Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 32 (2014) 1–17

Procedia Computer Science

Anisotropic Surface Meshing with Conformal Embedding

Zichun Zhong*[a], Liang Shuai*[a], Miao Jin[b], Xiaohu Guo†[a]

[a]*Department of Computer Science, University of Texas at Dallas, USA*
[b]*Center for Advanced Computer Studies, University of Louisiana at Lafayette, USA*

Abstract

This paper introduces a parameterization-based approach for anisotropic surface meshing. Given an input surface equipped with an arbitrary Riemannian metric, this method generates a metric-adapted mesh with user-specified number of vertices. In the proposed method, the edge length of the input surface is directly adjusted according to the given Riemannian metric at first. Then the adjusted surface is conformally embedded into a parametric 2D domain and a weighted Centroidal Voronoi Tessellation and its dual Delaunay triangulation are computed on the parametric domain. Finally the generated Delaunay triangulation can be mapped from the parametric domain to the original space, and the triangulation exhibits the desired anisotropic property. We compute the high-quality remeshing results for surfaces with different types of topologies and compare our method with several state-of-the-art approaches in anisotropic surface meshing by using the standard measurement criteria.

Keywords: Anisotropic Surface Meshing, Anisotropic Centroidal Voronoi Tessellation, Conformal Embedding

1. Introduction

Nowadays, anisotropic meshes are important for improving the accuracy of the numerical simulations as well as better approximating the shapes [1, 2]. The anisotropic meshes are designed as elongated mesh elements with desired orientations and aspect ratios as user specified. These types of meshes are used in movies, animations, computer-aided design (CAD), computer-aided manufacturing (CAM), architecture design, and scientific visualization, etc. They can provide more accurate approximations for the surface of the original object than the correspondent isotropic counterpart, in regions where the magnitudes of two principal curvatures are different. To accurately simulate the behavior of the physical phenomena, such as the flow of water, air across the earth, the deformation and wrinkles of clothes, the anisotropic meshes are preferred. In this paper, our goal is to generate the anisotropic triangle mesh with stretching ratios and directions conforming to the user-specified metric tensor field.

One typical way to generate the anisotropic triangular mesh is to compute Anisotropic Centroidal Voronoi Tessellation (ACVT) [3, 4] and its dual mesh. This method needs to compute an Anisotropic Voronoi Diagram (AVD) in the ambient 3D space and its intersection of a surface, with sites constrained on the surface. This technique may lead to disconnected Voronoi cells without the topology control, if two regions are very close in 3D space but are far away along the surface. Especially with large anisotropic stretching ratios, the computed constrained ACVT on surface tends to be incorrect. It is natural to use the geodesic distance to compute the constrained ACVT, but the computation of the geodesic distance is difficult to be accurate and efficient.

[1]*Joint first authors: the first two authors contributed equally to this work.
[2]†Corresponding author.

To overcome the above limitations, we propose a new method for anisotropic meshing of surfaces endowed with Riemannian metrics. Theoretically, the best way for anisotropic mesh generation is to compute a perfect isometric embedding of the Riemannian surface into a high-dimensional Euclidean space [5], where the metric becomes uniformly identity, and then directly compute Centroidal Voronoi Tessellation (CVT) on it. However, it is very challenging to find such a high-dimensional isometric embedding – so far we have not seen any method available to compute it. So we take an alternative approach: instead of isometrically embedding the Riemannian surface into a high-dimensional space, we "conformally" embed the surface into a 2D parametric domain, followed by a weighted CVT computation on such 2D parametric domain. Such conformal embedding is achieved by: (1) adjusting only the edge length of the input surface according to the given Riemannian metric, without explicitly computing the vertex coordinates (i.e. its embedding); (2) conformally deforming the edge length so that the surface can be embedded into a 2D parametric domain. Here the conformal embedding tries to preserve the angle of the input surface triangles, thus it preserves the local aspect-ratios of the input metric. Then we rely on the computation of weighted CVT on this 2D parametric domain with a density function applied to compensate for the introduced area distortion of surface parameterization.

An advantage of this method is its efficiency since the CVT computation is performed on a 2D parametric domain, as compared to the traditional approach of computing ACVT and its intersection with the surface in 3D space [3, 4]. Finally the dual triangulation of computed CVT is mapped back to the original domain and an anisotropic meshing result is gained. In Sec. 5 we show the comparison with existing approaches in anisotropic surface meshing by using the standard measurement criteria.

2. Backgrounds and Related Works

2.1. Anisotropic Metric

Anisotropy defines the distortion of the distances and angles. Consider the domain $\Omega \subset \mathbb{R}^m$, and a given point $\mathbf{x} \in \Omega$ with endowed metric $\mathbf{M}(\mathbf{x})$. The anisotropic squared length of a vector $\mathbf{a}$ at $\mathbf{x}$ can be measured by the dot product between $\mathbf{a}$ and itself, using the metric $\mathbf{M}(\mathbf{x})$ as follows:

$$||\mathbf{a}||^2_{\mathbf{M}(\mathbf{x})} = \langle \mathbf{a}, \mathbf{a} \rangle_{\mathbf{M}(\mathbf{x})} = \mathbf{a}^T \mathbf{M}(\mathbf{x}) \mathbf{a}. \tag{1}$$

The metric matrix $\mathbf{M}(\mathbf{x})$ is a symmetric positive-definite (SPD) $m \times m$ matrix, which can be decomposed with Singular Value Decomposition (SVD) into:

$$\mathbf{M}(\mathbf{x}) = \mathbf{R}(\mathbf{x})^T \mathbf{S}(\mathbf{x})^2 \mathbf{R}(\mathbf{x}), \tag{2}$$

where the diagonal matrix $\mathbf{S}(\mathbf{x})^2$ contains its ordered eigenvalues, and the orthonormal matrix $\mathbf{R}(\mathbf{x})$ contains its eigenvectors. If we denote $\mathbf{Q}(\mathbf{x}) = \mathbf{S}(\mathbf{x})\mathbf{R}(\mathbf{x})$, by combining Eqs. (1) and (2), we can understand the anisotropic squared length of vector $\mathbf{a}$ as the "isotropic" squared length of its transformed vector $\mathbf{b} = \mathbf{Q}(\mathbf{x})\mathbf{a}$, as:

$$||\mathbf{b}||^2_{\mathbf{I}} = \mathbf{b}^T \mathbf{b} = \mathbf{a}^T \mathbf{M}(\mathbf{x}) \mathbf{a} = ||\mathbf{a}||^2_{\mathbf{M}(\mathbf{x})}, \tag{3}$$

where $\mathbf{I}$ is the identity matrix. This means the vector $\mathbf{a}$ is rotated by $\mathbf{R}(\mathbf{x})$ and then scaled by $\mathbf{S}(\mathbf{x})$, before measuring its Euclidean length.

For the experiments given in this paper (Sec. 4), users start from designing a smooth scaling field $\mathbf{S}(\mathbf{x})$ and a rotation field $\mathbf{R}(\mathbf{x})$ that is smooth in regions other than some singularities, and then compose them to $\mathbf{Q}(\mathbf{x}) = \mathbf{S}(\mathbf{x})\mathbf{R}(\mathbf{x})$ and $\mathbf{M}(\mathbf{x}) = \mathbf{Q}(\mathbf{x})^T \mathbf{Q}(\mathbf{x})$, which is similar to the way of Du et al. [3]. The metrics are defined on the tangent spaces of the surface. Suppose s_1 and s_2 are the two diagonal items in $\mathbf{S}(\mathbf{x})$ corresponding to the two eigenvectors in the tangent space, and $s_1 \leq s_2$. Then we can define the *stretching ratio* as $\frac{s_2}{s_1}$ [6]. For the anisotropic meshing on 3D surfaces, we use the following metric tensor:

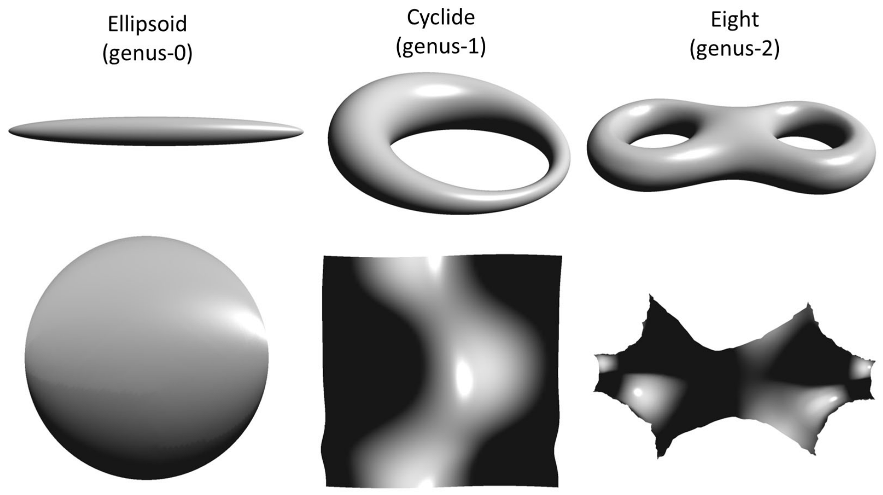
$$\mathbf{M} = [\mathbf{v}_{min}, \mathbf{v}_{max}, \mathbf{n}] diag({s_1}^2, {s_2}^2, 0) [\mathbf{v}_{min}, \mathbf{v}_{max}, \mathbf{n}]^T, \tag{4}$$

where $\mathbf{v}_{min}$ and $\mathbf{v}_{max}$ are the directions of the principal curvatures, $\mathbf{n}$ is the unit surface normal. s_1 and s_2 are two user-specified stretching factors along principal directions. To ensure smoothness of the input metric field, as suggested by Alliez et al. [7], we apply Laplacian smoothing to both the stretching ratios and directions.

2.2. Universal Covering Space and Its Mappings

Let Σ be a surface embedded in R^3. Let $\mathbf{g}$ be the Riemannian metric of Σ induced from its Euclidean metric in R^3. Suppose $u : \Sigma \rightarrow R$ is a scalar function defined on Σ. Then $\bar{\mathbf{g}} = e^{2u}\mathbf{g}$ is also a Riemannian metric on Σ. Angles measured by $\mathbf{g}$ are equal to those measured by $\bar{\mathbf{g}}$. Therefore, we say $\bar{\mathbf{g}}$ is conformal to the original metric.

According to the Uniformization Theorem [8], any surface admits a Riemannian metric of a constant Gaussian curvature, which is conformal to the original one. Such metric is called the uniformization metric. Specifically, surfaces with positive Euler characteristics (i.e., $\chi > 0$) exist spherical uniformization metric with +1 Gaussian curvature everywhere. Surfaces with zero Euler characteristics (i.e., $\chi = 0$) exist Euclidean uniformization metric with 0 Gaussian curvature everywhere. Surfaces with negative Euler characteristics (i.e., $\chi < 0$) exist hyperbolic uniformization metric with -1 Gaussian curvature everywhere.



(a) $\chi > 0,\ \mathbb{S}^2$ (b) $\chi = 0,\ \mathbb{E}^2$ (c) $\chi < 0,\ \mathbb{H}^2$

Figure 1: The universal covering space of a closed surface can be isometrically embedded into one of the three canonical domains: sphere, plane, or hyperbolic space with its uniformization metric. Here, we show the embedding of one domain based on the uniformization metric. χ is the Euler characteristic number of the surface.

A universal covering space of a surface, explained in an intuitive and informal way, is a simply connected periodic tessellation of the surface domain. Locally, the mapping between the surface and its universal covering space is one-to-one and continuous. Based on the uniformization metric, the universal covering space of a closed surface can be isometrically embedded into one of the three canonical domains: the spherical domain $\mathbb{S}^2$ for genus zero surfaces with positive Euler numbers, the planar domain $\mathbb{E}^2$ for genus one surfaces with zero Euler number, and the hyperbolic space $\mathbb{H}^2$ for high genus surfaces with negative Euler numbers (see Fig. 1).

In particular, our work appears to be mainly an extension of the techniques in [9, 10] to the anisotropic setting. Additionally, the use of conformal parameterization for anisotropic remeshing has been mentioned in [11], but it takes topological disk as input that means any given surface has to be cut into topological disks. This is a big limitation for parameterization process. Meanwhile, in this reference, no specific details are given to tell the reader how to solve the anisotropic meshing by using conformal mapping. Instead of using CVT optimization method, their proposed method cannot obtain an optimal meshing result, and there is no quality evaluation for the anisotropic meshing case.

2.3. Anisotropic Centroidal Voronoi Tessellation

Given a set of sites $\mathbf{X} = \{\mathbf{x}_i | i = 0...n-1\}$ to sample the surface domain Ω, Centroidal Voronoi Tessellation (CVT) is a special type of Voronoi Diagram such that each site $\mathbf{x}_i$ coincides exactly with the centroid of its Voronoi cell. CVT can generate regular hexagonal Voronoi cells [12], and its dual graph is a Delaunay triangulation of $\mathbf{X}$ with well-shaped isotropic triangles. There are two main methods to compute CVT: one is the Lloyd relaxation [13], and the other is a quasi-Newton energy optimization solver [14].

In surface meshing, it is possible to use a geodesic Voronoi Diagram over the surface [15] to generalize this definition. In order to simplify the computations, some researchers compute the Restricted Voronoi Diagram (RVD) or Restricted Delaunay Triangulation (RDT) [16], instead of the geodesic Voronoi Diagram. In [17], a Restricted Centroidal Voronoi Tessellation is provided, which requires all the sites to be constrained on the surface. Using the 3D Euclidean distance as an approximation, Yan et al. [18] computed the CVT in 3D space and then intersected it with the surface. However, this approach is not efficient. When computing CVT on a surface in 3D space, during optimization the sites need to be projected and constrained on the input surface and the gradients are computed approximately in the tangent plane in each iteration. An alternative approach is to compute the CVT in the 2D parametric domain of the surface [19, 9, 10].

CVT was further generalized to the anisotropic version by Du et al. [3] using the following definition of anisotropic Voronoi cell $\mathrm{Vor}_{\mathrm{A}}$:

$$\begin{aligned} \mathrm{Vor}_{\mathrm{A}}(\mathbf{x}_i) &= \{\mathbf{y} | d_{\mathbf{y}}(\mathbf{x}_i, \mathbf{y}) \leq d_{\mathbf{y}}(\mathbf{x}_j, \mathbf{y}), \forall j\}, \\ \text{where:} & \\ d_{\mathbf{x}}(\mathbf{y}, \mathbf{z}) &= \sqrt{(\mathbf{z} - \mathbf{y})^T \mathbf{M}(\mathbf{x})(\mathbf{z} - \mathbf{y})}, \end{aligned} \tag{5}$$

where the sites $\mathbf{x}_i$, $\mathbf{x}_j$, and points $\mathbf{x}$, $\mathbf{y}$, $\mathbf{z}$ are all in the domain Ω. An anisotropic Voronoi Diagram with the given Riemannian metric needs to be constructed in each Lloyd iteration, which is a time-consuming operation. To make the computation much faster, Valette et al. [20] proposed a discrete approximation of ACVT by clustering the vertices of a dense pre-triangulation of the domain, at the expense of degraded mesh quality.

Lévy and Bonneel [21] extended the computation of CVT to a 6D space in order to achieve a curvature-adaptation. The main idea of their method is to transform the anisotropic meshing on a 3D surface to an isotropic one embedded in 6D space, which can be efficiently computed by CVT with Voronoi Parallel Linear Enumeration.

So far, there is no prior research work, which gave detailed elaboration about using parameterization-based method to compute ACVT and its dual mesh on surfaces. In this paper, we provide such an anisotropic surface meshing method, which avoids the construction of AVD on the surface in the intermediate iterations of energy optimization, and it is more efficient since the computation is performed in a 2D parametric domain. Thus, our method demonstrates better performance than previous ACVT methods as shown in Sec. 5.1.

2.4. *Refinement-Based Anisotropic Delaunay Triangulation*

Refinement-based anisotropic Delaunay triangulation is the anisotropic version of point insertion in Delaunay triangulation, and has been applied to many practical applications [22, 23, 24]. Cheng et al. [25, 26, 27] applied Delaunay refinement to anisotropic surface meshing. Boissonnat et al. [28, 29] introduced a Delaunay refinement framework, which makes the star around each vertex $\mathbf{x}_i$ to be consisting of the triangles that are exactly Delaunay for the metric associated with $\mathbf{x}_i$. In order to "stitch" the stars of neighboring vertices, refinement algorithms add new vertices gradually to achieve the final anisotropic meshes. Note that the CVT/ACVT-based approaches are very different from the Delaunay refinement, as they optimize both the location and connectivity of vertices on the surface globally.

2.5. *Particle-Based Anisotropic Meshing*

Another way to compute anisotropic meshing is the particle-based approaches. Bossen and Heckbert [30] defined the distance function with the metric tensor to simulate the repulsion and attraction forces between particles. Shimada et al. [31, 32, 33] used a bounded cubic function of the distance to physically-based model the inter-bubble forces, and further extended it to anisotropic meshing by converting spherical bubbles to ellipsoidal ones. Both Bossen et al. and Shimada et al.'s works require dynamic population control schemes, that is to adaptively insert or delete particles/bubbles in certain regions. Thus if the initialization of the number of particles does not have a good estimation to fill the domain, it will take a long time to converge. To overcome this problem, Zhong et al. [6] introduced a particle-based anisotropic meshing approach by formulating the inter-particle energy optimization in a higher dimensional "embedding space" [5]. Such energy optimization strategy shows very fast convergence speed, without any need for the explicit control of particle population. But in their final step of mesh generation (i.e. to connect the particles to form an anisotropic mesh), they still need to compute the Restricted Anisotropic Voronoi Diagram on the surface in 3D space, which requires a fine tessellation of the input surface and is less robust and efficient, given some high stretching ratios of the input metric. The comparison with our approach is shown in Sec. 5.2.

3. Algorithm

3.1. Scheme

In this section, we illustrate the computational scheme of the proposed anisotropic surface meshing method based on conformal embedding. Fig. 2 shows the partial results of each step of our algorithm, taking the Cyclide surface as an example.

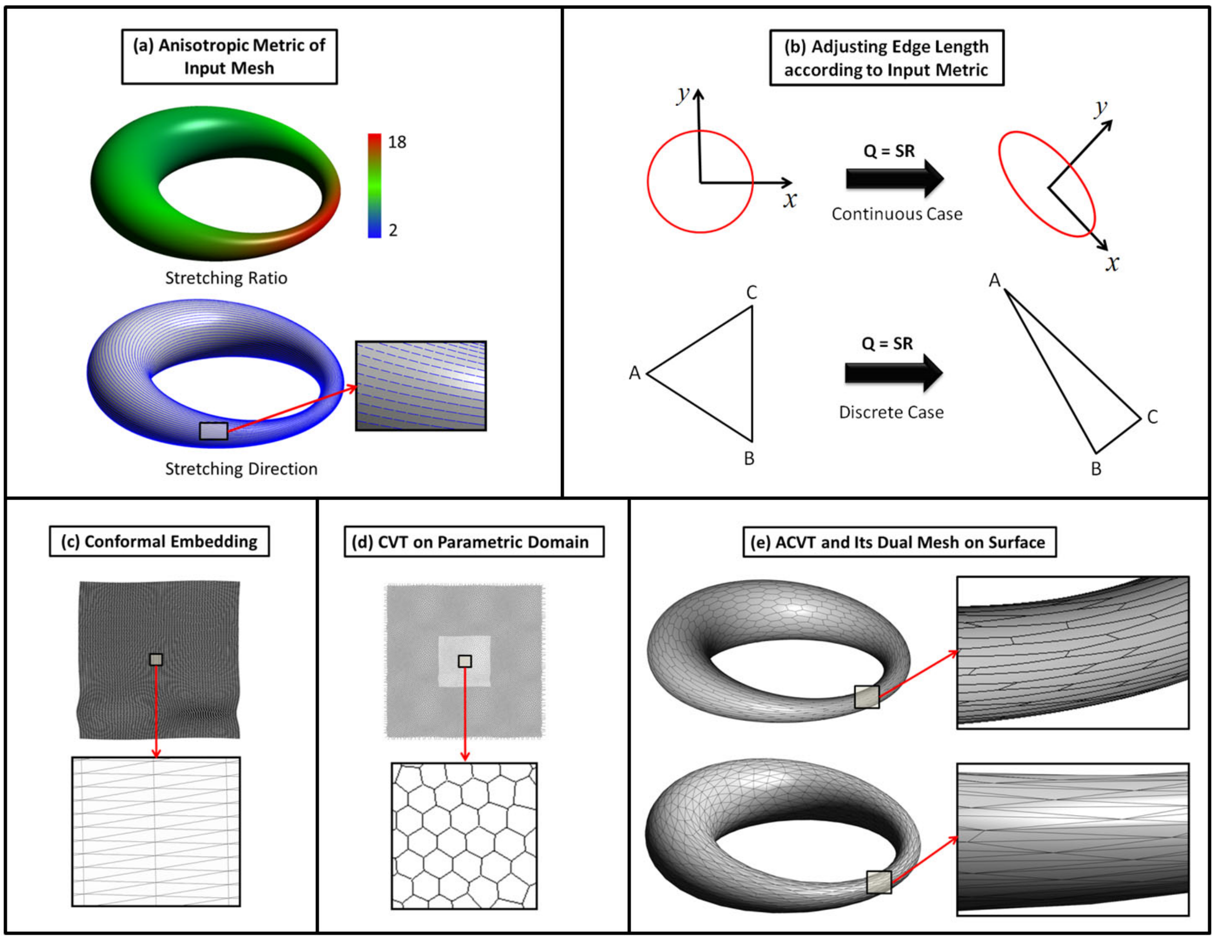

(a) Anisotropic Metric of Input Mesh
18
2
Stretching Ratio
Stretching Direction
(b) Adjusting Edge Length according to Input Metric
y
x
Q = SR
Continuous Case
y
x
A
B
C
Q = SR
Discrete Case
A
B
C
(c) Conformal Embedding
(d) CVT on Parametric Domain
(e) ACVT and Its Dual Mesh on Surface

Figure 2: The partial results in each step of our anisotropic meshing with conformal embedding scheme for the Cyclide surface.

(1) *Specifying Anisotropic Metric of Input Mesh* (Fig. 2 (a)): users specify a smooth metric field for the input mesh surface, as discussed in Sec. 2.1.

(2) *Adjusting Edge Length according to Input Metric* (Fig. 2 (b)): given the input metric, the edge length of the input surface mesh can be adjusted, so that its Euclidean length equals the desired anisotropic length as shown in Eq. (3). The details are provided in Sec. 3.2.

(3) *Conformal Embedding of the Mesh with Adjusted Edge Length* (Fig. 2 (c)): approximation of the angle-preserving parameterization can be computed by conformally adjusting the edge length, e.g. the discrete surface Ricci flow algorithm [34], so that the whole input mesh can be embedded in a 2D parametric domain. Details are introduced in Sec. 3.3.

(4) *Centroidal Voronoi Tessellation on 2D Parametric Domain* (Fig. 2 (d)): a weighted CVT can be computed in this conformally embedded 2D parametric domain, with a density function to compensate for the introduced area distortion during surface parameterization (Sec. 3.4).

(5) *Anisotropic Mesh Generation* (Fig. 2 (e)): finally, the anisotropic surface mesh is computed by simply mapping the dual mesh of the CVT from the 2D parametric domain to the original surface (Sec. 3.5).

3.2. Adjusting Edge Length

To compute the anisotropic meshing on surfaces equipped with Riemannian metric, we can utilize the concept of a higher dimensional "embedding space" [5] where the metric is uniform identity, and then compute CVT in this space. However, it is very challenging to find such a high-dimensional isometric embedding explicitly. In contrast, it has been widely studied in the surface parameterization literature, to "conformally" embed the given Riemannian surfaces into 2D spaces. Thus we can take an alternative approach – conformally embed the surface equipped with anisotropic metric into a 2D domain, and then compute a weighted CVT on it. This is in spirit similar to Rong et al.'s approach [9, 10], but they are only handling isotropic meshing of surfaces, while we generalize it to anisotropic cases.

In order to compute the conformal embedding of the surface equipped with anisotropic metric into a 2D domain, we need to know the edge length of the input surface under the given metric.

For each triangle $\triangle_{ABC}$ with vertex positions $\mathbf{x}_A$, $\mathbf{x}_B$, and $\mathbf{x}_C$, we average the metrics of three vertices $\mathbf{Q}(\triangle_{ABC}) = \frac{\mathbf{Q}(\mathbf{x}_A)+\mathbf{Q}(\mathbf{x}_B)+\mathbf{Q}(\mathbf{x}_C)}{3}$ to approximate the metric for the triangle $\triangle_{ABC}$. Then the edge lengths of each triangle can be deformed according to its own $\mathbf{Q}(\triangle_{ABC})$, but the neighboring triangles may not be able to give consistent edge lengths – their shared edge may have different lengths due to the differences of $\mathbf{Q_s}(\triangle_{ABC})$ across neighboring triangles. We compute the average of these two lengths, and use it as the final deformed edge length. Now, the neighboring triangles are consistent on the shared edge.

However, in some cases, the deformed edge lengths may result in an invalid triangle, i.e. the sum of two edge lengths is less than the third edge length. Here we provide two solutions for this degenerate case: (1) edge flipping, and (2) subdivision.

We modify the Delaunay criterion to remove potential invalid triangles. Suppose $\triangle_{ABD}$ and $\triangle_{BCD}$ are two adjacent triangles (Fig. 3), Delaunay retriangulation flips the edge to maximize the minimum angle. Equivalently, if $\angle DAB + \angle BCD > 180°$ and $\angle DAB + \angle BCD$ (before flipping) $> \angle ABC + \angle CDA$ (after flipping) hold, then we flip edge BD to edge AC. Here we use the averaged metric $\mathbf{Q}_{avg} = \frac{\mathbf{Q}(\mathbf{x}_A)+\mathbf{Q}(\mathbf{x}_B)+\mathbf{Q}(\mathbf{x}_C)+\mathbf{Q}(\mathbf{x}_D)}{4}$, where A, B, C, and D are four vertices being checked [30]. $\mathbf{Q}_{avg}$ is used to compute the lengths of edges formed by these four vertices, and the angles are computed from the law of cosines.

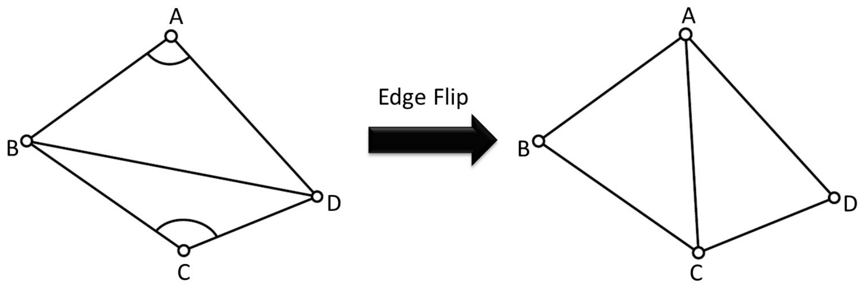


Figure 3: Edge flip.

After edge flipping, if there is still any invalid triangle, we can subdivide the longest edge into two, i.e. add a vertex at the middle of the longest edge, so that we can divide two adjacent triangles (sharing the same edge) into four smaller triangles. The metric of the new added vertex is the average of the corresponding two vertices' metrics.

Through this strategy, the subdivided triangles can be better to smoothly catch the variation of the adjusting edge length, so that it helps to remove the original invalid triangles.

One example (Fig. 4) shows the input triangular mesh before and after edge-flips, and it demonstrates that the edge flipping operation is required to help to remove all invalid triangle (originally 5,286 out of 207,368) and increase the minimal angle to 6.35°. This is because the original mesh connectivity cannot be well compatible with the input metric, so that we have to change some local triangular connectivity. This operation guarantees the following conformal embedding computation feasible.

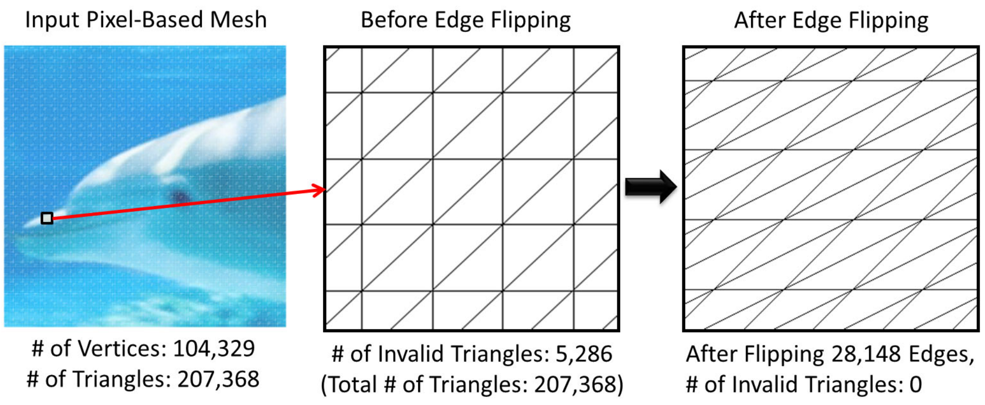

Input Pixel-Based Mesh
Before Edge Flipping
After Edge Flipping
# of Vertices: 104,329
# of Triangles: 207,368
# of Invalid Triangles: 5,286
(Total # of Triangles: 207,368)
After Flipping 28,148 Edges,
# of Invalid Triangles: 0

Figure 4: Edge flipping example of Dolphin image input mesh. Originally there are 5,286 invalid triangles; after edge flipping, there is no invalid triangle.

It should be noted that we can remove all of the invalid triangles by using the edge flipping and subdivision strategies, only if the metric is smooth enough.

As we know, the curvature computations on the piecewise-linear input mesh are always noisy, so an additional stage for smoothing over the preliminary resulting metric field is often most needed. Otherwise, these noisy curvature values will lead to large jumps of metrics between neighboring vertices, and it is highly possible to have illegal edge lengths (i.e. invalid triangles). We provide one example (Fig. 5) to illustrate the necessity of having an enough smoothed metric.

3.3. *Conformal Embedding*

We use edge length to approximate the Riemannian metric of a given discrete surface. We then compute the adjusted edge length as explained in Sec. 3.2. For a closed genus $g = 0$ surface, we apply spherical harmonic method [35] to compute its spherical uniformization metric and embed its universal covering space in a unit sphere. For a closed genus $g = 1$ surface, we apply discrete surface Euclidean Ricci flow [34] to compute its Euclidean uniformization metric and embed its universal covering space to a plane. For a closed genus $g > 1$ surface, we apply discrete surface hyperbolic Ricci flow [34] to compute its hyperbolic uniformization metric and embed its universal covering space to a hyperbolic space. For a topological disk surface, we can apply different tools to conformally map it to a planar domain [36, 34]. The metric in the embedded domain is angle-preserved with the specified surface anisotropic metric. Locally, there is only scaling. It is denoted as conformal factor.

3.4. *Weighted Centroidal Voronoi Tessellation*

Instead of computing ACVT, our meshing generation method is achieved by computing the weighted CVT on the embedding (parametric) domains (i.e. spherical plane $\mathbb{S}^2$, Euclidean plane $\mathbb{E}^2$ and hyperbolic plane $\mathbb{H}^2$ corresponding to different topologies of input surfaces). The density function used to compute weighted CVT is defined as the square of the conformal factor, which measures the scaling of the local area of a conformal embedding. Specifically, the conformal factor is defined as the ratio of incident triangle area sums on each vertex $\mathbf{v}$ of the surface in the original domain with adjusted edge length and in the embedding domain as in Eq. (6). The area of a triangle $\triangle_{ABC}$ with

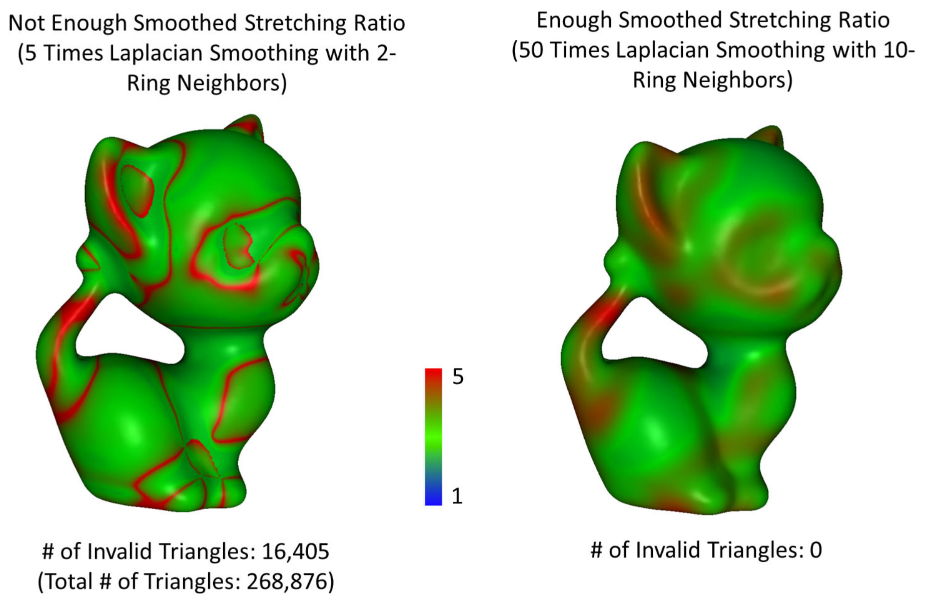

Not Enough Smoothed Stretching Ratio (5 Times Laplacian Smoothing with 2-Ring Neighbors)
Enough Smoothed Stretching Ratio (50 Times Laplacian Smoothing with 10-Ring Neighbors)
5
1
# of Invalid Triangles: 16,405 (Total # of Triangles: 268,876)
# of Invalid Triangles: 0

Figure 5: One example (Kitten surface model) illustrates the necessity of having an enough smoothed metric. The left one shows after 5 times Laplacian smoothing with 2-ring neighbors, there are 16,405 invalid triangles. The right one shows after 50 times Laplacian smoothing with 10-ring neighbors, there is no invalid triangle any more.

adjusted edge lengths a, b and c is computed by Heron's formula as in Eq. (7). The conformal factor is interpolated linearly within triangles when served for computing the weighted CVT.

$$cf(\mathbf{v}) = \frac{\sum \triangle_{adjusted}(\mathbf{v})}{\sum \triangle_{embedded}(\mathbf{v})}. \tag{6}$$

$$\triangle_{ABC} = \frac{1}{4}\sqrt{(a+b+c)(b+c-a)(c+a-b)(a+b-c)}. \tag{7}$$

As elaborated in Rong et al.'s work [10], setting the density function to $cf(\mathbf{v})^2$ will let the dual mesh of the weighted CVT satisfy a constant sizing field on the adjusted-edge-length surface, which means the sites are uniformly distributed on it. When mapping back the sites from the embedding domain to the original surface, their distribution exhibits the desired anisotropic property due to the adjusting edge length process as explained in Sec. 3.2.

The weighted CVT is computed with the Lloyd's algorithm [13]. Starting with arbitrary initial sites, the algorithm computes the Voronoi Diagram for these sites and updates their positions with the centroids of the Voronoi cells iteratively. Finally, the optimization is guaranteed to be converged for the weighted CVT on $\mathbb{S}^2$, $\mathbb{E}^2$, or $\mathbb{H}^2$, as proved in Du et al. [37] and Rong et al. [10].

For different topologies of the surfaces, we compute different types of Voronoi Diagrams w.r.t. their embedding domains. Closed genus-0 surfaces can be embedded on the spherical plane $\mathbb{S}^2$, and the spherical Voronoi Diagram is computed with the STRIPACK algorithm [38].

Closed genus-1 surface can be embedded on the Euclidean plane $\mathbb{E}^2$ periodically. To compute periodic 2D Voronoi Diagram, we make neighbor site copies around a chosen embedding mesh patch and compute the 2D non-periodic Voronoi Diagram with CGAL [39], as suggested in [10]. A topological disk surface can also be embedded on the Euclidean plane $\mathbb{E}^2$, without showing periodicity. So we just simply compute the 2D non-periodic Voronoi Diagram and project boundary sites (whose cells intersect with the disk boundary) to the disk boundary to preserve the original surface boundary.

Closed high-genus (genus >1) surfaces can be embedded on the hyperbolic plane $\mathbb{H}^2$ periodically. It is proved by Nielsen et al. [40] that the 2D hyperbolic Voronoi Diagram on Klein disk is equivalent to a certain *power diagram* [41] on the Euclidean plane. We use this conclusion and compute the corresponding power diagram with CGAL. The neighbor site copies are still needed to deal with the periodicity as suggested in [10]. Since the number of neighbor patches is much larger in high-genus surfaces ($16g^2 - 8g$ neighbor patches for a genus-g surface), we use the technique in [42] to reduce the unnecessary site copies.

The centroid (or center of mass) of a region with a given density function on the Euclidean plane $\mathbb{E}^2$ is well defined in mathematics. This definition has been extended to the spherical plane $\mathbb{S}^2$ and the hyperbolic plane $\mathbb{H}^2$ with the concept of *model centroid* [43] as suggested in [10]. With these definitions, we can compute the centroids of Voronoi cells numerically on different embedding domains after computing the Voronoi Diagram in each Lloyd iteration, by setting the density function as the square of conformal factor as explained at the beginning of this subsection.

3.5. Anisotropic Mesh Generation

After we compute the weighted CVT, the final mesh is generated as the dual of the Voronoi Diagram in the parametric domain. Using the barycentric coordinates of each output vertex, we can map the dual mesh from the parametric domain onto the original surface, and generate the final anisotropic meshing results.

3.6. Evaluation Criteria for Mesh Quality

According to the quality measurements of isotropic triangular mesh suggested by Frey and Borouchaki [44], we use the following criteria for isotropic case: The quality of a triangle is measured by $G = 2\sqrt{3}\frac{S}{ph}$, where S is the triangle area, p is its half-perimeter, and h is the length of its longest edge.

(1) G_{min} is the minimal quality of all triangles;
(2) G_{avg} is the average quality of all triangles;
(3) θ_{min} is the smallest angle of the minimal angles of all triangles;
(4) θ_{avg} is the average angle of the minimal angles of all triangles;
(5) $\%_{<30^\circ}$ is the percentage of triangles with their minimal angles smaller than 30°;
(6) The angle histogram is also provided to show the distribution of angles for all triangles.

To measure the quality of anisotropic surface meshes, for each triangle $\triangle_{ABC}$ we use its approximated metric $\mathbf{Q}(\triangle_{ABC}) = \frac{\mathbf{Q}(\mathbf{x}_A)+\mathbf{Q}(\mathbf{x}_B)+\mathbf{Q}(\mathbf{x}_C)}{3}$ to affine-transform the triangle $\triangle_{ABC}$, and then use the above isotropic mesh quality criteria (i.e. $G_{min}, G_{avg}, \theta_{min}, \theta_{avg}, \%_{<30^\circ}$, and angle histogram) to check how similar it is as compared to a regular triangle in the transformed space.

4. Experiments

We implement the algorithms using both Microsoft Visual C++ 2010 and Matlab R2013a. For the hardware platform, the experiments are implemented on a desktop computer with Intel(R) Xeon E5620 CPU with 2.40GHz, and 20G DDR3 RAM.

4.1. 2D Domain Meshing

Fig. 6 shows the meshing result of a 2D square domain $[0, 1]^2$ with 1,000 samples, given varying metric tensor $\mathbf{M}(\mathbf{x}, \mathbf{y}) = diag\{s_1^2, s_2^2\}$, where $s_1 = 1/(0.031 + x)$, and $s_2 = 1/(0.031 + y)$.

Fig. 7 shows the meshing result of a 2D square equipped with the circular anisotropic tensor field:

$$\mathbf{M}(\mathbf{x}) = \mathbf{R}(\mathbf{x})^T diag(Stretch(\mathbf{x})^2, 1)\mathbf{R}(\mathbf{x}), \quad (8)$$

with the rotation field $\mathbf{R}(\mathbf{x})$ and the stretching field $Stretch(\mathbf{x}) \in [1, 10]$ shown in Fig. 7. The final mesh has 4,000 samples.

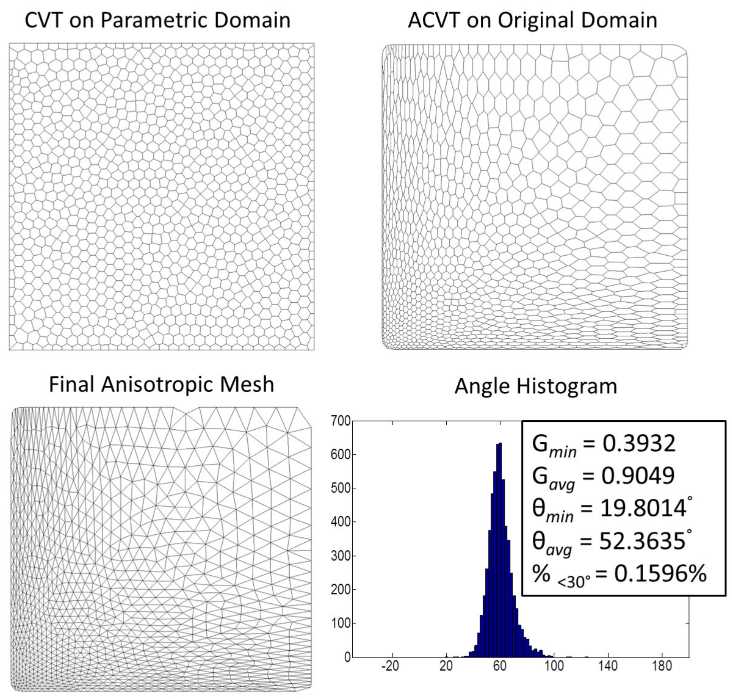


Figure 6: Anisotropic meshing with 1,000 samples on a 2D domain with metric $\mathbf{M}(\mathbf{x}, \mathbf{y}) = diag\{s_1^2, s_2^2\}$, where $s_1 = 1/(0.031 + x)$, and $s_2 = 1/(0.031 + y)$.

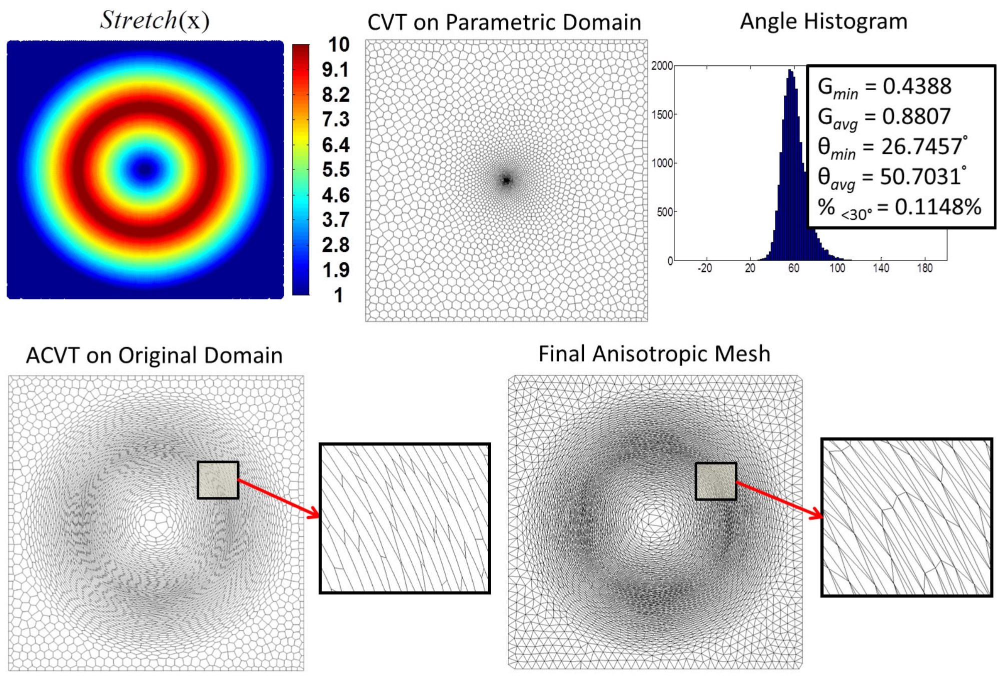


Figure 7: Anisotropic meshing with 4,000 samples on a 2D domain with the circular anisotropic tensor field $\mathbf{M}(\mathbf{x}) = \mathbf{R}(\mathbf{x})^T diag(Stretch(\mathbf{x})^2, 1)\mathbf{R}(\mathbf{x})$, where $Stretch(\mathbf{x}) \in [1, 10]$.

4.2. 2D RGB/Gray-Scale Image-Based Meshing

Our framework can be easily extended to handle the image-based case with RGB or gray-scale colors. Intuitively, it can be seen that the structure of the mesh conforms closely to the color/intensity patterns of the given image [45]. The mesh is fairly isotropic in the areas of constant or linear color/intensity variation; while near the areas of highly non-linear color/intensity variation, the triangles are denser and the edges of triangles are well preserved with the feature edges of the images. This idea is also similar to *Cañas* and Gortler's surface remeshing in arbitrary high-dimensions [46], and Lévy and Bonneel's 6D CVT work in order to achieve a curvature-adaptation [21].

The basic idea is to use the embedding $\phi : \Omega \to \mathbb{R}^6$ or $\mathbb{R}^4$ defined by:

$$\phi(\mathbf{x}) = [x, y, z, sr, sg, sb]^T or [x, y, z, si]^T, \tag{9}$$

where $\mathbf{x} = [x, y, z]^T$, $\mathbf{c}(\mathbf{x}) = [r, g, b]^T$ or $[i]^T$ is the RGB colors or gray-scale intensity to Ω, and $s \in (0, \infty)$ is a user-defined parameter specifying the desired anisotropy. In our framework we can simply compute the edge length in this $6D$ or $4D$ space and parameterize the original image domain. After CVT computation, the final mesh can be reconstructed by mapping the dual triangulation of the Voronoi Diagram from the $2D$ image parametric domain to the original 2D image domain.

Fig. 8 shows the anisotropic meshing result of a 323×323 RGB color Dolphin image with $3,000$ vertices, and $s = 0.1$.

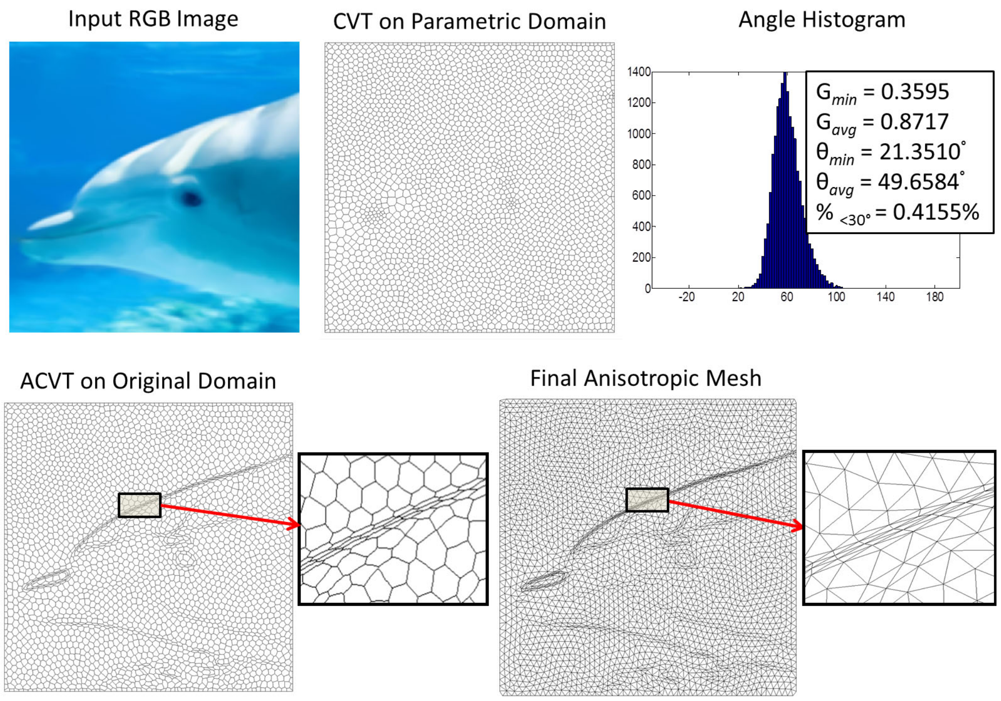


Figure 8: The anisotropic meshing with 3,000 output vertices of a 323×323 RGB color Dolphin image.

Fig. 9 shows the anisotropic meshing result of a 211×211 gray-scale Computed Tomography (CT) medical image with $3,000$ vertices, and $s = 0.5$.

4.3. 3D Surface Meshing

The experiments on the Ellipsoid (genus-0)(Fig. 10), Cyclide, Kitten (genus-1)(Fig. 11, Fig. 12), and the Eight (genus-2) (Fig. 13) surfaces are computed by curvature-based metric tensor fields. We use the metric of Eq. (4) with $s_1 = \sqrt{K_{min}}$ and $s_2 = \sqrt{K_{max}}$, where K_{min} and K_{max} are the principal curvatures. The Ellipsoid surface in Fig. 10 has the anisotropic stretching ratio $\frac{s_2}{s_1} \in [1, 10]$. As for the angle preservation, the only practical conformal mapping

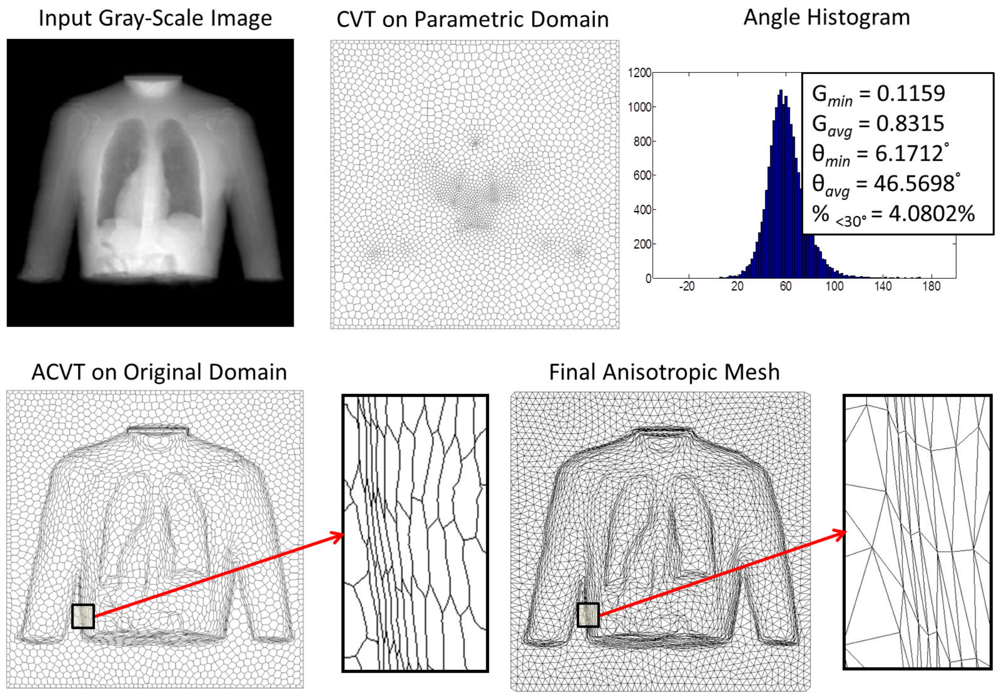


Figure 9: The anisotropic meshing with 3,000 output vertices of a 211×211 gray-scale CT image.

Table 1: Statistics of the 2D domains, 2D images, and 3D surfaces with given metrics and their computational times.

Model	**Figure**	**Input # Vertex**	**Output # Vertex**	**Stretch**	**Anisotropy**	**#Iteration**	**Para Time**	**CVT Time**
2D Varying Metric Domain	6	5,112	1,000	[1,32]	*NA*	100	$10s$	$388.11s$
2D Circular Metric Domain	7	25,717	1,000	[1,10]	*NA*	100	$34s$	$848.29s$
Dolphin Image	8	104,329	3,000	*NA*	0.1	100	$218s$	$6,405.80s$
CT Medical Image	9	44,521	3,000	*NA*	0.5	100	$117s$	$4,113.01s$
Ellipsoid Surface	10	10,242	1,000	[1,10]	*NA*	100	$225s$	$181.71s$
Cyclide Surface 1	11	25,920	8,000	[2,29]	*NA*	200	$36s$	$548.21s$
Cyclide Surface 2	15	21,600	1,000	[2,18]	*NA*	100	$24s$	$39.15s$
Kitten Surface	12	134,438	5,000	[1,5]	*NA*	100	$81s$	$271.79s$
Eight Surface	13	10,000	1,000	[1,5]	*NA*	100	$91s$	$315.59s$

Note: **Stretch**: The user specified stretching ratios $\frac{s_2}{s_1}$ in Eq. (4) for 2D domains and 3D surfaces. **Anisotropy**: The user specified anisotropy s in Eq. (9) for 2D images. **#Iteration**: The iteration numbers of Lloyd CVT computation. **Para Time**: Time of parameterization computation. **CVT Time**: Time of CVT computation.

method for genus zero surface is harmonic map. Harmonic map can tolerate much more degenerate triangles than Ricci flow, but with the price of angle distortion. So that, the mesh quality may be affected somehow. The Cyclide surface in Fig. 11 has the anisotropic stretching ratio $\frac{s_2}{s_1} \in [2, 29]$. Fig. 12 shows the anisotropic meshing of a Kitten surface with anisotropic stretching ratio $\frac{s_2}{s_1} \in [1, 5]$. Fig. 13 shows the anisotropic meshing of an Eight surface with anisotropic stretching ratio $\frac{s_2}{s_1} \in [1, 5]$.

Tab. 1 gives the statistics of all experimental examples, including the 2D domains, 2D images, and 3D surfaces meshed with given metrics.

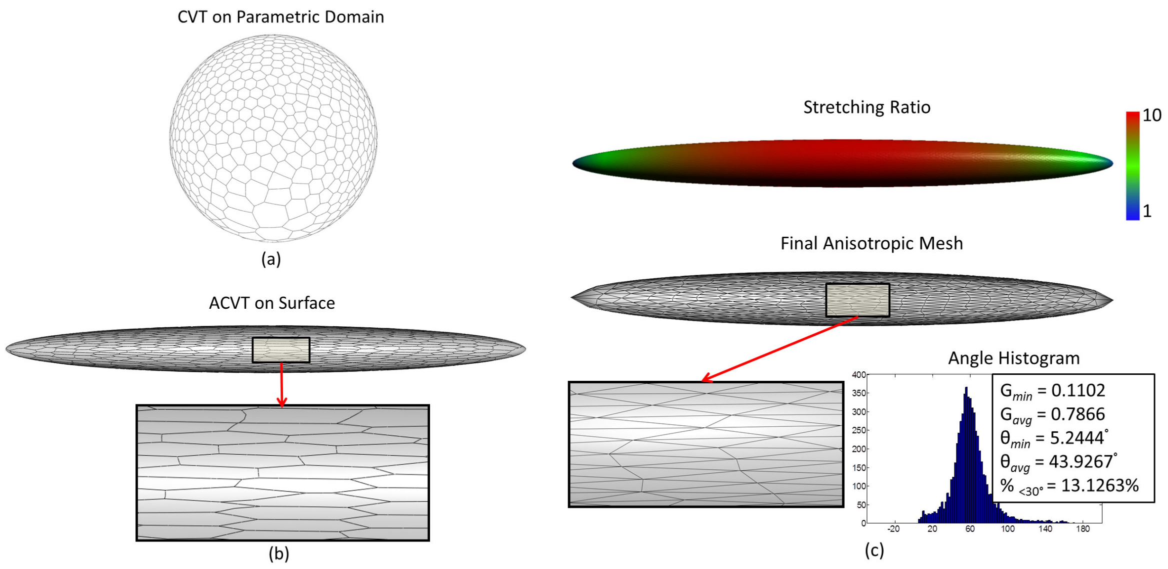


Figure 10: The anisotropic meshing with 1,000 output vertices of Ellipsoid surface with the stretching ratio $\frac{s_2}{s_1} \in [1, 10]$: (a) The CVT on parametric domain. (b) ACVT on surface. (c) Final anisotropic mesh.

5. Comparisons

In this section, we show our comparative analysis and experiments with other anisotropic meshing approaches, including Du and Wang's continuous ACVT approach, Valette et al.'s discrete ACVT approach (Sec. 5.1), and Zhong et al.'s particle-based approach (Sec. 5.2). To compare with other anisotropic triangulation methods, we use the same number of output vertices, the same initializations, and the same number of optimization iterations.

5.1. Comparison with Other ACVT Methods

We compare the generated surface mesh quality and computational speed between our method and two previous ACVT methods: Du and Wang's method with triangle clipping strategy [3] and Valette et al.'s discrete ACVT method [20].

Our implementation of Du and Wang's ACVT method is a little bit different from the original one proposed in [3]. They did some discrete approximation by adding sample points to the given triangular mesh, but our implementation of their method is by clipping the triangles, which is more accurate. The details are as follows: they approximated the anisotropic Voronoi region discretely. Firstly, some points are sampled from edges on a triangle. By testing the anisotropic distance from the sampled points to site points, the triangle is then partitioned into sub-regions (referred from Fig. 4.1 in their paper [3]). Comparatively, in our implementation of their method, firstly we classify each vertex of the input mesh according to the condition whether its incident triangle needs splitting or not. If a triangle is shared by two or more sites, we clip the triangle into sub-regions. There are two cases:

Case 1: A triangle is shared by two sites: when two vertices of an edge belong to different sites, there is a point on the edge with equal distance to these two sites. By representing this point as a linear combination of the two endpoints of the edge, we can compute the point. The other point on another edge can be found in the same way. For each triangle, we connect these two points on two edges; then the triangle is splitted into two regions: a triangle and a quad.

Case 2: A triangle is shared by three sites: we need to find an interior point within the triangle to split the triangle into three regions. It can be done in the same spirit as in Case 1, with the help of barycentric coordinates.

The clipping operation is exact, since the distances from the clipping point to its closest two/three sites are equal.

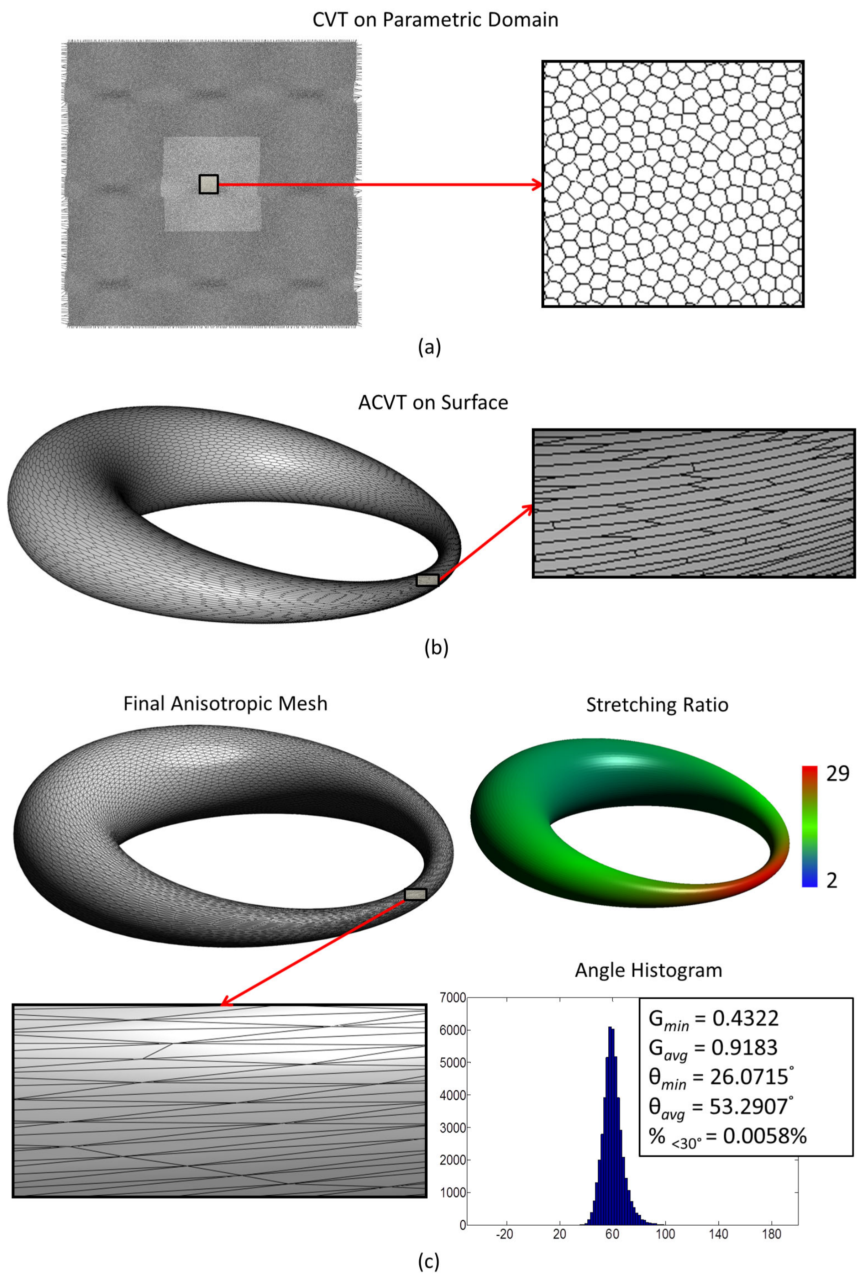


Figure 11: The anisotropic meshing with 8,000 output vertices of Cyclide surface with the stretching ratio $\frac{s_2}{s_1} \in [2, 29]$: (a) The CVT on parametric domain. (b) ACVT on surface. (c) Final anisotropic mesh.

The implementation of Valette's method is the same as the original paper: it is a discrete approximation of ACVT by clustering the vertices of a dense pre-triangulation of the domain, which makes the computation much faster. For each triangle, it can be assigned to at most one site and no clipping is needed.

In the following comparison, we run through 100 iterations, and re-mesh $8,000$ vertices on the Cyclide surface with stretching ratio $\frac{s_2}{s_1} \in [2, 29]$. The input mesh is $25,920$ vertices and $51,840$ faces.

Fig. 14 and Tab. 2 show the comparison results. On the one hand, our approach provides better mesh quality than other ACVT methods after 100 iterations optimization. There may be two reasons to explain why the meshing results of the proposed algorithm are much better in terms of both efficiency and element quality than that of Du and Wang's

Table 2: Comparison of meshing quality for the Cyclide surface.

Method	Time	G_{min}	G_{avg}	θ_{min}	θ_{avg}	$\%_{<30°}$
Ours	$310.10s^*$	0.3771	0.9068	22.5709°	52.4795°	0.0187%
Du and Wang's	$9,749.08s$	0.1989	0.8546	7.9089°	48.5434°	1.7860%
Valette et al.'s	$539.90s$	0.1421	0.7590	5.0111°	39.9945°	10.6049%

Note: 310.10s includes both the parameterization computation time: 36s and CVT computation time: 274.10s.

ACVT method: (1) one significant advantage of this method is its efficiency, since the CVT computation is performed in a 2D parametric domain, as compared to the traditional approach of computing ACVT and its intersection with the surface in 3D space; (2) when computing ACVT on a surface in 3D space, during optimization the sites need to be projected and constrained on the input surface and the gradients are computed approximately in the tangent plane in each iteration. These two factors may affect the convergence speed. Compared with Valette et al.'s ACVT method, because of its discrete nature, it does not work well for highly anisotropic stretching if the input triangulated mesh are not fine enough. We can see that in this example, $\frac{Output\#Vertex}{Input\#Vertex}$ is about $\frac{1}{3}$, so the sampling for Valette et al.'s approach is not dense enough, and this property leads to the poor results of their method in Tab. 2. On the other hand, our approach has faster computational speed. From Tab. 2, we can see that our method, which is a continuous CVT approach on 2D domain, is around 36 times faster than Du and Wang's continuous ACVT method with clipping strategy, and around 2 times faster than Valette et al.'s discrete ACVT method.

5.2. Comparison with Particle-Based Method

In this subsection, we compare the mesh quality between our method and one latest particle-based anisotropic surface meshing method: Zhong et al.'s method [6]. In the last step of their mesh generation, it needs to compute the AVD in 3D space by Du and Wang's method [3], and then intersect it with the surface. However, this approach may lead to disconnected Voronoi cells and non-manifold vertices and edges if the topology control strategy [18] is not considered.

Our alternative parameterization approach can compute the CVT in the 2D parametric domain of the surface, which does not have this problem even if without topology control.

Fig. 15 compares our method with Zhong et al.'s particle-based anisotropic surface meshing method, which uses Riemannian distance to compute a 3D AVD, and then finds the intersection between this 3D AVD and the surface. For the Cyclide surface with the anisotropic stretching ratio $\frac{s_2}{s_1} \in [2, 18]$, the slender parts with the high anisotropic stretching ratios are very close to each other in 3D space, but they are far away along the surface with Riemannian distance. So Zhong et al.'s method needs dense input and output vertices to capture the 3D surface shapes. Taking this Cyclide surface for example, if the numbers of the vertices of the input and output meshes are large enough (such as $Input\#Vertex = 129,600$ and $Output\#Vertex = 8,000$), Zhong et al.'s method with topology control can work well, as mentioned in [6]. If the numbers of the vertices of the input and output meshes are relative small, which does not meet Topological Ball Property [16], their methods may have problems. In this experiment, with $21,600$ input vertices and $1,000$ output vertices, we can see that our method using 2D Euclidean plane CVT has no problem, while their method without topology control generates some non-manifold vertices and edges (shown in Fig. 15).

6. Discussion of Limitations and Future Work

There are some limitations in our current method. If the input triangular surface mesh is not dense enough to catch the variation of the input anisotropic metric, it is highly possible to have illegal edge lengths for some triangles. Without a legal triangulation, we cannot compute the conformal embedding.

Conformal embedding provides a convenient way to embed a given anisotropic metric of a surface onto a 2D domain explicitly, but conformal parameterization methods are in general sensitive to initial triangulation quality and prefer equilateral triangulations due to their discrete approximation property. Specifically, discrete surface Ricci flow uses circle packing to approximate discrete conformal deformation. It is proved in [47] that discrete surface Ricci flow is a gradient flow of a convex energy so we can apply Newton's method when computing conformal embedding with discrete surface Ricci flow. The Hessian matrix is guaranteed positive if two circles associated with each ending vertex of an edge are between $[0, \frac{\pi}{2}]$. It is straightforward to find such circle packing for equilateral triangulations, but if the triangulation of a surface has too many skinny and degenerated triangles, such circle packing will be very difficult to construct. The induced triangulation from the anisotropic metric of a surface is highly possible to have many skinny and degenerated triangles, so we have to sacrifice the anisotropic metric by reducing its stretching ratio along maximal and minimal curvature directions for most of our testing surfaces. This is one biggest limitation of the proposed method, which cannot compute the input mesh with particularly large anisotropic stretching ratios. Moreover, there is only one tool - hyperbolic Ricci for high genus surfaces, and it is very sensitive to the triangulation. So it is the limitation of the conformal embedding method to compute the complicated topology models.

In the future, we would like to explore the possibility to embed the anisotropic metric of a surface in other s-paces. Meanwhile, to speed up the computation of CVT, we would like to take parallel computing of CVT in locally embedded charts of the surface.

Acknowledgements

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the paper. Zichun Zhong, Liang Shuai, and Xiaohu Guo were partially supported by Cancer Prevention & Research Institute of Texas (CPRIT) under Grant No. RP110329, and National Science Foundation (NSF) under Grant Nos. IIS-1149737 and CNS-1012975. Miao Jin was partially supported by NSF CCF-1054996, CNS-1018306 and CNS-1320931.

References

[1] F. Alauzet, A. Loseille, High-order sonic boom modeling based on adaptive methods, Journal of Computational Physics 229 (3) (2010) 561–593.

[2] R. Narain, A. Samii, J. F. O'Brien, Adaptive anisotropic remeshing for cloth simulation, ACM Transactions on Graphics 31 (6) (2012) 147:1–147:10.

[3] Q. Du, D. Wang, Anisotropic centroidal Voronoi tessellations and their applications, SIAM Journal on Scientific Computing 26 (3) (2005) 737–761.

[4] F. Sun, Y. Choi, W. Wang, D. Yan, Y. Liu, B. Lévy, Obtuse triangle suppression in anisotropic meshes, Computer Aided Geometric Design 28 (9) (2011) 537–548.

[5] J. Nash, C^1-isometric embeddings, Annals of Mathematics 60 (3) (1954) 383–396.

[6] Z. Zhong, X. Guo, W. Wang, B. Lévy, F. Sun, Y. Liu, W. Mao, Particle-based anisotropic surface meshing, ACM Transactions on Graphics 32 (4).

[7] P. Alliez, D. Cohen-Steiner, O. Devillers, B. Lévy, M. Desbrun, Anisotropic polygonal remeshing, ACM Transactions on Graphics 22 (3) (2003) 485–493.

[8] A. F. Beardon, A Primer on Riemann Surfaces, Cambridge University Press, 1984.

[9] G. Rong, M. Jin, X. Guo, Hyperbolic centroidal voronoi tessellation, in: Proceedings of Symposium of Solid and Physical Modeling (SPM 2010), 2010, pp. 117–126.

[10] G. Rong, M. Jin, L. Shuai, X. Guo, Centroidal voronoi tessellation in universal covering space of manifold surfaces, Computer Aided Geometric Design 28 (8) (2011) 475–496.

[11] E. Marchandise, J. Remacle, C. Geuzaine, Optimal parametrizations for surface remeshing, Engineering with Computers 12 (2012) 1–20.

[12] Q. Du, V. Faber, M. Gunzburger, Centroidal Voronoi tessellations: Applications and algorithms, SIAM Review 41 (4) (1999) 637–676.

[13] S. Lloyd, Least squares quantization in PCM, IEEE Transactions on Information Theory 28 (2) (1982) 129–137.

[14] Y. Liu, W. Wang, B. Lévy, F. Sun, D. Yan, L. Lu, C. Yang, On centroidal Voronoi tessellation – energy smoothness and fast computation, ACM Transactions on Graphics 28 (4) (2009) 101:1–101:17.

[15] G. Peyre, L. Cohen, Surface segmentation using geodesic centroidal tesselation, in: Proceedings of the 3D Data Processing, Visualization, and Transmission, 2nd International Symposium, 3DPVT '04, 2004, pp. 995–1002.

[16] H. Edelsbrunner, N. R. Shah, Triangulating topological spaces, in: Symposium on Computational Geometry, 1994, pp. 285–292.

[17] Q. Du, M. D. Gunzburger, L. Ju, Constrained centroidal Voronoi tessellations for surfaces, SIAM Journal on Scientific Computing 24 (5) (2003) 1488–1506.

[18] D. Yan, B. Lévy, Y. Liu, F. Sun, W. Wang, Isotropic remeshing with fast and exact computation of restricted Voronoi diagram, Computer Graphics Forum 28 (5) (2009) 1445–1454.

[19] P. Alliez, É. Verdiére, O. Devillers, M. Isenburg, Centroidal voronoi diagrams for isotropic surface remeshing, Graphical Models 67 (3) (2005) 204–231.

[20] S. Valette, J. M. Chassery, R. Prost, Generic remeshing of 3D triangular meshes with metric-dependent discrete Voronoi diagrams, IEEE Transactions on Visualization and Computer Graphics 14 (2) (2008) 369–381.

[21] B. Lévy, N. Bonneel, Variational anisotropic surface meshing with Voronoi parallel linear enumeration, in: 21st International Meshing Roundtable, 2012, pp. 349–366.

[22] H. Borouchaki, P. L. George, F. Hecht, P. Laug, E. Saltel, Delaunay mesh generation governed by metric specifications. part I. algorithms, Finite Elements in Analysis and Design 25 (1–2) (1997) 61–83.

[23] H. Borouchaki, P. L. George, B. Mohammadi, Delaunay mesh generation governed by metric specifications. part II. applications, Finite Elements in Analysis and Design 25 (1–2) (1997) 85–109.

[24] C. Dobrzynski, P. Frey, Anisotropic Delaunay mesh adaptation for unsteady simulations, in: 17th International Meshing Roundtable, 2008, pp. 177–194.

[25] H.-L. Cheng, T. Dey, H. Edelsbrunner, J. Sullivan, Dynamic skin triangulation, ACM-SIAM symposium on Discrete algorithms 25 (2001) 525–568.

[26] S. Cheng, T. Dey, E. Ramos, T. Ray, Sampling and meshing a surface with guaranteed topology and geometry, in: Proc. 20th Sympos. Comput. Geom., 2004, pp. 280–289.

[27] S. Cheng, T. Dey, E. Ramos, Anisotropic surface meshing, in: Proc. ACM-SIAM Sympos. Discrete Algorithms, 2006, pp. 202–211.

[28] J. Boissonnat, C. Wormser, M. Yvinec, Locally uniform anisotropic meshing, in: Proceedings of the 24th annual symposium on Computational geometry, SCG '08, 2008, pp. 270–277.

[29] J. Boissonnat, C. Wormser, M. Yvinec, Anisotropic Delaunay mesh generation, Research Report RR-7712.

[30] F. Bossen, P. Heckbert, A pliant method for anisotropic mesh generation, in: 5th International Meshing Roundtable, 1996, pp. 63–76.

[31] K. Shimada, D. C. Gossard, Bubble mesh: Automated triangular meshing of non-manifold geometry by sphere packing, in: Proceedings of the 3rd ACM Symposium on Solid Modeling and Applications, 1995, pp. 409–419.

[32] K. Shimada, A. Yamada, T. Itoh, Anisotropic triangular meshing of parametric surfaces via close packing of ellipsoidal bubbles, in: 6th International Meshing Roundtable, 1997, pp. 375–390.

[33] S. Yamakawa, K. Shimada, High quality anisotropic tetrahedral mesh generation via packing ellipsoidal bubbles, in: 9th International Meshing Roundtable, 2000, pp. 263–273.

[34] M. Jin, J. Kim, F. L. 0002, X. Gu, Discrete surface ricci flow, IEEE Transactions on Visualization and Computer Graphics 14 (5) (2008) 1030–1043.

[35] X. Gu, Y. Wang, T. F. Chan, P. M. Thompson, S. tung Yau, Genus zero surface conformal mapping and its application to brain surface mapping, IEEE Transactions on Medical Imaging 23 (2004) 949–958.

[36] B. Lévy, S. Petitjean, N. Ray, J. Maillot, Least squares conformal maps for automatic texture atlas generation, ACM Transactions on Graphics 21 (3) (2002) 362–371.

[37] Q. Du, M. Emelianenko, L. Ju, Convergence of the Lloyd algorithm for computing centroidal Voronoi tessellations, SIAM Journal on Numerical Analysis 44 (1) (2006) 102–119.

[38] R. J. Renka, Algorithm 772: Stripack: Delaunay triangulation and voronoi diagram on the surface of a sphere, ACM Trans. Math. Softw. 23 (3) (1997) 416–434.

[39] Cgal, Computational Geometry Algorithms Library, http://www.cgal.org.

[40] F. Nielsen, R. Nock, Hyperbolic voronoi diagrams made easy, in: Proceedings of the 2010 International Conference on Computational Science and Its Applications, ICCSA '10, 2010, pp. 74–80.

[41] F. Aurenhammer, Power diagrams: Properties, algorithms and applications, SIAM J. Comput. 16 (1) (1987) 78–96.

[42] L. Shuai, X. Guo, M. Jin, GPU-based computation of discrete periodic centroidal voronoi tessellation in hyperbolic space, Computer-Aided Design 45 (2) (2013) 463–472.

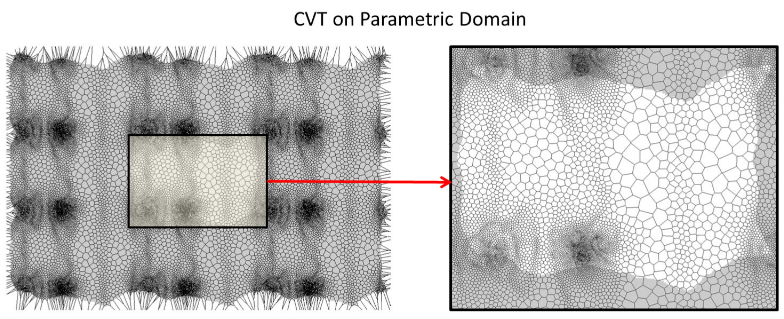
[43] G. Galperin, A concept of the mass center of a system of material points in the constant curvature spaces, Communications in Mathematical Physics 154 (1) (1993) 63–84.

[44] P. J. Frey, H. Borouchaki, Surface mesh evaluation, in: 6th International Meshing Roundtable, 1997, pp. 363–373.

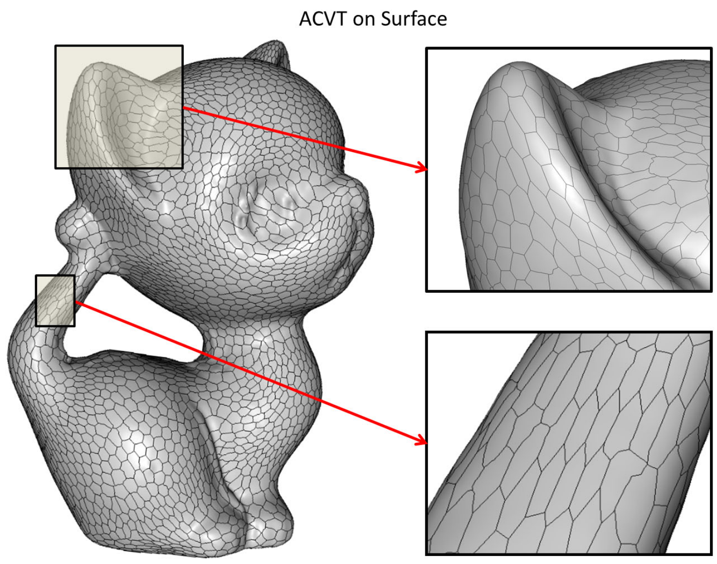
[45] M. Garland, P. Heckbert, Simplifying surfaces with color and texture using quadric error metrics, in: Proceedings of the conference on Visualization '98, 1998, pp. 263–269.

[46] G. *Cañas*, S. Gortler, Surface remeshing in arbitrary codimensions, The Visual Computer: International Journal of Computer Graphics 22 (9) (2006) 885–895.

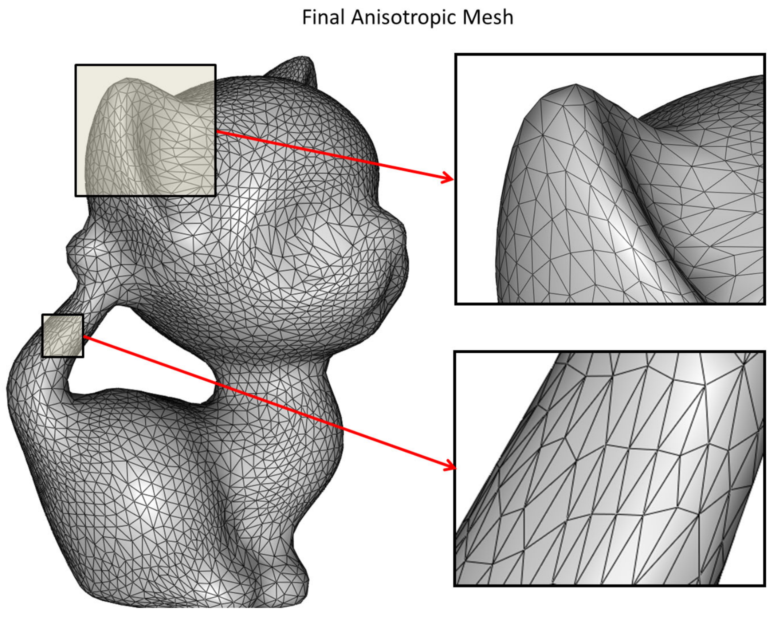
[47] B. Chow, F. Luo, Combinatorial ricci flows on surfaces, Differential Geometry 63 (1) (2003) 97–129.


CVT on Parametric Domain

(a)


ACVT on Surface

(b)


Final Anisotropic Mesh

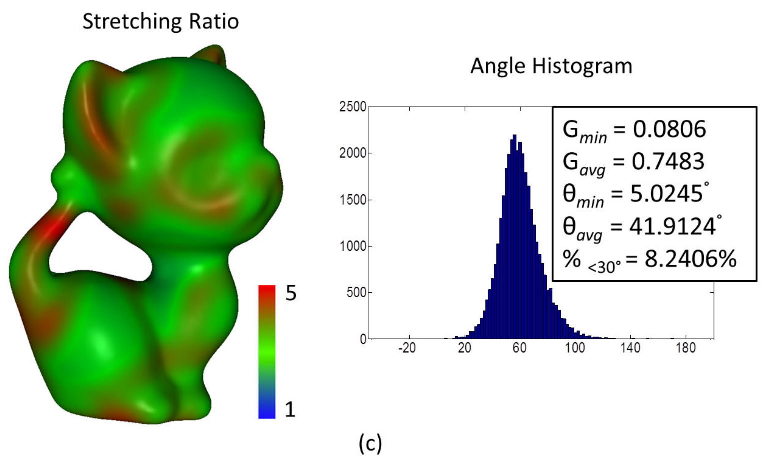


Figure 12: The anisotropic meshing with 5,000 output vertices of Kitten surface with stretching ratio $\frac{s_2}{s_1} \in [1, 5]$: (a) The CVT on parametric domain. (b) ACVT on surface. (c) Final anisotropic mesh.

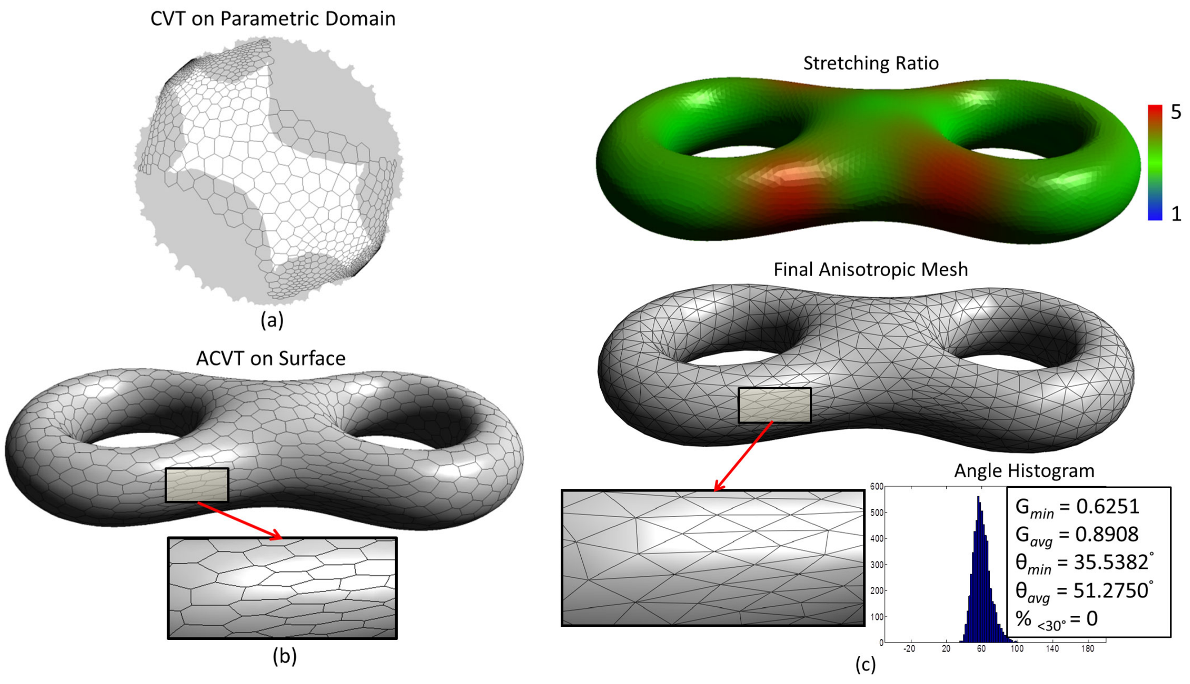


Figure 13: The anisotropic meshing with 1,000 output vertices of Eight surface with the stretching ratio $\frac{s_2}{s_1} \in [1, 5]$: (a) The CVT on parametric domain. (b) ACVT on surface. (c) Final anisotropic mesh.

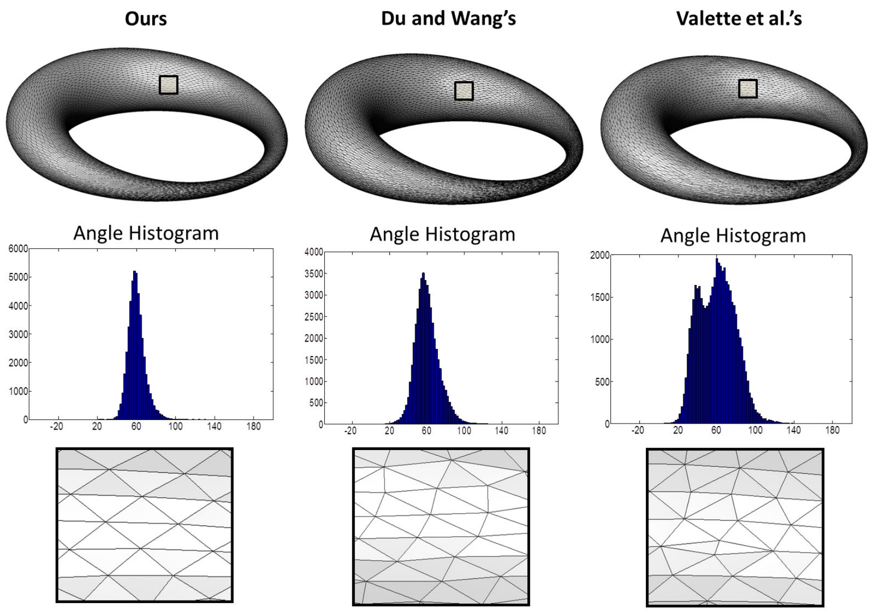


Figure 14: Comparison with other ACVT approaches with 8,000 output vertices.

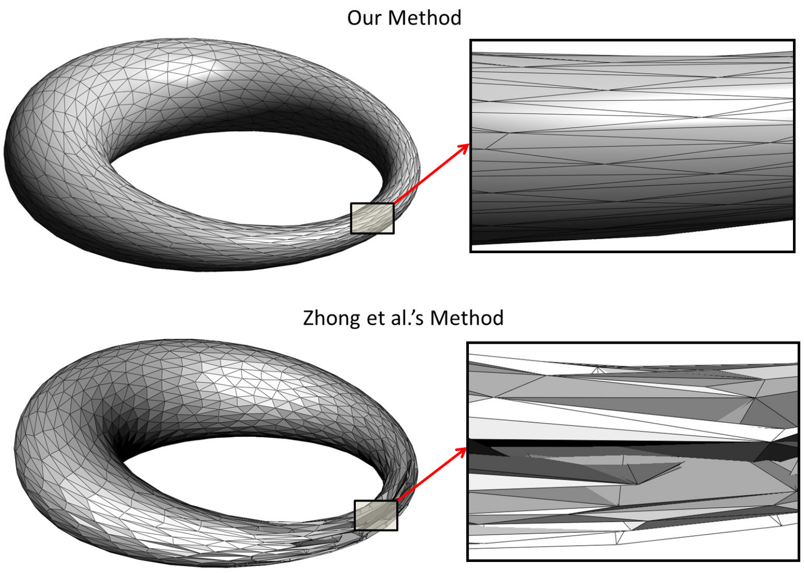


Figure 15: The comparison of anisotropic meshing with 1,000 output vertices of Cyclide surface with the stretching ratio $\frac{s_2}{s_1} \in [2, 18]$ by our method and by Zhong et al.'s method (both are without topology control). The zoom-in parts show the non-manifold vertex and edges existing in their results, while our method does not have.