy of Sciences
[3] Wuhan University, China [4] State University of New York at Stony Brook, USA [5] Dalian University of Technology, China
[6] Harvard University, USA [7] Capital Normal University [8] University of Science and Technology Beijing [9] Princeton University

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

Figure 1: The Buddha model deforms to a planar mesh.

Abstract

Based on mesh deformation, we present a unified mesh parameterization algorithm for both planar and spherical domains. Our approach can produce intermediate frames from the original meshes to the targets. We derive and define a novel geometric flow: ***unit normal flow (UNF)*** *and prove that if UNF converges, it will deform a surface to a constant mean curvature (CMC) surface, such as planes and spheres. Our method works by deforming meshes of disk-topology to planes, and spherical meshes to spheres. Our algorithm is robust, efficient, simple to implement. To demonstrate the robustness and effectiveness of our method, we apply it to hundreds of models of varying complexities. Our experiments show that our algorithm can be a competing alternative approach to other state-of-the-art mesh parameterization methods. The unit normal flow also suggests a potential direction for creating CMC surfaces.*

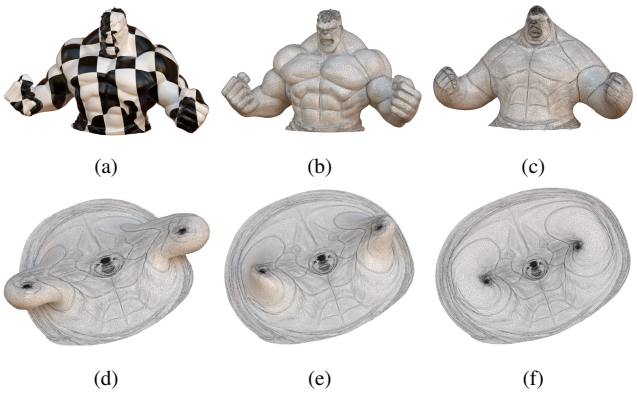
Keywords: mesh parameterization, deformation, constant mean curvature, rotation, unit normal flow

1. Introduction

In this paper, we present a simple and novel algorithm of planar and spherical mesh parameterization. Our methodology is not the same as previous ones: we do not compute a direct embedding of a mesh onto a planar or spherical domain. Instead, we deform it towards a planar and spherical shape. Our method unifies planar and spherical mesh parameterization into a single framework, which consists of the iterations of two steps alternately: averaging face normal and deforming the surface. For meshes of disk or spherical topology, they converge to planar or spherical shapes automatically during the iterations. Figures 1, 2 and 3 demonstrate the deformations and their planar and spherical parameterizations respectively. Our approach produces locally injective mappings in general, although no theoretical proof is given. Our method works by evolving the normal of vertices, which leads to the updating of the positions.

The design methodology of major parameterization algorithms in research community obtains the mapping by minimizing some kinds of energies, which are often based on the measurements of angle or area distortions. The conformal and area-preserving parameterizations are computed separately. Then the mixed result can be achieved by combining them. Although conformal mapping can preserve angles, it results in huge local area distortion, and the area-preserving one makes shapes twisted. The ideal solution to texturing applications should have both smaller area and angle distortions simultaneously. How to balance these two kinds of distortion is a complicated problem. Intuitively, the balance should be determined by the shape of a surface locally and globally.

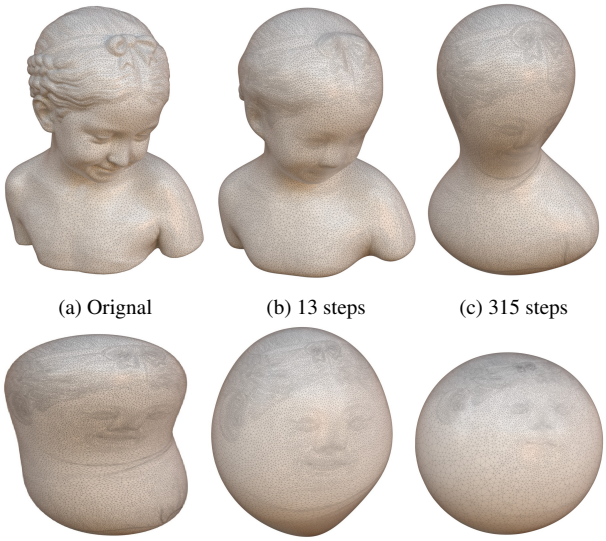
† e-mail:alanzhaohui@qq.com
‡ e-mail:skh@whu.edu.cn, corresponding author
§ e-mail:bz@math.princeton.edu
¶ e-mail:sjg@cs.harvard.edu
‖ e-mail:gu@cs.stonybrook.edu



(a) (b) (c)

(d) (e) (f)

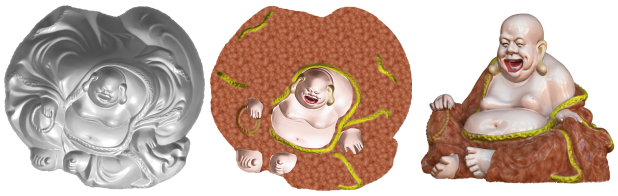
Figure 2: (a) texturing; (b) original mesh; (c),(d),(e) intermediate deforming frames; (f) final planar mapping.



(a) Orignal (b) 13 steps (c) 315 steps

(d) 665 steps (e) 3795 steps (f) Final sphere

Figure 3: The deforamtion of bimba model and its spherical parameterization.



(a) (b) (c)

Figure 4: The texturing of the buddha model.

Our algorithm works as follows. Firstly we compute the new normal of every point by averaging the normals of its neighbors. Secondly we reconstruct a surface which fits the current normals. By alternating these two steps we get a heat-like geometric flow on surfaces, which we call the **unit normal flow (UNF)**. Honestly, in this paper, we do not give a proof of convergence of unit normal flow. What we prove is, on a smooth surface, **if** the flow converges, it will produce a CMC surface, such as a plane or a sphere. This observation guides the design of our algorithm. Our experiments on hundreds of discrete meshes support our claim that, as far as sphere and plane are concerned, the flow can converge.

A surface must have constant mean curvatures on every point in order to be a CMC surface. Given a non-CMC surface, our unit normal flow deforms it to a CMC surface gradually. Every step of unit normal flow can be viewed as a special smoothing operation. Yet it is very different from the mean curvature flow and Willmore flow. It is well known that mean curvature flow has singularities [CMI12], therefore it needs to be modified to avoid the singularities in every step [KSBC12]. However, our normal flow based algorithm hasn't got any singularity in our extensive experiments.

The most important application of planar parameterization is texture mapping. An example of our approach is exhibited in Fig. 4. We first render the planar mesh shown in Fig. 1f using the normals of the original 3D mesh; Then we draw textures on the rendered image; Finally the textured 3D mesh is exhibited. In our attached video, we also show the special effect of *two dimensional foil* described in the science-fiction books [Liu14, Liu15, Mor17].

In summary, the contributions in this paper are: 1) defining a novel geometric flow on surfaces: unit normal flow; 2) establishing and deriving the relationship between unit normal flow and constant mean curvature surfaces; 3) proposing a robust, simple-to-implement algorithm to discretize and approximate the non-linear UNF; 4) applying the algorithm in the application of planar and spherical mesh parameterization. Further, our method has a special feature of mapping the selected partial parts of meshes onto a plane and keeping the remaining parts unchanged. Instead of computing positions directly, we work with a new quantity called : unit normal field.

2. Related Work

Parameterization of triangle meshes is a problem with a long history in computer graphics. Due to the abundance of literature on mesh parameterization, we will focus on the approaches which are the most relevant to ours. For in-depth surveys, we refer to [FH05, SPR*07].

By fixing the boundary of a topological disk mesh to a convex shape, the linear method [Flo03] produces bijective parameterization, but with huge angle and area distortion. The free-boundary linear conformal methods [Flo03, DMA02, LPRM02, BCGB08, MTAD08] fail in guaranteeing to have a bijective map. Nonlinear methods which minimizes the energies of conformal or isometric distortion [SdS01, CPSS10, SS15, APL14, LZ14, FLG15] can not handle large models. In contrast, every step of our method solves a linear system. The local-global approach [LZX*08] is similar to ours. However, their iterations only works on two dimensions and produces a lot of flipped faces. On the contrary, our method iteratively deforms meshes in 3D space and has no flips in practice. The recent method [LYNF18] requires an initial bijective mapping as starting step.

Some of injective [APL14, AL13, SKPSH13, FLG15, SS15,

RPPSH17, LGZ*16] and bijective [JSP17, SS15] mapping algorithms work either through a flip-less initialization or by enforcing a low bounded distortion. These methods need complicated and non-linear computation to guarantee injectivity or bijectivity. Our approach builds locally injective mappings automatically in most cases. For a few of complicated shapes, our mappings are locally injective and there are overlaps. However with a simple interactive initialization, we can adjust these injective mappings to be bijective.

The planar mapping algorithms in [JKLG08, Luo04, ZLG*18, SSP08] are founded on Ricci flow, Calabi flow, Yamabe flow respectively. These methods work on metric space and are intrinsic, in contrast, our algorithm is extrinsic and can produce in-between embedding. OMT based area-preserved algorithms are proposed in [SCL*16, SCQ*16, SCL*17], and the method in [YLZG18] can balance the area-preserved and angle-preserved mapping. However, these methods need to solve a non-linear Monge-Ampere equation.

For spherical parameterizations, the most related algorithms are modified mean curvature flow (cMCF) [KSBC12] and conformal Willmore flow [CPS13]. Mean curvature flow updates the positions of surface points gradually by minimizing the gradient of the surface embedding or the surface area [KSBC12, Tau95, DMSB99, Cho93, PP93]. They are usually used in the applications of mesh smoothing and minimal surfaces. This flow has singularity and only some simple convex surfaces can converge [H*84]. The cMCF updates the metric of the surface in every step and avoids singularity so that it is stable. In [CPS13], they design a special Willmore flow: conformal curvature flow. This flow runs in curvature space. Their "change of variables" methodology is similar to ours. [HFL18] decimates meshes to obtain bijective isometric spherical mappings. In contrast, ours is deformation based. In [WLL14], the authors propose a spherical parameterization which extends the planar ARAP method, however this algorithm leads to flips in high curvature sections.

The value of mean curvature of every point on a CMC surface [Kap90, XZ08, Ren15] is constant. Minimal surfaces [XPB06, PP93] are special CMC surfaces whose mean curvature value is zero. Normally CMC surface is constructed by minimizing the surface area under the fixed volume constraints [Bra92, PR02, DH06, Ren15], where a non-linear optimization needs to be solved. The algorithm [PCL*12] creates a CMC mesh from an existing one, however it changes the connectivity of the input mesh. Our method keeps the mesh connectivity unchanged.

Gauss map diffusion (or spherical harmonic flow) is discussed in [PKC*16], and normal field filtering is also exploited in [Tau01, YOB02, TWBO02, TWBO03, SRML07, ZFAT11]. However, they are only used in the application of mesh smoothing and denoising. Here we apply the flow to the mesh parameterization. The method in [ZLJW06] maps one source surface directly to another target surface of the same genus without the intermediate planar or spherical domain, but it can not guarantee that all source vertices lie on the target surface.

Poisson system based deformation [YZX*04] is a well-known geometry modelling technique. After the rotations of the triangle faces are known, they can be rotated into the new orientations, then a Poisson system is used to blend the triangles together and reconstructs a new shape. The rotations can be achieved by the interpolation from two corresponding meshes [XZWB06] or from the rotations of the constraint faces [ZRKS05]. For the application of mesh deformation [ZG16], a local-global method is used to compute the optimal rotations. In polycube construction [ZLL*17], the rotations are calculated according to the corresponding polycube face normals. Our algorithm also relies on this kind of methodology, however we compute the rotations by averaging face normals.

3. Unit Normal Flow

Our motivation is to deform surfaces by the following observation, that the derivatives of surface normals with respect to time should be equal to the Laplacians of the normals. In this section, we define the **unit normal flow** mathematically. This flow is different from the well-known mean curvature flow [KSBC12], averaged mean curvature flow [XPB06], Willmore flow [BS05, WBH*07], Ricci flow [JKLG08], and surface diffusion flow [SK01, XPB06]. All these kinds of flows can be modelled as geometric partial differential equations (PDE) [XPB06, XZ08]. As far as we know, this is the first time that this definition appears in the mathematical and graphical research literature.

Let S be a smoothly immersed surface in $\mathbb{R}^3$. Let g be the metric on S restricted from $\mathbb{R}^3$. Let n be the smooth unit normal vector field on S. Denote $\langle , \rangle$ as the inner product and $\Delta_g n$ as the Laplacians of the unit normals. The formal definition of **unit normal flow** is the following:

$$\frac{dn}{dt} = \Delta_g n - \langle \Delta_g n, n \rangle \cdot n. \tag{1}$$

Notice that the norm of n is preserved under this flow, since $\frac{d}{dt}\langle n,n\rangle = \langle n, \frac{d}{dt}n\rangle = \langle n, \Delta_g n\rangle - \langle n, \Delta_g n\rangle = 0$

Lemma 1 If the Laplacians $\Delta_g n$ of the unit normal field n is parallel to n, i.e., $\Delta_g n /\!\!/ n$, then the mean curvature H of S is constant.

Proof We prove it with a local calculation.
Use an isothermal coordinate $r(x_1, x_2)$ for S. Let $z = x_1 + ix_2$. Write

$$\frac{\partial}{\partial z} = \frac{1}{2}\left(\frac{\partial}{\partial x_1} - i\frac{\partial}{\partial x_2}\right),$$

$$\frac{\partial}{\partial \bar{z}} = \frac{1}{2}\left(\frac{\partial}{\partial x_1} + i\frac{\partial}{\partial x_2}\right).$$

Let $g = \lambda^2((dx_1)^2 + (dx_2)^2)$ be the metric of S under the isothermal coordinate. Define

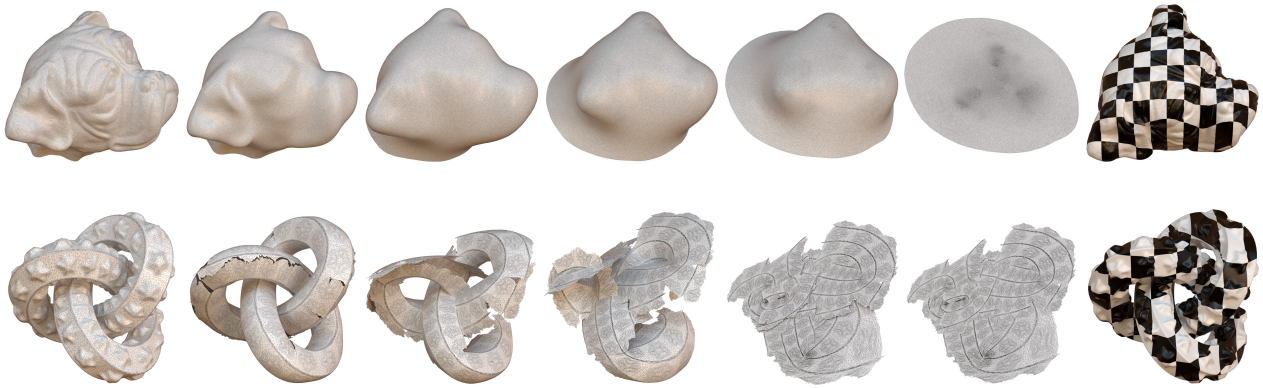
$$\Omega = \langle r_{zz}, n \rangle.$$

Here r_{zz} is a shorthand for $\frac{\partial^2 r}{\partial z^2}$.
The vectors $\{r_z, r_{\bar{z}}, n\}$ form a local frame of $\mathbb{R}^3$ on S. The next step is to calculate its equations of motion. Notice that

$$\langle r_z, r_z \rangle = \langle r_{\bar{z}}, r_{\bar{z}} \rangle = 0.$$

Therefore

$$\langle r_{zz}, r_z \rangle = \langle r_{z\bar{z}}, r_z \rangle = \langle r_{z\bar{z}}, r_{\bar{z}} \rangle = \langle r_{\bar{z}\bar{z}}, r_{\bar{z}} \rangle = 0.$$



(a) (b) (c) (d) (e) (f) (g)

Figure 5: (a) are the original meshes; (b), (c), (d), (e) are intermediate deforming meshes; (f) are the final planar parameterizations and (g) are the texturings. The meshes of "doghead" and "linkedring" have the number of faces of 4318, 100000 respectively.

On the other hand,

$$\langle r_z, r_{\bar{z}} \rangle = \frac{1}{2}\lambda^2.$$

Therefore

$$\langle r_{zz}, r_{\bar{z}} \rangle = \lambda \lambda_z.$$

Notice that the mean curvature

$$\begin{aligned} H &= -\frac{1}{2}\Big(\frac{1}{\lambda^2}\langle \frac{\partial r}{\partial x_1}, \frac{\partial n}{\partial x_1}\rangle + \frac{1}{\lambda^2}\langle \frac{\partial r}{\partial x_2}, \frac{\partial n}{\partial x_2}\rangle\Big) \\ &= -\frac{2}{\lambda^2}\langle n_z, r_{\bar{z}}\rangle \\ &= \frac{2}{\lambda^2}\langle n, r_{z\bar{z}}\rangle. \end{aligned}$$

By the definition of Ω,

$$\langle n_z, r_z \rangle = -\langle n, r_{zz}\rangle = -\Omega.$$

Combining the identities above, one has

$$\begin{cases} r_{zz} = \frac{2}{\lambda}\lambda_z r_z + \Omega n \\ r_{z\bar{z}} = \frac{\lambda^2}{2} H n \\ n_z = -H r_z - \frac{2}{\lambda^2}\Omega r_{\bar{z}} \end{cases}$$

Let g denote the metric. Notice that for any function f on S,

$$\begin{aligned} \Delta_g f &= \frac{1}{\sqrt{\det g}}\frac{\partial}{\partial x^j}\big(g^{jk}\sqrt{\det g}\frac{\partial f}{\partial x^k}\big) \\ &= \frac{4}{\lambda^2} f_{z\bar{z}}. \end{aligned}$$

Therefore

$$\begin{aligned} &\Delta_g n /\!\!/ n \Rightarrow n_{z\bar{z}} /\!\!/ n \\ &\Rightarrow -H_{\bar{z}} r_z - \frac{\lambda^2}{2}H^2 n - \big(\frac{2\Omega}{\lambda^2}\big)_{\bar{z}} r_{\bar{z}} - \frac{2}{\lambda^2}\Omega\Big(\frac{2}{\lambda}\lambda_{\bar{z}} r_{\bar{z}} + \Omega n\Big) /\!\!/ n \\ &\Rightarrow H_{\bar{z}} = 0 \\ &\Rightarrow H \text{ is a constant.} \end{aligned}$$

□

By lemma 1, the critical point of unit normal flow equation 1 leads to a CMC surface.

The proof of the singularity, existences, uniqueness and convergence of the solution of the UNF of surfaces will be investigated in our future study. They are still open problems. However, we prove this mathematics fact: "if the unit normal flow converges to a stable surface, the surface is a constant mean curvature surface." Experimentally, we demonstrate that the flows converge on most of discrete meshes of disk or spherical topology.

4. Our algorithms

Motivated by lemma 1 for unit normal flow and the following **Hopf** theorem, we design a simple, elegant, and practical algorithm for planar and spherical mesh parameterization.

Theorem 1 (Hopf) Every closed immersed constant mean curvature surface of genus 0 in $\mathbb{R}^3$ is a round sphere.

The unit normal flow we defined in Equation 1 is a highly nonlinear equation with respect to the position, which is hard to solve numerically in this format. In order to solve this issue, we propose and use the "change of variable" strategy which has also been applied in [CPS13]. That is, our algorithms do not compute the positions directly. Instead, we solve the current unit normal field in the first place, and then reconstruct the geometry of the surface from it.

We approximate the smooth unit normal field n by the discrete face normal field. Our unit normal flow updates face normals instead of vertex normals in every iteration.

The key point is the discretization of the Laplace operator. In graphics community, the well-known cotangent Laplace operator [PP93] is used for functions defined on the vertices of meshes. Therefore it can not be used for our face normals. In this paper, we propose a simple method to approximate the Laplacian operator of normal functions defined on faces by the following formula:

$$\Delta_d n_i(t) = \sum_{j \in Neighbor(i)} n_j(t) - n_i(t) \approx \Delta_g n(t), \tag{2}$$

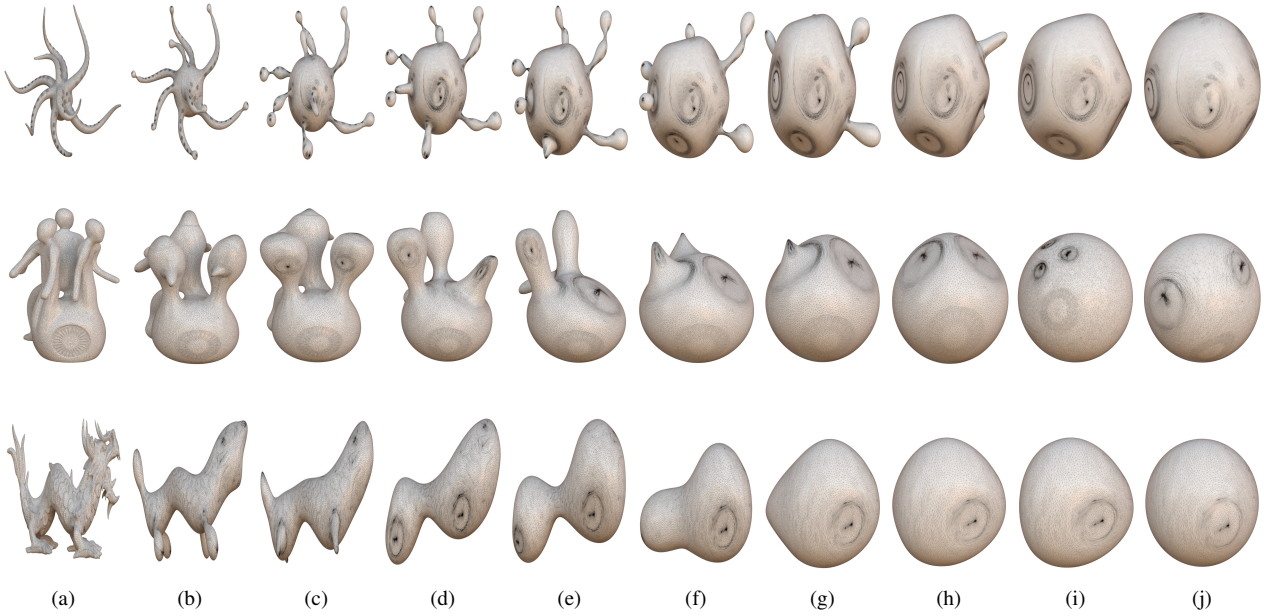


Figure 6: (a) are the original meshes; (b),(e),(c),(f),(g),(h) are the intermediate deforming meshes; (i) are the convergent shapes; (j) are the final normalized spherical parameterizaitons. Octupus, memento and dragon have the faces of 497236, 99932, 105298 respectively.

where $n_j(t)$ denotes the unit normal of face i at time t, $\Delta_d n_i$ represents the normal under the discrete Laplace operator, the $Neighbor(i)$ denotes the neighbors of the face i, which includes i itself. Then the new normal at time $t+1$ is computed like below:

$$n_i(t+1) = \Delta_d n_i(t) + n_i(t) = \sum_{j \in Neighbor(i)} n_j(t) \qquad (3)$$

Our flow is defined on **unit** face normals. Even though face normals n_i have unit length, $\Delta_d n_i$ are not guaranteed to be unit; we need to normalize it. In practice, the faces in $Neighbor(i)$ are not constrained to be one-ring neighbors, k-ring neighbors can also be used.

If we use the area-weighted average, then the flow will fail in our experiments. The intuitive reason is that the centre of the area-weighted average maybe not fall inside the face. For example, in Fig. 7, the average normal of neighbors of red face is not located inside the red face. In Fig. 8, we demonstrate the success of the simple average over the area-weighted average.

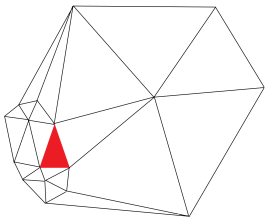


Figure 7: The area-weighted average

After the new face normals are computed in every step, we rotate all triangle faces from their old orientations to the current ones independently. However, the result triangle soup is not a valid mesh. We use the Poisson system based method [ZLL*17, ZG16] to reconstruct a valid manifold triangle mesh. This step can be regarded as solving a system of the unknown positions from the known normal variables.

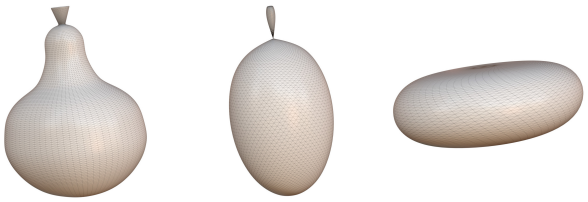


Figure 8: (a) is the original spherical pear mesh; (b) is the deformed result by area-weighted average; (c) is the deformed result by the simple average at the same number iteration with (b).

Let S be the original surface and S' be its deformed surface embedded in 3-dimension. We denote a 3-vector x_v be the position associated with vertex v of S, and a 3-vector x'_v with vertex v of of S'. On every triangle of the mesh, we define one rotation matrix variable referred to as $R(t)$. Let he_{vw} represent the halfedge from vertex v to w. We denote the angle of the corner opposite to the halfedge he_{vw} in its triangle as a_{vw}. Finally $R(t_{vw})$ represents the 3×3 rotation matrix associated with the triangular face whose halfedge is he_{vw}.

After fixing a vertex, we can obtain the unknown position variables x' by solving a single linear system as follows [LZX*08,

CPSS10, ZG16]:

$$\sum_{w\in N(v)}\big[\cot(a_{vw})+\cot(a_{wv})\big](x'_v-x'_w)$$
$$=\sum_{w\in N(v)}\big[\cot(a_{vw})R(t_{vw})+\cot(a_{wv})R(t_{wv})\big](x_v-x_w). \quad (4)$$

By defining the 3-vector at vertex v as:

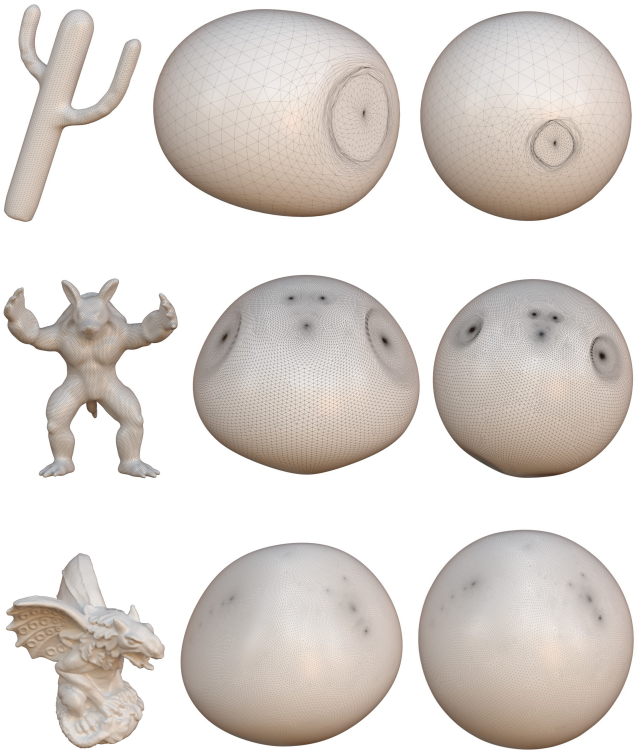
$$b_v := \sum_{w\in N(v)}\big[\cot(a_{vw})R(t_{vw})+\cot(a_{wv})R(t_{wv})\big](x_v-x_w), \quad (5)$$

we can rewrite the above system into matrix format as:

$$Lx' = b, \quad (6)$$

where L is the n-by-n Laplacian matrix, x' and b are n-vectors of 3-vectors.

Given two unit normals, the rotation matrix $R(t)$ between them can be computed by the algorithm of Rodrigues' rotation formula. We solve the equation 6 by fixing the position of one vertex.



(a) Original (b) Convergent shape (c) Normalized sphere

Figure 9: The convergent and normalization of the spherical deformations. (b) are the convergent shapes of unit normal flow and (c) are the normalized spheres.

In summary, our algorithm consists of two steps. In the first step, we average the unit face normals, and in the second step, we deform or reconstruct the surface by the constraints of the current unit face normals. After applying these steps, we get a new mesh which is smoother than the previous one. These two steps are carried out alternatively until the flow converges and the shape of the mesh does not change anymore. In every iteration step i, we recompute the rotation matrix $R_i(t)$ by averaging the face normals. With the new $R_i(t)$, we update L_i and b_i. The iteration could be written as:

$$L_i x'_i = b_i\,. \quad (7)$$

For small meshes, our algorithm converges quickly in a few iterations. But for larger ones, it needs several thousand steps. In the planar case, we speed up the algorithm by computing the average normals of all triangular faces and add it to the average results when updating the normals, as shown in equation 8. The bigger the parameter *weight*, the faster the convergence. However for meshes of bad quality, a big *weight* will lead to flipped faces. In practice, we set it to 0.02. Surprisingly, this simple method decreases the needed number of iterations to roughly a dozen for large meshes. The experimental details are exhibited in the next section.

$$\Delta_d n_i = \sum_{j\in Neighbor(i)} n_j + weight \times \sum_{k\in AllFaces} n_k \quad (8)$$

After the flow converges, we need to do some post-processing. For planar parameterizations, the convergent mesh probably does not fall on a perfect plane. We rotate it and align the average normal of all triangle faces to Z direction, then project it onto the XY plane to obtain their texture coordinates. And for spherical mappings, the convergent shapes are not perfect spheres as well. So we need to normalize the shapes, as exhibited in Figure 9. We first compute the mean center position C of all vertices. Then the average distance R between all vertices and the mean center C are calculated. Finally the normalized position p'_i of the vertex i is computed by the following formula:

$$p'_i = (p_i - C)\times\left(\frac{R}{\|p_i - C\|}\right) \quad (9)$$

5. Experiments

To demonstrate the efficiency and robustness of our algorithm, we apply our algorithm to hundreds of challenging meshes of disk and spherical topology, summarize its performance and compare with other methods. All the demonstrations shown in this paper are locally injective. We ran our experiments on a 12-core Xeon clocked at 2.7GHz, using the Eigen solver [GJ*10] as the linear system solution.

Table 1: Five kinds of distortions.

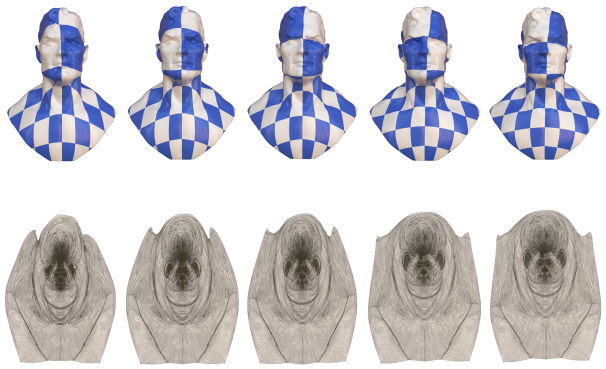
Symmetric Dirichlet	$\sum_{t=1}^{n}(\sigma_{1,t}^2+\sigma_{2,t}^{-2})$
Conformal AMIPS 2D	$\sum_{t=1}^{n}\frac{\sigma_{1,t}^2+\sigma_{2,t}^2}{\sigma_{1,t}\sigma_{2,t}}$
ASAP	$\sum_{t=1}^{n}A_t(\sigma_{1,t}-\sigma_{2,t})^2$
ARAP	$\sum_{t=1}^{n}A_t[(\sigma_{1,t}-1)^2+(\sigma_{2,t}-1)^2]$
Green-Lagrange	$\sum_{t=1}^{n}A_t[(\sigma_{1,t}^2-1)^2+(\sigma_{2,t}^2-1)^2]$

In the graphics literature, there are several common definitions of mapping distortions [LZX*08, RPPSH17, FLG15, ZMT05]. As-Rigid-As-Possible (ARAP) and symmetric Dirichlet(SD) are two well-known isometric measurements. While the zoo of conformal energies include As-Similar-As-Possible (ASAP), conformal AMIPS (cAMIPS) and Green-Lagrange(GL). Denote σ_1, σ_2 as the two singular values of the triangle mapping, A_t as the area of triangle t, then these five kinds of measurements are summarized in Table 1:

Besides these five kinds of distortions, we also calculate the change ratios of areas, edges and angles. We use the extensive experimental results to justify the robustness, applicability and performance of our method.

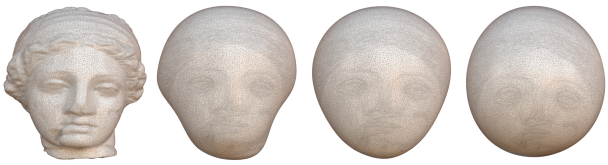
5.1. The ring size of neighbours

The size of the neighbouring area in our algorithm affects the convergence speed. The number of iterations decreases quickly when the ring size enlarges. The relationship is illustrated in Fig. 12.



(a) 1 (b) 17 (c) 29 (d) 81 (e) 100

Figure 10: The planar mapping and texturing of the same model by the ring size of 1, 17, 29, 81 and 100 respectively.



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

Figure 11: (a) is the original model; (b),(c),(d) are the intermediate deforming meshes at the iteration step of 200 for the ring size of 3, 10 and 20 respectively.

However, when the number of rings is bigger than a certain threshold, the mapping will not be flip-free. This value is not fixed and depends on the shape and size of the mesh. We list the statistics of iterations for several meshes with varying representative sizes in Table 2.

The mapping quality does not change dramatically with the ring size, as shown in Figure 13. Therefore we have a wide and stable spectrum for the ring size parameter. This fact makes our algorithms practical and we do not need to tweak the parameter for every mesh.

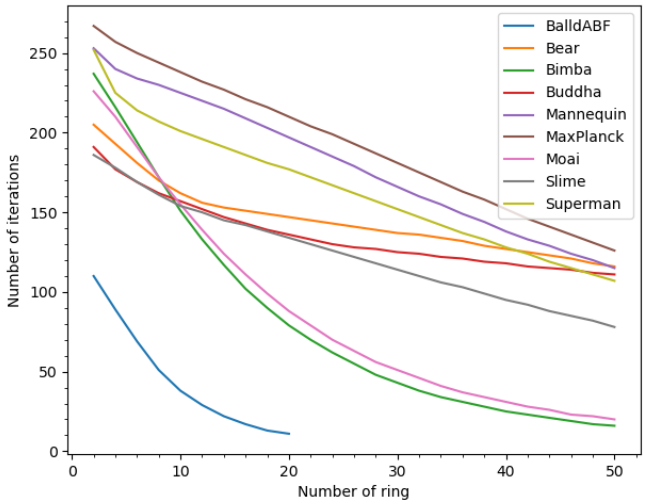


Figure 12: The number of iterations decreases when the ring size gets bigger.

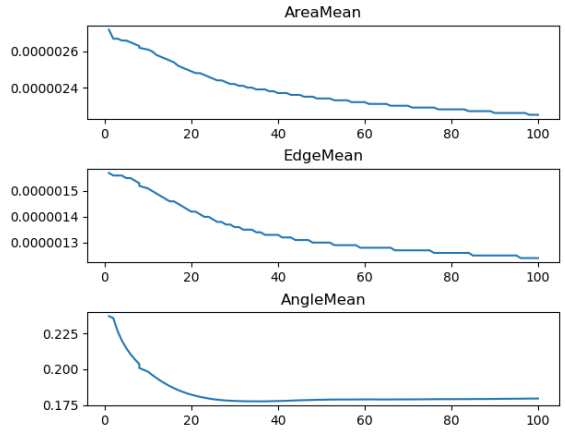


Figure 13: The mapping qualities of different ring sizes of the superman mesh.

According to our extensive experiments on hundreds of meshes, the above parameter works successfully for most of them. Furthermore, we can always obtain flip-less parameterization by shrinking the neighbourhood in practice if the parameter fails in a few cases. Fig. 11 and 10 exhibit the effects of ring size for the meshes of spherical and disk topology respectively. In most of cases, disk meshes can converge by dozens of iterations and the spherical ones needs thousands of steps. We recommend the following empirical values for the ring size:

Model	#Faces	Iter(2)	Iter(10)	Iter(24)	Iter(50)
BallABF	1k	110	38	F	F
Oni	3K	1481	97	F	F
Hand	7k	262	103	F	F
Bimba	11k	237	151	62	16
Moai	16k	226	155	70	20
MaxPlanck	84k	267	238	199	126
Slime	103k	186	154	126	78
Mannequin	109K	253	225	185	115
Superman	190k	252	201	167	107
Bear	296k	205	162	143	116
Buddha	471k	191	157	130	111

Table 2: The number of iterations in case of ring size of 2, 10, 24 and 50; F=Flip.

$$ring = \begin{cases} 5 & faces < 1000 \\ 10 & faces < 10000 \\ 20 & faces \geq 100000 \end{cases}$$

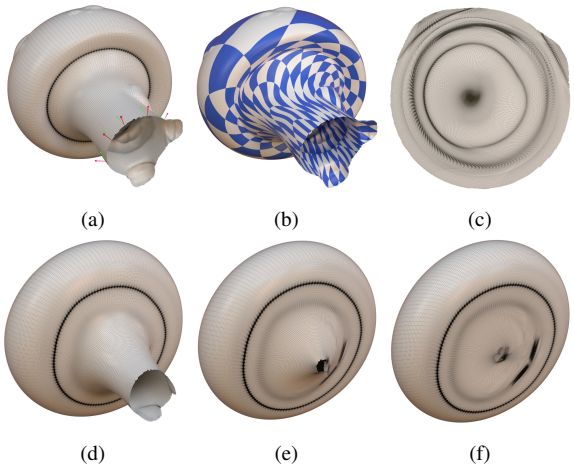
5.2. Disk topology

We demonstrate our mapping and corresponding texturing of disk-topology meshes with a single boundary in Fig. 5 and multiple boundaries in Fig. 14. The intermediate deforming frames are also exhibited in Fig. 1, 2.

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

Figure 14: The planar mapping and texturing of four multi-boundary meshes.

In our experiments, the normals of the boundary face are not fixed. Therefore the unfolded result of our algorithm is affected by the initial normals of boundary faces. When the boundary of a mesh is small and tight, possibly it can not be unfolded towards a plane driven by its natural initial face normals. The mesh in Figure 15a will be deformed to the non-planar shape of Figure 15f after 500 iterations through Figure 15d and 15e with its natural boundary face normals. We solve this problem by assigning it a set of specific boundary normals to pull faces apart, such that it can stretch to a plane in Figure 15c whose corresponding texturing is shown in Figure 15b.



(a) (b) (c)

(d) (e) (f)

Figure 15: The red arrows represent the natural face normals on the boundary of the original mesh (a) and the green ones are the normals we assigned; (d) (e) and(f) are the deforming meshes by red normals; (c) is the planar mesh deformed by the green normals; The texturing is exhibited in (b).

Our experiments produce bijective mappings for most of the meshes with the natural initial face normals. In some complicated cases, the naturally unfolded result may be injective, such as in Fig. 16b. By adjusting its boundary normals manually and interactively, we can turn this injective mapping into bijective, as shown in Fig. 16d.

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

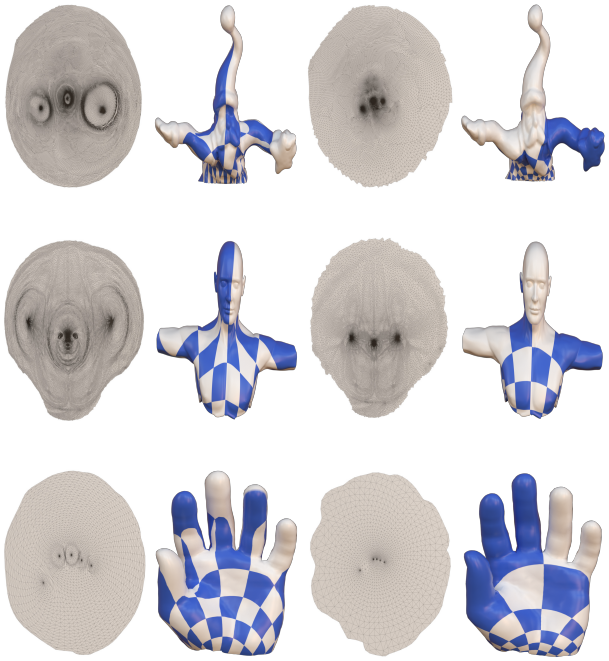
Figure 16: (a) is the original mesh cutted; (b), (c) are the injective mapping and its texturing; (d), (e) are the bijective mapping and its texturing.

The options for Laplacian matrix. On smooth surfaces, the Laplacian operator is dynamic and depends on the current deforming surface. In our setting, we approximate it with the Laplacian matrix of the initial mesh and keep it unchanged. Another alternative is updating the Laplacian matrix of the equation (6) in every iteration by the current geometry of the deforming mesh. The two kinds of mappings and the distortions are manifested in Figure 17 and Table 3. We concludes that the updated Laplacian matrix preserves the area and edge length better, and the fixed one produces better conformal mapping results.

The partial parameterization. Our deformation based method has a special feature, which is deforming parts of a mesh to planar and keeps the rest unchanged, as shown in Figure 26. As far as we

Table 3: The distorions by the fixed and updated Laplacian matrice; (Fix) denotes the fixed Laplacian matrix.

Model	AreaSum	EdgeSum	AngleSum	Symm Dirichilet	Conformal AMIPS2D	ASAP	ARAP	Green Lagrange
Buddha	0.30	0.20	$2.72*10^5$	$2.35*10^6$	$1.04*10^6$	0.22	0.21	1.11
Buddha(Fix)	0.56	0.33	$1.45*10^4$	$4.72*10^6$	$9.41*10^5$	$5.36*10^{-4}$	0.32	1.17
Hand	0.89	0.63	$6.76*10^3$	$1.89*10^6$	$2.41*10^4$	1.06	1.34	16.54
Hand(Fix)	1.38	1.02	$5.36*10^3$	$1.28*10^{11}$	$1.36*10^4$	0.01	1.67	11.70
Santa	0.84	0.63	$1.57*10^5$	$6.89*10^7$	$5.42*10^5$	2.10	2.19	79.93
Santa(Fix)	1.43	1.06	$1.38*10^5$	$2.96*10^{27}$	$2.74*10^9$	$9.00*10^{-4}$	1.74	13.82
Slime	0.35	0.29	$5.18*10^4$	$5.59*10^6$	$2.27*10^5$	0.24	0.24	1.34
Slime(Fix)	0.66	0.43	$2.47*10^3$	$1.81*10^6$	$2.06*10^5$	$3.34*10^{-4}$	0.39	1.41
Torso	0.68	0.55	$8.55*10^4$	$6.44*10^6$	$2.91*10^5$	1.46	1.47	28.11
Torso(Fix)	1.17	0.87	$9.38*10^3$	$1.25*10^6$	$1.87*10^5$	$9.55*10^{-4}$	1.31	8.07



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

Figure 17: (a) and (b) are the planar mappings and corresponding texturings by updating Laplacian matrix; (c) and (d) are by fixing Laplacian matrix.

know, this is the first approach which can mix two pieces of 3D and 2D mesh together seamlessly.

The comparisons. As same as some recent developed algorithms [FL16, SPSH*17], we also compare all kinds of mapping distortions of our algorithm with the ones in [RPPSH17], which is the standard benchmark of bijective parameterizations.

For a multi-core implementation, we report the running time of some of our experimental meshes of disk topology in Table 4. The number of iterations and timing are the same magnitude as the [RPPSH17]. On most of the meshes, our approach converges in a few dozens or a hundred iterations. Our algorithm also scales well on meshes in huge size as long as the linear system solver can process it.

Table 4: Timing of our planar mapping algorithm.

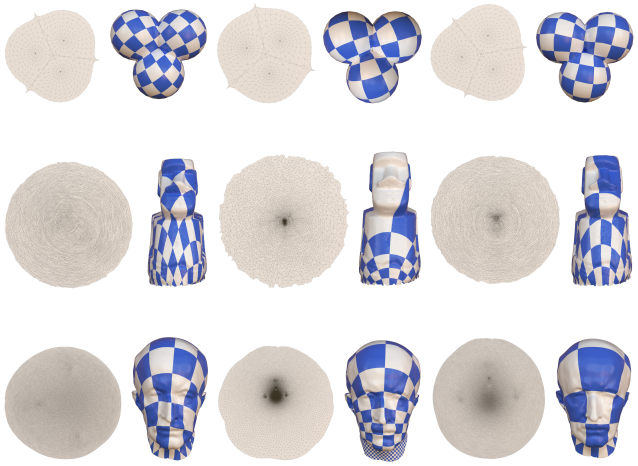
Model	Faces	Iteration	Time(s)	Vertices
Bimba	11k	26	1.089	6k
Beetle	39k	14	1.777	20k
MaxPlanck	85k	29	8.458	42k
Gargoyle	99k	57	18.178	50k
Bunny	100k	45	15.375	50k
NicoloDaUzzano	100k	17	5.599	50k
Mannequin	109k	27	9.119	55k
Surperman	190k	25	15.144	96k
Bear	296k	21	19.31	148k
Buddha	471k	20	30.657	236k

Due to the scope of the paper, we demonstrate the mapping and texturing of our method and [RPPSH17] for several models in Fig. 18 and list the distortions in Table 6. More demos are provided in the supplemental material. We observe that our method is less isometric than SLIM.

5.3. Spherical topology

In Fig. 6, we demonstrate the deformation and parameterization of several meshes of spherical topology. More exhibitions are shown in the supplemental and attached video. Unlike the planar one, the spherical unit normal flow converges slower and needs several thousands iterations for most meshes.

The method of conformal Willmore flow [CPS13] requires the meshes discretized without obtuse angles, which is a highly strict requirement. But our method and cMCF can be applied to most meshes. We exhibit the deformations of our method, cMCF [KSBC12] and conformal Willmore flow [CPS13] on the same bunny model of 28576 faces in Figure 25. The measurements are summarized in the Table 5. We notice that our method shows better edge preserving and isometric performance than cMCF and conformal Willmore flow.



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

Figure 18: (a) and (b) are the parameterization and texturing of SLIM [RPPSH17]; (c), (d) are of the results of our unit normal flow by fixed Laplacian matrix; (e), (f) are the ones by updated Laplacian matrix.

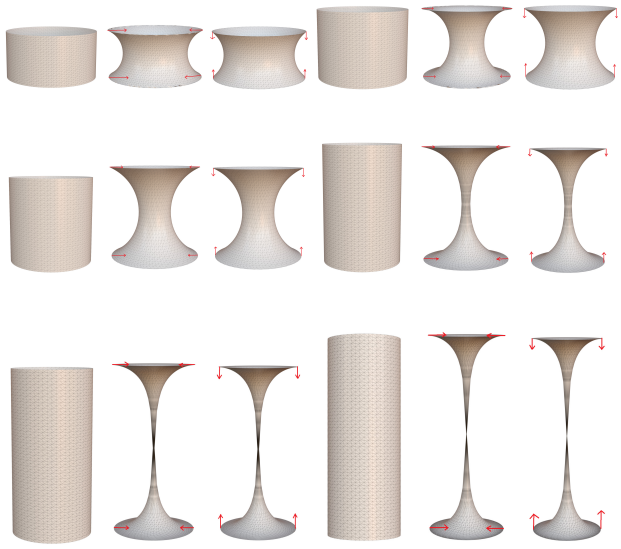
Table 5: The distortions of our method(UNF), cMCF [KSBC12], and conformal Willmore flow [CPS13].

	cMCF	Willmore	our method
AreaMean	$2.08*10^{-5}$	$2.09*10^{-5}$	$2.38*10^{-5}$
EdgeMean	$1.01*10^{-5}$	$1.01*10^{-5}$	$9.56*10^{-6}$
AngleMean	0.04	0.14	0.12
Symmeric Dirichlet	$4.18*10^{9}$	$3.7*10^{9}$	$5.79*10^{7}$
Conformal AMIPS 2D	$5.72*10^{4}$	$6.03*10^{4}$	$6.14*10^{4}$
ASAP	$7.31*10^{-4}$	0.10	0.04
ARAP	0.46	0.52	0.48
Green Lagrange	1.07	1.62	1.77

In Fig. 24, Table 7 and the supplemental material, we show the spherical mappings and distortions of our method, cMCF and the harmonic approach [GWC*04]. In all fourteen meshes, the harmonic approach can not obtain bijective parameterization; the mappings flip in only one model by our algorithm, six models by cMCF. We observe that our method is more conformal than the harmonic approach.

5.4. CMC-like surfaces

Plane and sphere are special and simple constant mean curvature surfaces. Our approximation and discretization of unit normal flow work successfully on them. For other kinds of CMC surfaces, our algorithm can also drive the flow to deform the corresponding discrete meshes. However, the convergent shapes are not CMC surfaces in an exact mathematical sense. We call them CMC-like surfaces.

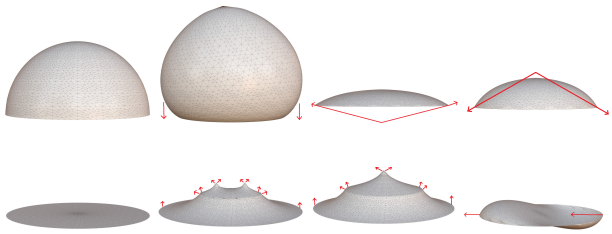


(a) (b) (c) (d) (e) (f)

Figure 19: The CMC-like surfaces generated by unit normal flow. The radii and heights of the cylinders are (10,10),(10,15),(10,20),(10,30),(10,40),(10,50) respectively.

In Figure 19, we demonstrate the convergent shapes of the cylinders of a set of different radii and heights, constrained by two sets of the different boundary face normals, under our unit normal flow. In this experiment, the positions of the boundary vertices of the cylinders are fixed. The red arrows are the representatives of the first set of the boundary face normals, and the green arrows are from the second set. The convergent shapes are catenoid-like surfaces, while the radii and heights we uses does not satisfy the exact mathematical formula of catenoids.

The cylinders with different top and bottom radii are exhibited in Figure 23. In Figure 20, we deform a half-sphere and a unit disk to the different CMC-like surfaces under varying face normals constraints. Unduloid-like surfaces are demonstrated in Fig. 21 and in them, all the boundary face normals are towards the centres. More demons are revealed in Figure 22.



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

Figure 20: The half-sphere and unit disk (a) are assigned three different set of boundary face normals; (b), (c), (d) show their corresponding convergent shapes.

On the one hand, CMC-like surfaces suggest that unit normal

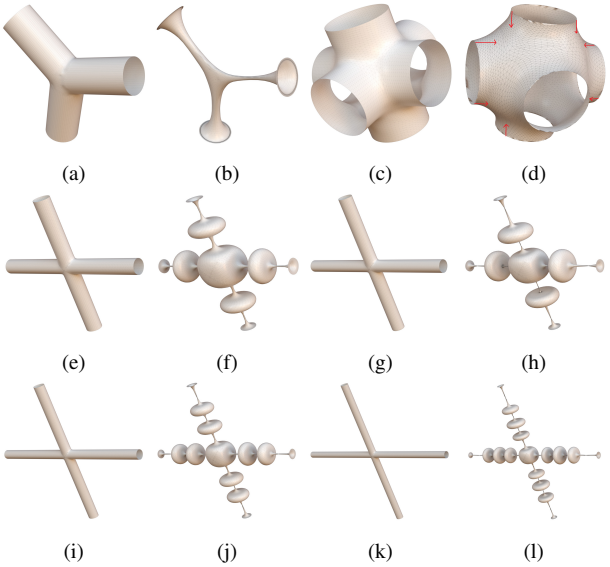


Figure 21: The first and third columns are the original meshes, the second and fourth columns are the convergent CMC-like surfaces. The radii and heights in (e),(g),(i),(k) are (10,120),(10,160),(10,200),(10,300).

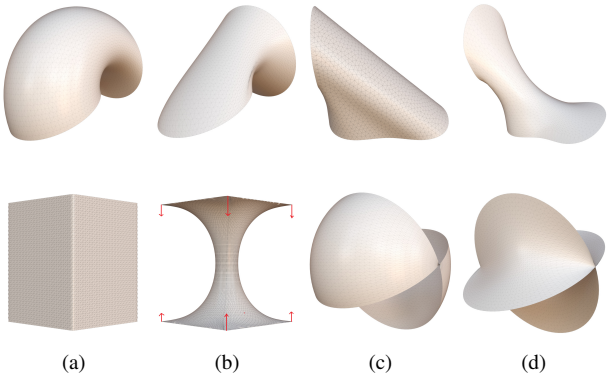


Figure 22: (a),(c) are the original meshes; (b),(d) are the convergent CMC-like surfaces.

flow could be mathematically convergent on smooth surfaces for all CMC surfaces. On the another hand, how to design a more accurate discrete unit normal flow for other kinds of CMC surfaces is a challenging problem which remains as a future work.

6. Conclusion and future work

We propose a special unit normal flow(UNF) to deform surfaces. This flow averages the normals of a smooth surface, and reconstruct the geometry to fit the smoothed normals. We define the mathematical equation of unit normal flow, and prove that the convergent surface has constant mean curvature if the flow is stable and converges. We also present an approximation method on discrete meshes and apply it to the applications of planar and spherical mesh parameterization. Our algorithm provides locally injective mappings.

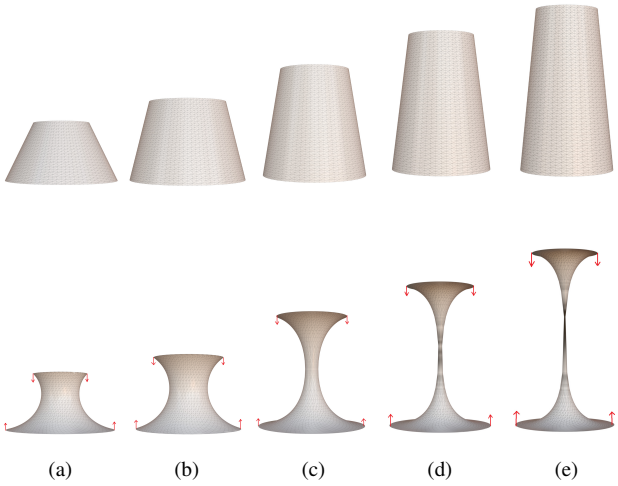


Figure 23: The CMC-like surfaces generated by unit normal flow. The top and bottom radii and heights of the cylinders are (10,20,20),(10,15,20),(10,15,30),(10,15,40),(10,15,50) respectively.

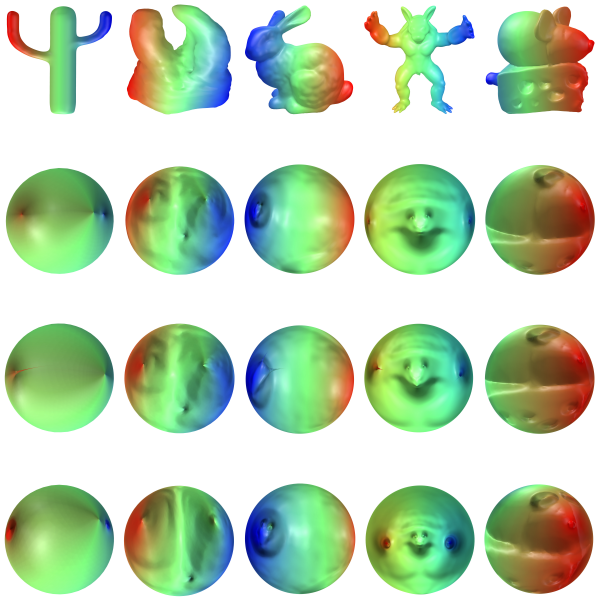
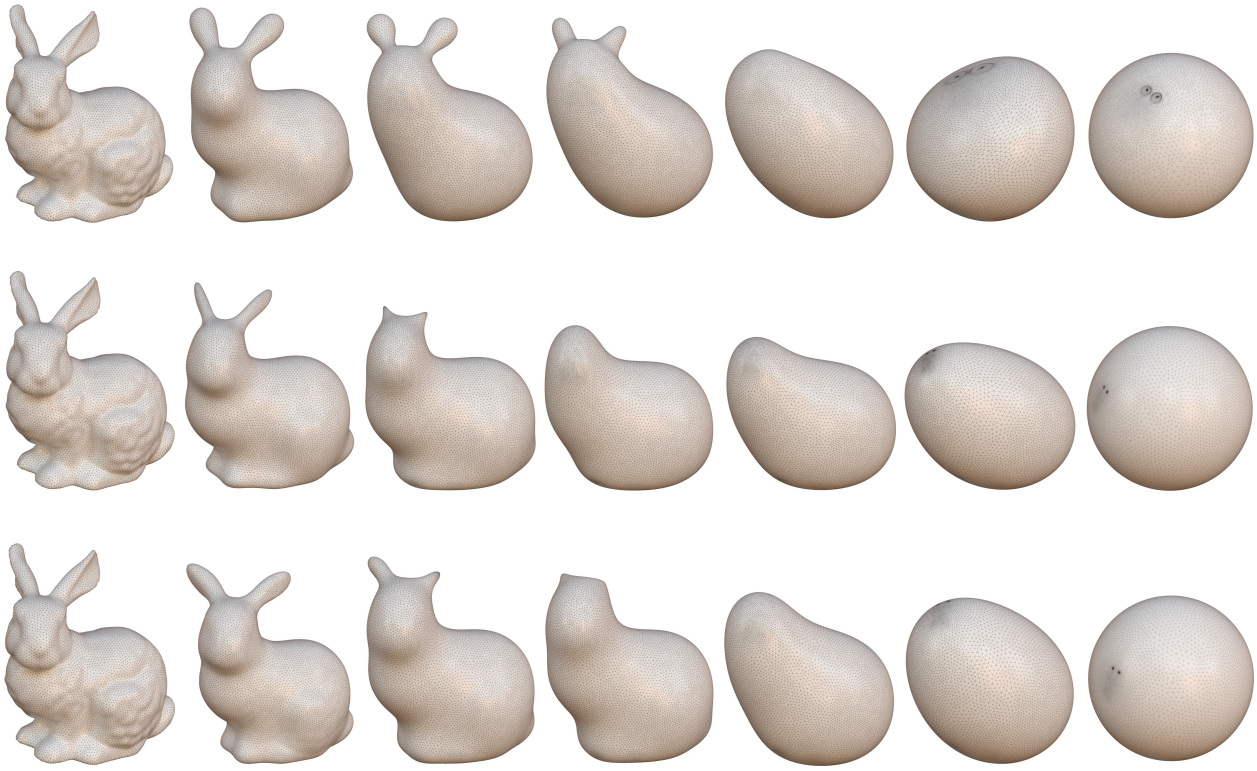


Figure 24: The spheric mapping of cMCF(the second row), harmonic(the thrid row) and UNF methods(the fourth row).

Some important problems remain untouched and will be explored in the future. The reconstruction step is affected by the quality of meshes and will fail if the mesh quality is extremely bad. The convergence, singularity, existences and uniqueness of the unit normal flow are waiting to be proved. It is also a great challenge to design an efficient, stable and accurate discrete algorithm to construct other types of constant mean curvature surfaces besides the planes and spheres.



(a) (b) (c) (d) (e) (f) (g)

Figure 25: The deforming comparision of our method (the first row), cMCF [KSBC12] (the second row) and conformal Willmore flow [CPS13] (the third row).

Model	Method	Area Mean	Edge Mean	Angle Mean	Symm Dirichlet	Conformal AMIPS	ASAP	ARAP	Green Lagrange
Bimba(11k)	SLIM	$4.76*10^{-5}$	$2.65*10^{-6}$	0.42	$8.51*10^{4}$	$3.48*10^{4}$	1.47	1.12	13.76
	UNF(Fix)	$9.89*10^{-5}$	$5.65*10^{-5}$	0.03	$5.71*10^{6}$	$2.26*10^{4}$	$1.45*10^{-3}$	1.23	3.90
	UNF	$6.21*10^{-5}$	$3.26*10^{-5}$	0.31	$1.58*10^{5}$	$3.06*10^{4}$	1.19	1.25	17.71
Bunny(100k)	SLIM	$3.85*10^{-6}$	$2.46*10^{-6}$	0.47	$7.67*10^{5}$	$3.24*10^{5}$	1.31	0.90	9.19
	UNF(Fix)	$1.05*10^{-5}$	$4.58*10^{-6}$	0.09	1.21E+11	$2.00*10^{5}$	$6.68*10^{-4}$	0.96	4.13
	UNF	$5.50*10^{-7}$	$2.69*10^{-6}$	0.35	$9.24*10^{6}$	$2.93*10^{5}$	0.80	0.78	6.71
Buddha(471k)	SLIM	$5.1*10^{-7}$	$2.6*10^{-7}$	0.20	$2.22*10^{6}$	$1.04*10^{6}$	0.22	0.17	0.81
	UNF(Fix)	$1.18*10^{-6}$	$4.6*10^{-7}$	0.01	$4.68*10^{6}$	$9.41*10^{5}$	$4.94*10^{-4}$	0.31	1.06
	UNF	$6.2*10^{-7}$	$2.8*10^{-7}$	0.18	$2.32*10^{6}$	$1.03*10^{6}$	0.19	0.19	0.92

Table 6: The distorions of our unit normal flow (UNF) method and SLIM [RPPSH17]. UNF and UNF(fix) denote the updated and Fixed Laplacian matrice respectively.

References

[AL13] AIGERMAN N., LIPMAN Y.: Injective and bounded distortion mappings in 3d. *ACM Transactions on Graphics (TOG) 32*, 4 (2013), 106. 2

[APL14] AIGERMAN N., PORANNE R., LIPMAN Y.: Lifted bijections for low distortion surface mappings. *ACM Transactions on Graphics (TOG) 33*, 4 (2014), 69. 2

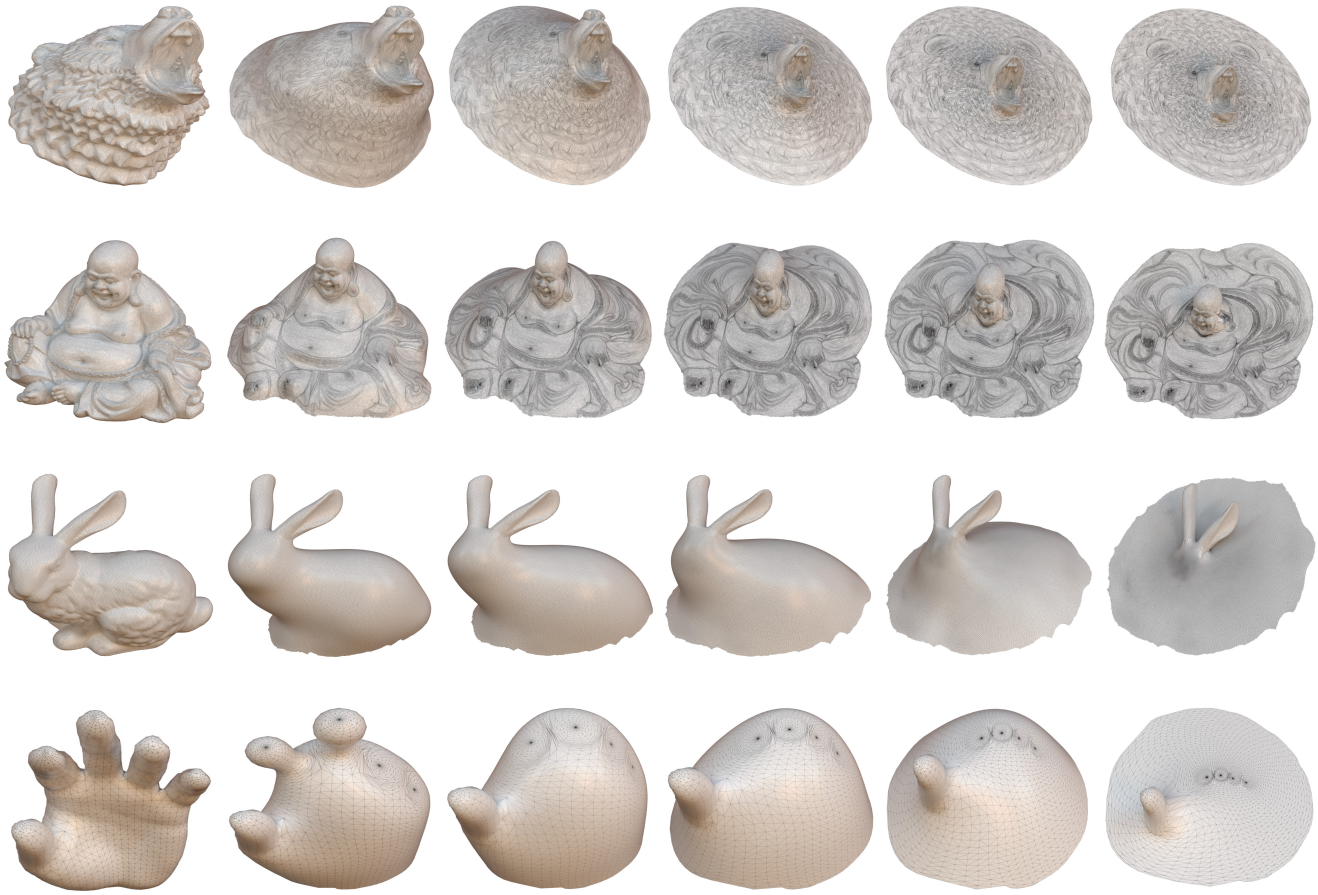
[BCGB08] BEN-CHEN M., GOTSMAN C., BUNIN G.: Conformal flattening by curvature prescription and metric scaling. In *Computer Graphics Forum* (2008), vol. 27, Wiley Online Library, pp. 449–458. 2

[Bra92] BRAKKE K. A.: The surface evolver. *Experimental mathematics 1*, 2 (1992), 141–165. 3

[BS05] BOBENKO A. I., SCHRÖDER P.: Discrete willmore flow. In *ACM SIGGRAPH 2005 Courses* (2005), ACM, p. 5. 3

[Cho93] CHOPP D. L.: Computing minimal surfaces via level set curvature flow. *Journal of Computational Physics 106*, 1 (1993), 77–91. 3

[CMI12] COLDING T. H., MINICOZZI II W. P.: Generic mean curvature flow i; generic singularities. *Annals of mathematics 175*, 2 (2012), 755–833. 2



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

Figure 26: The center faces in the original meshes (a) are selected and kept unchanged in our deformation; (f) exhibits the final partial planar mesh mapping.

Model	Method	Flip	Area Mean	Edge Mean	Angle Mean	Symm Dirich	Conformal AMIPS2D	ASAP	ARAP	Green Lagrange
flow(29k)	UNF	0	$2.38*10^{-5}$	$9.56*10^{-6}$	0.12	$5.79*10^{7}$	$6.14*10^{4}$	0.04	0.48	1.77
	cMCF	0	$2.09*10^{-5}$	$1.01*10^{-5}$	0.04	$4.22*10^{9}$	$5.72*10^{4}$	$8.03*10^{-4}$	0.46	1.08
	Harmonic	1584	$2.21*10^{-5}$	$8.60*10^{-6}$	0.27	$2.31*10^{10}$	$3.60*10^{5}$	0.14	0.45	1.87
armadillo(60k)	UNF	0	$1.95*10^{-5}$	$8.66*10^{-6}$	0.23	$9.81*10^{6}$	$1.43*10^{5}$	0.10	1.20	6.00
	cMCF	0	$1.90*10^{-5}$	$1.04*10^{-5}$	0.13	1.56E+12	$1.21*10^{5}$	$7.09*10^{-4}$	1.24	3.77
	Harmonic	7897	$1.82*10^{-5}$	$8.99*10^{-6}$	0.50	1.35E+12	$1.74*10^{6}$	0.11	1.19	5.62
bimba(100k)	UNF	0	$1.16*10^{-5}$	$3.88*10^{-6}$	0.23	$1.29*10^{6}$	$2.29*10^{5}$	0.56	1.39	34.92
	cMCF	0	$1.26*10^{-5}$	$4.53*10^{-6}$	0.02	$2.23*10^{6}$	$2.00*10^{5}$	$2.87*10^{-3}$	1.14	9.10
	Harmonic	0	$9.50*10^{-6}$	$3.74*10^{-6}$	0.42	$6.16*10^{6}$	$4.94*10^{5}$	1.99	2.02	56.14

Table 7: The distortions of our method(UNF), cMCF [KSBC12] and harmonic spheric mapping [GWC*04].

[CPS13] CRANE K., PINKALL U., SCHRÖDER P.: Robust fairing via conformal curvature flow. *ACM Transactions on Graphics (TOG) 32*, 4 (2013), 61. 3, 4, 9, 10, 12

[CPSS10] CHAO I., PINKALL U., SANAN P., SCHRÖDER P.: A simple geometric model for elastic deformations. In *ACM transactions on graphics (TOG)* (2010), vol. 29, ACM, p. 38. 2, 5

[DH06] DZIUK G., HUTCHINSON J. E.: Finite element approximations to surfaces of prescribed variable mean curvature. *Numerische Mathematik 102*, 4 (2006), 611–648. 3

[DMA02] DESBRUN M., MEYER M., ALLIEZ P.: Intrinsic parameterizations of surface meshes. In *Computer graphics forum* (2002), vol. 21, Wiley Online Library, pp. 209–218. 2

[DMSB99] DESBRUN M., MEYER M., SCHRÖDER P., BARR A. H.: Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques* (1999), ACM Press/Addison-Wesley Publishing

Co., pp. 317–324. 3

[FH05] FLOATER M. S., HORMANN K.: Surface parameterization: a tutorial and survey. In *Advances in multiresolution for geometric modelling*. Springer, 2005, pp. 157–186. 2

[FL16] FU X.-M., LIU Y.: Computing inversion-free mappings by simplex assembly. *ACM Transactions on Graphics (TOG) 35*, 6 (2016), 216. 9

[FLG15] FU X.-M., LIU Y., GUO B.: Computing locally injective mappings by advanced mips. *ACM Transactions on Graphics (TOG) 34*, 4 (2015), 71. 2, 7

[Flo03] FLOATER M.: One-to-one piecewise linear mappings over triangulations. *Mathematics of Computation 72*, 242 (2003), 685–696. 2

[GJ*10] GUENNEBAUD G., JACOB B., ET AL.: Eigen v3. http://eigen.tuxfamily.org, 2010. 6

[GWC*04] GU X., WANG Y., CHAN T. F., THOMPSON P. M., YAU S.-T.: Genus zero surface conformal mapping and its application to brain surface mapping. *IEEE Transactions on Medical Imaging 23*, 8 (2004), 949–958. 10, 13

[H*84] HUISKEN G., ET AL.: *Flow by mean curvature of convex surfaces into spheres*. Australian National University, Centre for Mathematical Analysis, 1984. 3

[HFL18] HU X., FU X.-M., LIU L.: Advanced hierarchical spherical parameterizations. *IEEE transactions on visualization and computer graphics 24*, 6 (2018), 1930–1941. 3

[JKLG08] JIN M., KIM J., LUO F., GU X.: Discrete surface ricci flow. *Visualization and Computer Graphics, IEEE Transactions on 14*, 5 (2008), 1030–1043. 3

[JSP17] JIANG Z., SCHAEFER S., PANOZZO D.: Simplicial complex augmentation framework for bijective maps. *ACM Transactions on Graphics (TOG) 36*, 6 (2017), 186. 3

[Kap90] KAPOULEAS N.: Complete constant mean curvature surfaces in euclidean three-space. *Annals of Mathematics 131*, 2 (1990), 239–330. 3

[KSBC12] KAZHDAN M., SOLOMON J., BEN-CHEN M.: Can mean-curvature flow be modified to be non-singular? In *Computer Graphics Forum* (2012), vol. 31, Wiley Online Library, pp. 1745–1754. 2, 3, 9, 10, 12, 13

[LGZ*16] LIU T., GAO M., ZHU L., SIFAKIS E., KAVAN L.: Fast and robust inversion-free shape manipulation. In *Computer Graphics Forum* (2016), vol. 35, Wiley Online Library, pp. 1–11. 2

[Liu14] LIU C.: *The three-body problem*, vol. 1. Macmillan, 2014. 2

[Liu15] LIU C.: *The dark forest*, vol. 2. Macmillan, 2015. 2

[LPRM02] LÉVY B., PETITJEAN S., RAY N., MAILLOT J.: Least squares conformal maps for automatic texture atlas generation. In *ACM transactions on graphics (TOG)* (2002), vol. 21, ACM, pp. 362–371. 2

[Luo04] LUO F.: Combinatorial yamabe flow on surfaces. *Communications in Contemporary Mathematics 6*, 05 (2004), 765–780. 3

[LYNF18] LIU L., YE C., NI R., FU X.-M.: Progressive parameterizations. *ACM Transactions on Graphics(SIGGRAPH) 37*, 4 (2018). 2

[LZ14] LEVI Z., ZORIN D.: Strict minimizers for geometric optimization. *ACM Transactions on Graphics (TOG) 33*, 6 (2014), 185. 2

[LZX*08] LIU L., ZHANG L., XU Y., GOTSMAN C., GORTLER S. J.: A local/global approach to mesh parameterization. In *Computer Graphics Forum* (2008), vol. 27, Wiley Online Library, pp. 1495–1504. 2, 5, 7

[Mor17] MORRISON M. A.: Death's end. *World Literature Today 91*, 1 (2017), 74–75. 2

[MTAD08] MULLEN P., TONG Y., ALLIEZ P., DESBRUN M.: Spectral conformal parameterization. In *Computer Graphics Forum* (2008), vol. 27, Wiley Online Library, pp. 1487–1494. 2

[PCL*12] PAN H., CHOI Y.-K., LIU Y., HU W., DU Q., POLTHIER K., ZHANG C., WANG W.: Robust modeling of constant mean curvature surfaces. *ACM Transactions on Graphics (TOG) 31*, 4 (2012), 85. 3

[PKC*16] PRADA F., KAZHDAN M., CHUANG M., COLLET A., HOPPE H.: Motion graphs for unstructured textured meshes. *ACM Transactions on Graphics (TOG) 35*, 4 (2016), 108. 3

[PP93] PINKALL U., POLTHIER K.: Computing discrete minimal surfaces and their conjugates. *Experimental mathematics 2*, 1 (1993), 15–36. 3, 4

[PR02] POLTHIER K., ROSSMAN W.: Discrete constant mean curvature surfaces and their index. In *Visualization and Mathematics* (2002), Citeseer. 3

[Ren15] RENKA R. J.: A simple and efficient method for modeling constant mean curvature surfaces. *SIAM Journal on Scientific Computing 37*, 4 (2015), A2076–A2099. 3

[RPPSH17] RABINOVICH M., PORANNE R., PANOZZO D., SORKINE-HORNUNG O.: Scalable locally injective mappings. *ACM Transactions on Graphics (TOG) 36*, 2 (2017), 16. 2, 7, 9, 10, 12

[SCL*16] SU K., CHEN W., LEI N., CUI L., JIANG J., GU X. D.: Measure controllable volumetric mesh parameterization. *Computer-Aided Design 78* (2016), 188 – 198. SPM 2016. 3

[SCL*17] SU K., CHEN W., LEI N., ZHANG J., QIAN K., GU X.: Volume preserving mesh parameterization based on optimal mass transportation. *Computer-Aided Design 82* (2017), 42 – 56. Isogeometric Design and Analysis. 3

[SCQ*16] SU K., CUI L., QIAN K., LEI N., ZHANG J., ZHANG M., GU X. D.: Area-preserving mesh parameterization for poly-annulus surfaces based on optimal mass transportation. *Computer Aided Geometric Design 46* (2016), 76 – 91. 3

[SdS01] SHEFFER A., DE STURLER E.: Parameterization of faceted surfaces for meshing using angle-based flattening. *Engineering with computers 17*, 3 (2001), 326–337. 2

[SK01] SCHNEIDER R., KOBBELT L.: Geometric fairing of irregular meshes for free-form surface design. *Computer aided geometric design 18*, 4 (2001), 359–379. 3

[SKPSH13] SCHÜLLER C., KAVAN L., PANOZZO D., SORKINE-HORNUNG O.: Locally injective mappings. In *Computer Graphics Forum* (2013), vol. 32, Wiley Online Library, pp. 125–135. 2

[SPR*07] SHEFFER A., PRAUN E., ROSE K., ET AL.: Mesh parameterization methods and their applications. *Foundations and Trends® in Computer Graphics and Vision 2*, 2 (2007), 105–171. 2

[SPSH*17] SHTENGEL A., PORANNE R., SORKINE-HORNUNG O., KOVALSKY S. Z., LIPMAN Y.: Geometric optimization via composite majorization. *ACM Trans. Graph 36*, 4 (2017). 9

[SRML07] SUN X., ROSIN P., MARTIN R., LANGBEIN F.: Fast and effective feature-preserving mesh denoising. *IEEE transactions on visualization and computer graphics 13*, 5 (2007). 3

[SS15] SMITH J., SCHAEFER S.: Bijective parameterization with free boundaries. *ACM Transactions on Graphics (TOG) 34*, 4 (2015), 70. 2, 3

[SSP08] SPRINGBORN B., SCHRÖDER P., PINKALL U.: Conformal equivalence of triangle meshes. In *ACM Transactions on Graphics (TOG)* (2008), vol. 27, ACM, p. 77. 3

[Tau95] TAUBIN G.: A signal processing approach to fair surface design. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques* (1995), ACM, pp. 351–358. 3

[Tau01] TAUBIN G.: Linear anisotropic mesh filtering. *Res. Rep. RC2213 IBM 1*, 4 (2001). 3

[TWBO02] TASDIZEN T., WHITAKER R., BURCHARD P., OSHER S.: Geometric surface smoothing via anisotropic diffusion of normals. In *Proceedings of the conference on Visualization'02* (2002), IEEE Computer Society, pp. 125–132. 3

[TWBO03] TASDIZEN T., WHITAKER R., BURCHARD P., OSHER S.: Geometric surface processing via normal maps. *ACM Transactions on Graphics (TOG) 22*, 4 (2003), 1012–1033. 3

[WBH*07] WARDETZKY M., BERGOU M., HARMON D., ZORIN D., GRINSPUN E.: Discrete quadratic curvature energies. *Computer Aided Geometric Design 24*, 8 (2007), 499–518. 3

[WLL14] WANG C., LIU Z., LIU L.: As-rigid-as-possible spherical parametrization. *Graphical Models 76*, 5 (2014), 457–467. 3

[XPB06] XU G., PAN Q., BAJAJ C. L.: Discrete surface modelling using partial differential equations. *Computer Aided Geometric Design 23*, 2 (2006), 125–145. 3

[XZ08] XU G., ZHANG Q.: A general framework for surface modeling using geometric partial differential equations. *Computer Aided Geometric Design 25*, 3 (2008), 181–202. 3

[XZWB06] XU D., ZHANG H., WANG Q., BAO H.: Poisson shape interpolation. *Graphical Models 68*, 3 (2006), 268–281. 3

[YLZG18] YU X., LEI N., ZHENG X., GU X.: Surface parameterization based on polar factorization. *Journal of Computational and Applied Mathematics 329* (2018), 24–36. 3

[YOB02] YAGOU H., OHTAKE Y., BELYAEV A.: Mesh smoothing via mean and median filtering applied to face normals. In *Geometric Modeling and Processing, 2002. Proceedings* (2002), IEEE, pp. 124–131. 3

[YZX*04] YU Y., ZHOU K., XU D., SHI X., BAO H., GUO B., SHUM H.-Y.: Mesh editing with poisson-based gradient field manipulation. *ACM Transactions on Graphics (TOG) 23*, 3 (2004), 644–651. 3

[ZFAT11] ZHENG Y., FU H., AU O. K.-C., TAI C.-L.: Bilateral normal filtering for mesh denoising. *IEEE Transactions on Visualization and Computer Graphics 17*, 10 (2011), 1521–1530. 3

[ZG16] ZHAO H., GORTLER S. J.: A report on shape deformation with a stretching and bending energy. *CoRR abs/1603.06821* (2016). URL: http://arxiv.org/abs/1603.06821. 3, 5

[ZLG*18] ZHAO H., LI X., GE H., LEI N., ZHANG M., WANG X., GU X.: Conformal mesh parameterization using discrete calabi flow. *Computer Aided Geometric Design* (2018). 3

[ZLJW06] ZHANG L., LIU L., JI Z., WANG G.: Manifold parameterization. In *Advances in Computer Graphics*. Springer, 2006, pp. 160–171. 3

[ZLL*17] ZHAO H., LEI N., LI X., ZENG P., XU K., GU X.: Robust Edge-Preserved Surface Mesh Polycube Deformation. In *Pacific Graphics Short Papers* (2017), Barbic J., Lin W.-C., Sorkine-Hornung O., (Eds.), The Eurographics Association. doi:10.2312/pg.20171319. 3, 5

[ZMT05] ZHANG E., MISCHAIKOW K., TURK G.: Feature-based surface parameterization and texture mapping. *ACM Transactions on Graphics (TOG) 24*, 1 (2005), 1–27. 7

[ZRKS05] ZAYER R., RÖSSL C., KARNI Z., SEIDEL H.-P.: Harmonic guidance for surface deformation. In *Computer Graphics Forum* (2005), vol. 24, Wiley Online Library, pp. 601–609. 3